DUBLIN CITY UNIVERSITY
SCHOOL OF ELECTRONIC ENGINEERING

Academic Performance Analysis using the Google Visualization API

Peter Watters B.Eng MIEI,
August 2010

MASTER OF ENGINEERING

IN

Telecommunications

majoring in Network Implementation

Supervised by David Molloy

Acknowledgements

I would like to acknowledge Mr. David Molloy for his support and help during the completion of this project. I would also like to sincerely thank my family and friends for their continual support and encouragement.

Declaration

I hereby declare that, except where otherwise indicated, this document is entirely my own work and has not been submitted in whole or in part to any other university.

Signed: ..... Date: .....

Abstract

This dissertation is an analysis of academic performance analysis methods of student academic data using the Google visualisation API. The Google Visualisation API is a dynamic charting tool becoming more popular recently and the objective of this project was to investigate the types of academic visualisations that could be represented using the API focusing on the study of individual academic performance, performance of modules and university registration figures.

The research for this project was three fold and encompassed research into best practice statistical techniques to use when analysing student academic data, the latest visualisation technologies available for charting student academic data and the latest web development tools currently available for building interactive user interfaces.

The results of the research show that there are better alternatives to the Google Visualisation API currently available for graphically representing statistical data in the form of the High Charts API and that there are plenty of excellent JavaScript framework libraries currently available to enhance the graphical display of the user interface. The research of academic data analysis showed that linear regression analysis is one of the best means of predicting student academic performance based on previous results and can be a very useful tool to use when analysing student performance.

Table of Contents

Table of Figures

Chapter 1- Introduction

The goal of this project was to research new and innovative ways of analysing and interpreting student academic data. The project was divided into several areas of research and the end product of the research is an interactive visualisation web application. Several areas of research were covered and research was done into:

- Statistical data analysis and data mining
- Analysis of student academic performance
- Plotting visualisations using the Google Visualisation API
- Researching new and innovative ways of displaying graphic data to the user.

The first steps taken were to perform a full analysis of the problem at hand and to come up with a project plan and timeline for completion. This is discussed in further detail in the following subsections.

1.1 Problem analysis

The Google Visualization API allows users to select from sources of structured data to display data in a range of different representations. The Google Visualization API enables users to represent their data, stored on a data source, as a visualization compliant data source. This allows users to generate reports and dashboards as well as to analyse and display data effectively.

The aim of this project was to Use the Google Visualisation API to analyse student registration and academic data. The focus was on key aspects such as registration growth/decline, module average marks and individual performance. Integration of existing data sources and extraction of data from these sources into suitable formats for visualisation was required.

1.2 Project planning

This project entailed a huge amount of research and development work so an effective plan needed to be decided upon in order to ensure all aspects were covered effectively. Tasks were divided and a week by week plan was decided upon. A week entailed 40 hours work which could be spread out over a longer time but allowed the 400 hours total work required to be broken up into 10 week chunks to make the workload easier to manage.

As there is a lot of different research involved in this project it was quite easy to get side tracked and dwell too much on certain things, over-emphasising particular areas of research. This plan came in useful to prevent this from happening as the overall goal of the project was always in focus and the steps to get there clearly defined.

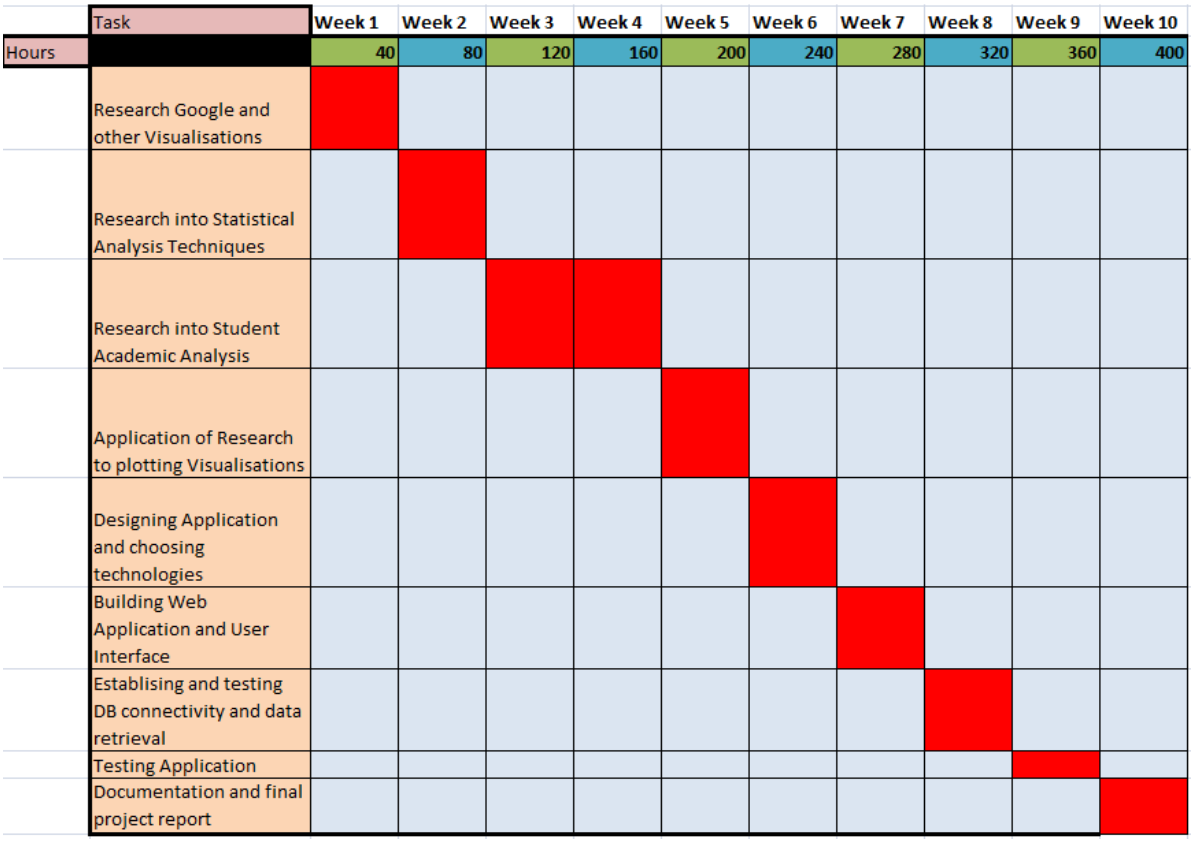


Figure 1 Project Plan

As you can see in Figure 1 above the work was divided into 40 hour weeks with a week dedicated to each task. The only task that took longer than this was the research done into techniques used to analyse student academic data as this is the foundation of this project. There is a vast amount of information available on this topic and research could have continued for months in this area so it was difficult to focus on the key factors suitable to this paper. The most relevant information found is covered in depth in section

Chapter 2- Technical Background

There are some technical details that are referenced throughout this paper that are important to understand in order for the reader to realise the full potential of the research. Many of these details are discussed in the sections below where the Google Visualisation API and basic statistical analysis techniques are explained and discussed in order to improve the understanding this research of and to make it as globally useful as possible.

2.1 Google Visualisation API

The Google Visualisation API is a java script code library that allows manipulation of structured data to be displayed using a wide selection of visualisations. Users can represent their existing data source as a visualization compliant data source and display the information on web pages or dashboards. The Google Visualisation API allows for a better graphical representation of information and allows for effective analysis of data. There are a huge range of visualisations to choose from for many purposes and Google visualisations are dynamic supporting user interaction to aid in bringing the data to life.

Google Visualisations require internet access to work effectively as they call java script code from http://www.google.com/jsapi (1). This has both advantages and disadvantages in that although you are always running the latest version of the API it simply will not work at all if not connected to the internet. Google visuals also use Adobe Flash to display in the browser window. This means that users will need to install this plug-in to be able to view it in their browser, which again would affect users without internet access as they will not be able to download the latest version.

While it does have an over-reliance on plug-ins and the internet there are many benefits found using Google visuals. There is a very high quality of some of the dynamic visualisations such as the annotated timeline and motion chart and they can really bring data to life. Example screenshots of these are shown in Figure 2 and Figure 3 but as these are static images they don't show the full interactivity available.

Google Annotated timeline:

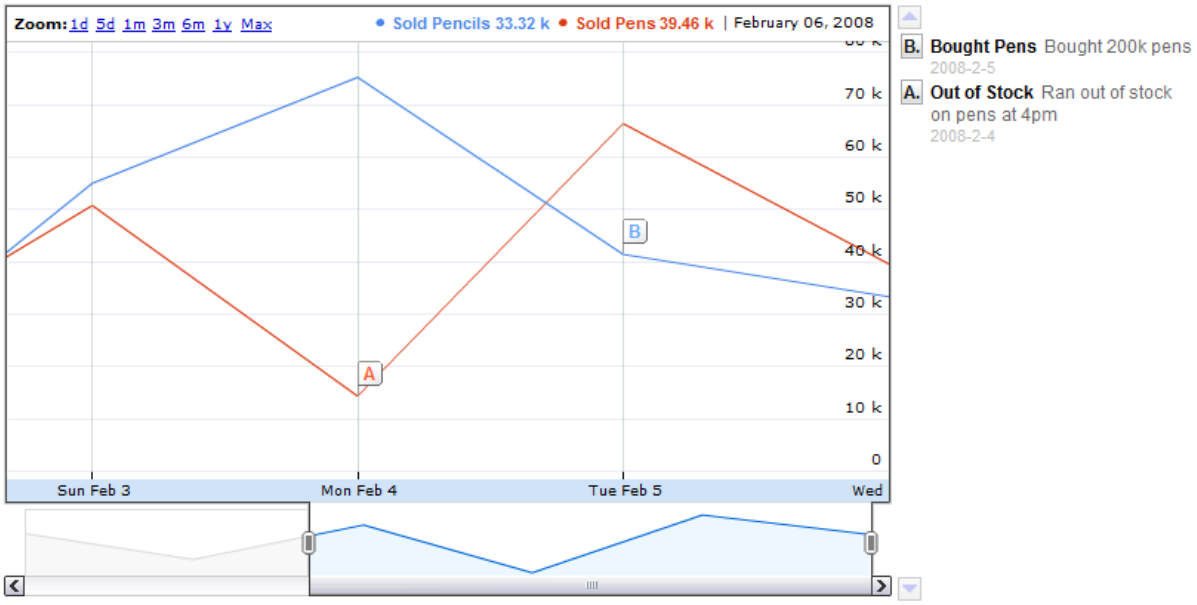


Figure 2 Google Annotated Timeline (2)

The annotated timeline is a dynamic chart that allows the user to track variables over a period of time. There are various dynamic features such as the ability to add annotations and zoom to different scales. Users can hover over a point on one of the lines to view the value of these particular points and use the dragabble timeline at the bottom to focus on a particular period in time.

Google Motion chart:

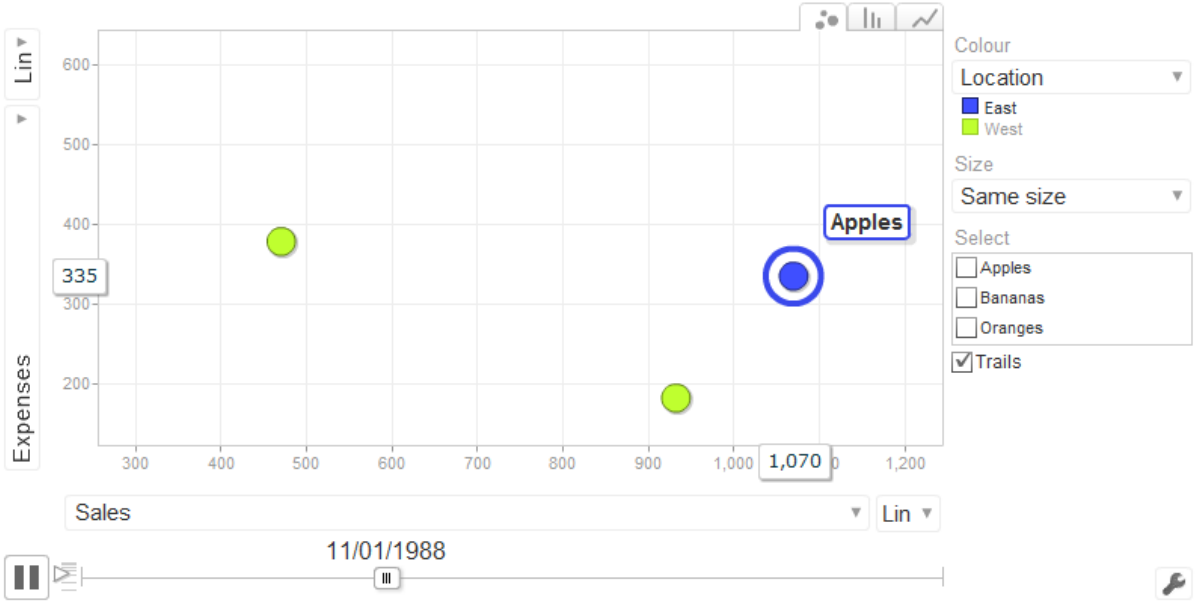


Figure 3 Google Motion Chart (3)

The motion chart is the best visualisation available as it offers viewing of multiple variables and their relationships in one place. It has been successfully used by GapMinder (4) to show important world trends about populations. Grouping by country and geographical region they offer a range of analysis metrics such as C02 emission per person, children's math scores, income per person and infant mortality rate. The possibilities are in fact endless and this is a great example to show that this one chart can be used to examine all possible aspects of any analysis in this or any field.

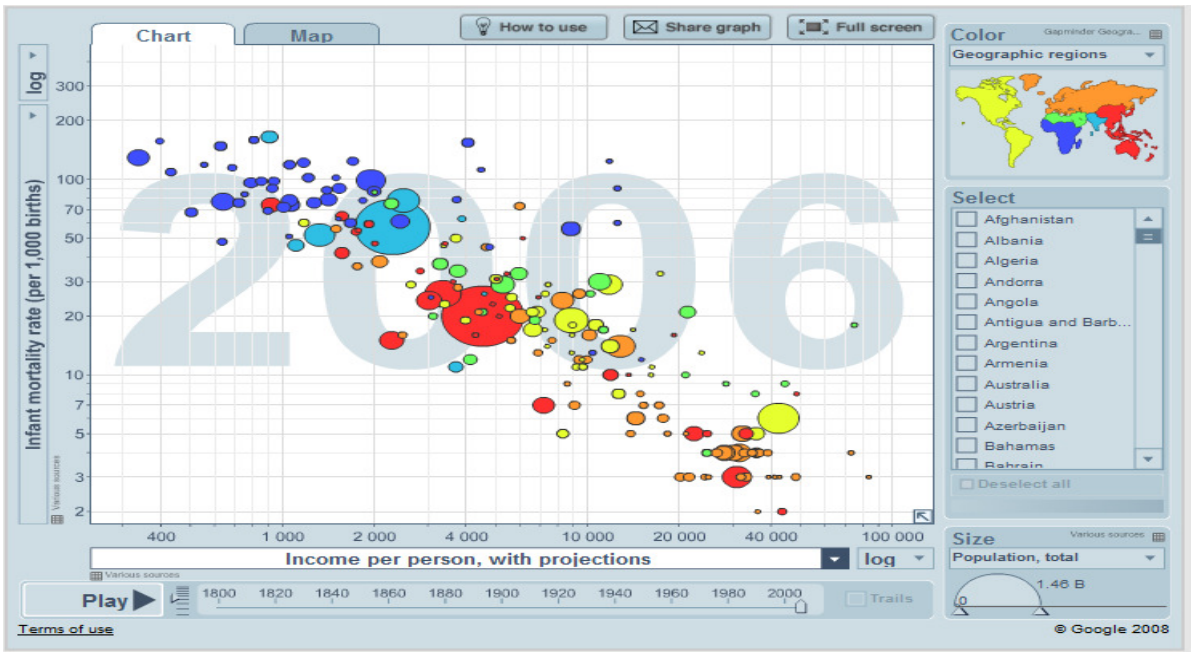


Figure 4 Gapminder Motion Chart Overview (4)

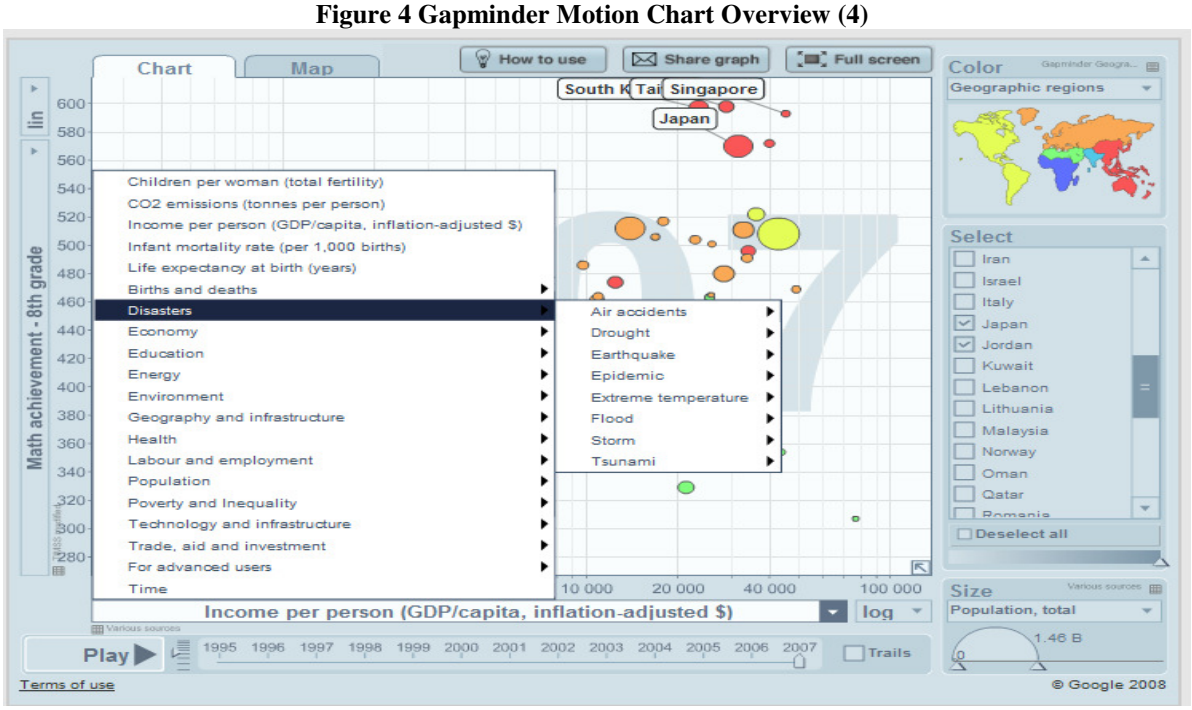


Figure 5 Gap Minder Motion Chart Drill-Down (4)

The API also offers a wide range of more basic visualisations with a high focus on graphic quality. Visualisations can read data from a wide variety of live data sources using various languages. Data can even be loaded directly from Google spreadsheets and used for visualisations. An example of some sample image charts available freely to download from the Google Chart homepage (5) are shown in Figure 6 Google Visualisation API: Area Chart (6) and Figure 7 Google Visualisation API: Bar Chart below.

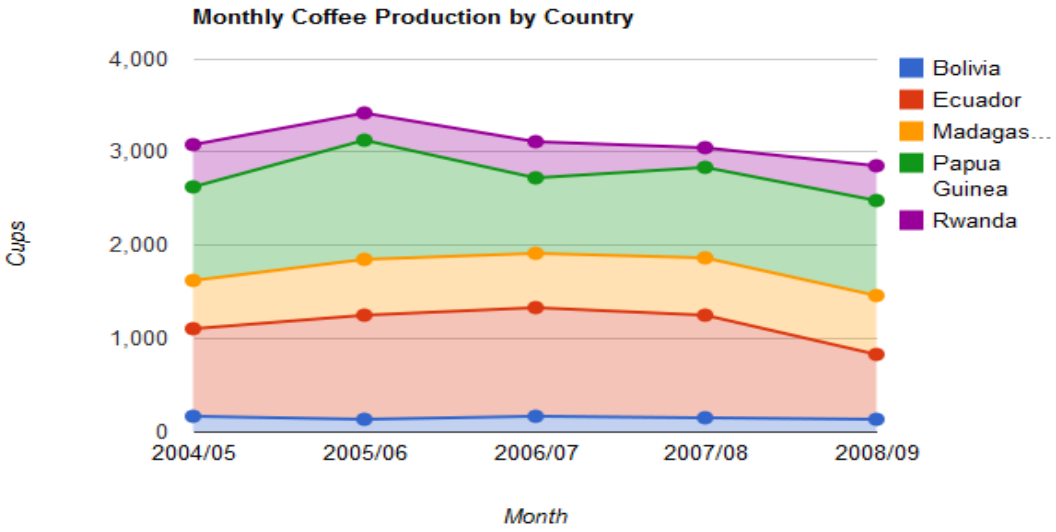


Figure 6 Google Visualisation API: Area Chart (6)

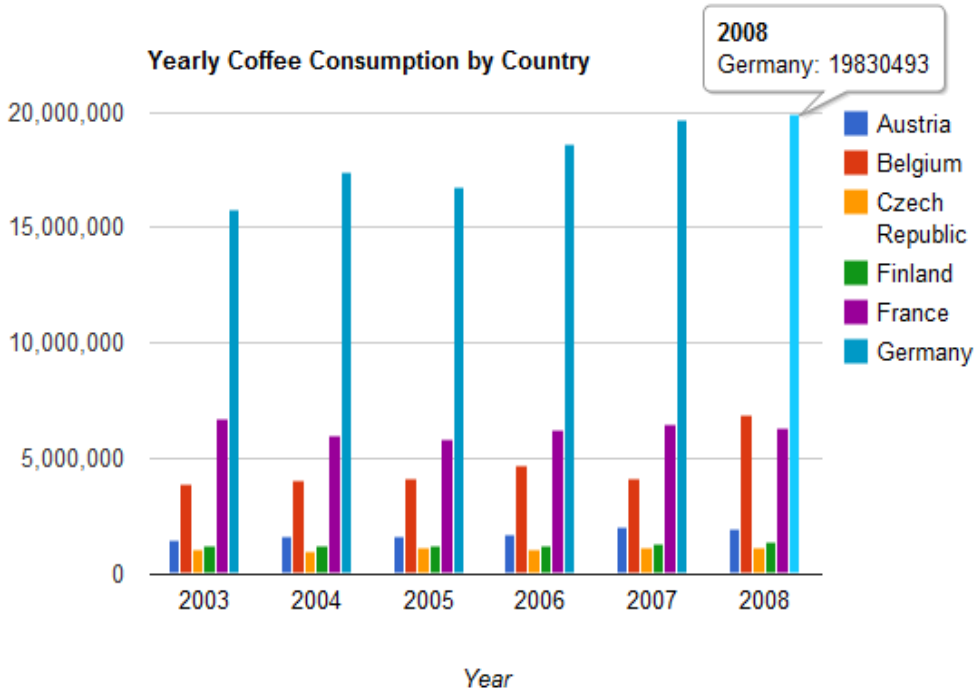


Figure 7 Google Visualisation API: Bar Chart (7)

Another benefit of Google Visualisations is the availability of the Google code playground (8). This provides an online AJAX based editor where visualisations can be coded and viewed live so developers can edit the visualisation code here, preview the results, output the resultant HTML and then copy this to their web server. This can save a lot of development time and is a really good way of seeing all of the available visualisations in one place along with the background code. One drawback of the Google code playground is that it is not supported by Microsoft Internet Explorer however this is almost irrelevant as most serious web developers who will use the tool undoubtedly stay clear of IE for all purposes except final testing.

Use of the Google code playground also gives access to examples of a lot of other Google API's such as Google Language which allows for translation of web pages and other libraries which allow users to load java script libraries such as jQuery and MooTools (9) directly from Google rather than call them locally. This places additional data transmission overhead which can slow down the application however where high speed internet is available it is suitable and it ensures the latest libraries are always used.

2.2 Statistical Analysis Techniques

Visualisations are excellent ways of displaying data but it is very important to plot the most useful things that can be inferred from the data rather than plotting meaningless graphs that don't really convey any real worth. Statistics can be used to prove many things and bias situations so an important part of the research for this project was to research the subject of statistical analysis in order to have an educated opinion on what data would be the most useful in an analysis of student performance.

In order to analyse and predict student results there is a need to use statistical inference which is basically using information obtained from a data set to make observations about the entire data set. Some good background reading for someone interested in learning more about the theory behind these techniques explained can be found in "Probability and Statistics" (10) by Murray R. Spiegel and "Statistics for Dummies" (11) by Deborah Rumsey both of which provide numerous worked examples and are very thorough in their explanations.

Statistical inference allows for prediction of future values of unknown quantities by using data already available (12) . In the case of this project the data set is the student academic database and the aim of this thesis is to use the results and other academic information available there to both analyze current trends and predict future variables such as student grade point average(GPA) and pass and fail rates. An extremely useful reference paper in this field and for analysing data in an academic context is "Statistical data: The underestimated tool for higher education Management" (12).

This paper (12), addresses the issue of using statistical data to aid universities in their analysis of student performance particularly focusing on Makerere University in Uganda. Their research highlighted the importance of universities collecting relevant data to help them make informed decisions about student performance. They found that there were several problems with the way the university was currently managing its data including problems with duplication, presence of useless data and the lack of good data. This paper is a great overview of studying the performance of both students and universities and highlights the need for continued data collection and analysis of university performance to help improve it for all concerned. Statistical analysis techniques are a key feature of academic analysis so it is important to give an overview of the techniques mentioned extensively throughout the research.

2.2.1 Mean

The mean of a data set is a commonly used analysis technique and can be more easily explained as the average value of a series. The mean can be an extremely useful tool when inferring data from statistics as it allows visibility of how common a particular value is or how uncommon it is. For example a student may have an average GPA of 60% which is quite respectable however if the average GPA of the class is 80% this student is below average and their average GPA is not actually as good as it would have initially seemed.

2.2.2 Mode

Mode is a statistical technique that helps distinguish more frequent variables in a data set. The mode is a value in the data set with a frequency of no less than that of other values in the set. This is more easily explained by using an example of a student. If a student were to receive the following grades in a semester: 40%, 55%, 60%, 40% the *mode* is 40% as it occurs twice in the dataset and the others occur just once. Another way of putting this is to

say that 40% has a frequency of 2 and the others have a frequency of 1. This frequency of 2 is more commonly known as the *modal frequency*.

2.2.3 Standard Deviation

Standard deviation is literally defined as "*the square root of the variance*" (13). In lay mans terms the standard deviation of a data set is described as an estimation of the variation of values in the data set about the mean value. If the standard deviation is small it can be inferred that values in the data set are close to the mean value. If the standard deviation of a data set is large it means that more values are distributed at a greater distance about the mean. In the example of a student having a large standard deviation of GPA would mean inconsistency in results and having a low standard of GPA would mean the student is performing consistently however well or poorly they are performing.

2.2.4 Standard Error

Standard error is an extremely useful statistical technique and allows analysts to determine the accuracy of statistical predictions. It is a measurement of the accuracy of a statistical technique based on the sample size used to perform that prediction. The standard error of the mean of a sample can be calculated by dividing the standard deviation of the population by the square root of the sample size. It can be used to help understand the accuracy of graphs and estimate errors in calculation and data reliability.

2.2.5 Confidence Interval

A confidence interval is an interval in data within which there is a high probability the mean of the population will fall, or in other words it's a section where the average of the sample population will be represented. When analysing samples of data, in the case of this project student data, each sample taken is likely to produce a different mean depending on the parameters used to limit the sample. These variations in mean are usually limited to the interval of the mean population and thus a confidence interval may be formed*." If you've computed a mean for a sample, you can compute a confidence interval based on the probability that your mean is also applicable to the population as a whole.*" (14)

2.2.6 Correlation Co-efficient

A correlation co-efficient can be used to measure the strength of a relationship between two variables and could be used in this project to measure relationships between student

performance. A scale of -1 to 1 is used to measure how related two samples are. 1 means the samples are fully correlated and 0 means they are not related at all. If the correlation co-efficient is -1 they are oppositely correlated, which means that for every change in one sample size there is an equal and exact opposite change in the other sample.

2.2.7 Linear Regression

Linear regression is a form of regression analysis which can prove extremely useful in data analysis. The relationship between one or more independent variable and another variable is modelled by a least squares function called a linear regression equation. A linear regression equation with one independent variable represents a straight line when the predicted value is plotted against the independent variable as shown below. (15)

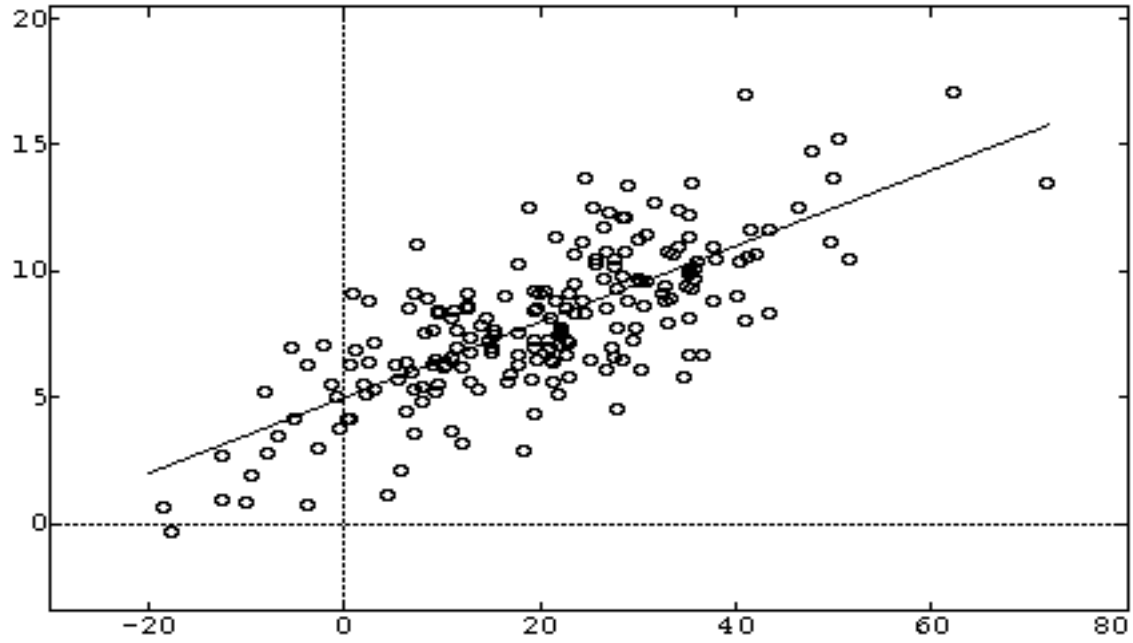


Figure 8 - Linear Regression Analysis (15)

Linear regression is a key analysis technique used when modelling student data and appeared frequently in various papers reviewed as part of this project. It is an excellent way of predicting the future value of a variable based on that of another.

2.2.8 Linear discriminant analysis

Linear discriminant analysis is a common technique used to find linear combinations or patterns to characterise classes of objects or events. It was first used in the financial world to help characterise which banks would fail and succeed: *"In bankruptcy prediction based on*

accounting ratios and other financial variables, linear discriminant analysis was the first statistical method applied to systematically explain which firms entered bankruptcy vs. survived." (16). The same techniques could be employed to student analysis to determine which students pass and fail their courses or drop out altogether.

2.2.9 Prediction/ Probabilities

Probability and probability theory form a key part of statistical analysis and are a vital tool when analysing any data source. Probability theory is a specific branch of mathematics dedicated to the analysis of random phenomena which is quite useful in the prediction of future data (17). Probabilities can be used when analysing students to predict how likely they are to succeed or fail based on those who have gone before and similar characteristics to those students.

An example of this would be if 90% of male students aged 18-23 fail a particular course then there is a high probability that a 19 year-old male student currently taken the module will fail. If calculated these probabilities can give a more informed prediction of how well students will do in exams and really aid in prediction accuracy.

2.2.10 Decision Trees

A decision tree is a tool used to help make decisions by forming a tree-like model of all the possibilities of a scenario. These techniques are commonly used when analysing data and can be applied to student behavioural analysis very easily. An example decision tree is shown in Figure 9 Example Student Decision Treewhere a student's risk of failure is decided upon using predefined variables.

The decision tree makes some assumptions and uses these to classify the student's failure risk. Examples used are:

- Male students have a higher risk of failure
- 1st year students are more likely to fail than those in later years
- Students aged 18-22 are more likely to fail
- Students who have failed before are more likely to fail
- Those with a poor continuous assessment average are more likely to fail

Figure 9 is a simple example of how a decision tree could be used in analysing student data and could be extended to include more detailed analysis of specific student traits. It is not a

good way of plotting student behaviour but helps in the analysis stage when deciding what factors may affect student performance. It was useful in this project when deciding on analysis metrics for student academic data.

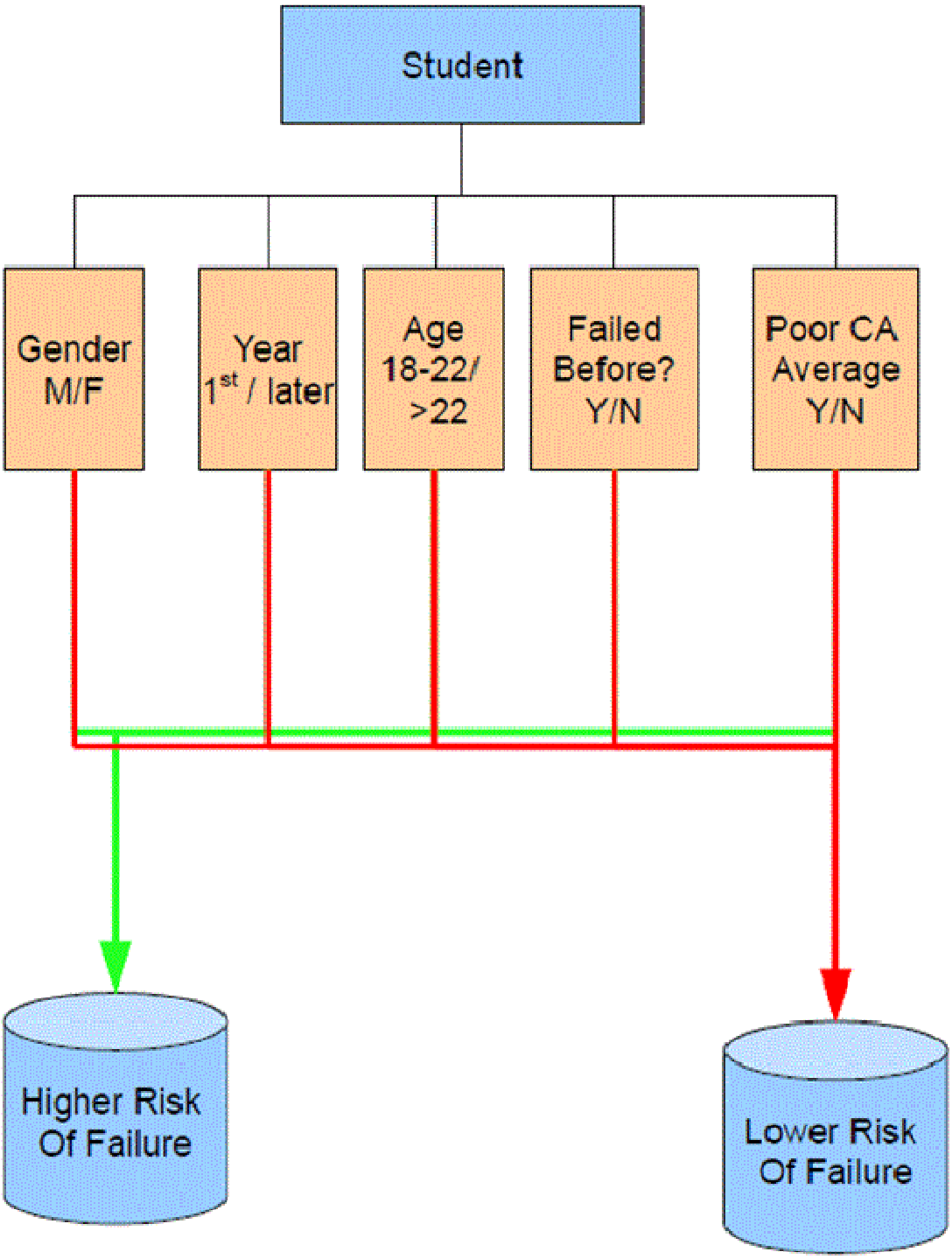


Figure 9 Example Student Decision Tree

2.2.11 Conclusion

After initial research into background technologies was performed a clearer idea of the project unfolded and aided greatly in the research that followed. The best and worst aspects of the Google visualisation API were examined in detail and this gave way for research into its competitors in the visualisation market. Other alternatives with improved performance on and off-line and with improved graphics without the requirements for browser plug-ins were sought to improve the accessibility and usability of the final application.

The initial research into statistical analysis proved vital to aid in understanding further papers specific to student academic performance covered in the latter stages of the research. Initial research was needed to pinpoint the most accurate ways of mapping and predicting statistical data available to help narrow down the fields of research when looking at analysis student and university academic data specifically. The techniques discovered appeared quite frequently and the initial research into linear regression analysis in particular proved to be extremely worthwhile.

Chapter 3- Research

The research for this project was divided into three fronts as a lot of things needed to be taken into consideration to decide on the key factors to building this application. The three key areas of research were:

- Visualisation technologies
- Student data analysis and prediction
- New web technologies for enhanced user interaction

These areas were vital as the end product of the project is an application to plot student data that needs to look and perform well and present interesting ways of analysing student data. There are vast amounts of visualisation technologies available and in order to fully appreciate and see the disadvantages of Google visualisations a detailed comparison needed to be done of them compared to their competitors.

Student data analysis was an extremely important part of the research for this application as it is a very active field of research and in order to make the most of the data provided and try to help DCU improve the performance of students a comprehensive study of other research in this area needed to be done. The end product of a visually appealing application isn't of as much benefit to the university as a more intelligent and well researched analysis of the best methods for analysing and predicting student performance so research into case studies in this area was crucial to this project.

Further research was also done into new web technologies such as jQuery and the jQuery UI which help improve the user experience on a web application and also into advanced statistical techniques for analysing and predicting data. This was to ensure completeness and universal relevance of the research and to make this paper as useful as possible for future researchers in these areas.

3.1 Visualisations

The initial research began by working with Google Visualisations to learn how they work and understand how to use them. Some disadvantages were found immediately for example the Google API needs to access googlecode.com to work. This means if there is no internet access the visualisation application will not load and will not work. Unlike other visualisation libraries it is not possible to download Google Visualisation code directly to a

web server to be called locally. It is not good enough to produce software that overly relies on external things like an internet connection for example if it only worked in one browser there would be outrage.

The application should perform well even without internet access, as long as a user is on the local network and has sufficient authentication to access the database this should be good enough. For security reasons regarding the security of academic data it would be better to use this application without internet access and behind the DCU firewall as this would be a lot safer and would prevent people from snooping and trying to crack into the database. If someone gained access to the DCU student database they could cause a lot of damage giving people excellent grades or terrible grades and correcting damage done would take a lot of work.

For these reasons and to obtain a general overview of the types of visualisations out there and what they are capable of the first step taken in the research for this project was to analyse the visualisation software currently on the market. Several technologies were investigated including:

- Google Charts
- Visifire
- HighCharts
- AM Charts
- Axiis
- RGraph
- Fusion Charts
- HTML 5 Canvas

Each of these technologies is discussed in further detail in the following sections.

3.1.1 Google Charts

Google chart is a more primitiive version of the Google Visualisation API offering more basic non-dynamic image charts to represent data. Even though they are more basic versions of Google visualisations the loading time is no better and they still perform poorly at slow internet speeds. However over high speed connections they offer an excellent means of charting data and much more. Various interesting applications of Google Charts can be

found in “50 Google Charts Tricks for Your Next Classroom Presentation” (18) which proved a valuable resource in understanding just what could be achieved using the API.

Unlike Google visualisations Google Charts can be embedded on top of one another and any chart type can be embedded inside a bar, line, radar or scatter chart. This is useful when several things need to be analysed in detail on one chart and an example of this is shown below:

Figure 10 Google Chart API Line Chart (19)

Charts can be fed parameters using a URL or even generating using pre-existing html tables on a page (5). A sample URL to display a chart is:

```
http://chart.apis.google.com/chart?chs=250x100&chd=t:12,16,16,24,26,28,41,
51,66,68,13,45,81|16,14,22,34,22,31,31,48,71,64,15,38,84&cht=s&chm=o,0000F
F,0,-1,0|o,FF0000,0,0:9:,5|D,000000,1,10:,1,-1
```

If entered into a browser it will display the chart in Figure 11 Google Chart Example Line Chart.

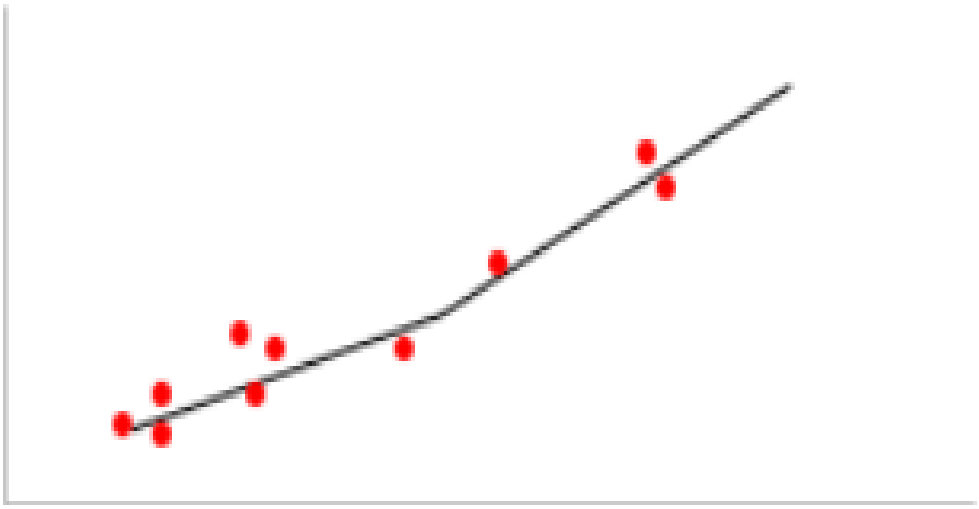


Figure 11 Google Chart Example Line Chart

This makes it extremely easy to feed data into the charts however they still need to reference Google's online code repository and therefore rely on internet access. An application shouldn't have to rely on external services to function correctly and this began to lead the research away from Google code to see what competitors were currently available and their advantages.

3.1.2 Visifire Visualisations

Visifire visualisations (20) are an equivalent of Google's however they do not rely on internet connectivity to work. They require the local inclusion of a java script file which much be present in the same directory for every page that wishes to use Visifire visuals. They use the Microsoft Silverlight plug-in to draw visuals in the browser. This is Microsoft's equivalent of adobe flash player and can produce some highly aesthetically pleasing graphs are shown in Figure 12 Visifire Column Chart Figure 13 Visifire Area chart Figure 14 Visifire Candlestick chart Figure 15 Visifire bar chart with trend line .

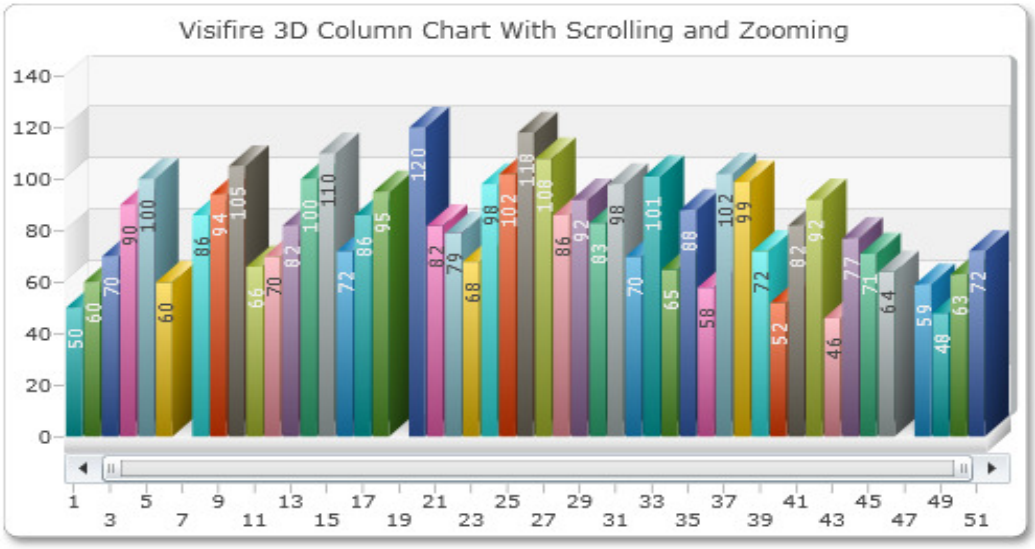


Figure 12 Visifire Column Chart (20)

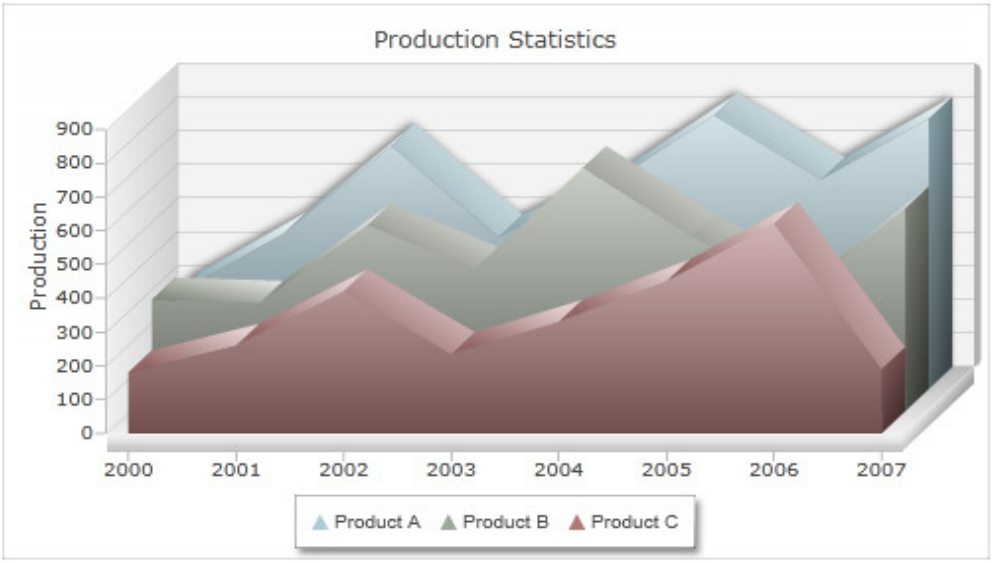


Figure 13 Visifire Area chart (20)

A distinct advantage of Visifire is its ability to work on local machines without internet access however there is still a reliance on the Silverlight plug-in. If the user doesn't have

this already they will need internet access to install and similarly to update it. The necessary file is included on the CD accompanying this report, see Appendix 5 – Contents of accompanying CD for further details.

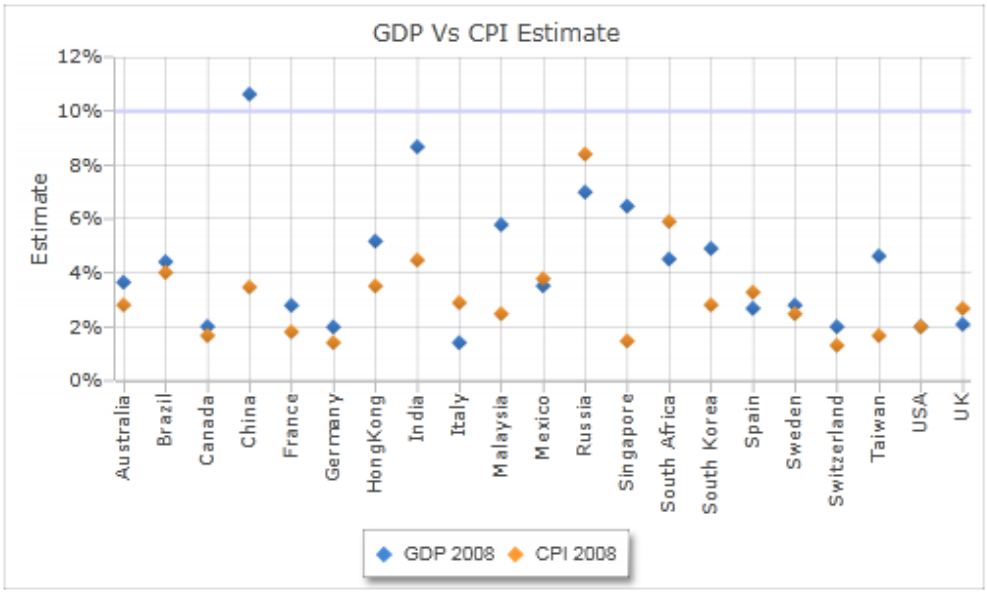


Figure 14 Visifire Candlestick chart (20)

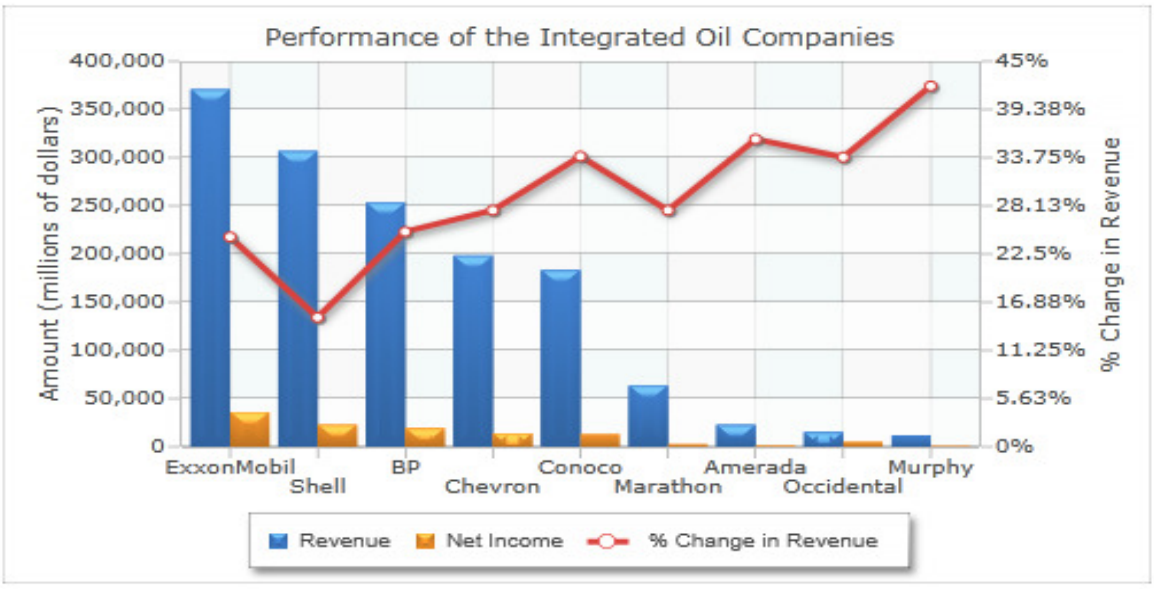


Figure 15 Visifire bar chart with trend line (20)

There is however a lot of repetition introduced if Visifire charts are used on multiple pages and in a large scale application which is well designed and sections of pages are separated into folders for security. If Visifire was to be used in such a setting its underlying JavaScript and .dll files would need to be present in every single directory and this leads to a lot of

repetition. Referencing them a directory above or below current in your web page simply won't work; they need to be in the same folder.

3.1.3 HighCharts

During the course of the visualisation research one of the best solutions found was HighCharts (21). This is a pure JavaScript library freely available for plotting charts without the need for any internet access or browser plug-ins. All that's needed is the inclusion of a JavaScript file anywhere on the web server and this can be referenced to plot graphs anywhere in the application. Another key advantage is that it's cross browser compatible including iPad and iPhone.

The charts are rendered using the HTML5 canvas tag in most browsers and using VML in Internet Explorer as this does not as yet fully support HTML5 and the canvas tag. No client-side plug-ins are needed which breaks from dependence on Flash or Silverlight. All that's needed to render HighCharts is the HighCharts core java script file and either the jQuery or MooTools framework (both are common web development java script libraries).

There is a huge library of charts including some very interesting combination charts like the chart below which comprises of a bar chart with a trend line and an embedded pie chart. HighCharts allow the developer to draw charts with numerous axes and plot several different chart types over one another which can really help convey as many trends as possible on one chart. An example of this chart is shown in Figure 16 High Charts Combination Chart below.

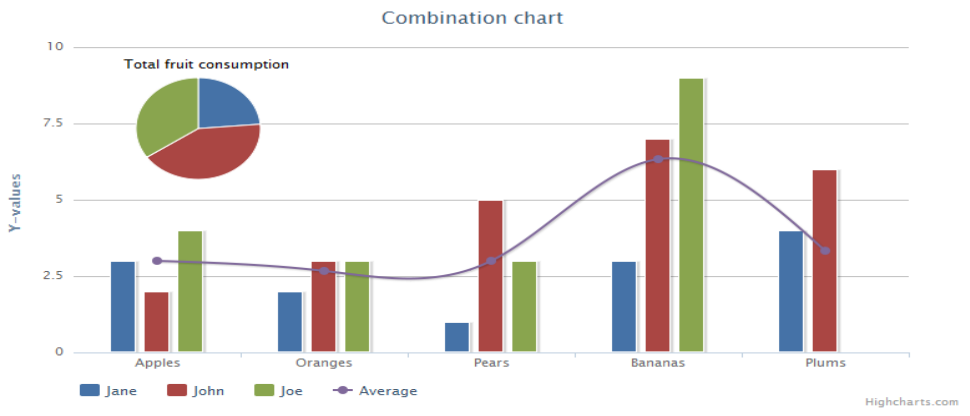


Figure 16 High Charts Combination Chart (20)

The chart in Figure 17 High Charts Dual Axis Combination Chart is an example of a dual axis chart with trend line. The user can click on the variable title to show/hide the bars or trend line as required.

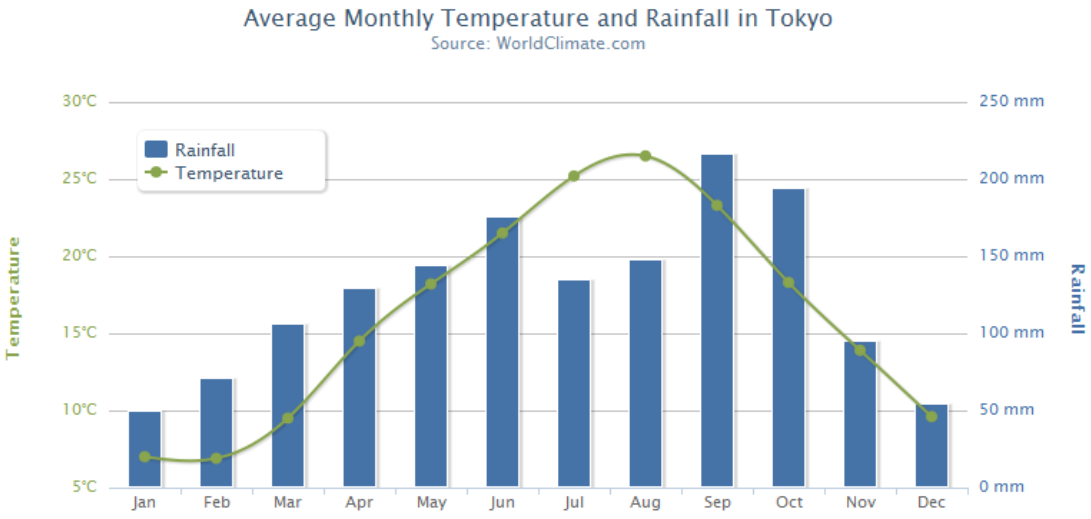


Figure 17 High Charts Dual Axis Combination Chart (20)

The API also provides a ready built chart for plotting regression which is shown in Figure 18 High Charts Scatter Plot with regression line , this is something which would have taken a lot of work to develop using Google Visualisations. This is ideal for use in linear regression analysis of student grades and can use multiple trend lines.

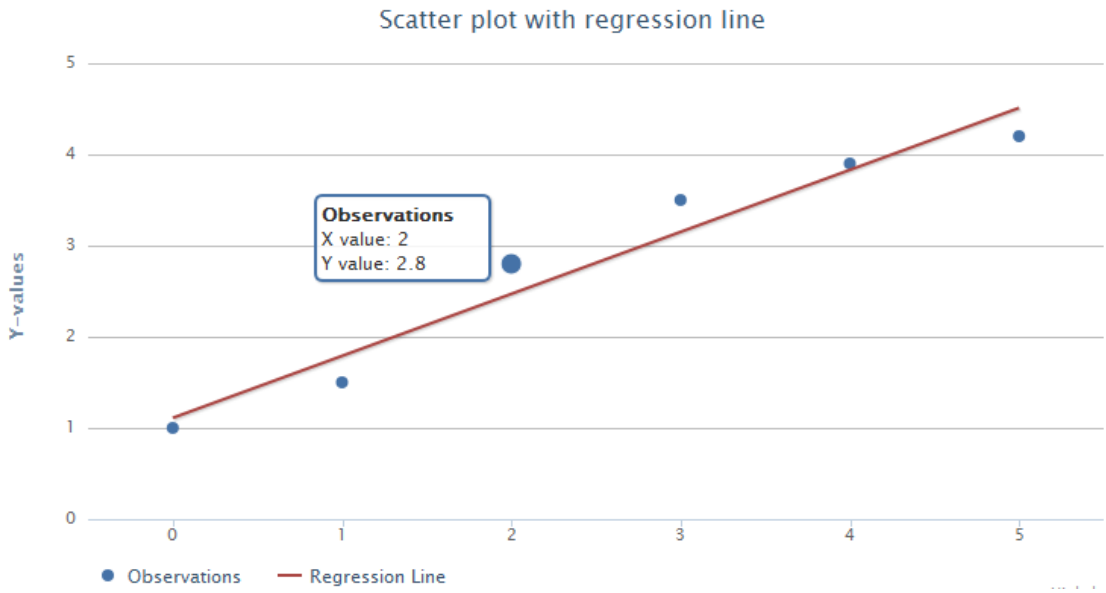


Figure 18 High Charts Scatter Plot with regression line (20)

A key benefit of HighCharts is its performance i.e. loading time. The graphs are drawn really quickly and there is little to no page load overhead caused by their inclusion. In the

case of Google Visualisations and Visifire there is a substantial delay in loading the page as the source API or browser plugin needs to be referenced and this can lead to an unpleasant user experience. Once HighCharts are referenced in the application and the core files included it can be used on/offline on any platform in any browser which is a key advantage.

Figure 19 High Charts Column Chart using values from HTML table below shows how a graph can be plotted using the values in a HTML table already existing on the page. Using data from a HTML table to draw charts as illustrated in Figure 19 is just one of the ways data can be passed to HighCharts but is extremely relevant to an application such as this as data tables will be used anyway and this is an easy way of drawing charts using them. Data can also be supplied to the charts in the form of a JSON array or even be supplied in conjunction with jQuery or MooTools to provide real-time live data supplied either by a database or directly by a user.

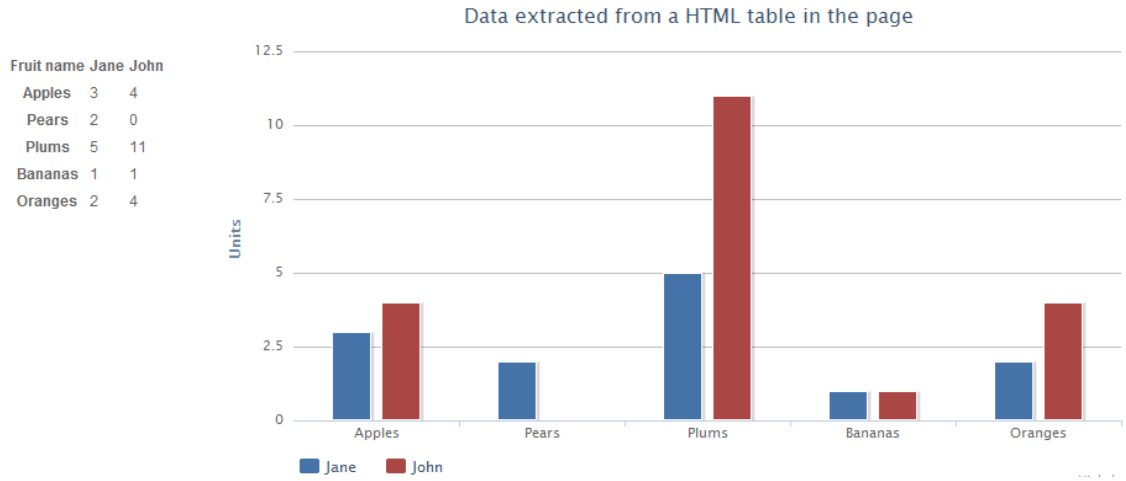


Figure 19 High Charts Column Chart using values from HTML table (20)

The term JSON here refers to Java Scipt Object Notation which is a common format used for data transfer in javascript applications. It allows a set of variables to be encoded and decoded from object to string format which makes it easy to pass information from one point to another through the use of session variables.

The only drawback of HighCharts is that they are not as dynamic as the Google Motion Chart or Annotated Timeline however they make up for that with their reliability, load speed and overall ease of use. They make far better use of the technology currently available and

don't rely on flash or the internet to work correctly so from the research they are the best and most developer and user friendly solution currently available on the market.

3.1.4 AM Charts

AM Charts (22) were another chart API studied during the research for this project and provide some crisp and user friendly charts. However as is the case with too many visualisations on the market they rely on Adobe Flash player to graphically plot into the browser. Some examples can be found in Figure 20 and Figure 21 to show the quality of the charts however they were not used as part of this project as their reliance on flash doesn't give any further benefit to the visualisations like for example the Google Motion Chart which is innovative and dynamic. HighCharts can do much the same thing as AM Charts but in a more efficient way so they would the preferred choice in most applications.

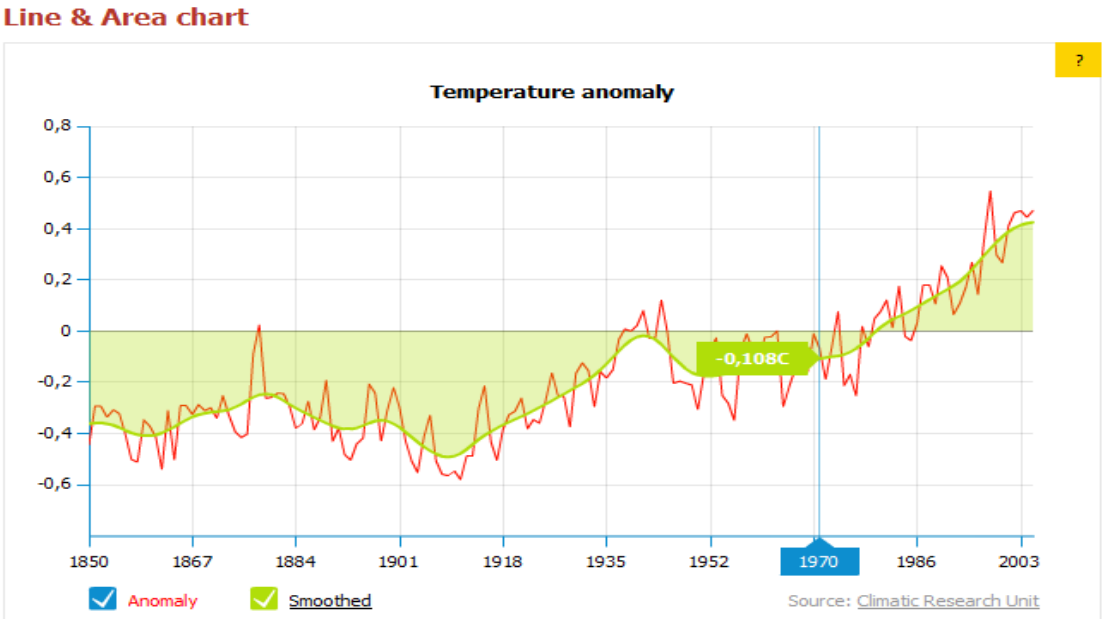


Figure 20 AM Charts – Area and Line (22)

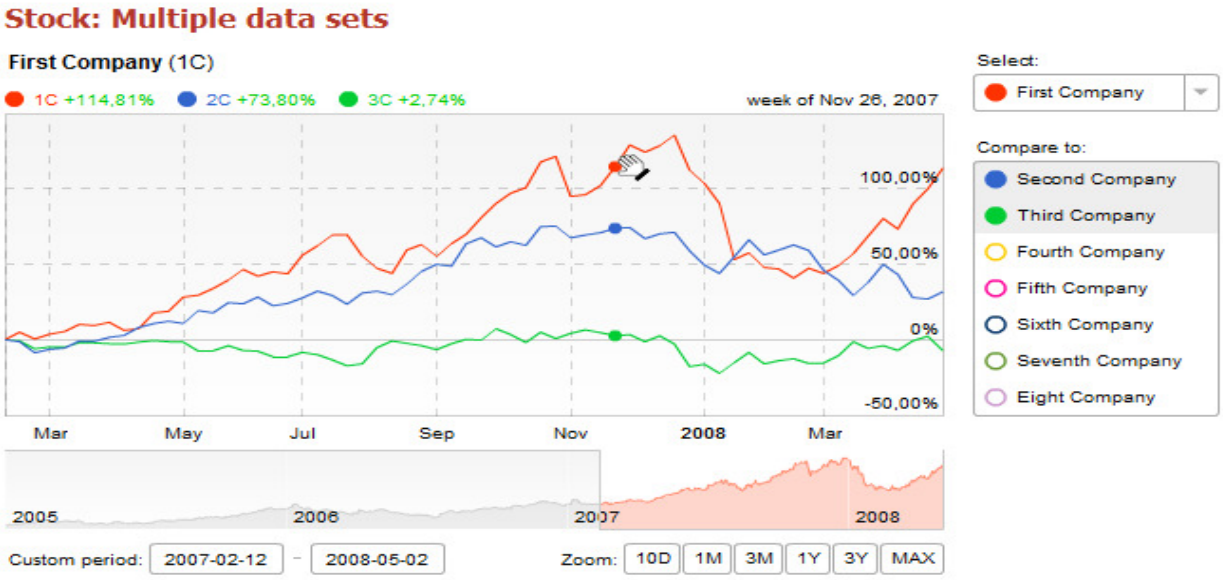


Figure 21 AM Charts Timeline (22)

3.1.5 Axiis

Axiis charts (23) are another chart type that was encountered during the research and they are the exception to the rule in that it is worth needing Adobe Flash player as the quality of the visualisations are built to such a high standard. Axiis visualisations are built upon the Degrapha graphics framework and using Adobe Flex. There is a battle going on in the web development world as to whether or not flash should be accepted as a standard feature and Flex is one of the key reasons why some developers feel it should be. Flex is a framework for building Rich Internet Applications (RIA) using Adobe Flex Builder as opposed to AJAX web sites.

This allows for greater control of the user experience and better graphic quality than is available otherwise. RIA are becoming more common now that the internet is so prominent as a business tool and with more and more software solutions becoming web based it is a great solution for people who need dynamic but stylish web applications. There is also new technology available from Adobe called Adobe AIR which allows for the creation of similar style desktop applications using flex. Some examples of the rich visualisations available using Axiis charts are shown overleaf in figures 22-26, they are taken from the Axiis demonstration page (24).

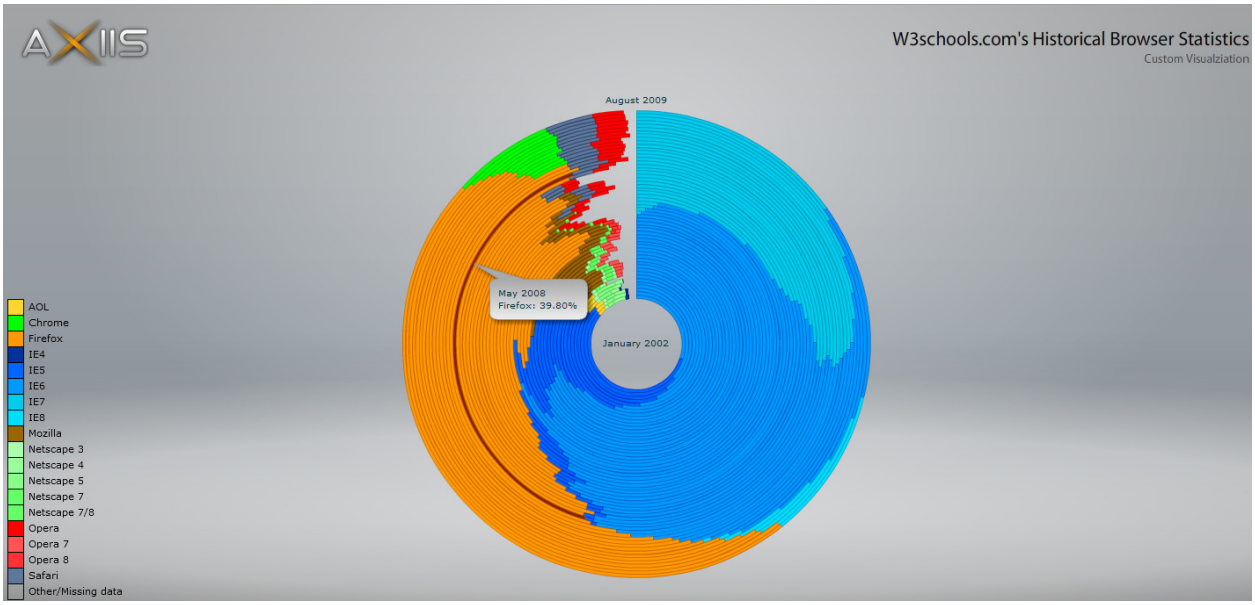


Figure 22 Axiis Browser usage distribution example (24)

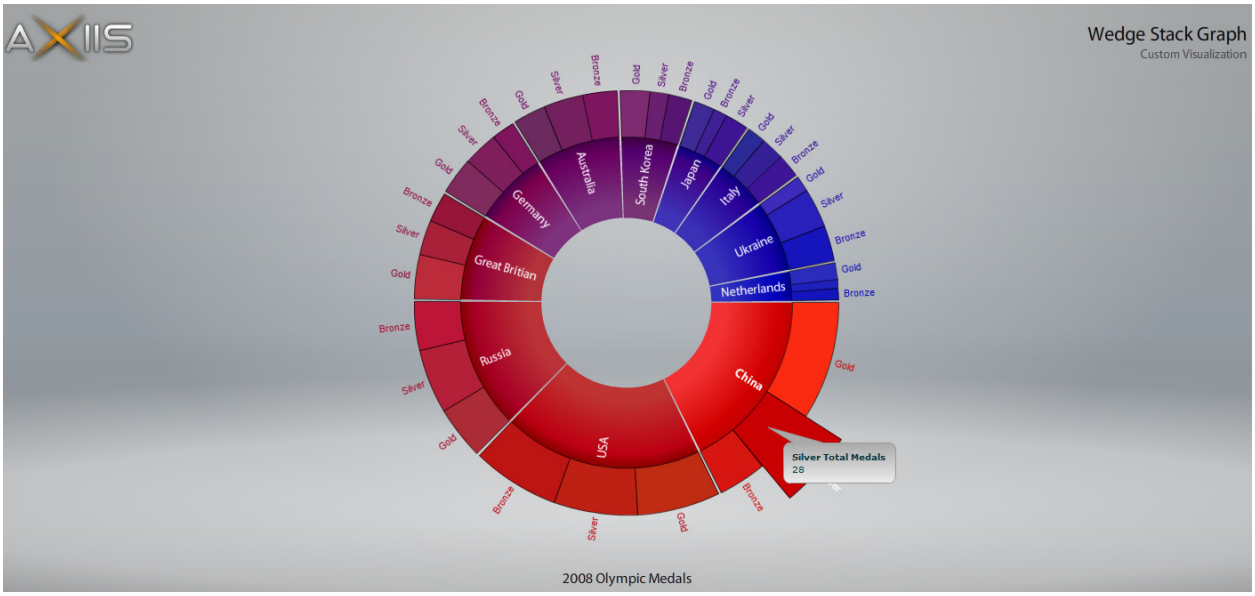


Figure 23 Axiis Wedge Stack graph (24)

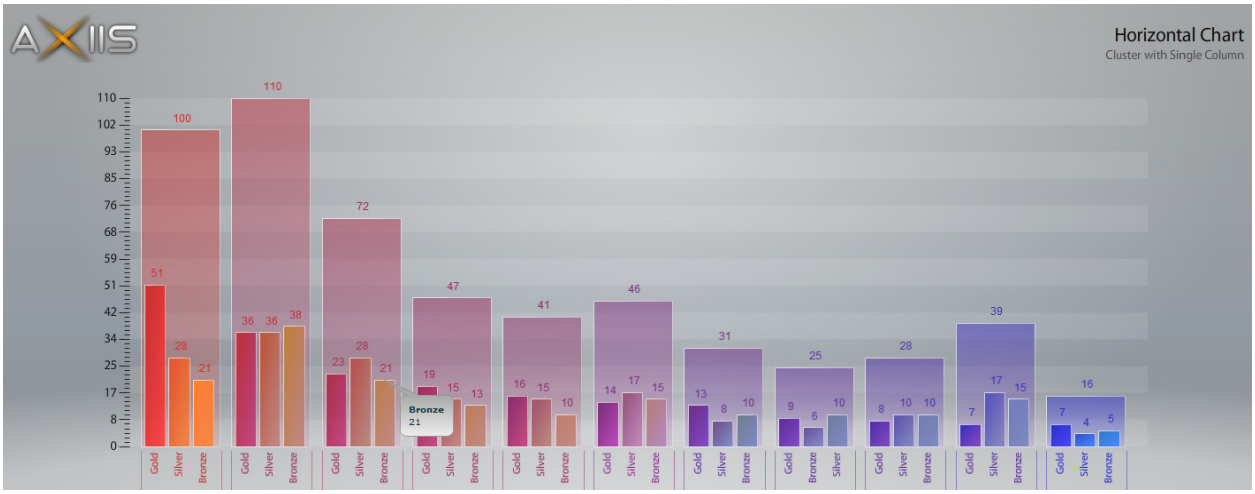


Figure 24 Axiis Horizontal Chart (24)

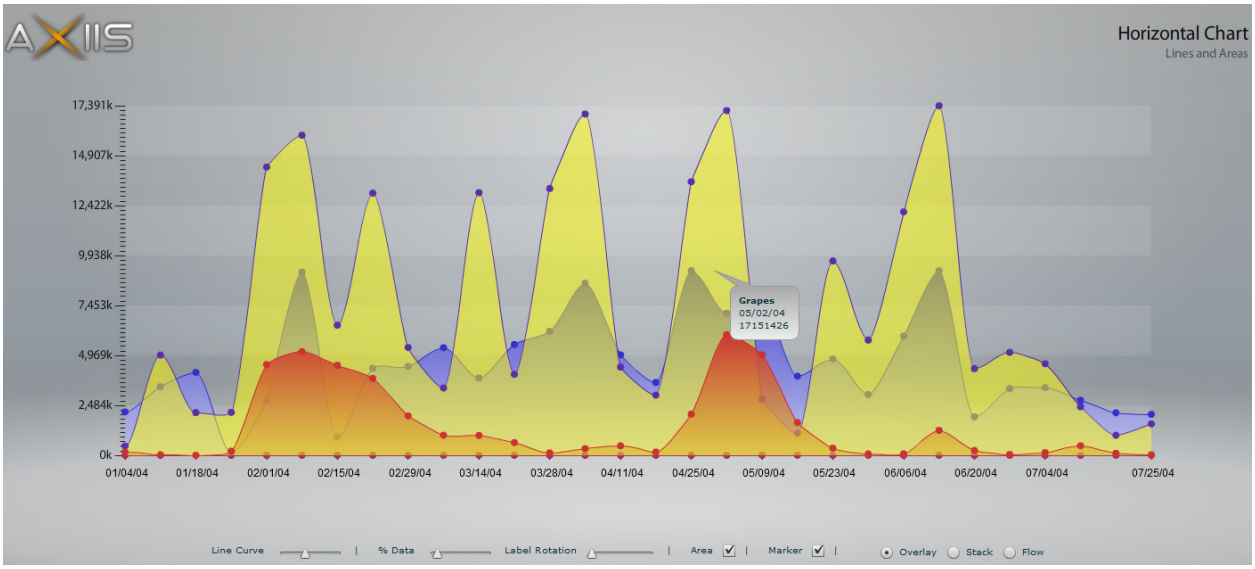


Figure 25 Axiis Horizontal Chart using area lines (24)

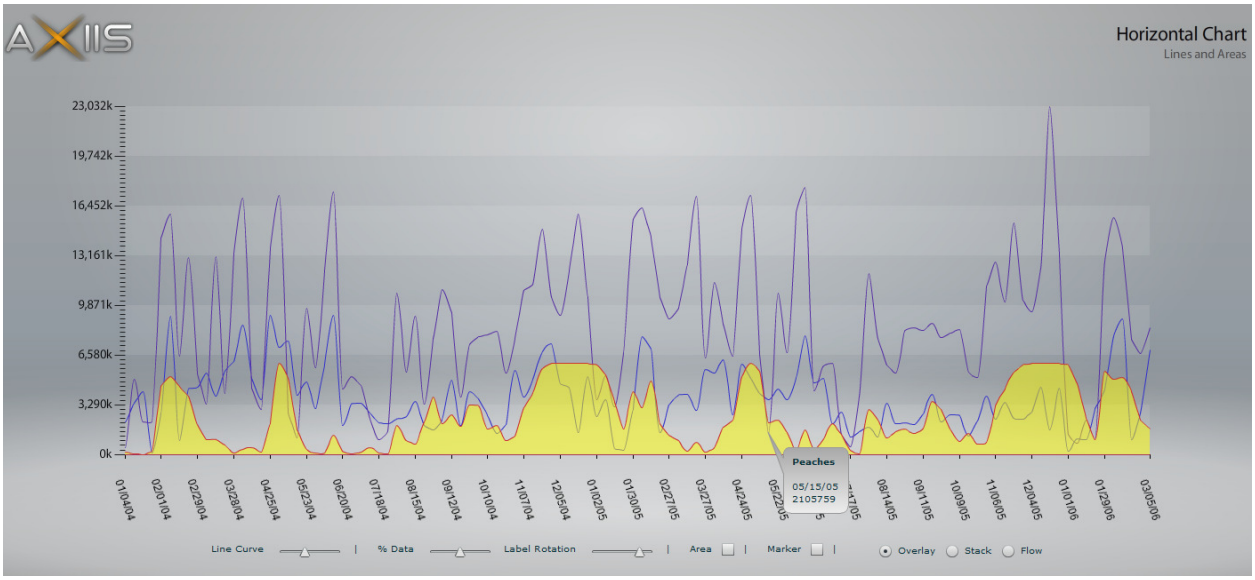


Figure 26 Axiis Horizontal Chart using area lines – some areas toggled off (24)

The graph in Figure 25 Axiis Horizontal Chart using area lines Figure 26 Axiis Horizontal Chart using area lines – some areas toggled off is extremely interactive and allows the user to toggle different features of the chart on and off such as lines, the percentage of input data used, whether to display as an area chart and various other possibilities. These graphs would have been great to use in this project however even with the emergence of RIA's as a real contender for web applications there is still not a world-wide acceptance of this reliance on Adobe. Software is better when it is open source and independent to any external platforms and there is simply too much reliance on Flash in Axiis charts.

Initially further research was performed into Adobe Flex with a view to using these visualisations in this project however a judgement call had to be made as in the scheme of

things the time spent learning a new language was not worth the value of the same time spent researching further student analysis techniques. A key point is that it is not about how visually appealing a visualisation may be a much more important feature is the quality and innovativeness of the data it's displaying.

3.1.6 RGraph

RGraph (25) is a chart library that makes full use of the HTML5 canvas tag to plot its graphs. It is extremely lightweight and offers some excellent charts with good features such as zooming and annotations however the graphics are simply not as visually appealing as the HighCharts charts. RGraph uses JavaScript and HTML to plot charts that work both on and off-line in any browser supporting the canvas tag.

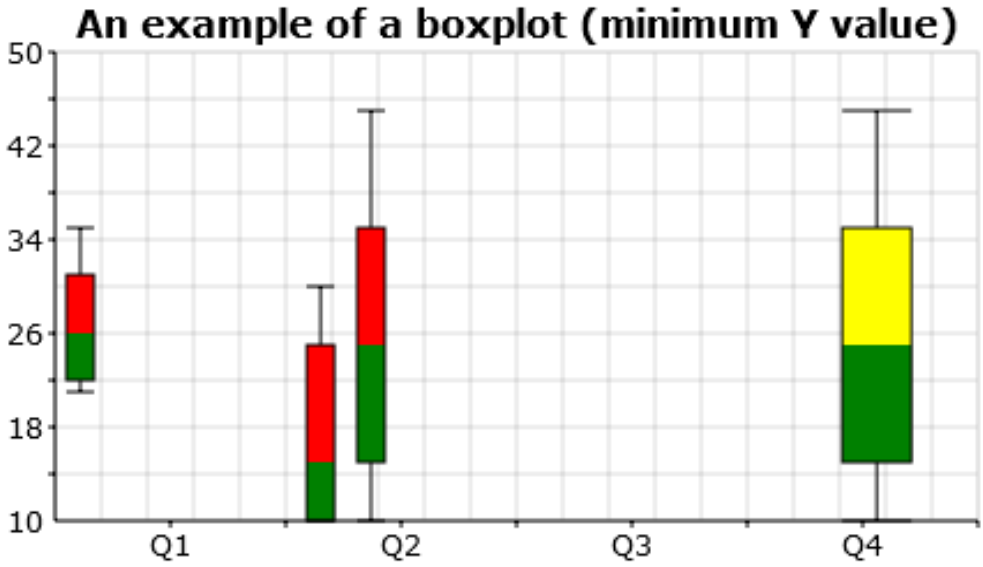


Figure 27 RGraph : Box plot chart (25)

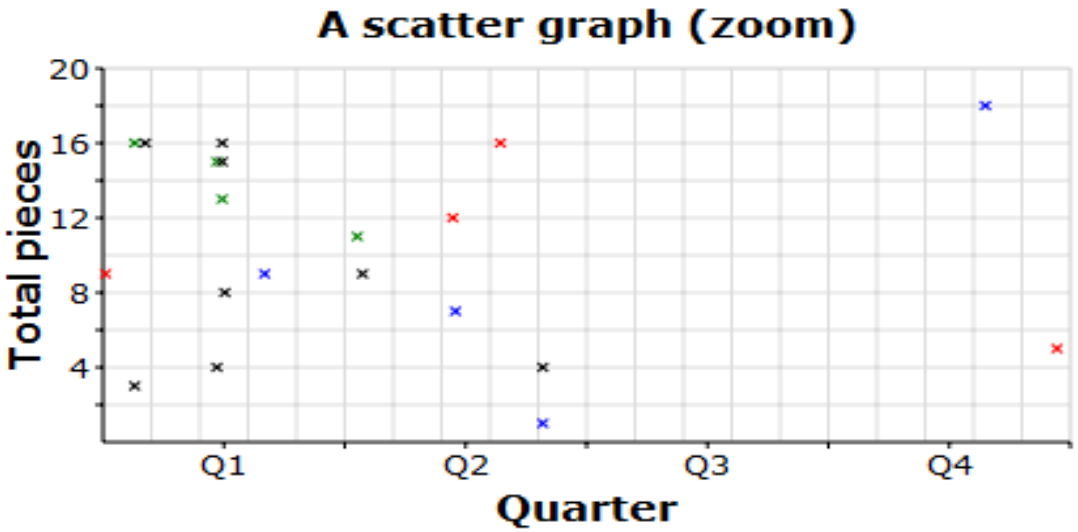


Figure 28 RGraph :Scatter chart (25)

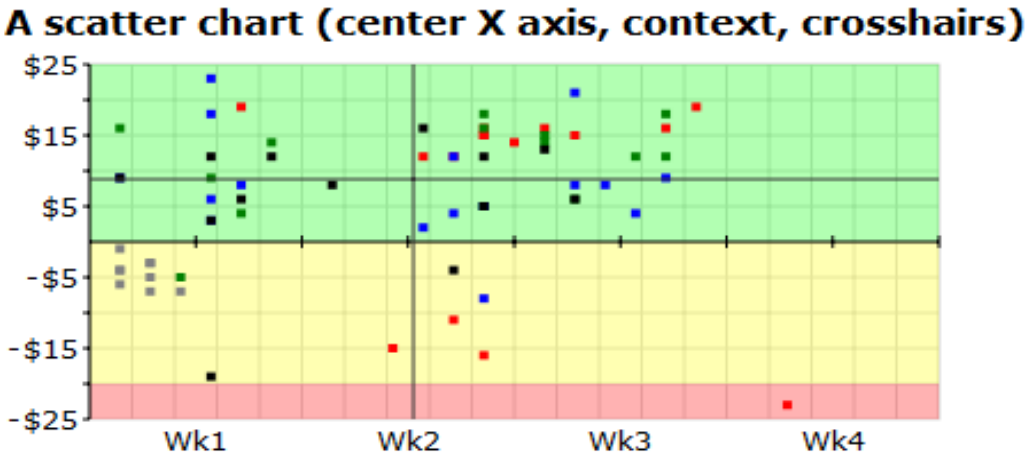


Figure 29 RGraph : Scatter chart with zoom (25)

RGraph charts would have been used in this project however HighCharts offer better functionality when combined with jQuery and can provide the same level of client interactivity as a RIA application without the need for Flash. A key focus of this application was to make it as user friendly as possible and to make the best of new web technologies to do so and while RGraph provides some interesting chart ideas their execution graphically leaves a lot to be desired.

3.1.7 HTML5 Canvas

As a lot of the chart tools covered in the research makes use of the HTML5 canvas tag it was a natural progression to do some further research on this to assess how difficult it would

be to develop a bespoke charting API specifically for the purposes of this project. Several areas were covered to investigate HTML5 and its advantages but the main focus was on the canvas tag. The power of HTML5 and particularly the canvas tag is that visualisations can now be designed and built by web developers themselves without relying on using chart API's or getting into Flash or Silverlight territory.

The HTML5 canvas enables web developers to draw HTML in whatever way they wish on the section of the page enclosed by the canvas tags. This enables any graphs or other visualisations a user could possibly want to be written in JavaScript and HTML5 solely and will work in any browser supporting HTML5. This is basically all browsers except for Microsoft Internet Explorer which does partially support it but has not fully accepted it yet. HTML5 is backward compatible with previous versions of HTML so can be integrated into existing systems without an overhaul.

The canvas element finally brings scalable vector graphics to HTML that can rival the performance of flash based solutions such as flex and as they are written in pure HTML and JavaScript they are far better suited to accessible applications. While Flex lets you manipulate individual pixels or apply filter and effects to them canvas still allows you to do this but it is more difficult as all you can access is raw pixel data.

Scalable Vector Graphics employed by HTML5 canvas allow web developers to draw directly onto the area of the page surrounded by the canvas tag as if they would in for example Microsoft Paint. Using JavaScript and type of graph or image possible can be drawn and this can be extremely useful when plotting data as shown below:

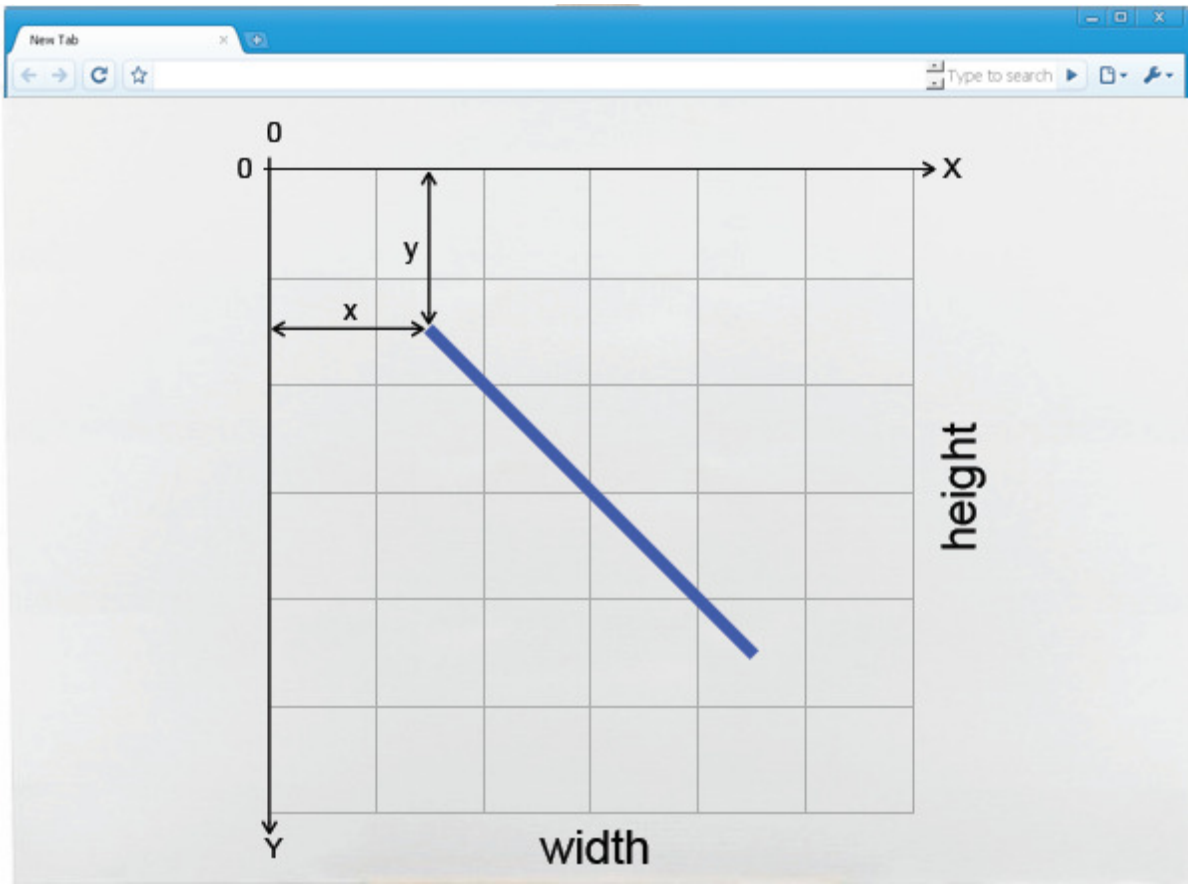


Figure 30 HTML 5 Canvas overview (26)

Before the introduction of the HTML5 canvas tag developers were restricted to technologies such as Vector Mark-up Language (VML), Visifire and Flash to produce the same kind of graphic quality in browsers. The canvas tag presents a distinct advantage as both canvas and SVG are intrinsic to the web and are supported in the transparent web technology stack. It is supported by the official web standards and is therefore an integral feature that must be included in all future browsers even if Microsoft are a bit slower than most to develop full acceptance.

HTML5 has some other innovations such as the availability of native browser support for video which is something that almost always relied on Flash in the past. YouTube for example is the largest video hosting site and they use flash for their videos. The inclusion of the <video> tag in HTML5 really adds to the arsenal HTML5 has to rival Flex when it comes to building RIA's. HTML5 also supports user history features such as the undo of a text edit in a form which is a useful feature to help ease user frustration if they made a mistake which is not supported at all in Flex applications.

HTML5 will be a serious rival to Adobe Flash in the not too distant future if not the de-facto standard for high level web graphics. Some amazing apps have been developed already to show its power once some research and effort is put into unleashing its full potential. A terrific example of this is the Sketchpad app (27) developed by ColorJack (28) which can be seen over leaf in Figure 31. It is a Microsoft Paint style application that offers far more functionality that is written using the HTML5 canvas element and JavaScript. It can produce some stunning graphics which could more than rival flash applications. Other features of HTML5 are its easier mark-up format to make it easier to organise web pages and web applications. Specific DIV tags are pre-defined for areas of the page to make layout a lot easier to design and maintain:

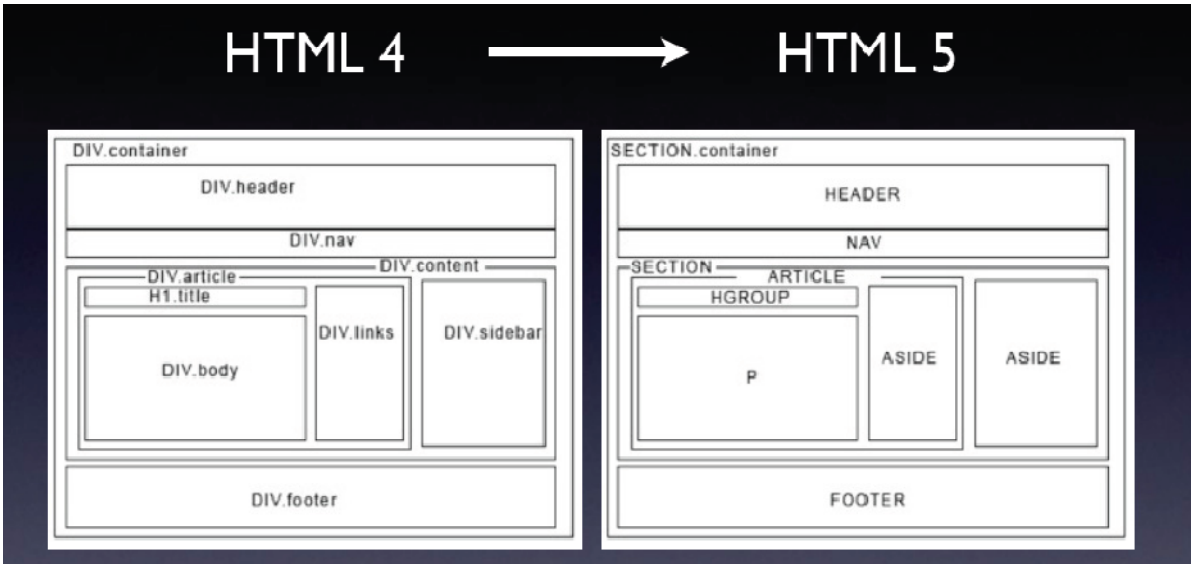


Figure 32 Document structure differences between HTML 4 and 5 (29)

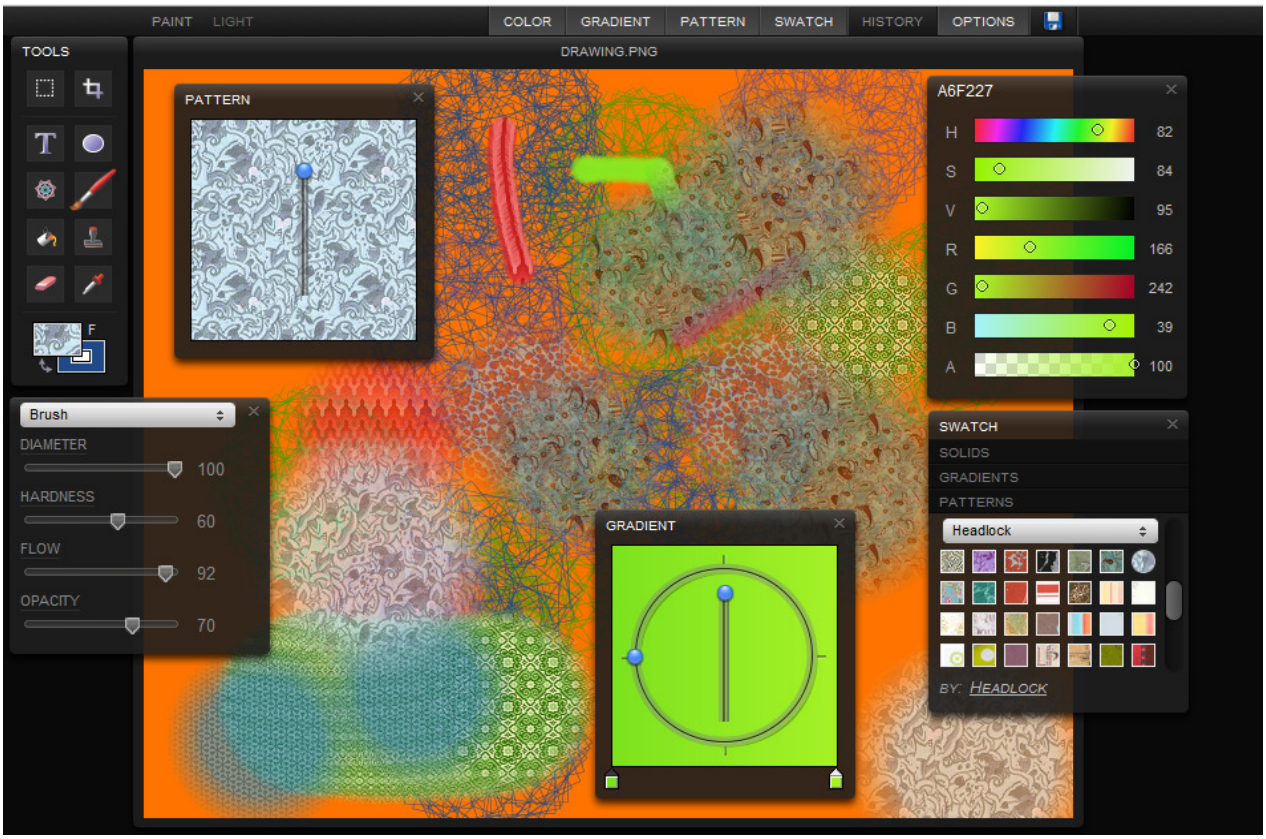


Figure 33 Sketchpad drawing application using HTML5 Canvas (27)

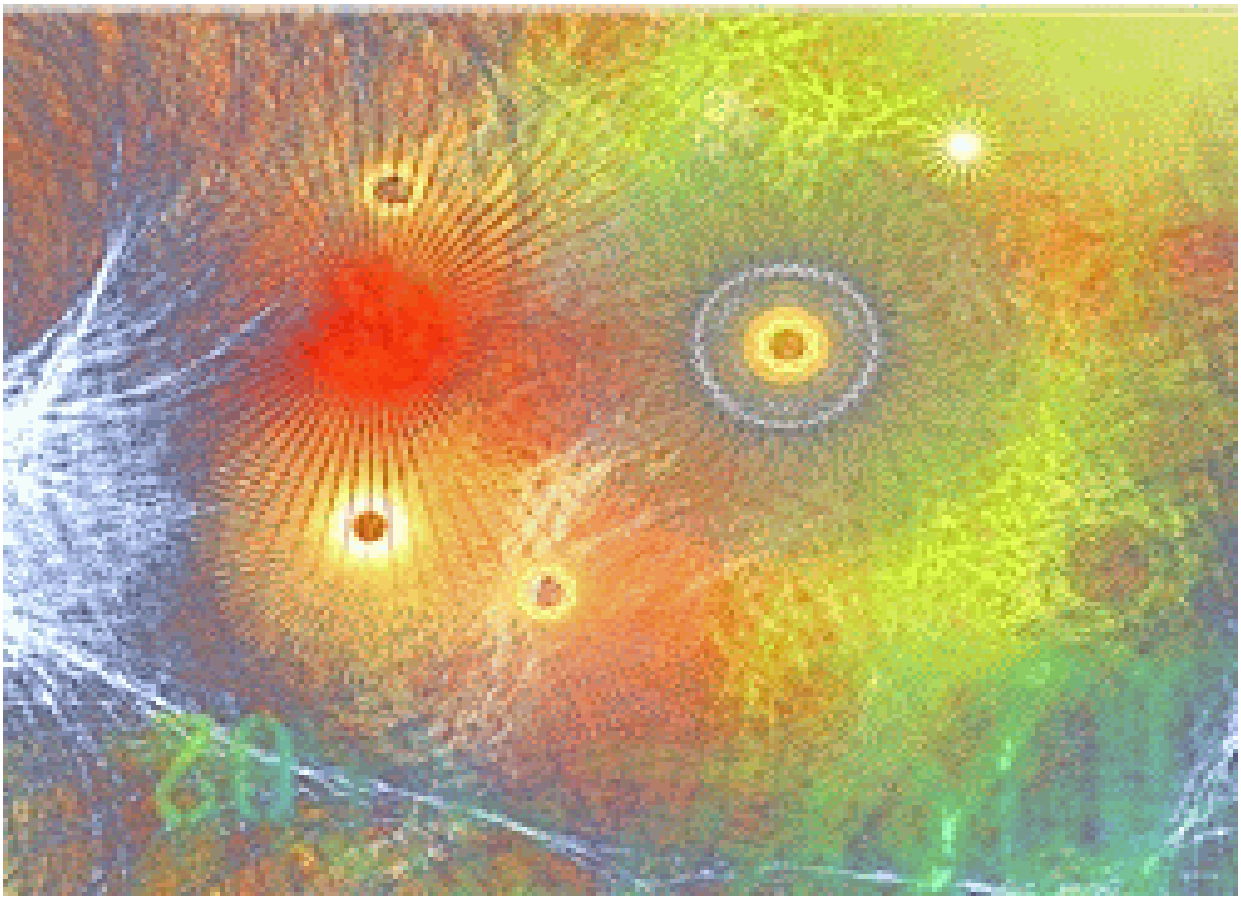


Figure 34 Sample Sketch pad graphic (28)

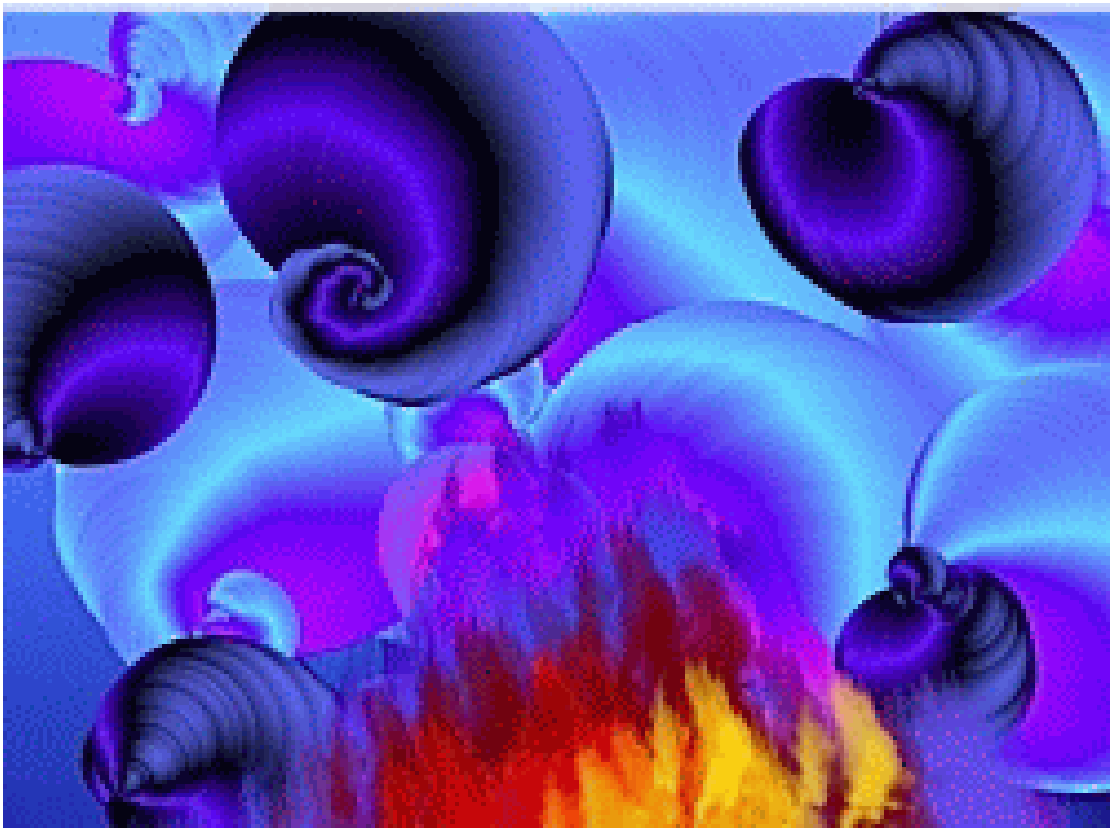


Figure 35 Sample Sketch pad graphic (28)

ColorJack (28) see HTML5 as the number one standard for drawing web graphics as it finally breaks away from a reliance on flash files. It offers improved performance as users can undo mistakes they have made and is easier to code and edit further as it relies on HTML style tags to operate. Flash is far more difficult to edit and code and requires the use of an Adobe IDE rather than a plain old text editor. The main reason Flash was preferred in most applications was due to the improved graphical quality however now HTML5 has overtaken it and looks the best option for these kind of applications. It's easy to code and requires no browser plug-ins to work which may not seem but the truth is there is nothing as good currently on the market.

3.2 Student Performance analysis

3.2.1 Student academic analysis

This project focuses on analysing and displaying student academic data so it was vital to research several case studies in this field to help guide further research and to get ideas on the most useful data to analyse to help plot academic performance in graph form. There are a lot of ways to plot the information available and graphs could be used to plot anything so focusing in on picking the most useful information to display and having key logic behind this was an important part of this project.

3.2.1.1 Informative General Case Studies

Neuro fuzzy analysis of academic performance was a topic present in a lot of papers on student academic performance analysis and a great example of its use is provided by "Neuro Fuzzy Reasoner for Student Modelling" (30) by Zoran Sevarac. Sevarac discusses the neuro-fuzzy approach to student analysis which allows categorization of students using information found in their characteristics. This type of model can be quite useful for evaluating student behaviour as it is quite easy to understand and implement. Linguistic variables are used to make it easier to follow the analysis and these variables are applied to fuzzy sets for classification. This is known as a fuzzy rule and an example of this is:

... IF (TEST_RESULT IS LOW) THEN STUDENT_CLASS IS BAD [.]This rule says that if a student has low result on a test, he is classified as a bad student. The expression (TEST_RESULT IS LOW) is the premise, and the expression (STUDENT_CLASS IS BAD) is the consequence of this fuzzy rule... (30) pg 1

This example shows the basic idea behind the fuzzy logic approach. This particular paper uses it to classify student quality based on the grade obtained and the time taken to complete a test. The neuro-fuzzy reasoning (NFR) model is shown in Figure 36 - Fuzzy logic classification (30).

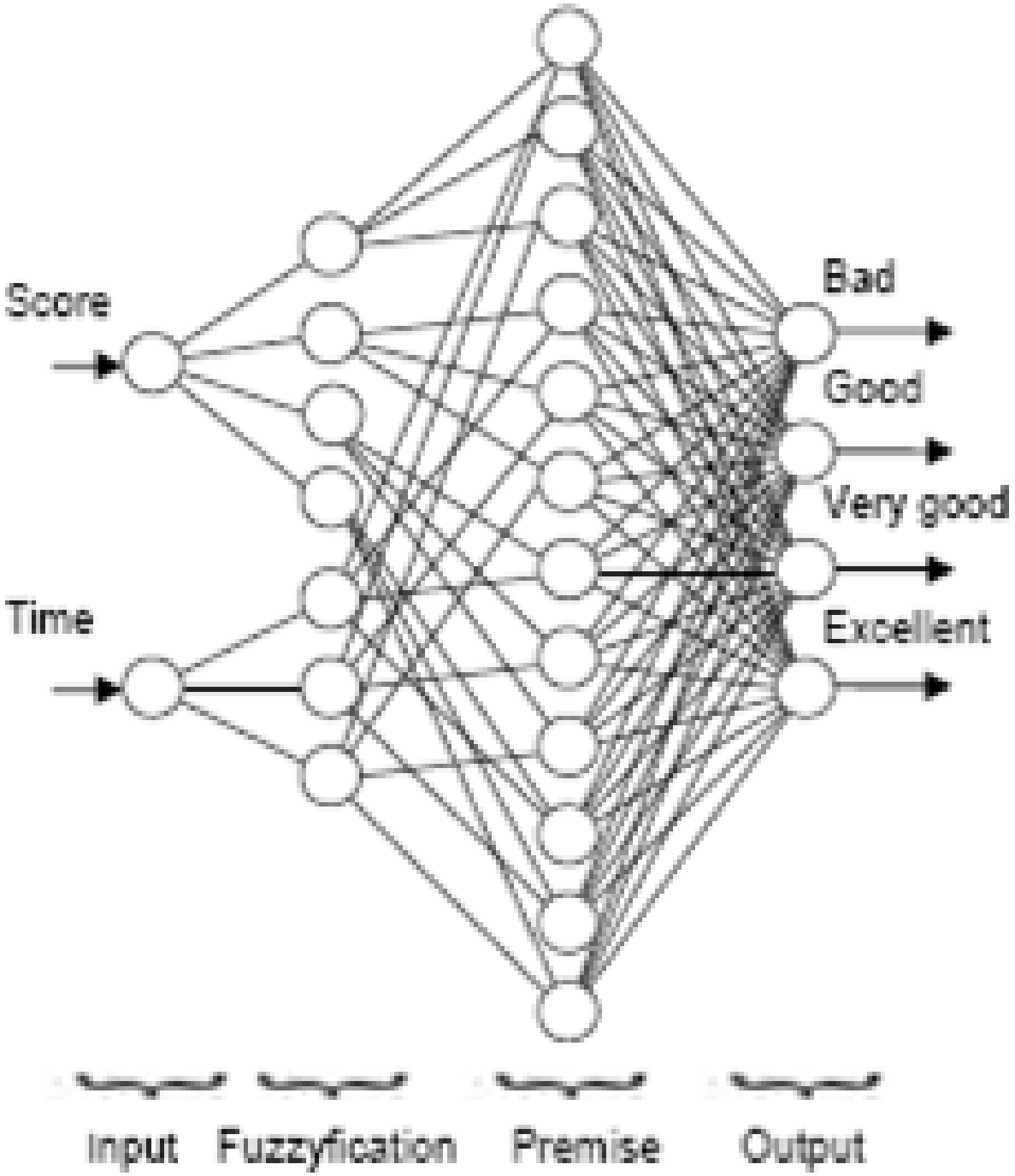


Figure 36 - Fuzzy logic classification (30)

A publication by the Journal of the Statistical and Social Inquiry Society of Ireland on "The Leaving Certificate and First Year University Performance" (31) was great background reading on student academic data analysis and examines the significance of leaving certificate examination results on first year under-graduate performance at University College Cork. Although it is not directly relevant to this project (from the data provided

only course entry points rather than specific leaving certificate point scores can be considered) it is a source of some excellent information on how to go about modelling student academic performance.

The paper examines the role that the leaving certificate plays as a gauge of how well students will perform at university level. It explores alternatives to this such as aptitude and personality tests as an indicator of how a student will perform which is something that could be useful to a university when accepting course applications. For example just because a student achieves the maximum result of 600 points at leaving certificate level doesn't necessarily mean that they would make an excellent nurse, scientist or engineer. It would benefit both students and the university to do this as students taking courses would have a natural aptitude for that field and would perform better.

This paper was a true source of inspiration and provided some great ideas that could be applied to this project. An analysis of the pass rate percentage of faculties at the university versus point score was performed that could be easily adapted to plot module pass rate against various things.

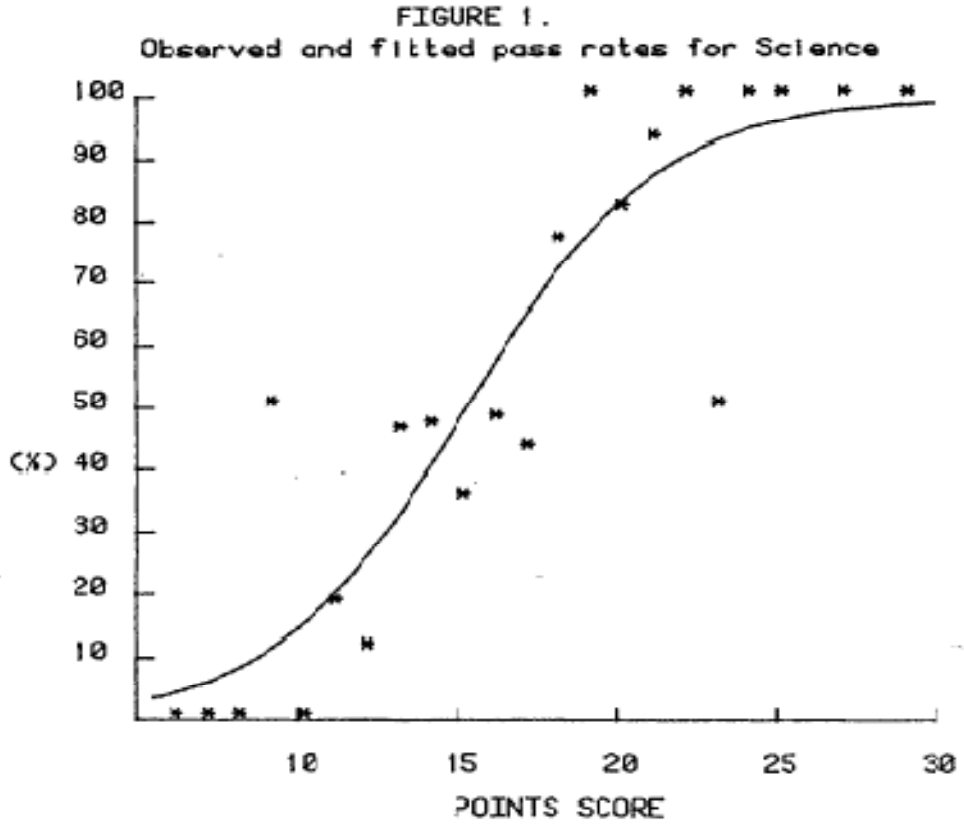


Figure 37 Observed Vs fitted Science pass rates (31) Pg 8/9

3.2.1.2 Analysis of student performance based on gender

Gender analysis is a good way of analysing student data as this can prove a useful tool both for analysing current student gender distribution and corresponding GPA and for predicting gender distributions in future courses. Some courses have a very biased load towards a particular gender for example the majority of engineering students tend to be male and the majority of nursing students tend to be female.

Comparisons between the performance of majority and minority groups can be very useful as if for example only 10% of engineering students are female yet make up the majority of high grades the overall grade for all students would be biased by them. This could help lecturers more accurately predict how well individual students will do as they can compare their performance to others with similar gender. The more closely you can match the student being analysed to a group of the overall population through classification the more reliable predictions of future performance will be.

A paper analysing the 'gender gap' in UK universities gave an interesting analysis of the existing majority of males who achieved a 1.1 in their courses (32). This study took a huge sample size of approximately 1.7 million students and found that in general males achieved more first class honours than females. However they found that males tended to pick courses that were first-rich, or in other words had a much higher percentage of firsts.

This shows how easily misleading statistics can be if you don't take as much data as possible and think beyond this data. At first glance this could indicate males are doing better in college in general however this is not the case at all as more females are taking more difficult courses and those who take the first-rich courses that males do could do just as well there. This paper provided great background on analysis by gender and prompted some useful graphing ideas such as Ratio analysis of males to females by year as shown in Figure 38 Ratio of male to female first class honours 1994/95 – 2001/02 s pg 12.

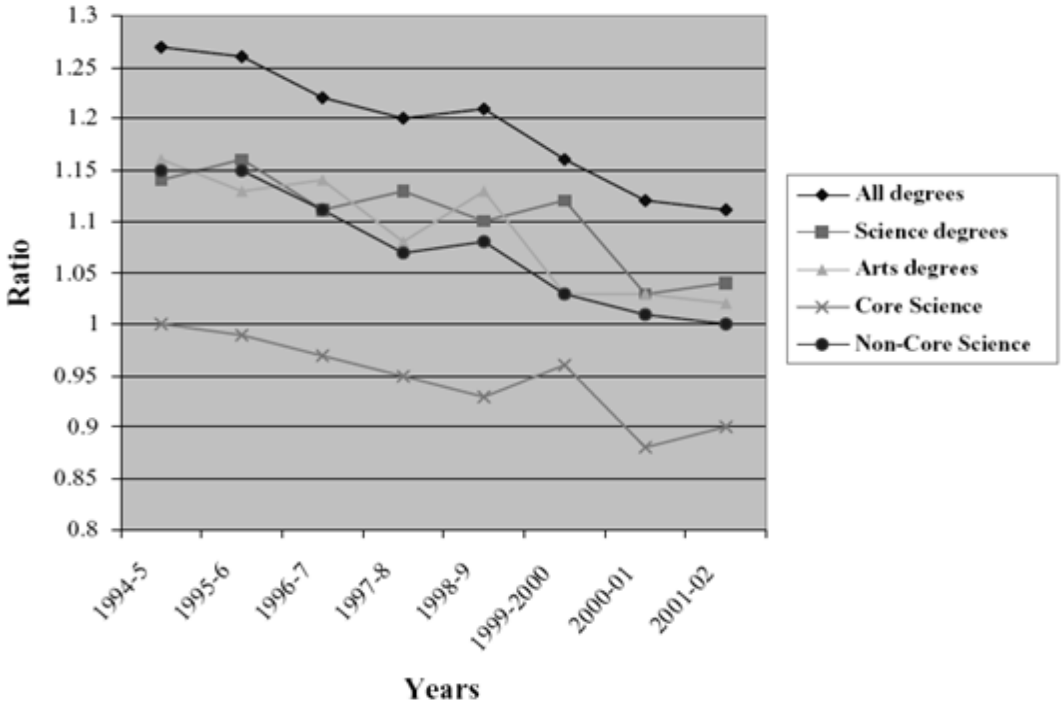


Figure 38 Ratio of male to female first class honours 1994/95 – 2001/02 s (32)pg 12

Other useful papers of not that were encountered during the research into student analysis were an analysis of US Army linguistics students: "*Research on Integrated Performance Assessment at the Post-Secondary Level: Student Performance Across the Modes of Communication.pdf "* (33)and an assessment of student academic performance by the association of Student Examination performance: *"The Association of Student Examination Performance with Faculty and Resident Ratings Using a Modified RIME Process"* (34)

3.2.1.3 Analysis of student performance based on circumstances

Student performance can be closely related to circumstances and as a lot of courses are now being offered remotely it would be a good performance predictor to see if remote access students have a higher risk of failure. and an excellent case study of this is found in "Student Performance Online Vs. Onground: A Statistical Analysis of IS Courses" (35). They compare the performance of a large sample set of student data to try and find differences in student performance between on-campus students and those taking courses online. This would be very useful for DCU to do as it's a good assessment of how well a module is doing.

This analysis could be used to study the differences between those learning remotely and on-campus, for example if all the students who are taking a module online have poor results perhaps the online material for that module needs to be reviewed and improved. Similarly if the students taking the course online do better than those attending class then the lecturers performance may need to be reviewed.

Previous studies of this kind focused on a much smaller set of sample data but here a much larger sample of 1300 students was analysed across many courses analysing various factors about students most notably grade point average, student load(credits taken per semester) and whether the student attended class or took the course online.

... The study includes more than 1300 observations spread across seven courses that are part of the computer science and information systems curriculum at Northwest Missouri State University. Student performance was compared by grade point average, ACT composite scores, number of credit hours completed, instructor, and delivery method... (35) pg .1

The results of this study showed that in 3 of the 7 courses analysed there were significantly lower grade point average scores for students taking the course online than those attending class. The other 4 courses showed no major difference in performance as a result of delivery method (attendance type). This shows that there may be some courses more suited to online learning or similarly some courses where the student really needs to be on campus. If DCU could analyse this kind of information it would greatly improve resource management and the overall efficiency of the university. They would help get the best out of both their resources and their students by researching into this. There are several other factors to consider however – students doing poorly on these remote courses may not be putting in the full hours required or there may not be enough relevant significant material online to enable them to grasp the module fully.

This data is available analysis on the DCU student database as there is a field for course attendance type, enabling classification and graphing of this information. This paper gives a great overall background on the things to look for when analysing student data. They produced bar charts to show GPA against time for several courses and a summary of all courses that helped inspire some of the graphs used in this project. Examples of the graphs

used in this paper can be found in Figure 39 Comparison of Average Grades for Online and Onground Students pg 5Figure 40 Semester Comparison for CSIS 140 pg. 5and Figure 41 Semester Comparison for CSIS 317 140 pg. 5.

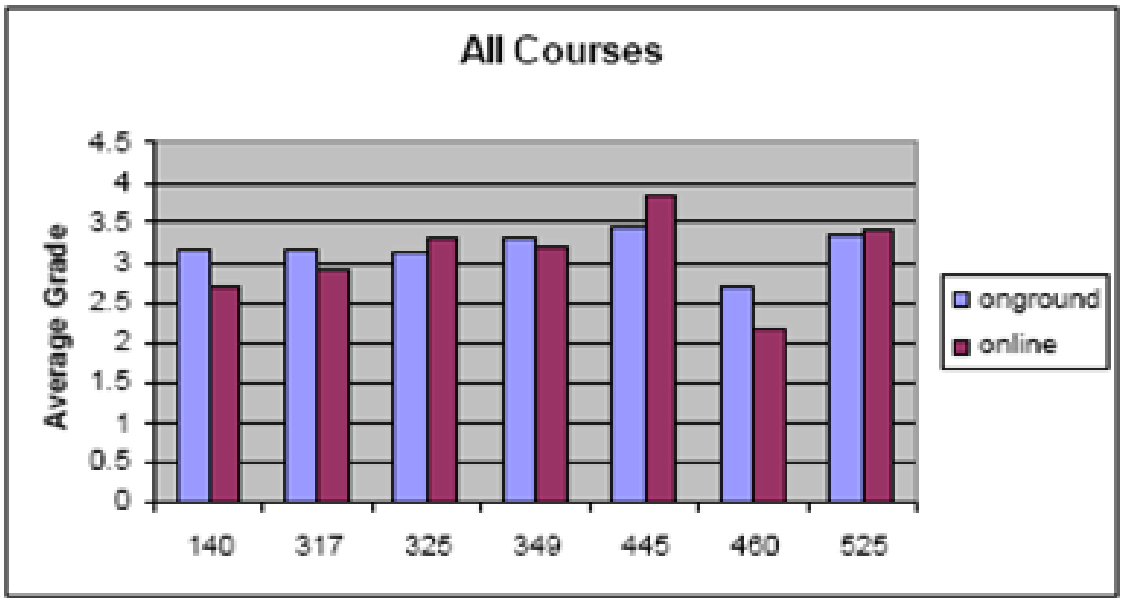


Figure 39 Comparison of Average Grades for Online and Onground Students (35) pg 5

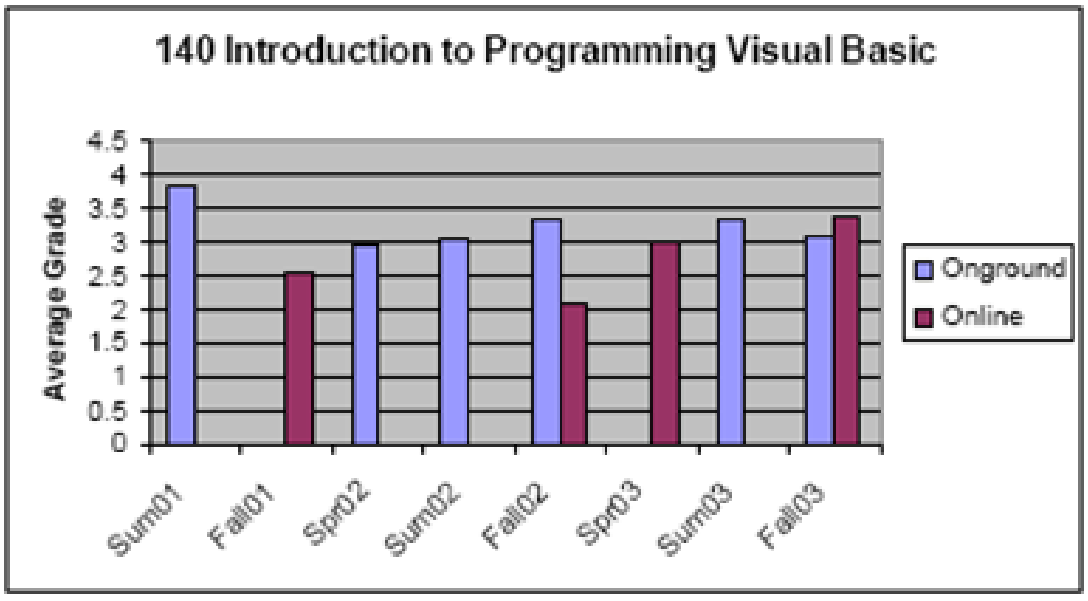


Figure 40 Semester Comparison for CSIS 140 (35) pg. 5

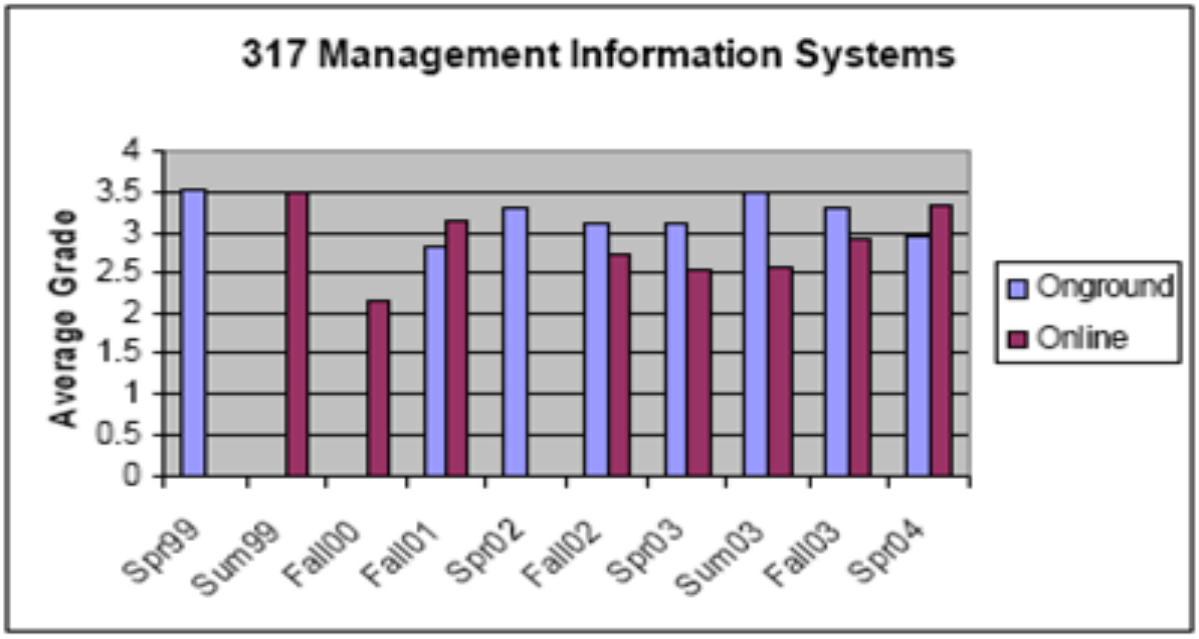


Figure 41 Semester Comparison for CSIS 317 140 (35) pg. 5

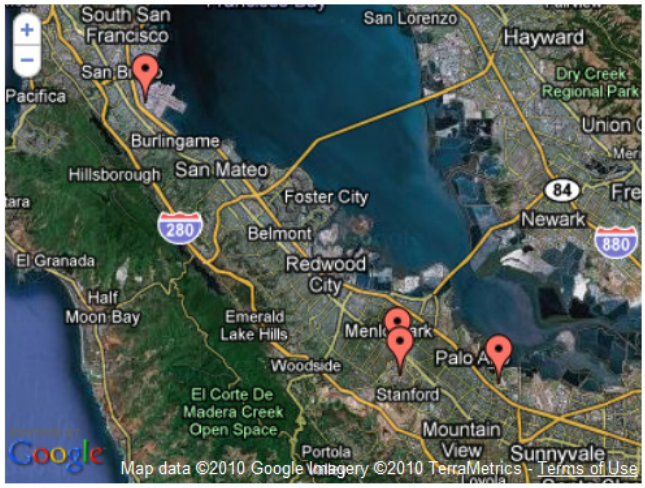
The graphs in this study provided inspiration to use bar charts to plot on a per semester basis. It is possible to create a student's module results combination chart on a per semester basis with a line of average using HighCharts and this research was a help in coming up with that idea. It also provided the idea to plot Exam Vs CA averages per semester with average CA, Exam and overall lines.

A similar analysis into student performance based on circumstance was performed at Simon Fraser University and analysed the difference in performance between students residing on and off campus and is entitled "Academic Performance at Simon Fraser University:
SFU Residence Students vs. Non-Residence Students" (36). This study compares the performance of under-graduate students on the basis of whether or not they reside on campus. This is something that could be useful to DCU for many reasons.

If students living on campus are performing worse than those who aren't then the management of residence buildings needs to be analysed to help improve performance of university residents. Similarly if students residing on campus perform better in exams this could be used by the college as a marketing tool to encourage more students to live on campus therefore providing the university with more income for development.

The ideas presented in this paper are not directly applicable to this project as although access is given to student address information it is not considered as being not reliable enough to draw conclusions from analysing them. The accuracy of such an analysis is questionable as a lot of students may be living on campus but using their parent's home address. If access was granted to the on campus residents database these students could be analysed against the overall student population in order to gauge the differences in performance between the two however at this moment in time this information is not available for analysis.

It would be useful to gather such information in future years and a possible visual implementation could involve the Google Visualisation MAP API (37) which enables linking of a data set to location values in Google Maps such as shown in the Figure 42.



	Lat	Lon	Name
1	37.4232	-122.0853	Work
2	37.4289	-122.1697	University
3	37.6153	-122.39	Airport
4	37.4422	-122.1731	Shopping

Figure 42 Google Visualization: Map (37)

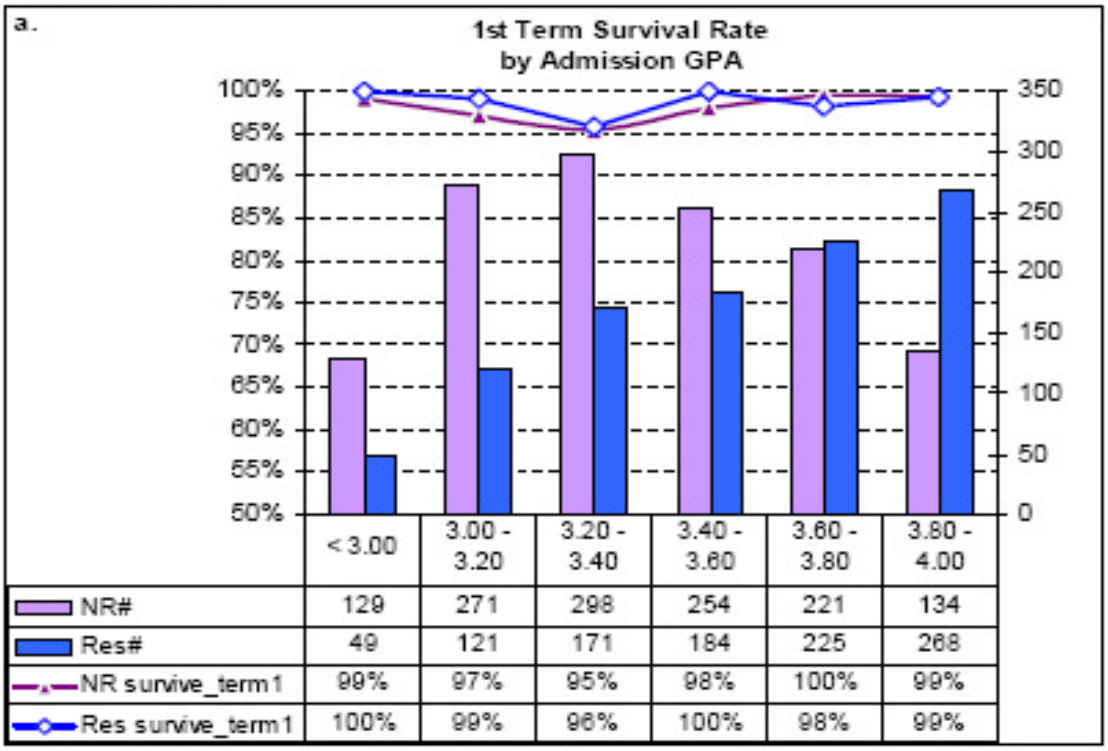
Although the performance metrics used are not applicable to this project this study provided some attention-grabbing graphs of academic performance which prompted the idea to use different visualisations in combination for better effect which can be achieved using HighCharts Combination Charts. The paper examines a number of academic performance measures which are applicable to all types of student performance analysis including:

- First term survival rate
- Entry shock
- Degree completion rate
- Graduation GPA
- Time to degree completion

(36) pg 3

Appendix B: Academic Performance of Residence and Non-Residence Students by Admission GPA

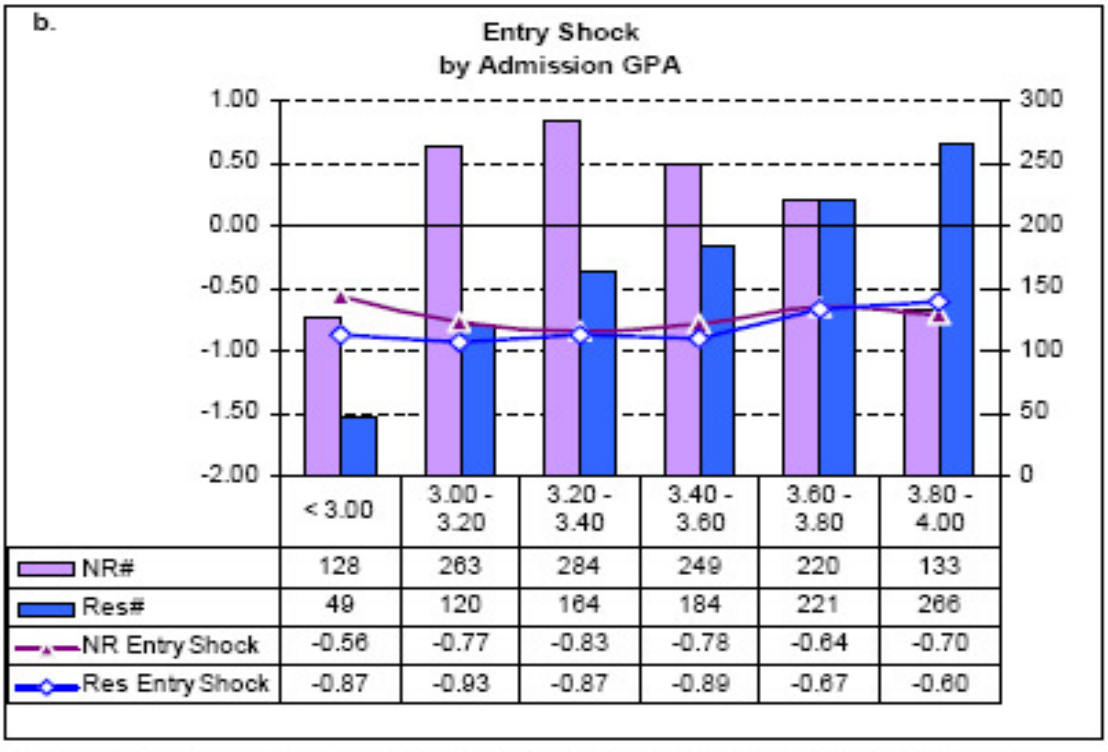
a.



	< 3.00	3.00 - 3.20	3.20 - 3.40	3.40 - 3.60	3.60 - 3.80	3.80 - 4.00
NR#	129	271	298	254	221	134
Res#	49	121	171	184	225	268
NR survive_term1	99%	97%	95%	98%	100%	99%
Res survive_term1	100%	99%	96%	100%	98%	99%

Residence students who were in residence in their first term are compared to non-residence students.

b.



	< 3.00	3.00 - 3.20	3.20 - 3.40	3.40 - 3.60	3.60 - 3.80	3.80 - 4.00
NR#	128	263	284	249	220	133
Res#	49	120	164	184	221	266
NR Entry Shock	-0.56	-0.77	-0.83	-0.78	-0.64	-0.70
Res Entry Shock	-0.87	-0.93	-0.87	-0.89	-0.67	-0.60

Residence students who were in residence in their first term are compared to non-residence students.

Figure 43 Academic performance of Residence and Non-Residence Students by Admission GPA (36) pg. 9

Auxiliary research into student performance analysis based on circumstances performed at the University of Warwick, in the paper “Determinants of degree performance in UK universities: a statistical analysis of the 1993 student cohort” (38) was another muse in the formulation for analysis ideas. This is a study into the performance of undergraduate students leaving UK universities in 1993. An investigation is done into the root causes of good and bad performance of students based primarily on their gender, social class previous academic record. If DCU could build up a database of such information it could prove very useful in future in aiding data analysis and visualisation.

If this type of information can improve the accuracy of student grade predictions and hence improve the universities ability to help hem then it should be collected as a priority due to the key benefits that can be obtained from its analysis. If it can be inferred that for example students who come from a lower social class achieve higher grades then efforts could be made to help motivate other students from higher social class backgrounds to push themselves harder.

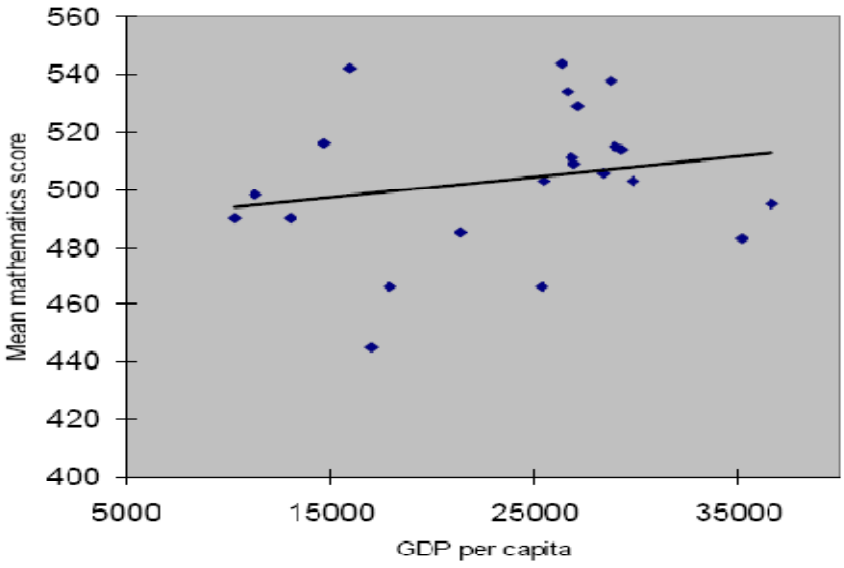
A good university should be run like a good business with a strategic focus on continual improvement and evolvement, which Henry Rosovsky brushes on in his witty and satirical guide to running a university based on his experience at Harvard University (39). The goal should be to retain as many students as possible to increase revenues from registration fees and therefore increase funding for further improvement of facilities at the university to continue the cycle of improvement and continually raise the bar of expectation.

The more useful data that can be collected about students to aid in this quest the better and the initial funding spent in initial research would pay for itself in years to come. Increased precision and accuracy in predicting student performance will lead to increased revenue from less students dropping out as lecturers will be able to quickly make more informed decisions on the paths certain students seem to be taking and intervene before it becomes too late.

3.2.1.4 Regression analysis

Regression analysis is a theme that was encountered on numerous occasions in the research for this project and is a tremendously useful tool in predicting the future value of one variable based upon the value of another. An excellent study into analysing performance was performed by the Danish Technological Institute (40) where data from three international surveys of students' skills were analysed to try and help improve basic skill levels across the EU. The skills they analysed were important but basic level skills such as reading and mathematics but the same principles could easily be applied to different data. This research also served as a good analysis of what data matters most when analysing student performance and how it can be used effectively to help educators improve their own class performance and the system in general.

Several behavioural and social aspects of student life are covered in these surveys to help study student performance from every angle. The paper also presents some very interesting ideas and uses linear regression analysis to help analyse and predict the average test scores in science, mathematics and reading on standardised tests compared to GDP per capita of their country in U.S. dollars. The goal was to see if there was any direct correlation between the wealth of a country and the performance of its students. An example of this is shown in the plot below (40). Although no direct correlation can be seen this backs up the effectiveness of linear regression analysis in predicting student performance.



Source: 2003 PISA dataset, OECD 2004a.

Figure 44 - Regression analysis on a country wide basis of mean mathematics score against GDP per capita (40) pg 67

A particularly interesting idea they proposed was to measure student confidence level in subjects by asking them to predict what result they would get in their exam. This kind of data if collected could be extremely useful when trying to make decisions about a modules performance overall. Perhaps a lot of students did badly but thought they would do a lot better and the core values of the course need to be emphasised more in future semesters. This would be worth doing in any university as it would ensure that when reforms need to be made regarding module performance all factors are taken into consideration and an educated decision is made.

Another interesting paper found during the course of this projects research was an analysis of whether a student's neighbourhood will help is an accurate predictor of academic performance (41). The paper was written by Franklin T Thompson and investigates the effect a student's neighbourhood has on their academic performance. using Student biographical information regression analysis is used to see if neighbourhood is an accurate predictor of student performance. This could be used by the university to try and spot patterns in poor performance from students who come from a similar area or even currently reside on-campus. Steps could be taken to try and reach out to and help these students and improve the overall GPA of students attending the college.

Regression analysis is an excellent way of predicting the outcome of variables but can be quite difficult to understand at first as the computation required is quite complicated and extensive. A first-rate resource that played a vital role in gaining an initial understanding of how to apply regression techniques to student data is Milton Smiths paper entitled "Statistical Analysis of Performance of Learning Disabled Students" (42). The paper includes a full worked example of linear regression analysis of students on a per semester basis and was a useful resource in this project for this reason.

3.2.1.5 Discriminant analysis

Discriminant analysis was decided against for this project at an early stage as the results offered by linear regression are just as accurate and there is no need to employ this as an additional technique. Some papers were covered as background reading and one of those that stood out was a paper by Blaženka Divjak and Dijana Oreški study at the University of Zagreb entitled "Prediction of Academic Performance Using Discriminant Analysis" (43).

They make some key points about the significance of importance of prediction of performance as it can also effect student attitude towards their studies which if left un attended could lead to even worse grades if a student is performing badly and eventually becomes frustrated and gives up altogether.

3.2.1.6 Benefits of student data analysis

There are many benefits to be obtained from academic analysis of students and it can prove to help more than just the university if it is used to aid students who may be struggling. The expense of university and the strain this can place on social and family relationships can put enormous pressure on students to perform well and this have devastating detrimental effects if the student is struggling with the workload or simply the difficulty of their course. This stress can cause students to become disillusioned with the education system and feel alienated when they perform poorly and need to repeat a module or indeed a year of their course.

In a lot of cases students simply give up as they have had enough of the pressure and seek other careers with a low probability of returning to education ever again. Prediction of student performance can help pre-empt this and prioritise resources towards the students who need it most. A study into preventing drop-outs at the university of Groningen goes into great detail about the complexities of these issues and the importance of predicting performance. (44) It is a study into premature withdrawal from university due to academic failure of students and the causes and consequences of it. This is an issue affecting many people including students, their families and their teachers.

... Early college leaving is often associated with negative consequences for students, their families, and university administration, It can cause heavy unrealized costs to universities and families. A student leaving university without having completed his/her study may also be exposed to various psycho-social problems. Dissatisfaction with college experience, disruption of life plans, and being jobless or being engaged in minor jobs to earn much less over a life time are some among others.... (44) *pg. 4*

3.2.1.7 Academic burnout:

Student academic burnout is a topic which is commonly mentioned in studies into student behaviour and the causes of drop-outs from both schools and universities. Student burnout

occurs when the student is overloaded with the amount of work expected and becomes disillusioned with their study.

"Well Being And Performance In Academic Settings- The Predicting Role Of Self Efficiency" (45) by Edgar Bresó Esteve is a comprehensive study of the key factors leading to student burnout and analyses variables such as student cynicism, self-efficiency, exhaustion, dedication and performance in order to help predict burnout. Student burnout is defined as

> *... a syndrome that is commonly measured by three dimensions (i.e. exhaustion, cynicism, and lack of efficiency). It refers to feeling exhausted because of study demands and having a cynical and detached attitude towards ones study...* (45) *pg 41*

Their research studied the effectiveness of measures taken to try and combat burn-out. This was done using a sample group of students who had an intervention from the university, a sample control group of stressed students and of healthy happy students. This kind of behavioural information is not currently available for research at DCU however if collected could provide an interesting additional student analysis method. Even without access to such behavioural data some good lessons can be learnt from this paper.

Student burnout proves to be a key cause of drop outs from many universities and could be an explanation as to why so many students on a full-time module may be failing. Or similarly for a part-time module which demands too much of its students.

Predicting student burnout and putting procedures in place to deal with it should be a vital part of any modern university. Even if intervening only worked in a small number of cases it would still help the college by providing additional registration fees and a better student pass rate. It would also benefit society as a whole as more graduates would be produced and less people will leave the education system disillusioned and never to return. Stress levels can become quite high among students and even though there is help available students may not feel they are able to seek it. Becoming stressed and overloaded with work in university can have a big effect on students sometimes even leading to self-harm and suicide. If student data analysis of previous students can prevent this happening to just one student in the current population then it is a great success.

3.2.2 Prediction of student performance

Analysing previous results with an aid to improving modules is good but what about students already in the university? Surely we should try to help them as much as we can while we can. This is why prediction of current student performance is a vital part of analysis. Using results obtained by students in previous years we are able to predict how well a student will do in coming exams/degree based on their current performance and similarities between that and those who have gone before.

3.2.2.1 Case Studies

Several case studies were examined to enable an educated decision to be made on the best way of predicting student performance at DCU. One of the most useful papers encountered was "Predicting Academic Performance" (46) by Marcos Gallacher at the Universidad del Cema. This paper examines the accuracy of university admissions tests as a predictor of academic performance.

Admission test scores are used to predict how well students will do in university exams and then actual student results are compared to this to examine the accuracy of the predictions. Their conclusion was that admission tests can be a useful indicator of how well a student will do but are far from perfect predictions and should be viewed with a degree of caution. This paper raised some interesting points on the topic of student data analysis and provided good inspiration for this project.

Linear regression is used to good effect and some interesting results were found- "Linear regression model results (Annex 1) suggest that student performance (as measured by GPA) can be predicted using as independent variables the program that the student is enrolled in (economics or management) and admission test results in verbal and quantitative ability." (46) Pg 6 Similar techniques were applied in this project to try and replicate the linear regression analysis to model DCU academic data using different scales. The performance graph shown in Figure 45 was a good source on how to model student data using linear regression analysis techniques.

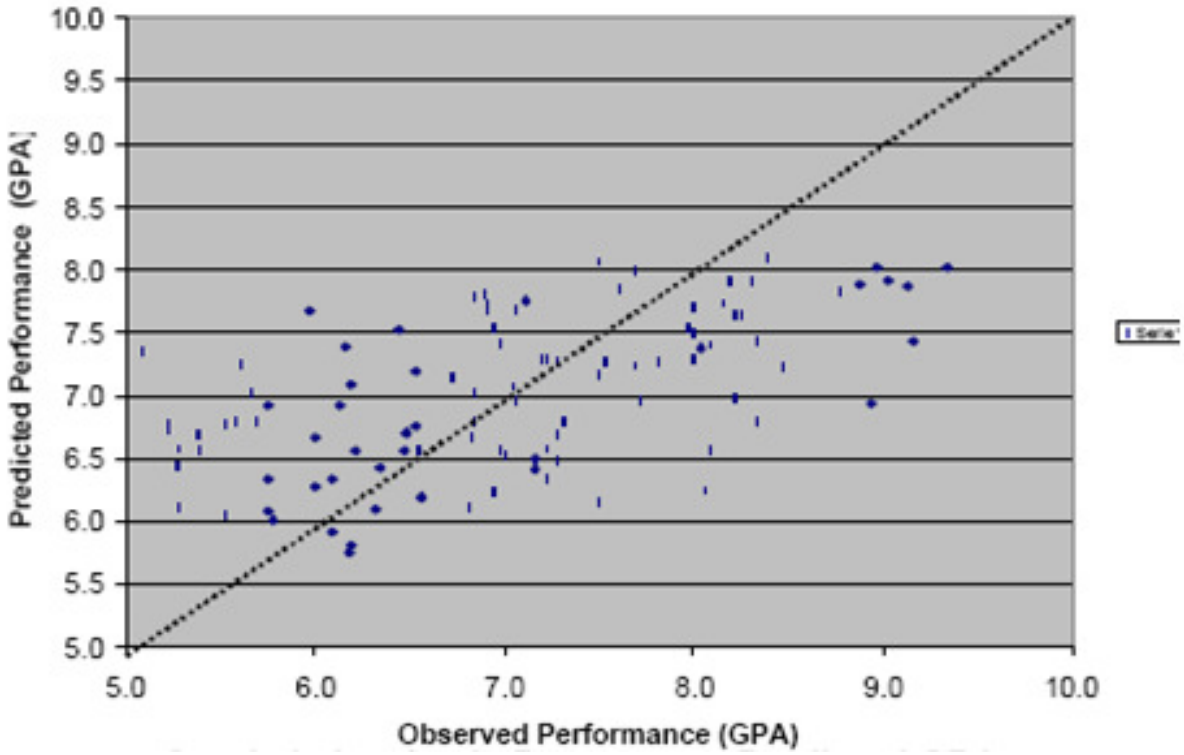


Figure 45 Student regression analysis Observed performance Vs. Predicted performance (46) pg. 13

This paper also examines some other interesting topics such as student improvement over time. An interesting example is posed of two students A and B taking the same course. "Student A shows a higher (graduating) overall GPA than B. However, B´s GPA in the last year of study is identical to A." (46) Pg 8 This example is a scenario many educators may come across and have to analyse and is a useful introduction into how to interpret student data. Many things could be derived from this. One theory could be that student B didn't perform well initially through lack of expertise but has raised their performance over time to match that of student A.

Student B may also have just realised the importance of their studies and began to make more of an effort in the past semester. Both these ideas look at things from student B's point of view but another valid theory could be that student B has made no improvement and student A could have experienced student burn out from over effort. Student B is steadily progressing but student A overloads themselves and cannot improve anymore and begin to lose interest and see grades decline. This kind of analysis could be vital to DCU to help predict students who are experiencing burnout and give them the help they need.

A metric for student improvement is formed to chart student progress as a more accurate indicator of whether a student is doing better or worse than in previous semesters. This analysis is very useful as it will provide more information on a students' progress through the university and even if they grades are still low if they are steadily improving they may not be as high risk as students whose grades are steadily in decline. This type of analysis could be useful when analysing data from technical courses such as science and engineering.

Science and Engineering courses are usually spread over 4 years with the first two years generally being slightly more theoretical and the last two years being more practical and hands on. Some students are more hands on and suited to practical aspects of the course so if a huge increase is noted in the GPA of a high risk student in the first semester of third they could be seen as less high risk and more of a practical than theoretical student. This could be backed up by further reference to their continuous assessment performance.

The scatter chart below provides an interesting look at student improvement versus predicted GPA.

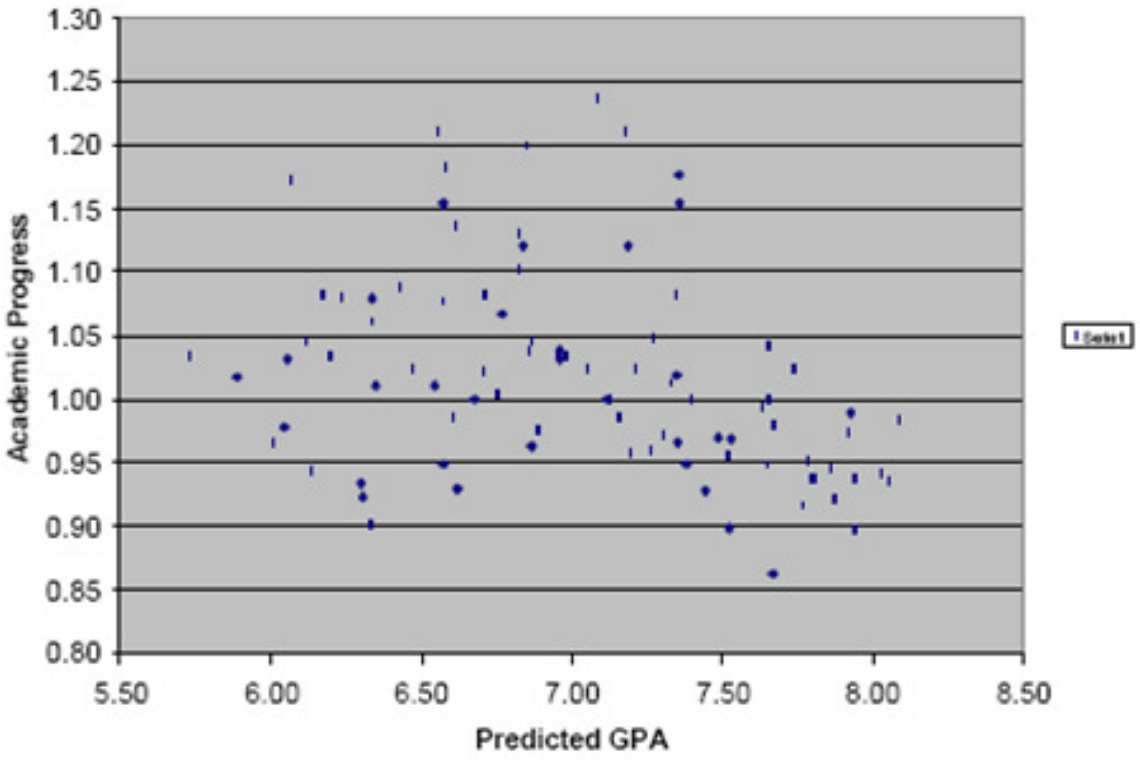


Figure 46 Scatter chart of Academic progress Vs Predicted GPA (46) pg. 13

In a lot of cases students will perform better in later years of the course than in the first year so accurate prediction of their grades can be helped by analysing their improvement or progress between years 1 and 2. The results of academic progress analysis need to be viewed with a broad mind though as weaker students can have a tendency to pick “easier” programs with less hours and demands. Student load also needs to be examined if results are to be accurate. It is also difficult to apply these types of analyses across courses as “comparisons of "progress" across programs can be difficult as the concept of "learning" may be interpreted differently for students being evaluated in "harder" as compared to "softer" subjects”. (46) Pg 8

3.2.2.2 Prediction using mapping:

Prediction of student performance involves the analysis of various scenarios and sometimes trying to map these scenarios graphically is a good way of understanding how best information can be interpreted to meet the requirements of these scenarios. For example when considering how likely a student is to fail and students who have a high fail risk happen to be male aged 22-24 and from Dublin then these things should be the first things to check when analysing a student’s performance. A good way of mapping these scenarios is through the use of Classification and Regression Trees which are discussed in length in “Predicting Student Retention and Academic Success at New Mexico Tech” (47) by Julie Luna.

This paper is a very useful resource as a guide to predicting student performance and retention and used many common methods to analyse student performance. Linear regression analysis, discriminant analysis and decision trees were all used to help discover the key variables that can be used to predict academic performance of first year students. An extremely interesting part of their research was the decision trees they used; Classification and Regression Trees or CART. CART is a non-parametric method for predicting an outcome based on facts about the data. They provide an excellent example of a system for predicting how likely a patient is to have a heart attack:

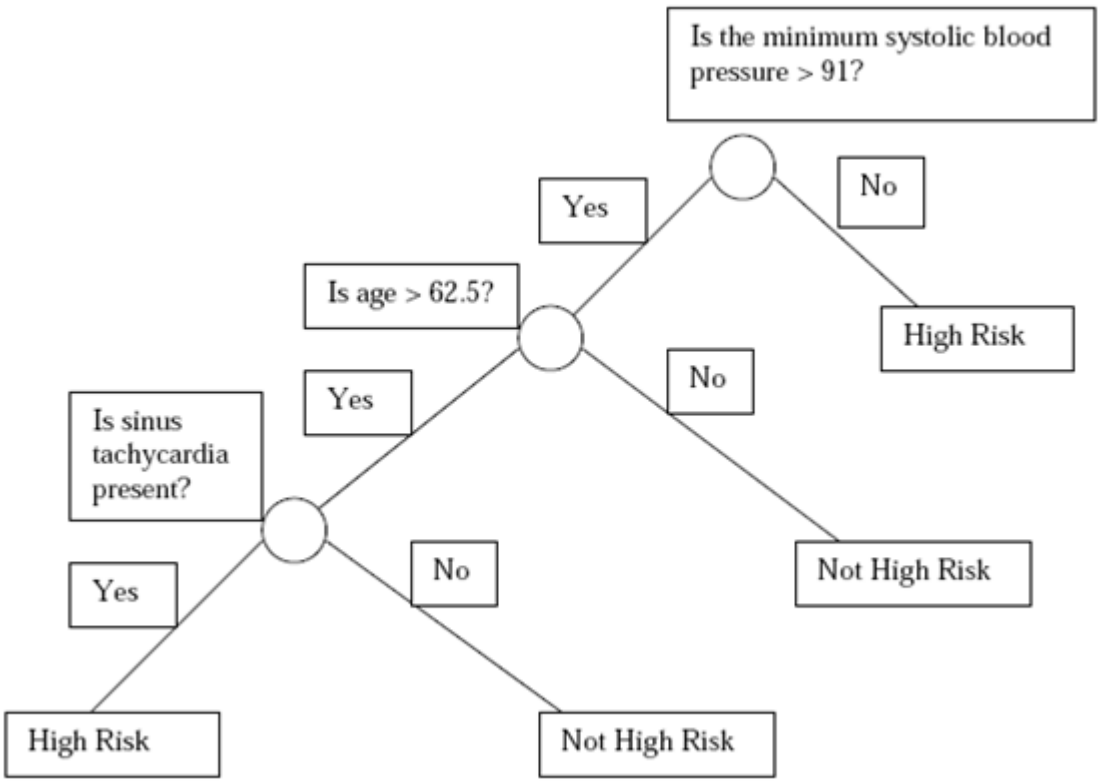


Figure 47 - CART example illustrating patient diagnosis (47) pg 9

This set-up could be easily modified to help predict how likely a student is to fail or drop out based on details about their academic history or other factors similar to diagnosing a patient. If for example Men aged under 23 and who have already failed 2 modules in their course have a 99% drop out rate than another student how falls into that category may be at high risk of dropping out too. CART analysis is a great way to think about patterns in data and understand their meaning in order to exploit correlations and predict future values.

3.2.2.2 Prediction using decision trees:

Additional different techniques were sought to provide a better overall idea of the types of student visualisations possible and some good examples of the use of various techniques for analysing academic performance including decision trees and Bayesian networks. An analysis of student data from Can Tho University in Viet Nam and the Asian Institute of Technology in Thailand (48) with students from 86 countries in order to formulate algorithms for accurate prediction of student performance.

In their research two techniques for prediction of student performance are compared; a decision tree and a Bayesian network. The details of these algorithms are beyond the scope of this project as they involve very detailed analysis of student attributes such as income, English skill level, marital status and gross national income that are not possible to analyse using the data currently available.

A similar style decision tree or Bayesian network would not be applicable in this project but the way the data is represented encouraged better representation of the data that is available. Predicted performance was measured using an interesting grouping of whether a student would achieve a fail, a fair result, a good result or a very good result in certain modules and courses. The groups were colour coded to improve the representation of the data to good effect. These groupings could be easily mirrored to the universities marking system of fail, pass, 2.2, 2.1 and 1.1. The bar chart below shows how effective this can be when comparing modules grade distributions:

{ Fail,
Fair,
Good,
VeryGood}

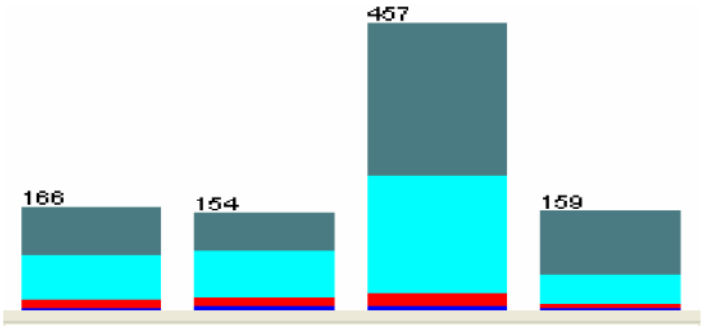


Figure 48 Grouped predicted performance results (48) Pg. 3

In their analysis they measure the percentage accuracy which is a great way to ensure accuracy of predictions and to just how often they may be wrong. Predictions were formed for students by repeatedly changing input parameters and measure differences in prediction compared to differences in input. This would be very useful to implement in this project as without knowledge of percentage error in predictions there is no way of knowing how accurate predictions will be and the validity of the predictions will be in doubt.

"Determination of factors influencing the achievement of the first-year university students using data mining methods" (49) by J.F. Superby is another paper that uses the neural network and decision tree based approach to student analysis. This study aims to predict academic performance of first year under-graduate students and where possible to pre-empt poor performance. They grouped students based on their predicted result and used a scale of

high, medium and low risk of failing. Those with a high risk of failure were singled out for extra tuition and special attention and the groups were formed as early in the semester as possible. A similar system could be used in DCU and assignment results in the first semester of first year students could be used as an initial vague prediction along with any other student data available. At the moment there is no source of student personal details and this would be needed to improve the accuracy of initial predictions.

Currently the only real predictor of results available is an analysis of the student's previous results or a comparison to the student body. The more details available about students and a better profile grouping within the student body can lead to more accurate results. For example, perhaps students male students from Dublin who enter the university aged 17 usually do poorly in first year exams and 17 year old male students from Donegal do a lot better in the same exams. If we were are analysing only basic student information we would think there is a 50:50 chance of 17 year old males failing these exams whereas if we had further data and the student we were analysing happened to be from Dublin their risk of doing poorly would rise and the university could make an effort to correct this while there was still time.

3.2.2.3 Prediction of performance using student profile:

Classification of students by risk profile is an innovative and creative idea that provides a useful way of predicting general student results and prioritising resources. Figure 49 Student Classification by risk scatter chart with risk lines pg. 3 (49) is an excellent example of how this could be applied in an easy to understand visual form. The plot details the results students obtained at the end of year in first year compared to their average grade in exams at the end of semester 1.

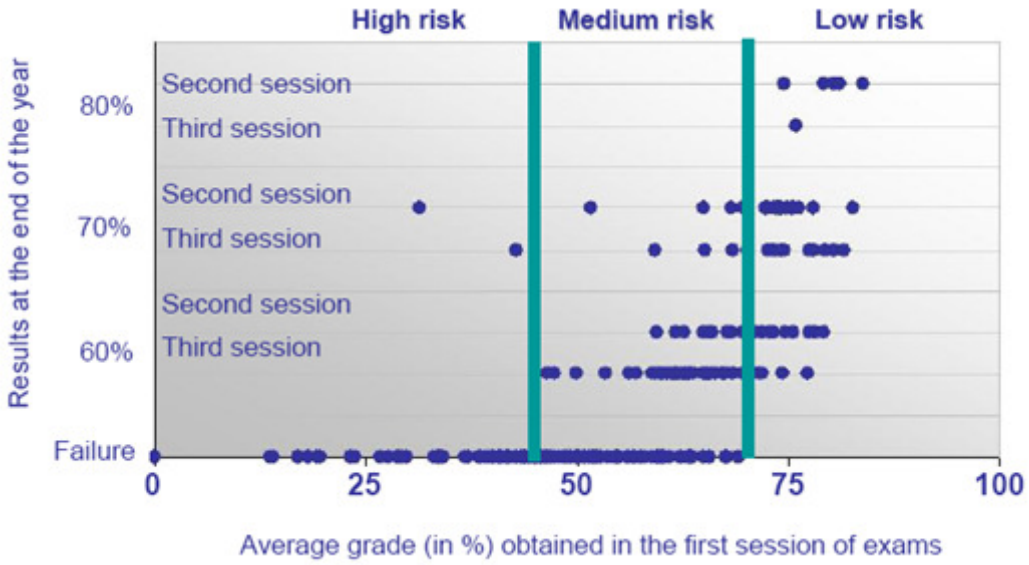


Figure 49 Student Classification by risk scatter chart with risk lines (49) pg. 3

This could easily be altered to update as the student progresses and the grade at the end of first semester could initially be their assignment 1 averages in the first semester as the results of these are usually available during the semester. This would enable student classification as early as possible to help get help to the high risk students as soon as possible. This study based their predictions on analysis of student attitude, behaviour, previous school results and other data through the use of a survey of 533 students. This data is not currently available at DCU but would be useful if collected. They analysed the survey results using discriminant analysis, decision trees and neural networks.

A topic encountered frequently in academic analysis papers is analysis of first year students to gauge their initial reaction to university examinations as a means of assessing how their progress will continue. This is not always an accurate prediction model though as student results often improve significantly after the initial first year teething stage.

A paper which backs up this way of thinking strongly is “A Comparison of the Academic Experiences and Achievement of University Students Entering by Traditional and Non-traditional Means” (50) This study examines performance of students but including previous qualifications as a way of classifying first time students against those who are returning to education and have already completed a degree or a course at another university. This removes the bias of higher averages from the students who are more experienced learners and enables for more accurate performance prediction.

This paper also examines the concept of analysis of student load to help predict results. This is a good idea as students with a lower load should be able to perform better than those with an extensive workload. Various factors could be used for analysis such as how many modules a student takes per semester or how many hours how many hours per week they have to attend lectures. This study used student credits to gauge **student load** and took this into account when predicting performance, giving a more accurate prediction when comparing students as more comparable metrics were being examined and the predictions narrowed down to more similar cases.

A study that was of good reference in formulating thinking in this area was "Predicting Academic Performance of Pharmacy Students: Demographic Comparisons" from the University of Texas at Austin (51). They developed several hypotheses of which metrics predict performance best and why that is quite useful when deciding upon the metrics to use in this project.

3.2.3 Module analysis

Analysis of individual modules and comparing their student grade statistics can be a very informative way of assessing how well or badly a module may be doing and it's effect on individual student performance and the course overall. If universities can find ways of improving the output of modules it can hugely benefit the grades of students on the course especially if it is seen as a core module as for example the object oriented programming(OOP) module offered to engineering students at DCU at both under-graduate and post-graduate levels.

Many further modules rely on the students' ability to program as most assignments require the submission of a programming exercise to show the understanding of an algorithm. In some cases the student may find they understand the course assignment and the algorithm extremely well but when posed with solving the problem programmatically they can have extreme difficulties resulting in poor performance in the module overall.

A study into the "Effects of Peer Tutoring, Attitude and Personality on Academic Performance of First Year Introductory Programming Students" by Paul Golding, Lisa Facey-Shaw and Vanesa Tennant (52) is an excellent reference point to this subject and is

an analysis of the performance of first-year Computer Science students at the University of Technology in Jamaica. They state that "*Programming is the core of any computer science degree. Failure to grasp programming concepts such as logistical reasoning and will have a negative impact on students academic performance*" (52) pg. 1. This study investigated the causes of poor performance in an introductory programming module in an attempt to improve student performance in other modules and in the course overall.

Two groups of students were selected: one experimental group and one control group. The two groups were given similar programming exercises with the control group working alone and the experimental group working with a peer tutor and with access to the course tutor. The results showed that peer tutoring had a significant impact on students performance in the module and that there was no significant impact on performance caused by personality type.

The information gathered from this study proved very useful to the Jamaican University and shows how important improving a single module could improve the performance of a student at degree level. It shows that particular attention needs to be paid to core modules such as OOP when analysing a student's performance as poor performance in these modules could be a strong predictor of poor overall performance.

An indicator of this could be if a student is doing poorly in all of their continuous assessment work yet doing well in exams on a computer science degree or similar. If the student was doing excellently at continuous assessment it could indicate they are not doing the work themselves but to pass the exam takes real understanding of the course and they could be struggling with the programming aspects of the continuous assessment.

"The Dynamics of Early School Leaving" (53) by Delma Byrne and Emer Smyth presents an interesting analysis of student retention at secondary school level and analysed some key performance indicators that could prove useful when analysing module and course performance. Student retention on particular courses could be analysed to see if they have a high dropout rate and to assess the difficulty of the course.

Figure 50 could be applied to DCU students who drop out grouped by various factors such as age to help predict future drop outs, implement steps to reduce this figure and then to

analyse the effect of the steps taken. Figure 50 could also be used to assess retention of students in the university as a whole and spot variation between courses. If course a course has a high dropout rate it could be classified as more difficult and an individual student's chance of dropping out would increase.

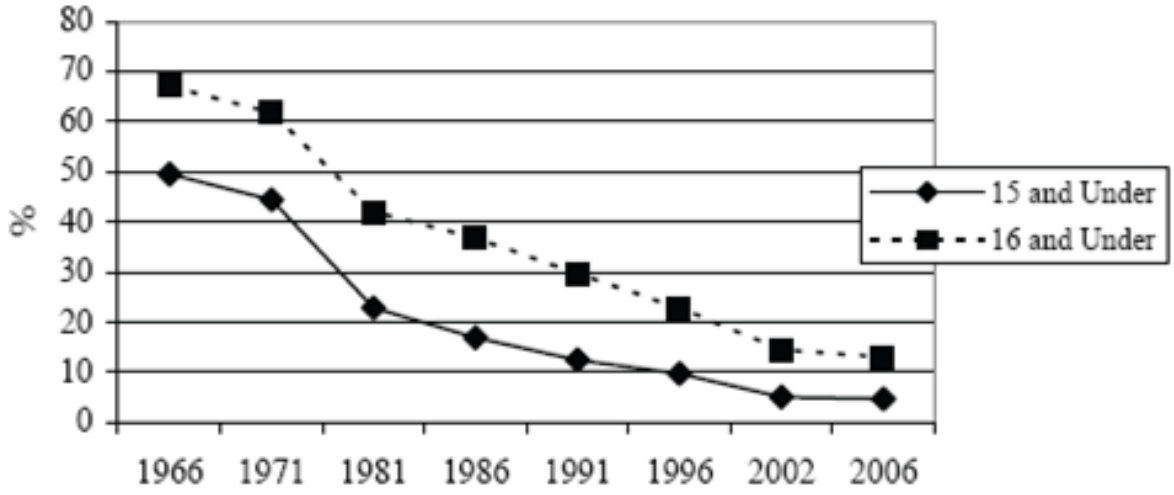


Figure 50 Line chart – Percentage of students who drop out by age (53)

Analysis of this would help the university focus their attention on the modules and courses that need most attention. Figure 51 below is a useful template for modelling this:

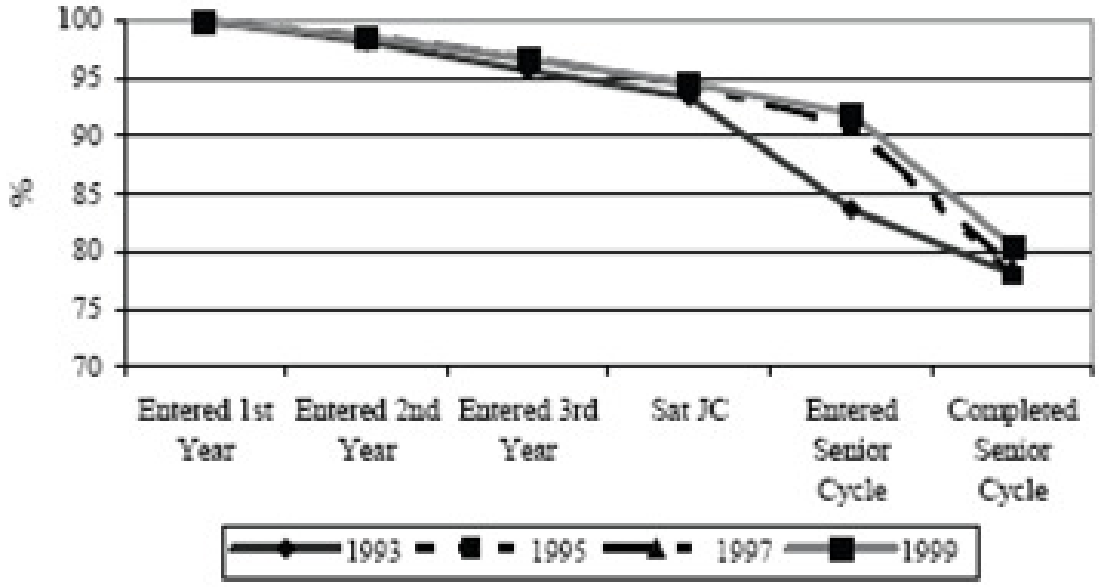


Figure 51 Retention over the School Career, Entrants to second level education 1993 to 1999 (53) pg. 21

Figure 51 compares student retention by year but could easily be modified to show retention by courses or modules. This paper also explores other interesting analysis parameters including gender analysis of qualification by gender which could be used to spot trends in

DCU students. Figure 52 shows qualification by gender where F represents Female, M represents male and:

- NQ: No Qualifications
- JC: Junior Certificate
- LC: Leaving Certificate

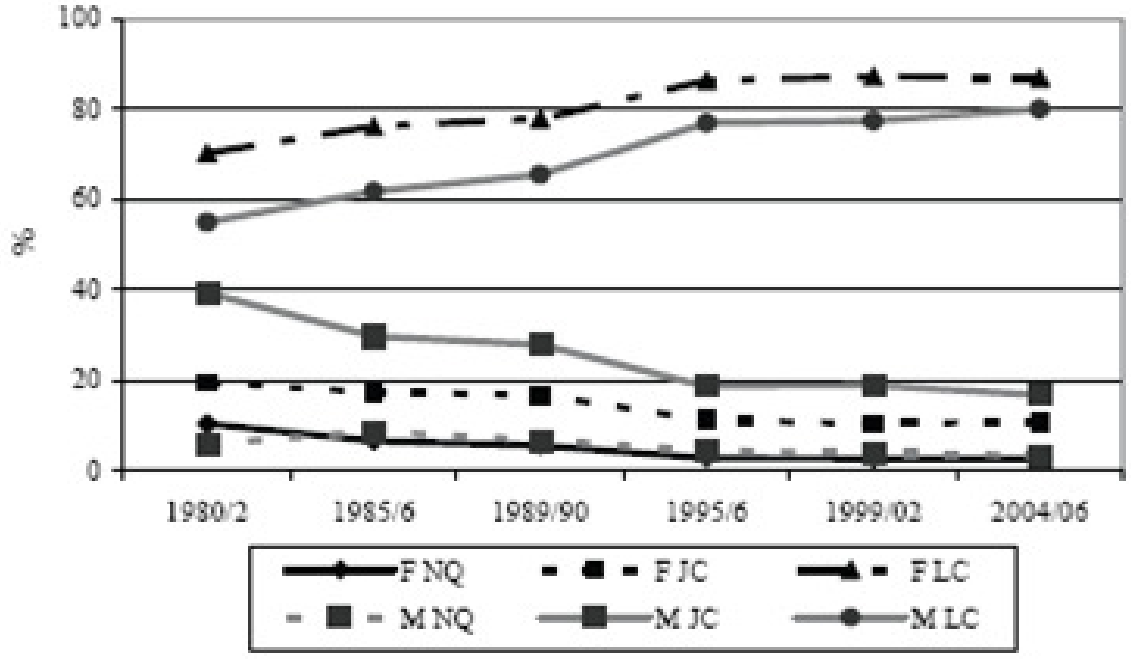


Figure 52 Educational qualifications by Gender, 1980-2006 [3.2.2.41] pg. 22

This could be used to plot drop out, undergraduate and postgraduate students from the university by gender, course, module or various other possibilities. It could also be used to show the percentage of students obtaining first class honours, second class honours, pass and fail grades in both module and courses. This paper also did some interesting research into the effect truancy and being late for class effected student dropouts.

Figure 53 shows how a greater proportion of students who are late for school end up dropping out. While this type of data is not available for this project it may be useful to record data such as labs or lectures missed for some courses. Engineering practical laboratory classes are very important to overall grades so should be treated as a high priority when predicting student performance. Similarly for courses like nursing where lecture attendance can be mandatory it would be useful to predict a student who has already missed nearly the maximum quota of missed classes is at high risk of missing more classes and failing the course.

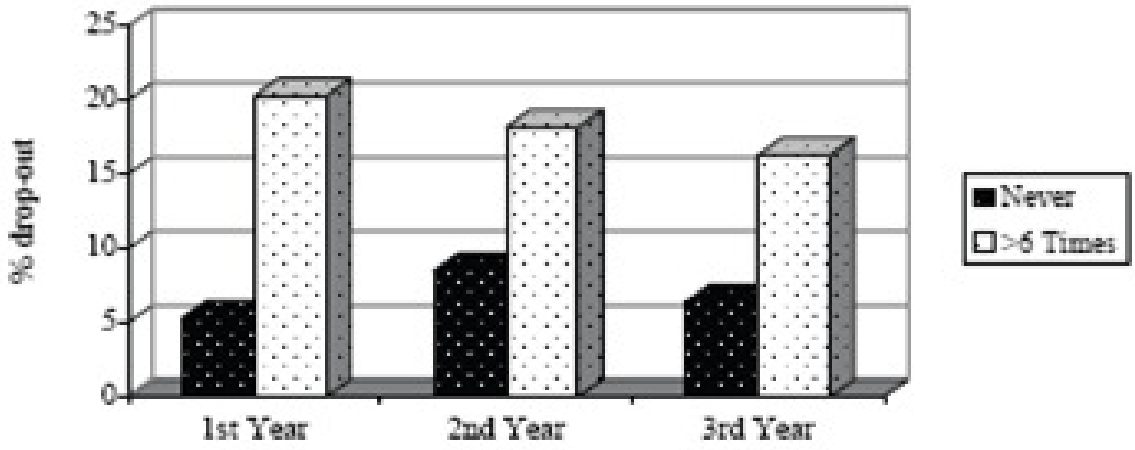


Figure 53- Early school leaving by prevalence of being late for school more than six times in Junior Cycle (53) pg. 68

3.2.4 Analysis of Course and Overall University Registration

Analysis of registration figures for particular courses and the university can indicate which courses are the most popular and can be a useful indicator as to how well the university is doing overall and in each specific course. Various factors can be analysed but the key performance indicators are registration count, drop out count and grade distribution count i.e. the count of first class honours, second class honours grade 1, second class honours grade 2, pass and fail results of students. By doing an in-depth analysis o each individual course better advice can be given to student when deciding on a course to take and this can help students choose modules that they are better at and therefore do better in.

Some key background on this subject relates to how students choose which course to take and The Times "Good University Guide 2010" was a good reference point for this. It contains a detailed listing and description of all courses and universities in the UK using a ranking system to help students find the course that suits them best. This book is a tool for prospective university students in the U.K. and is an insight into the factors that decide where they study. This kind of assessment of university performance can provide some useful lessons in how to improve the performance of DCU by focusing on how module and course performance is judged by prospective students. (54)

A lot of academic papers on the subject of university analysis were examined and an excellent foundation for this type of analysis is provided in a publication from Stanford University called “Multiple Choice: Charter School Performance in 16 States” (55).The study takes a detailed look at the performance of American private schools, or in other words charter schools. It presents some good analysis on the topic of analysis of institution performance and provides some innovative graphing methods that were of great use in this project. The study has a huge, broad sample set of data with hundreds of thousands of samples from all across America. This improves the universality of the study and the results can be more trusted for use by other parties. (55) pg 20-21

They introduce the idea of a control student which is an excellent idea to use to measure validity of predictions and analyse student performance overall. A control student could be classified by gender or time span depending on the type of analysis being done and would represent the average GPA of a sample of students to be compared with the performance of an individual. Several of these can be included on one combination chart to try and spot patterns in performance.

They have access to far more specific student data than is covered in the scope of this project such as student race, non-native English speakers and those students from a poorer background which helps them in their analysis. It could be a good idea for DCU to collect this type of data in future years as the more we can classify patterns in student behaviour and increase the accuracy of grade predictions the more we can make educated decisions and help students who might potentially struggle. Figure 54 shows how this type of data was used to show that black and Hispanic students were performing more poorly than students on average. The scaling of this bar chart is discussed in further detail in the next paragraph. (55) pg 32

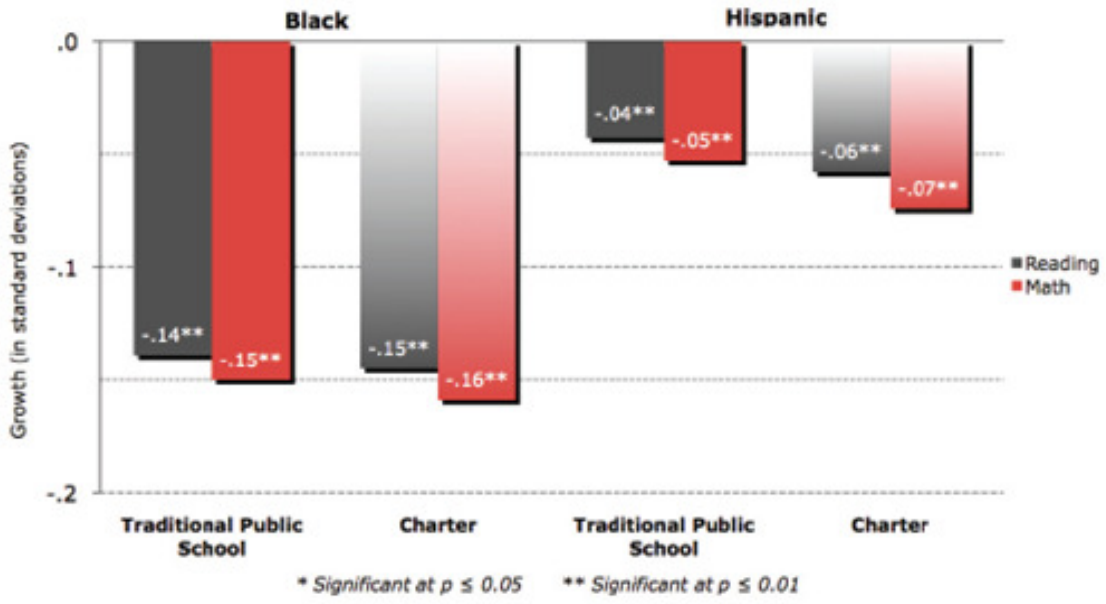


Figure 54 Charter School effect on Black and Hispanic Students (55) pg. 32

An extremely interesting student GPA analysis scale is used to graph data whereby students are scored based on whether or not they were above or below the state average. "Their score was 0 if they scored at the state average on the achievement test for that year and subject. Negative scores indicate their score was below the state average (with a -1 showing they were one standard deviation below the state average) and positive if their score was above. (55) pg 25

This proved great inspiration for this project as rather than plotting GPA using percentages all the time we can look at the broader picture of how students are performing compared to their peers. Several peer groups can be formed to provide different types of comparisons to see if a student is for example above average for males but sub-par overall or doing excellently for an engineer but poorly compared to the overall student population. The bar chart in Figure 55 is an example of how this kind of information can be plotted visually.

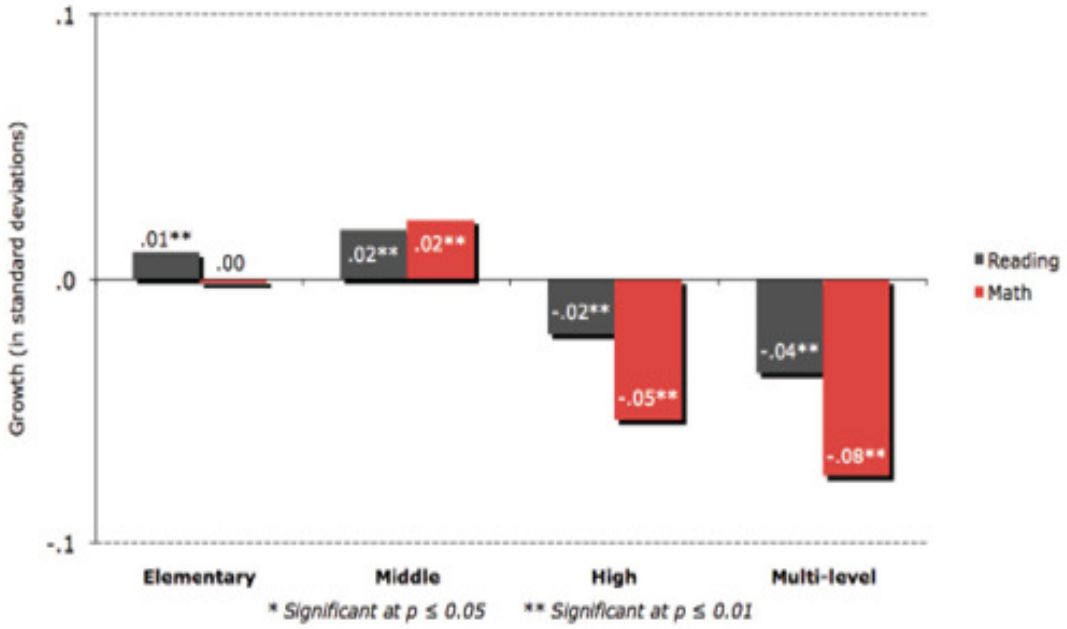


Figure 55 Charter School Effect by Grade Span (55) pg. 29

This prompted an idea for this project to plot student performance in modules for continuous assessment, exam mark and overall mark using this scale of how far above or below the standard deviation of the student population they are. It can also be applied to module analysis to see how well the module averages are doing compared to averages across all modules.

A broader example of overall school performance is done that transfers well when analysing modules, courses and faculties. A scale of "charter growth" is used, which means the GPA improvement/decline of each school. This is a useful analysis because if for example a module is doing poorly and steps are taken to change this, the improvement as a result of these steps can be modelled here to see how effective they were. Universities would also be able to see which faculties have the best increasing and decreasing standards. Figure 56 shows student average reading score against charter growth or improvement in standard deviations grouped by school.

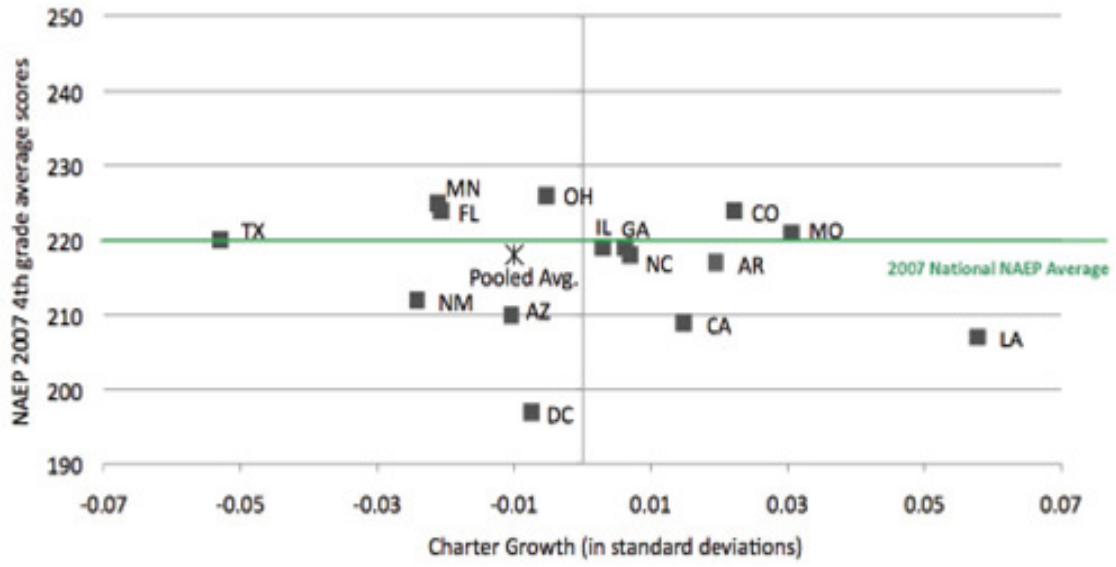


Figure 56 Charter Growth Compared to 2007 NAEP State by State – Reading (55) pg. 9

This paper provided many ideas in how this project could analyse modules and faculties and gave way for the idea to plot module average GPA improvement to help analyse if a module/course is doing better than previous years. Using a percentage difference value rather than a raw GPA makes it far easier to judge the key point – whether the metric is performing better or worse. Pre-calculating these values and visualising the difference values as opposed to the raw data can make it easier to infer performance without over complicating things.

Another way of doing this is to plot degrees above and below the standard deviation rather than in general and the scale can be reduced to a much smaller size between 0 and 1 where

- 0 is at standard deviation,
- 1 is above standard deviation,
- -1 is below standard deviation.

The way they analyse their data also gave some useful ideas for future additions to this project such as their use of school entry requirements to account for different difficulties in schools and ensure any predictions aren't biased. The paper also introduces the idea of the use of a control student and this was an excellent idea that is used in this project. A control student for various student population types such as all students all years, all male students all female students, all students taking a particular module etc. can be used as a benchmark for comparing individual results to profiles of control students in the overall student body.

This can help pinpoint correlations between subsets of students that perform poorly and aid in attempts to focus on the most troublesome profile groups.

Further before undiscovered analysis techniques were found in another paper by Alma Laurea which is a company run by a consortium of Italian universities and backed by the Ministry for Education, University and Research. (56) They perform innovative analysis on

This study performs analysis of university performance before and after reforms was made to a university to improve performance and provides great visualisations of the data that can help show improvement or not. They focus on several interesting factors including:

- student graduates per year by enrolment age,
- student age at graduation,
- graduates by prescribed course duration,
- graduates degree completion times,
- Months taken to complete dissertation/final exam.

They used colourful bar charts to display this data visually and this paper was one of the most innovative found during the research for this project when it came to new ideas about analysing student performance for graphing. The plots mentioned above are shown below:

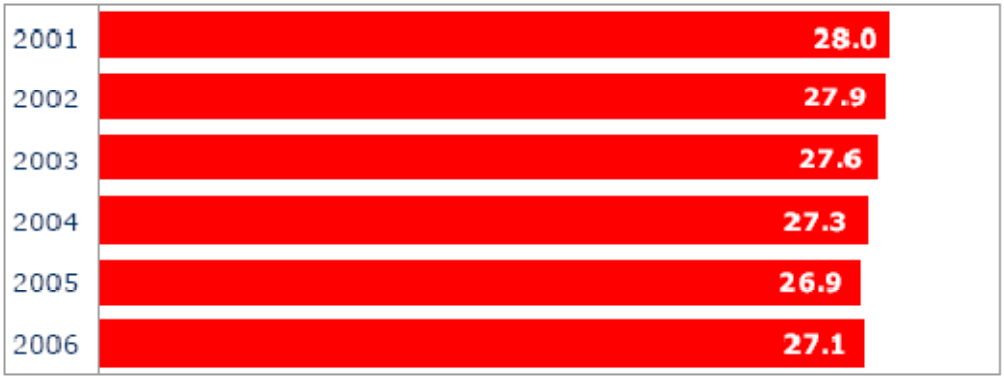


Figure 57 Age at graduation column chart (56)

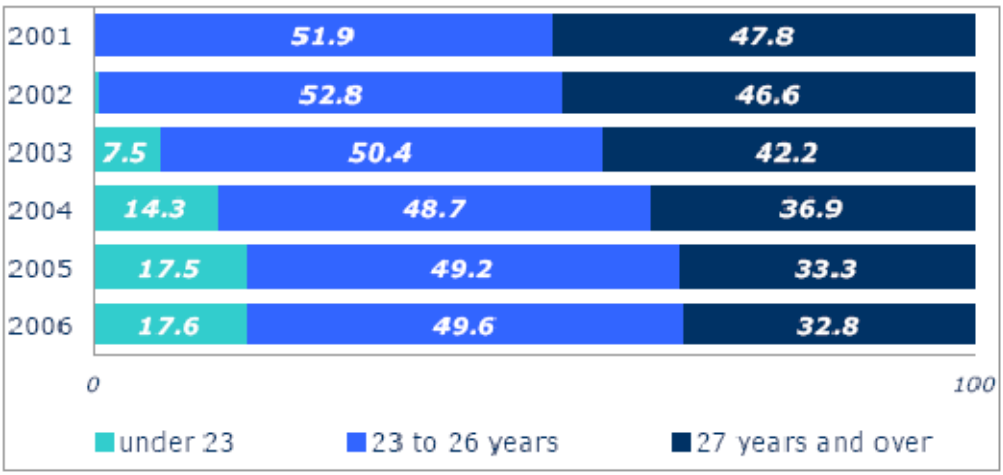


Figure 58 Distribution of graduates by age (56)

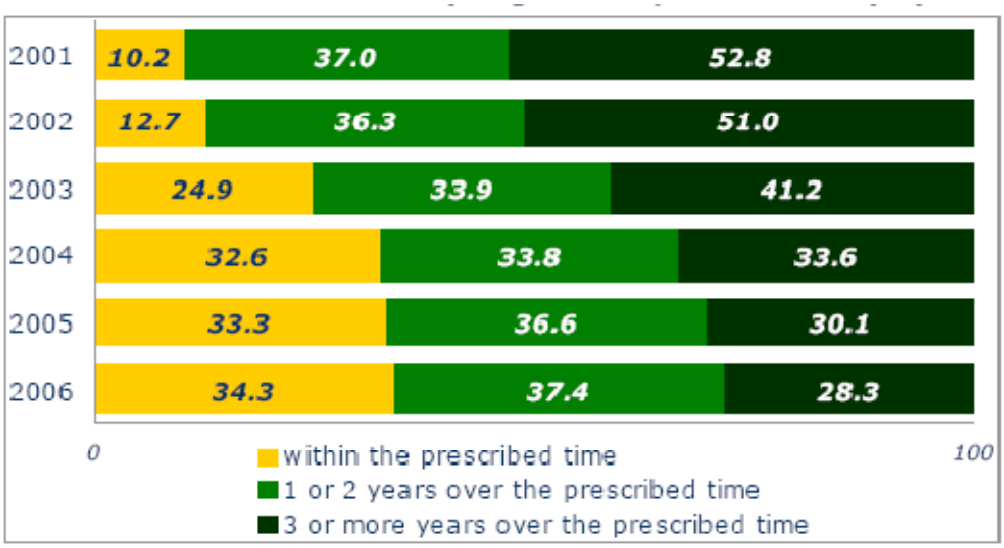


Figure 59 Graduates by degree completion times (56)

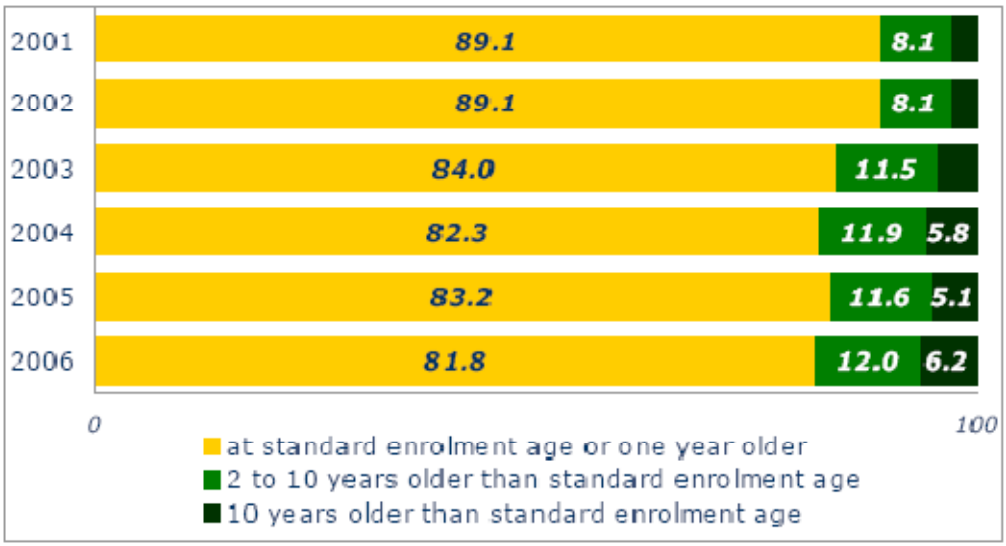


Figure 60 Graph of percentage graduates by enrolment age (56)

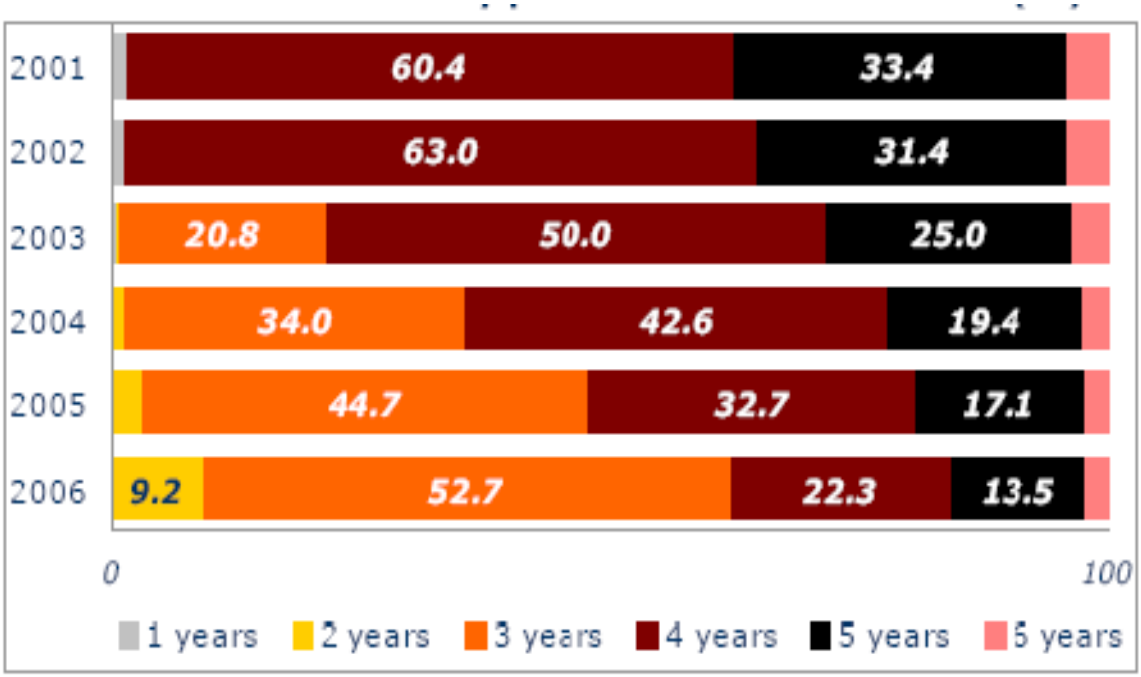


Figure 61 Graph of percentage graduates by course duration (56)

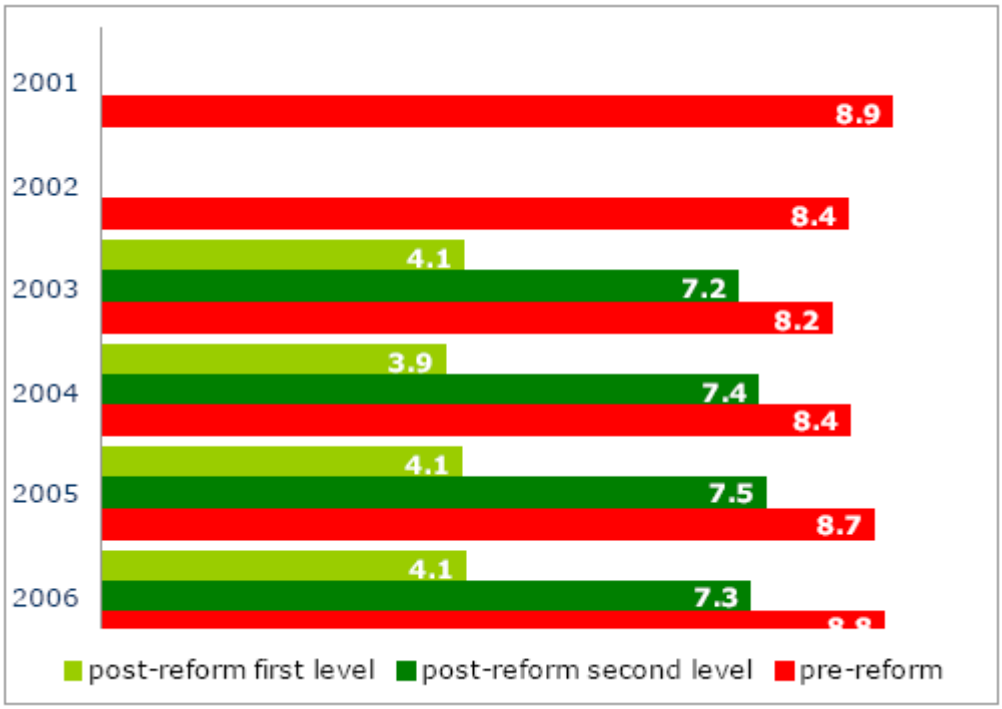


Figure 62 Graph of months taken by students to complete their dissertation/final examination (56)

These plots gave some great ideas on further ways of interpreting the data provided for this project and are put to good use in the application. The use of percentages rather than numbers gives the user a better immediate impression of the distribution of the values than

by using raw count values and this is a feature that is sought to be used in the analysis application for this project.

Judging the academic performance of a university or school can be an extremely difficult task and many variables need to be taken into consideration. A great source of reference on this topic was this paper written by The Maine Education Policy Research Institute at the University of Southern Maine (57). They state "Assessing the academic performance record of a school is a complex task that requires good data, thoughtful analysis, and sound professional judgment" and have developed six metrics or indices of performance that they recommend in assessment of school performance.

They measure student proficiency using a scale of four categories: Exceeds, Meets, Partially Meets and Does not Meet. This could be applied to a university using the 1.1, 2.1, P, F scale. They measure performance based on:

1. *Average MEA Composite Scale Core Index:*
   The three year school average for reading, writing, math and science.
2. *Percent At Least Meets MEA Proficiency Index:*
   The percentage of students who at east meet state proficiency average over 3 years.
3. *Percent At Least Partially Meets MEA Proficiency Index:*
   This represents the percentage of students who are in the top 3 categories of proficiency.
4. *Economically Advantaged Students Average MEA Index:*
   2 year average score of students not eligible for a free or reduced price school lunch.
5. *Economically Disadvantaged Students Average MEA Index:*
   2 year average score of students who are eligible for a free or reduced price school lunch.
6. *Average MEA Above or Below Predicted MEA Index:*
   A measure of how far above and below the predicted MEA a schools Average MEA is.

Whilst these parameters cannot be applied directly to this project they do provide some good ideas. Perhaps there is a pattern that students who are less well off in the university do better than those who are well off as they are more hard working and have less money for socialising. It could also be the opposite but without the data available for analysis no true

judgement can be made. This may be another thing worth adding to a student information database for future analysis.

More interesting ideas can be derived from the scale they use for analysis and the metrics examined. Metric 2 could be used to show the distribution of students who at least pass a module or even the course out right and can be grouped into 1.1, 2.1 and Pass groupings to see how well students are doing overall rather than just look at pass and fail rates. Many students may be on the border of a fail and the subject may be just too difficult or being taught poorly. This paper uses bar charts to plot their results and provided some interesting ideas for this project as they use both positive and negative axis values when plotting to indicate whether a value is above or below the average. Some examples of the graphs are provided in Figure 63 and Figure 64.

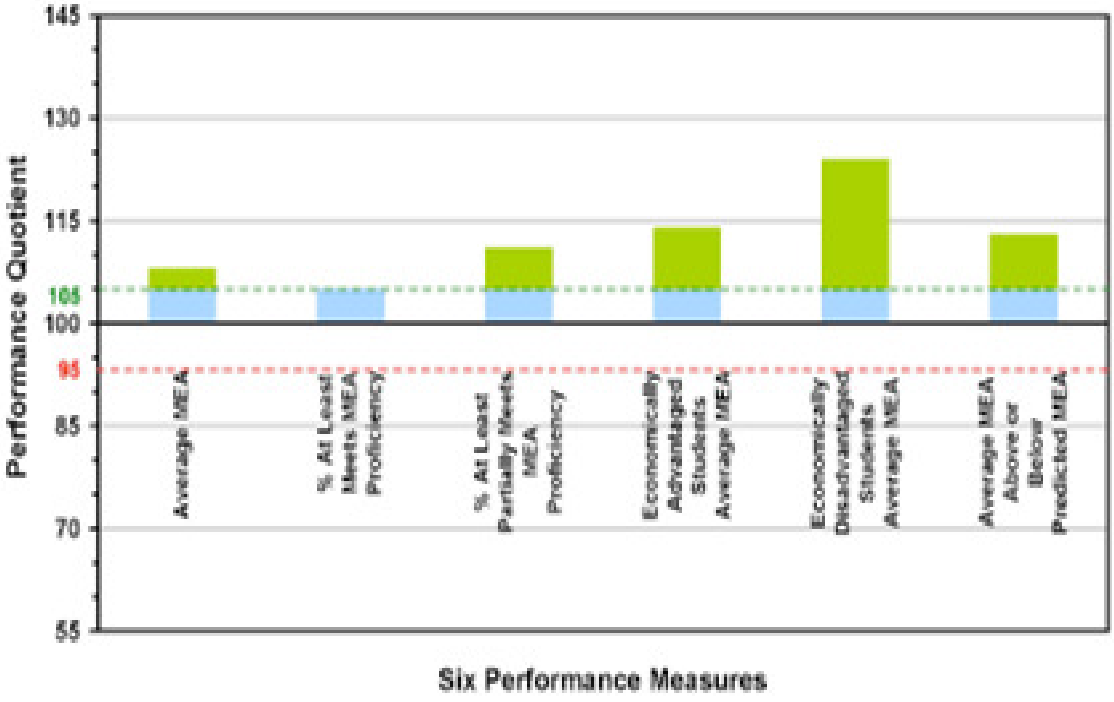


Figure 63 Indices for assessing student performance – Student performing well (57) pg. 3-4

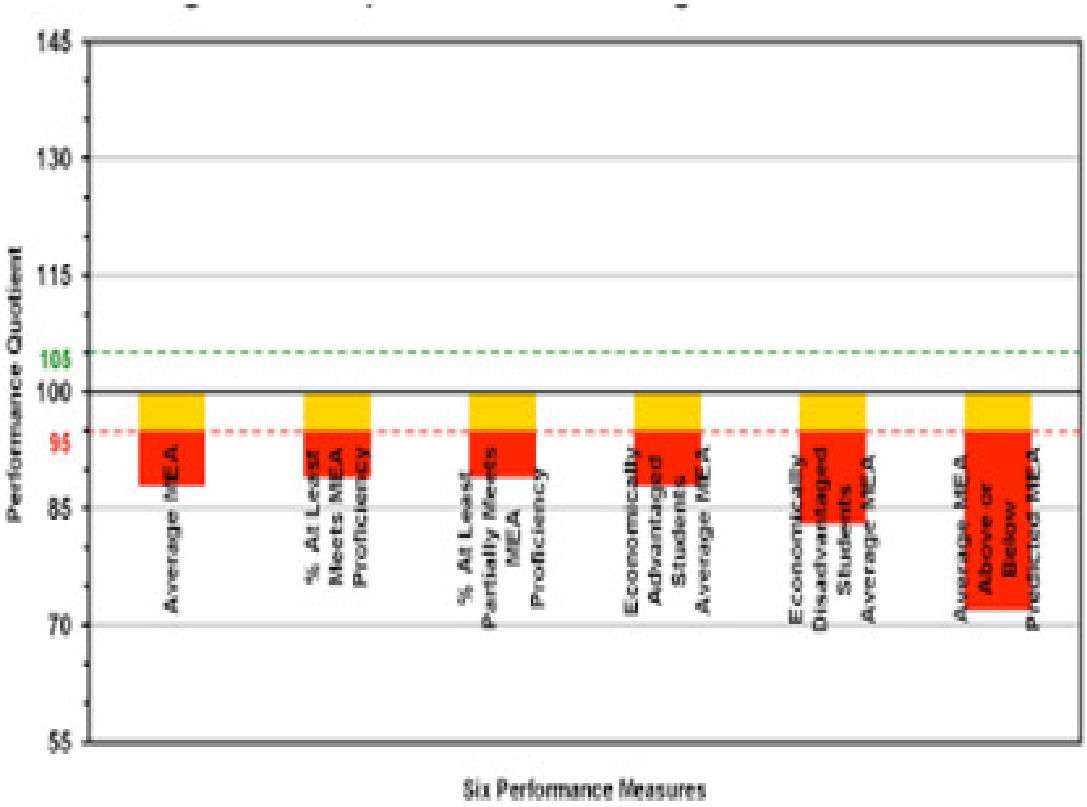


Figure 64 Indices for assessing student performance – Student performing poorly (57) pg. 3-4

These graphs inspired further ideas for this project as to what could be interesting things to plot including:

- Analysing degree completion times across all faculties
- Analysing the months on average taken to complete the thesis
- Graduates by enrolment age per module and course

All of these and many more other metrics could be conveyed far more effectively using these kinds of bar charts and this paper was of excellent help towards improving the quality of data being visualised and ensuring as many useful and accurate deductions as possible can be easily interpreted from it.

3.3 Other research of note

3.3.1 Other student performance graphing systems

Overview:

As part of the research for this project other methods and applications available for analysing student academic information were also investigated to try and learn as much about what is currently on the market as possible. Two systems are discussed in detail, the first being the ePortal (58) system used in some Irish second-level schools and the PADDS (59) system which has a very similar aim to this project. Both systems are discussed in further detail in the sections below and good points and bad points from each are analysed.

ePortal:

A system used to analyse performance of secondary school students was used to inspire work on this project and provided some useful ideas. Some screenshots of this application are given. All confidential student information has been removed. This system is built using the ePortal system (58) and offers a high level analysis of student performance using very basic static visualisations. Access to the system for academic purposes was kindly temporarily granted by Grange Community School in Donaghmede, Dublin 13 (60).

This system gave some useful ideas to be used as part of this project such as the ability to add comments on student performance to a student's data file. This is not possible in this project at the moment as the data source contains pre-defined tables and to insure it will work without any changes on the database side no new fields are added to pre-existing tables or new tables created.

It would be useful to add additional fields to store notes about student behaviour or particular personal problems they may be having that could indicate the cause of a decrease in student performance. This would help lecturers to give a more personal approach to student analysis and could make students feel more accepted and in a way wanted. Poor academic results can often spiral out of control as they can cause disillusionment with and resentment of studying and if students beginning to show these patterns are helped early on in the process it could make a big difference in them continuing to graduate or not.

The student analysis overview below was a good template for how the presentation of this application could be represented and shows best practices for conveying a lot of information to the user in a clear and easy to understand manner.

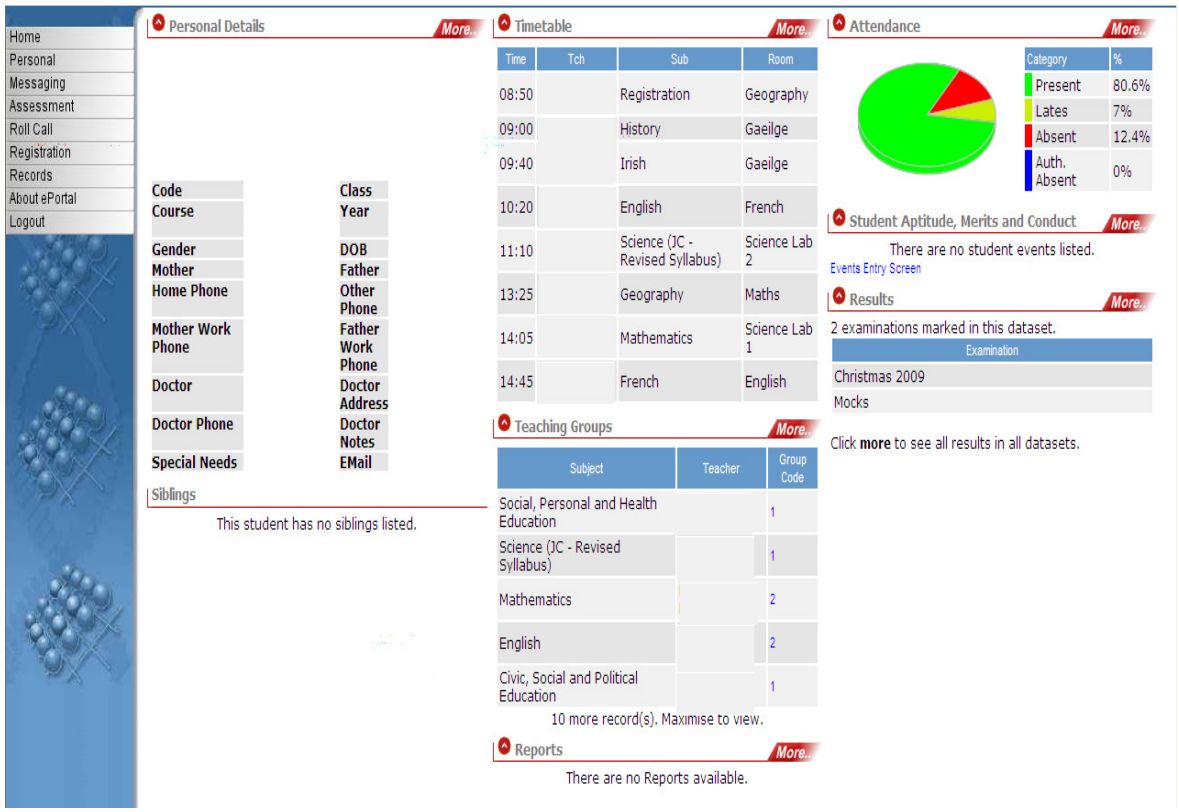


Figure 65 ePortal Student Analysis – Student analysis homepage (58)

Attendance is a key performance indicator at secondary school level and an example screenshot of attendance analysis along with an attendance data table is provided in Figure 66. Although not currently directly applicable to this project the techniques used can be modified to suit the analysis of student registration and drop out figures. It could be used in a sense to monitor the attendance of students to mandatory sessions for examples engineering laboratories or compulsory lectures for courses like nursing.

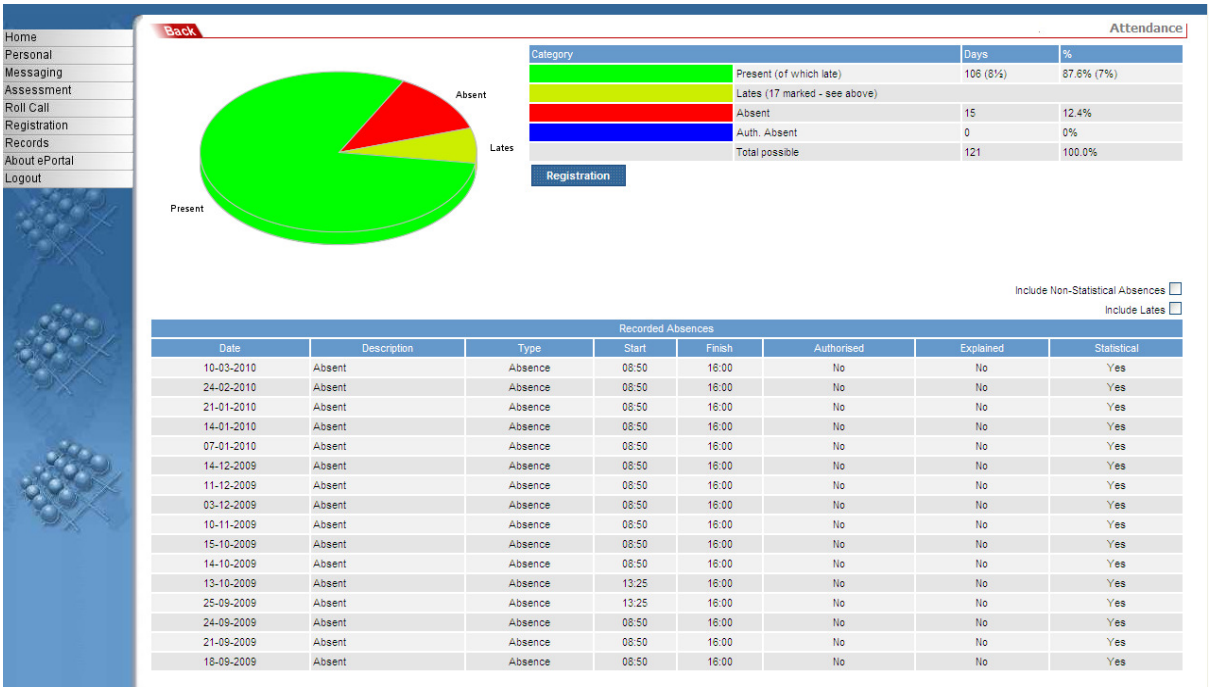


Figure 66 ePortal Student Analysis – Analysis of attendance (58)

This type of data is not currently stored in the student data base so is unavailable for analysis however a possible further area of research could be a simple widget built in to the visualisation application that would allow lecturers to mark class attendance and have the results directly populated in the database. This would be of great help to pre-empt some situations like for example a student who hasn't attended many lectures and has failed their first assignment may need immediate attention and priority care to try and give them the best chance possible of passing.

The ePortal application has further innovations in its offering of a chart plotting tool shown in Figure 67 where user specified variables can be used to plot various metrics. This is an excellent idea and something that would be useful to include in this project however it is a lot more complicated when using higher level visualisations such as the Google Visualisation API than it is to produce simple static graphics as provided in ePortal.

Users are more used to far more visually appealing and interactive online experiences now so this system suffers in that respect. The system does however emphasise the point that it's better to graph informative data in a visually poor way than it s to graph uninformative data in a visually appealing way. The graphs in this application are out-dated by newer technology but the data they are conveying to the user is as relevant now as it ever was.

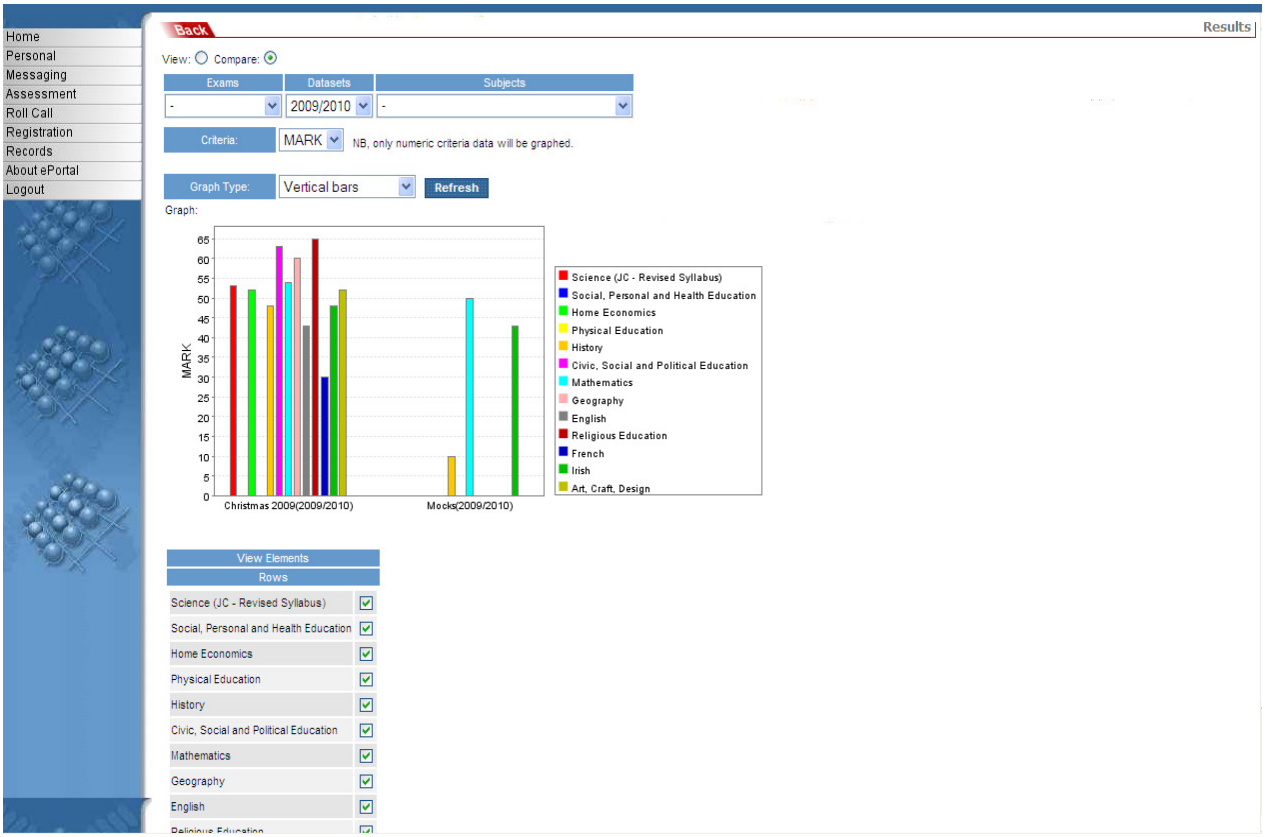


Figure 67 ePortal Student Analysis – Draw bespoke chart (58)

Performance-based Academic Decision-Support System:

Another student analysis system considered is a system called the Performance-based Academic Decision-Support System (PADSS) (59) developed at the Department of Electrical and Electronic Engineering, Eastern Mediterranean University in Turkey by DERVIS Z. DENIZ and IBRAHIM ERSAN. It was developed to show the different ways in which student performance data can be analysed and presented to aid in academic decision making. They offer very similar performance metrics to those used in this project and analysis can be viewed at university, department, school and individual student levels. They present some interesting ideas regarding the use of analysis and visualisation techniques to monitor the effectiveness of decisions made by the university along with traditional student analysis techniques.

This system provided some excellent performance indication metrics that are used in this project and helped to rationalise the path the research was taking. The same types of analysis with slightly different metrics were found in several papers and eventually a few key indicators of performance stood out. PADSS allows users to generate their own

visualisations and the column chart in Figure 68 below inspired the use of column charts to display grade distributions in this project:

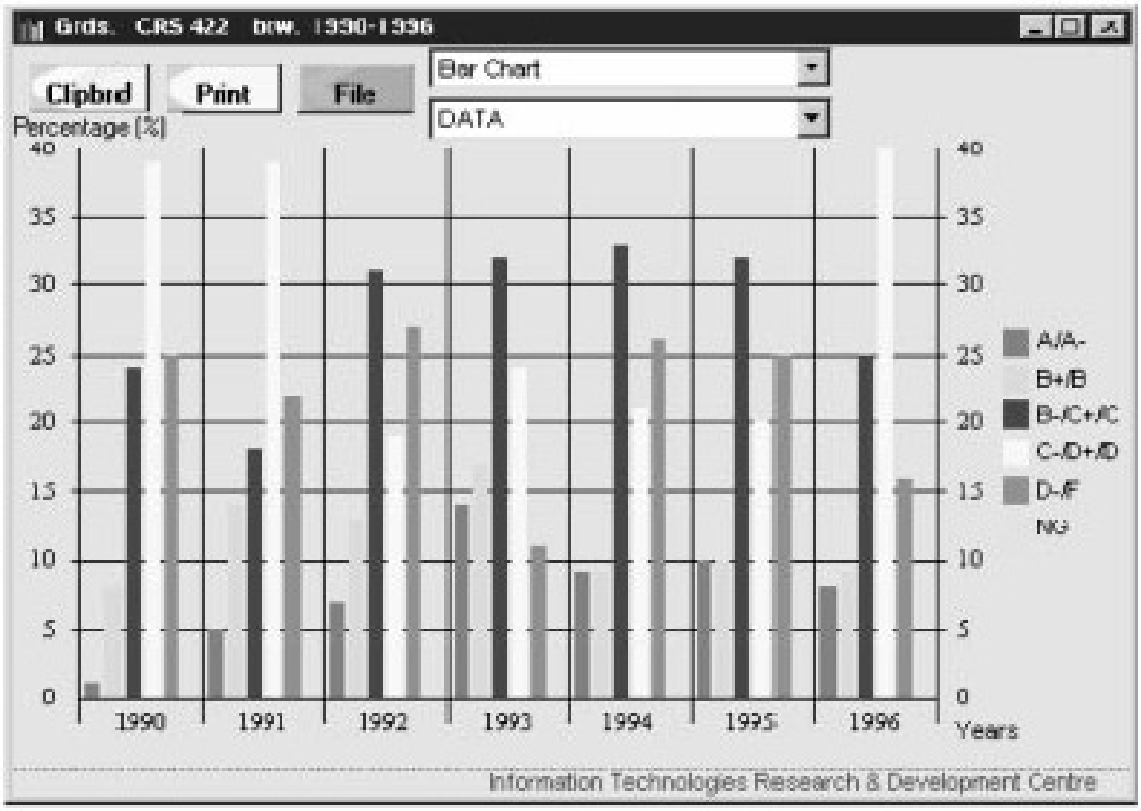


Figure 68 Grade distribution of a given course in any period of time (59)

Data tables are used in PADDS to provide a detailed overview of the data behind the visualisations and the effectiveness of their use helped emphasise the importance they would play in this application. Figure 69 shows how the effective use of data tables to convey as much information to the user about the student as possible.

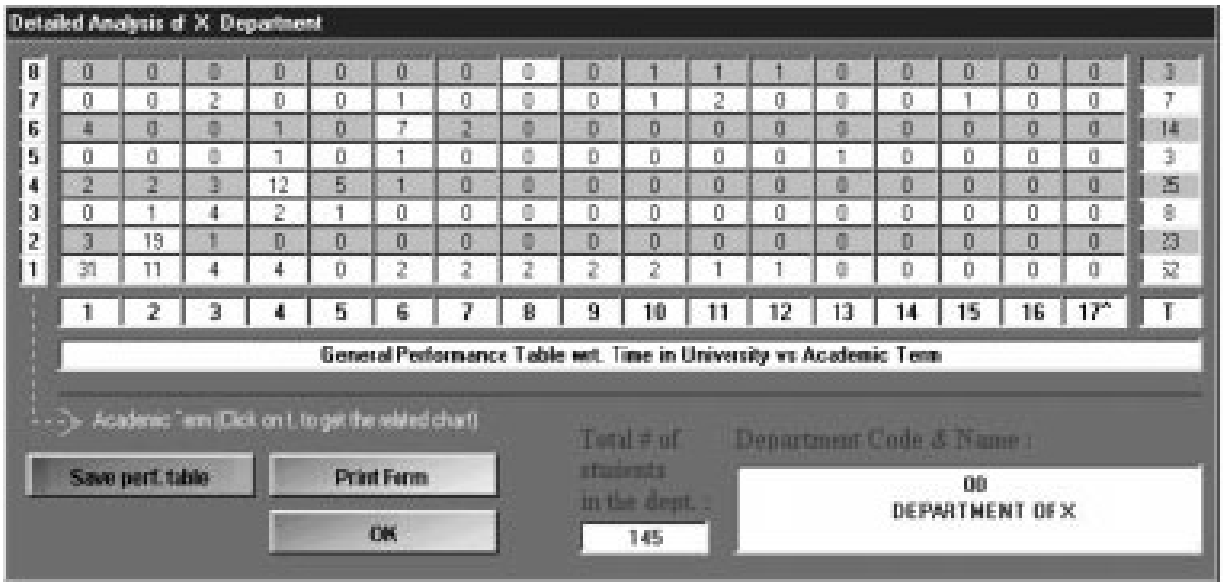


Figure 69 General performance table (59)

The layout isn't great and it's not very visually appealing to the user however the concept of displaying visualisation data in tables along with plots as this enables users to easily analyse the results in further detail. This can be useful if for example there is a corrupted entry in the database that is throwing off the graph output. A `null` value may cause unexplained operation in a graph but could be difficult to pinpoint however could be easily spotted in a sort able data table.

Providing tables also allows the users to easily export the output of the analysis by simply selecting the table value text, copying it to their clipboard and pasting it into their editor of choice. This means that data can be easily taken offline and used in programs such as Microsoft Excel to draw bespoke graphs that are not currently available in the application. This means the application will continue to be useful as a data source even if some better graphing techniques are discovered in future.

Conclusion:

Researching currently available student analysis systems was an extremely worthwhile exercise as it:

- Provided inspiration for new graphs and ways of organising data,
- Increased confidence in methods already chosen for the project that were found mirrored in both systems,
- Increased confidence in the general path the research for this project was taking due to the vastly superior visualisation and user interface tools available.

Both systems propose good ideas for graphing student academic data and one thing they share in common is the poor visualisation technology. They are older systems which were at the time state of the art however the web and web technologies have moved on a lot since then. They both analyse similar aspects of student performance

3.3.2 Initiatives to improve performance at universities

As part of the research into academic performance analysis a lot of useful and informative papers were found describing the analysis of student results after the university had made a change to the system or were trialling a new way of teaching. One of these papers provided some excellent ideas that could be used to help improve the performance of students in DCU, in particular engineering students, and is included for this reason. The study examines the performance of engineering students at a Chilean university and the recent increase in the grades of freshman students after the introduction of remedial teaching programs to help them get up to speed. (61)

This paper is an excellent source of information on student drop-out analysis and also proved to be an inspiration for a scheme DCU could undertake to give additional help to engineering students who may be struggling. The paper gives a detailed analysis of the outcomes of a remedial program for engineers and shows that there was a marked improvement in the performance of the students attending the program. It would be an excellent idea for DCU to run a similar scheme to aid their engineering students.

Remedial programs were run in mathematics, physics, engineering fundamentals and study techniques amongst others and the results proved quite positive with the student retention levels significantly improving. First year can be a harrowing time for new university students as there can be a lot of adjusting to do especially if they are coming directly from secondary school. A lot of students do poorly in their first year and a lot do too poorly to progress further and need to drop out.

In a lot of cases students simply don't know how to prepare themselves properly and that can be the cause of their demise. If they have to drop out this can also make them become disillusioned with the education system and they may not return to re-attempt another course. This study shows that by paying more attention to the basic fundamentals and ensuring all student know how to teach themselves and prepare for university exams can significantly improve student retention which benefits the university overall as they are more students continuing to pay registration fees.

A key focus of engineering modules is programming and it is something that is required in most assignments. Engineering courses can be extremely technical in their content and a student may be required to submit assignments detailing how complex systems work and provide a simple program to demonstrate their operation. This puts those with weaker programming skills at a significant disadvantage and inhibits their learning of the actual topic the assignment is on as they need to spend most of their time grappling with writing the program. This also may unfairly lower their rank in the class as they may be the best student but a better programmer with little knowledge of the module may get a higher grade.

If a remedial group was established in this area in DCU to help students gain the skills needed to perform their assignments it could easily improve the overall course grades of students who may have been struggling otherwise. If the results of this study are anything to go by it would be definitely worth giving it a try even with an additional charge placed on the students to cover the cost of the program. This may not even be needed though as if the program is a real success the extra registration fees gained from the decrease in student drop-outs would probably be more than enough to cover it.

3.3.3 New web technologies

This project is using various new visualisation technologies so it was important to research into new and better ways of developing the application as a whole and this involved a study of the new standards in user interaction, mainly AJAX techniques and Rich Internet Applications (RIA's). The web and how it's presented has developed extremely quickly in recent years and there are now far more powerful tools available to developers to enrich the quality of presentation of their application.

Rich Internet Applications (RIA):

RIA's are essentially web based applications that offer the same functionality as a desktop application. An excellent example of this would be a comparison between the web based Google Spreadsheets and the desktop based Microsoft Excel. User's have gotten used to this new improved web experience and have grown to expect it however it is quite a difficult task to produce them from a development perspective. RIA's were initially mostly usually

Adobe Flex based applications utilizing Adobe Flash to display visuals to the user and enable the rendering of graphical visualisations of an extremely high quality.

Adobe Flex is a framework that allows application developers to utilize the power of Flash in their applications through the use of various tools such as plug in widgets. It can produce stunning graphic quality and a great aesthetic value for web applications however it relies on flash so is incompatible with newer mobile devices such as the Apple i-Phone and i-Pad. The accessibility of these applications also suffers as a result and there is no hope of somebody using a screen reader trying to use the application as its output is a flash file that is rendered in the browser.

Learning to use flex would be worthwhile if graphic quality is all that is needed in an application however it is quite complicated and takes a lot of adjusting when compared to standard programming languages. It requires the use of Adobe Flex Builder which is built on the Eclipse IDE however unlike Eclipse it is not free and will cost around $249 for the standard edition. This is acceptable for a large organisation if a quick fix is sought however there are other open source alternatives that can offer the same functionality if not as graphically stunning. A good overview of Flex and HTML5 and a comparison of the two can be found in an interesting presentation by Pamela Fox (62).

AJAX:

The term AJAX doesn't refer to a specific technology in itself but to the use of several web technologies together to provide for a more interactive and user friendly experience. AJAX means asynchronous JavaScript and XML and is a serious rival to RIA's when it comes to web page dynamics. AJAX enables for a more interactive user experience and allows the developer to build intelligent web applications that don't require moving to different pages every time a user has an interaction or wants to see more content.

In the past it had been almost as difficult to create an AJAX based application as it was an RIA however with the advent of HTML5 there has been a strong focus on JavaScript programming and a number of extremely useful UI libraries have emerged. AJAX technologies are open source technologies and are therefore collaborated on more by various developers around the world. They also conform to W3C web standards if coded

properly and can be parsed by a screen reader such as Lynx or Google Robot so they provide accessibility and usability in an application while keeping to web standards.

jQuery:

jQuery (63) is a JavaScript library that is open-source and enables the developer to make use of hundreds of functions and plug-ins and was designed to change the way developers write JavaScript. The project began in 2005 and has been a huge success since then with many developers adding to the ever increasing plug-in library. It simplifies the core features JavaScript is commonly used for such as event handling, AJAX interactions and animations.

jQuery is a set of open source JavaScript libraries that make it easier for developers to write the code they need for their applications. Many developers have struggled with the same problems in their application and jQuery provides an easy way to reference a set of predefined functions and UI tools. There is also thousands of jQuery plug-ins available for download to serve any purpose needed with new ones being added daily. To use jQuery all that is required is the download of the source JavaScript file and inclusion in your HTML page. This then enables the use of the core functionality and additional files can be included for different plug-ins as needed. All that is needed to use jQuery is this single JavaScript file and once referenced in your web page you can begin to utilize it as an engine and write your application rapidly.

jQuery makes developing AJAX applications far simpler and their tag line is even "*jQuery, the write less, do more Java Script Library*"[www.jquery.org] which sums up what the project is all about. The jQuery UI is even better and is a user-interface tools library built upon the jQuery framework. It is easy to learn and understand and enables the use of various prewritten tools such as user validation, tabs, accordion menus and dialogs. It is perfect to use for widgets or even for the whole application.
A lot of issues encountered when developing applications have been common to many web developers and this library allows developers to utilise pre-existing code to provide the functionality they need without having to start from scratch themselves.

Another key feature of jQuery is that it is cross browser compatible and supports:

- Mozilla Firefox versions 2.0+
- Internet Explorer versions 6+
- Safari versions 3+
- Opera versions 9+
- Google Chrome versions 1+

It is tested and supported across all browsers by the jQuery code team themselves and eliminates a lot of headaches for web developers building their own JavaScript application from the bottom up. It enables developers to use features for example menus without the need of cross browser testing. It still should be performed however there can be a lot of issues encountered trying to ensure cross browser compatibility especially with internet explorer.

Internet explorer is probably the most commonly used browser, due to the worldwide presence of Microsoft, and the fact that it is preinstalled on Windows desktop machines, however is also one of the worst performing ones. When new versions are released backwards compatibility back to Internet Explorer version 6 needs to be ensured so any issues in those versions are likely to be still present today. Like it or not Internet Explorer is probably the most popular browser so when developing web applications its support needs to be a priority. With jQuery eliminating these issues it further strengthens its appeal as a quick way to get a visually appealing and dynamic application up and running with minimum time spent on such trivial issues.

jQuery is becoming increasingly popular and it already boasts some big name companies among its users. Google use it in their code search (http://code.google.com/) and other big names such as Amazon, IBM, Microsoft, Twitter and Dell all make extensive use of it on their web sites. jQuery itself is the core library in use but has also led to the development of another library specifically for user interface known as the jQuery UI.

jQuery User-Interface (UI):

The jQuery user-interface (UI) library is used extensively in the front end design of this project and provides developers with a set of excellent tools to aiding them in application design. A custom built package can be downloaded to suit your exact needs and can be

themed using one of the pre-defined themes or a bespoke theme can be developed using the ThemeRoller which allows users to develop a custom theme and preview before downloading (64).

Some of the best built in features are the accordion, tabs and dialog all of which are used extensively in this project. These features are excellent for helping build a dynamic web page using the DOM and div tags effectively to minimise the amount of times a user needs to change page. In historic web pages links would more often than not bring you to a new page which would require loading. Using the jQuery UI developers can easily organise their page into clear sections using `<div>` tag's and can offer a lot of functionality without requiring the user to re-load the page.

Along with these layout features there is a vast amount of other AJAX animation features available to really help bring the application to life. Complex interactions such as dragging, dropping and resizing elements can be performed along with various animations such as colour animation, fading in and out of div elements, hiding elements and many more. With a bit of work initially soon a highly interactive, cross browser compatible AJAX web application can be developed and this is why it was chosen for this project.

The jQuery UI is free to download and is a JavaScript source file accompanied by CSS files to control the style of the UI tools. There is even the ThemeRoller application on their website that allows you to interactively try new colours and view them instantly then downloaded your design once completed. This makes it an extremely attractive means of developing a high level application extremely quickly once you have gotten to grips with the code syntax. It also frees developers form the frustration caused by cross browser incompatibility as all tools in the UI are tested as cross browser compatible.

The UI is not just easy to use and tailor-make to your design standards and great functionality is offered particularly in the form of the tabs and accordion menus which make organising the web page a lot easier. The entire page content can fit in the browser window and the user is able to change the content quickly and without reloading the page or moving to another page. This greatly enhances the user experience and reduces frustration at lading times especially when working with a poor internet connection. Once the page is

downloaded and rendered client side all interactions are performed locally and there is no need to reload the page to achieve different effects.

Another extremely useful tool is the dialog which is basically a pop up box dialog that displays the contents of a element once an action is performed, usually a button being pressed. Te dialog is far more powerful than opening a pop-up window as there is an option to set it as a modal dialog which means that the user cannot perform any other action until the event required is performed or the dialog is closed. This is a great way of replicating the RIA style of controlling the user experience and is extremely useful for search and login forms. Dialogs are also a great way of organising content on the page as it allows a large amount of content to be available at ease and upon request without cluttering the page.

Display tools aren't the only benefit of the jQuery UI and as it is built on the core jQuery library you can utilise any of the other plug-ins currently available. This makes it far easier to approach new problems that occur when developing an application such as how to perform user validation or display data un a graphically pleasing table as the first step can be to check the plug-ins to see if anything is available that could help solve your problem or even give you the knowledge needed to get to the solution more quickly. This allows developers to focus on the business logic of the application rather than wasting time trying to re-invent the wheel and do something that has already been done thousands of times and in a hundred better ways than what is likely to be achieved when attempted again from scratch.

The jQuery UI is a great tool and is one of the best new technologies found during the research for this project. Google actually even use some components themselves for some of the visualisation components and host the library in the Google Libraries API which is a content-distribution network for the latest JavaScript libraries which means it can be referenced any time its needed without downloading locally which can be useful as the most up to date version of the library will always be used[(65)

An excellent resource for getting started with jQuery is to try out the demo pages at (66) or to use the excellent guide "jQuery Novice to Ninja" (67) which provides an abundance of useful tips when getting started with jQuery. Time spent getting to understand how it works

will reap great benefits in future and development time of other projects will be significantly reduced.

3.3 Conclusion

The research for this project took an in-depth look into various visualisation technologies currently available, advanced techniques for student academic data analysis and new web technologies for developing user friendly, rich graphic interfaces. The research into visualisation technologies proved to be very worthwhile as it discovered a number of interesting ways of viewing data including High Charts and Heat Colour tables. This added to the value of visualisations in the final application as the best available solutions were trialled and tested in full and an informed decision on the best visualisations to be used could be made.

Research into student academic performance as a topic itself was at the forefront of the research as the interpretations that can be drawn from the data being visualised is a lot more important than the ergonomics of the visualisation. Various aspects of academic analysis were investigated with a key focus on statistical analysis techniques used to derive patterns from student academic information. Many interesting case studies were analysed which gave an in-depth look into how various institutions analysis academic data and highlighted its importance at a lot of major universities.

Statistical analysis techniques and in particular linear regression analysis was used on many occasions and this inspired its use in this project. This lead to further analysis on prediction of student results which is something that would be worth investigating further. A lot of case studies in this field examined social and behavioural aspects attributing to academic performance which could not be used in this project as the data was not available. If a database of this information was collected at DCU it could be very useful in making key decisions in planning and anticipating student results and would greatly benefit the university and its students.

Research into new web technologies was also a useful aspect of the research performed and new techniques were discovered to enable a better user experience in the final application. The computing world has changed recently and with the advent of cloud computing light

weight, usually web based, applications are taking over. To compete in this market and conform to standard is difficult but through the use of jQuery and the jQuery UI it is possible to create a flash style rich internet application that conforms to web consortium standards, something which seemed ludicrous when work initially began on this project.

With the vast quantity of research covered and the quality discovered in some of it the application design phase was entered with an array of ideas to use and this made it a lot easier to think about how to work with the data available. A lot of work has been done into discovering the key factors that help show and predict student performance and by performing such thorough research in this area the universal appeal and the relevance of this project is increased. The design of the application is discussed in further detail in Chapter 4 - Application design overleaf.

Chapter 4 - Application design

An important aspect of any application is its design as it is important to have a clear idea of what is to be created and how this will be achieved. A clear and structured design makes it a lot easier to build a project in the long run and can significantly cut down on development time. It is extremely important to plan the application thoroughly as this will enable making educated decisions on what to do and how to do it. It was clear at the start of this project that a clear plan of action needed to be made and stuck to religiously as there was a lot of material to cover and it would be very easy to get side tracked.

After analysis of the various visualisation technologies encountered during the course of the research, High Charts were chosen as the main graphing tool. There were multiple reasons for this including the fact that they will work on or offline, require no browser plug-in, are cross-browser compatible and perform well rendering graphs with minimum loading time. Other charts such as Visifire and Google Visualisations are also used where they offer significant benefits such as the Google Motion Chart.

A decision was made to develop the user interface using a mix of technologies to enhance the user experience as much as possible and provide RIA style user interaction. jQuery, HTML and CSS are used to develop the Web Application front end. Java servlets were chosen for the back end business logic as this will allow for further extension to the application in the future. Initially PHP was a contender to use but Java is far more secure and extensible so it was chosen instead.

4.1 Application Structure

After some initial research into the Google Visualisation API a decision was made to develop a J2EE web application as it would ensure the application will be platform independent and portable. This means it could easily be integrated to pre-existing systems in the university with minimal effort and can be easily extended upon by further researchers. A decision was made here to choose J2EE over PHP even same kind of application could be written a lot faster using PHP. This is because J2EE applications are far better suited to

enterprise applications such as this one and by using it various java libraries can be used to enhance the functionality in future.

The initial design of the application was decided on and a three tiered web application structure was chosen. This structure is illustrated in Figure 70. The top level tier is the data tier which consists of the student information database. Below this is the application tier which is an Apache Tomcat web server. This is where all business logic is performed and the servlets connect to the database to gather the data and format it on here.

The lowest level tier is the client tier or the client's web browser. The client makes a request from the client tier to the application tier to display visualisations at which point the servlets at the application tier obtain the necessary information from the database and return the completed visualisations to the client. By separating the application in this way it provides a clear separation between the applications presentation and back end logic which is extremely important both for development and security reasons.

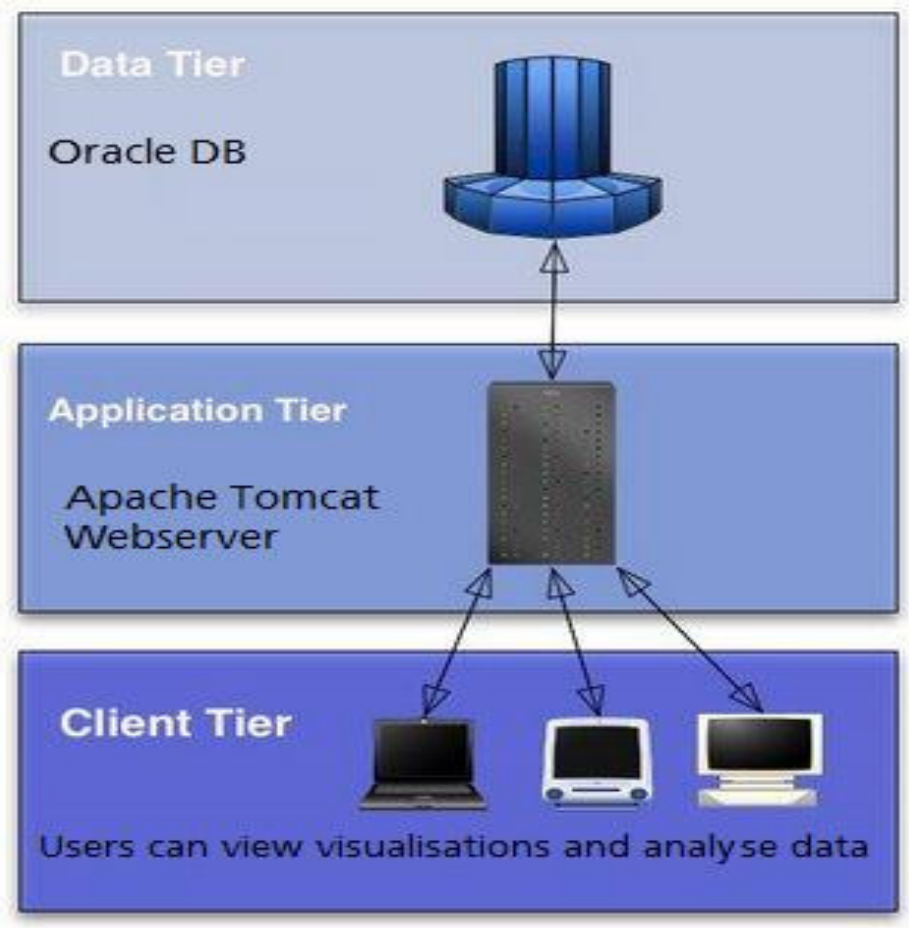


Figure 70 Application structure overview

Defining a clear structure for the application makes it easier for another developer to re-use my code and also helps a lot when de-bugging as issues can be tracked down more quickly by eliminating variables. If a visualisation is not being displayed you can check to see if it's a problem with the database or the servlets or simply the output to be displayed in the browser. This makes it far quicker to pinpoint bugs and a lot quicker to resolve them.

This type of structure is also important from a security perspective as in a web based application anything that is sent to the browser can be viewed by the user so it is important to hide business logic that may make your application vulnerable to attack. Common forms of attack on web applications include SQL injection where hackers input SQL syntax directly into HTML input forms to try and attack applications and gain access to the database. Separating presentation and business logic reduces the risk of the hackers being able to get in by providing them with less clues about how the application works. To further enhance database security stored procedures should be used to run queries as opposed to direct SQL statements as validations can be added inside to prevent unwelcome user input as a failsafe to the validation performed on client and server side. Using stored procedures also hides the database structure so we are not exposing any database information at the application tier apart from the name of the stored procedure.

4.2 User Interface Structure

Several factors were considered when designing the user interface structure and a discussion of each area of research can be found in the following sub sections. The aim of the visualisation interface is to provide as detailed and informative analysis of students, modules and faculties in as usable a manner possible. Some other applications were researched in order to get an understanding of the best way of displaying multiple visualisations on a single interface including a telecommunications software tool called Insights Pro produced by an Irish software company called The Now Factory (68). Sample screenshots are shown in Figure 71 and Figure 72 and give a great idea of how best to convey multiple data sets on a single display.

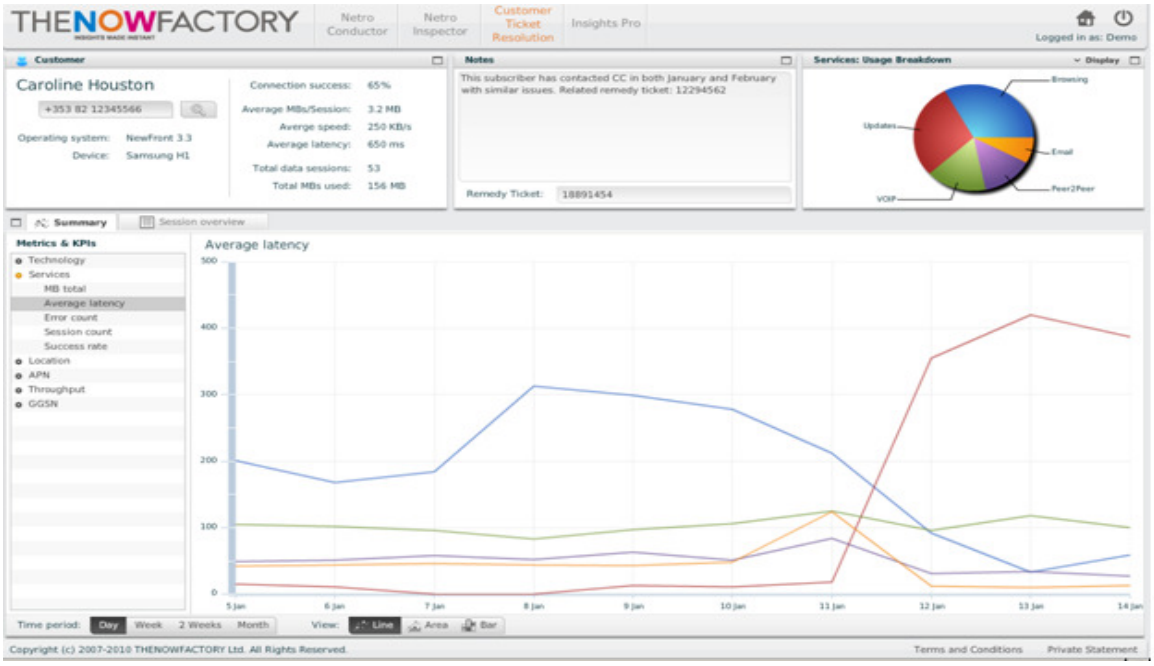


Figure 71 Insights Pro Screenshot 1

This application shows how a high level of interaction is possible once a clear layout is designed and the user can go where they want with minimal effort and was a good source of inspiration in the design of the application in this project.

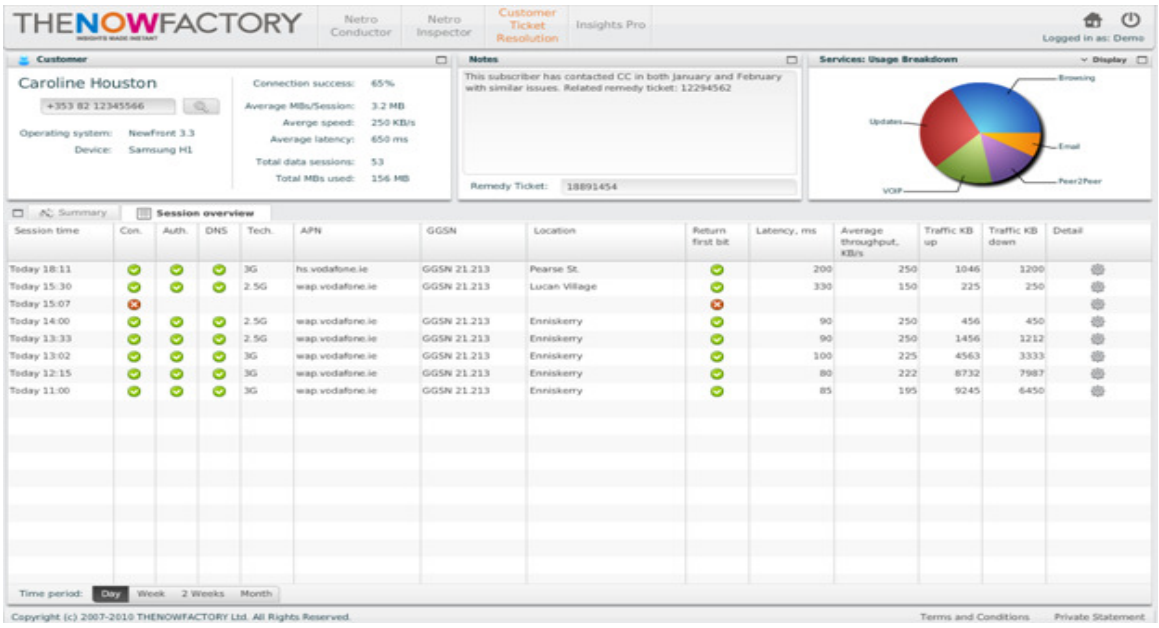


Figure 72 Insights Pro Screenshot 2

4.2.1 Accessibility and Usability:

One of the key aspects considered in the initial design was the usability of the application. An extremely useful point of reference in the design was Steve Krug's web usability bible "*Don't make me think*" (69). This is one of the best sources of information currently available on the topic of web usability and how to design a user friendly interface. A key message he tries to relay to the reader is that a web site should be as easy to understand as possible and that users should be able to figure out how to use it with little or no thinking required.

It helped with the UI layout design immensely and was a great insight into how user's think. It is essential to think like a basic user to ensure that anybody can use the system as far too many web applications and sites today are full of cluttered information that no-one reads. Due to information overload it can often be difficult to find what you are looking for even though it is right in front of you. Some important lessons were learned and some excellent ideas formulated on the basis of Krug's research and it's highly recommended reading for developing any kind of application not just a web based one.

This project focuses on analysing individual students, modules and course registration so the web application was divided into three areas to represent each area of analysis. After much research into the latest and greatest web techniques it was decided that jQuery and namely the jQuery UI.

The front end user interface is built using a mix of technologies mainly HTML, JavaScript and CSS for presentation and Java Server Pages (JSP) to present the content. The visualisations themselves are written in JavaScript and are generated by the back end server to be included in the `<head>` tag of the JSP files. Initially raw HTML files were used along with server side includes for organisation of file structure and interaction with servlets however problems were encountered further in the development stages when the graphs were using live database data. This is discussed in more detail in the application logic section 4.4.

This structure allows for separation between presentation, structure and data retrieval. A goal of the application was to make use of the best new technologies available and this is done extensively through the use of jQuery and the jQuery UI. HTML is used to define the

main structure and data, CSS is used for layout and style and jQuery is used to provide animation and AJAX effects.

4.2.2 Front-End UI:

The user interface for all analysis areas is designed to display a vast amount of information to the user in as organised and easy to access manner as possible. This is achieved through the use of a simple menu structure using a main navigation menu at the head of the page, a narrow jQuery accordion side menu to the left of the page and a jQuery tabbed interface as the main content area as shown in the screenshot below:

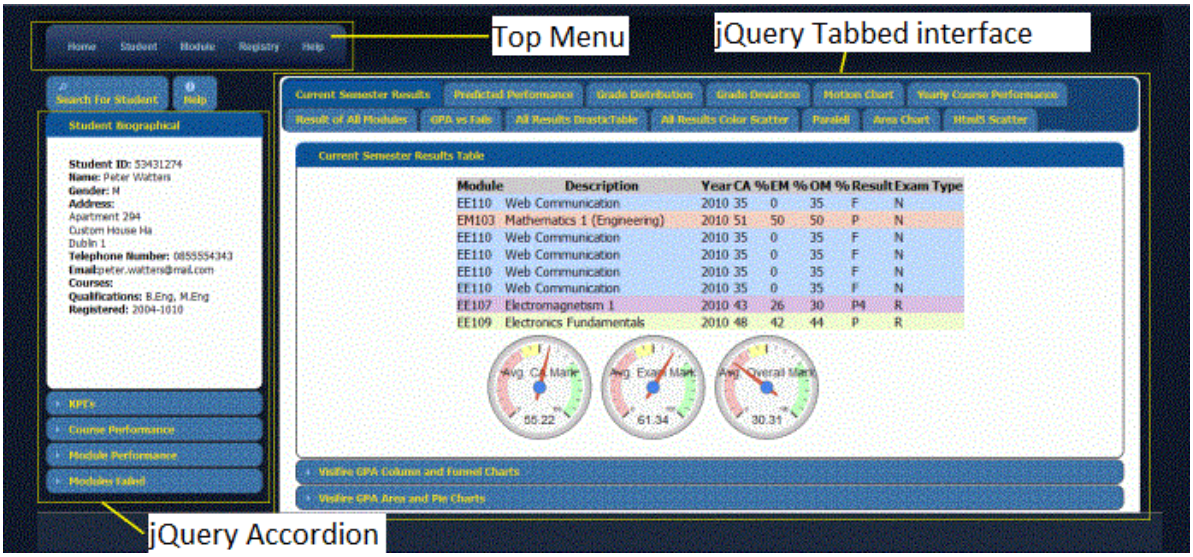


Figure 73 Front-End UI Design

The top menu used is a free to download jQuery plug-in by APYCOM (70) which provides a rich level of user interaction along with cross browser compatibility. It was one of many menu's that were tried and offered the best solution with its unobtrusiveness and ease of set-up as it uses plain HTML un-ordered list tags to define the structure. CSS styling is used to provide the style and colour and JavaScript is used to display the animation such as highlighting.

The menu is probably better described as almost cross browser compatible as operation is a little strange outside of Firefox and namely in Internet Explorer(IE) when it can be difficult to access a sub menu item at times as the animation is too sensitive to the movement of the mouse and this can be frustrating to users. This only occurs in IE and the effects of this problem are reduced by keeping the menu structure simple which meets the accessibility concept of the design anyway. It is not perfect by any means to not be cross browser

compatible but it was the menu with the most appealing design and the least compatibility issues of all those researched so a decision was made that it would be sufficient for use.

The side bar accordion menu to the left gives an overall indication of the performance of the metric being analysed be it student, module or course. The jQuery UI accordion is used to good effect here to provide access to as much content as would clog an ordinary static HTML page in an organised and aesthetically pleasing manner. The accordion menu is one of the best components provided in the jQuery UI and an additional benefit of it is that it can also be embedded within a jQuery tab to enable as many full page views as you could ever use in an easy to define and reliable manner.

The main of the page uses the tab interface to allow the user to select various individual visualisation pages analysing different metrics of performance. Everything in this page is rendered upon first download and any subsequent manipulation of data is performed solely on the client side which results in a more responsive application. This also reduces the load time moving between different pages to see different visualisations and provides AJAX style live interaction to the user.

4.2.3 Search Dialog:

The search dialog utilises the jQuery dialog module with its property set to modal. The basic setup of the dialog is to declare a new dialog and dialog button jQuery object in JavaScript and a div element to be displayed upon calling the dialog and a button to call it with HTML. By setting the property to modal the background of the screen is locked while the dialog is active which is more clearly shown by Figure 74 Search Dialog Design.

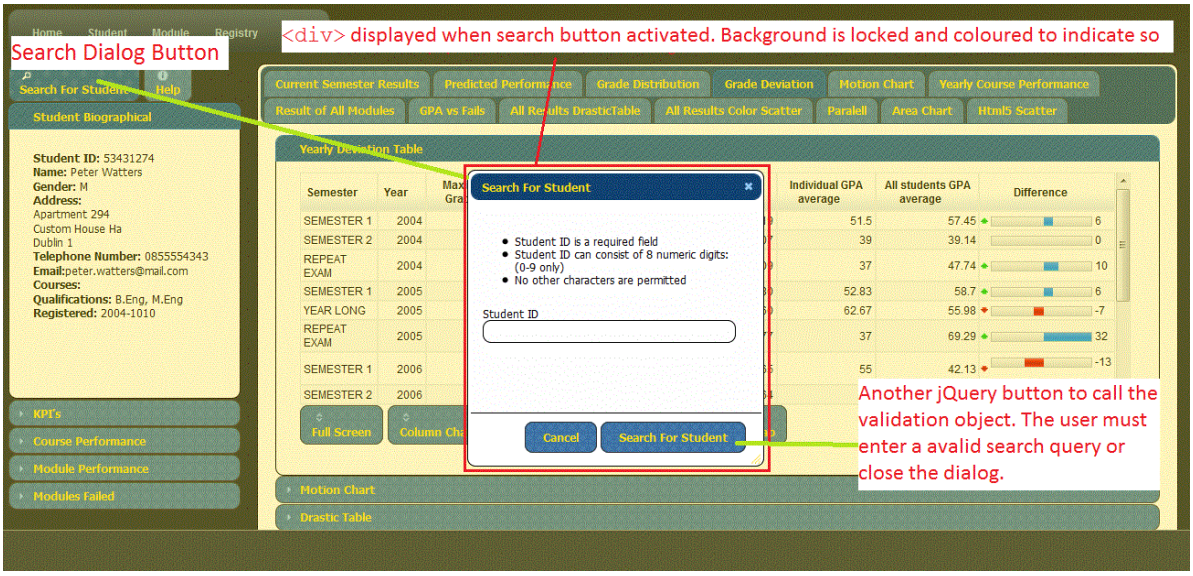


Figure 74 Search Dialog Design

This use of the dialog when the modal attribute is true and a form is incorporated enables developers to control the user experience and to offer similar functionality to flash based RIA's in a far more developer and user friendly manner. Various options can be set for the dialog such as calling it on initialisation of the page so the user has no choice but to deal with the dialog contents before continuing. This could be easily modified to provide user login functionality but as this project is intended to be a portable addition to pre existing analysis methods it is assumed such user validation is already in place.

4.2.4 Organisational Dialogs:

Dialogs are also used to organise the display and full screen visualisations can be called in the same way as the search dialog to give users ease of access to multiple visualisation types of the same data source with as little effort as possible. Tabs and accordions are useful for organising the layout into sections and categories but too many subsections and categories can clutter and confuse the layout so the use of pop out dialogs really enhances the user experience. An example of this is shown in Figure 75 and Figure 76.

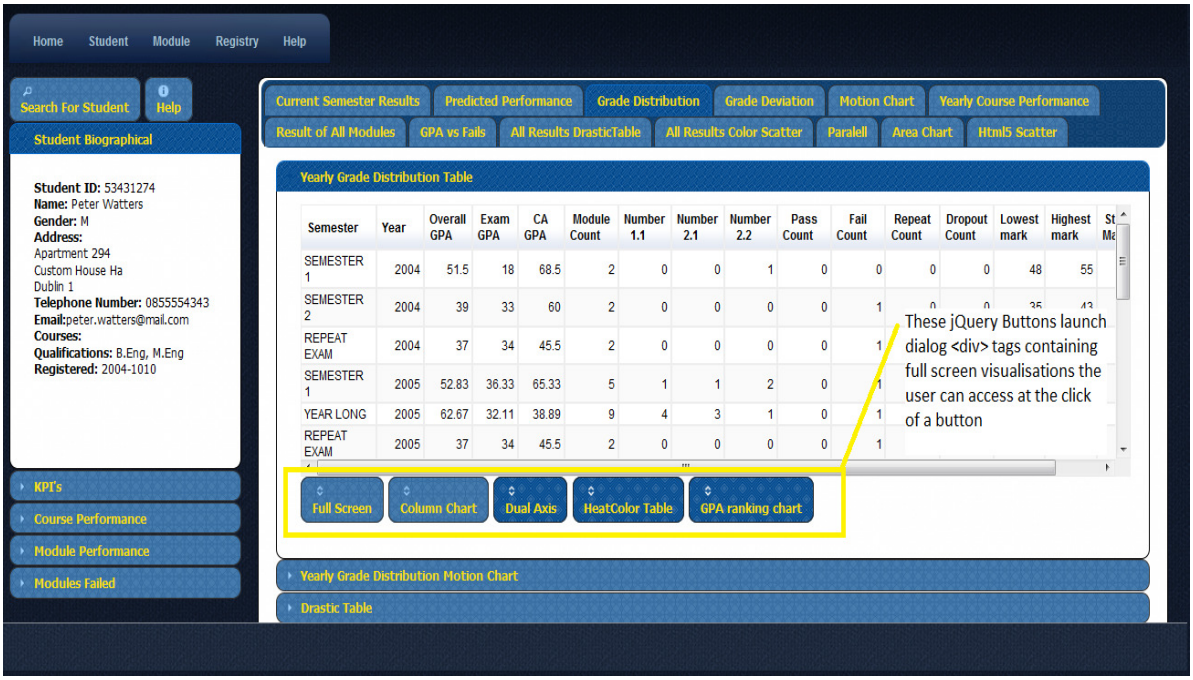
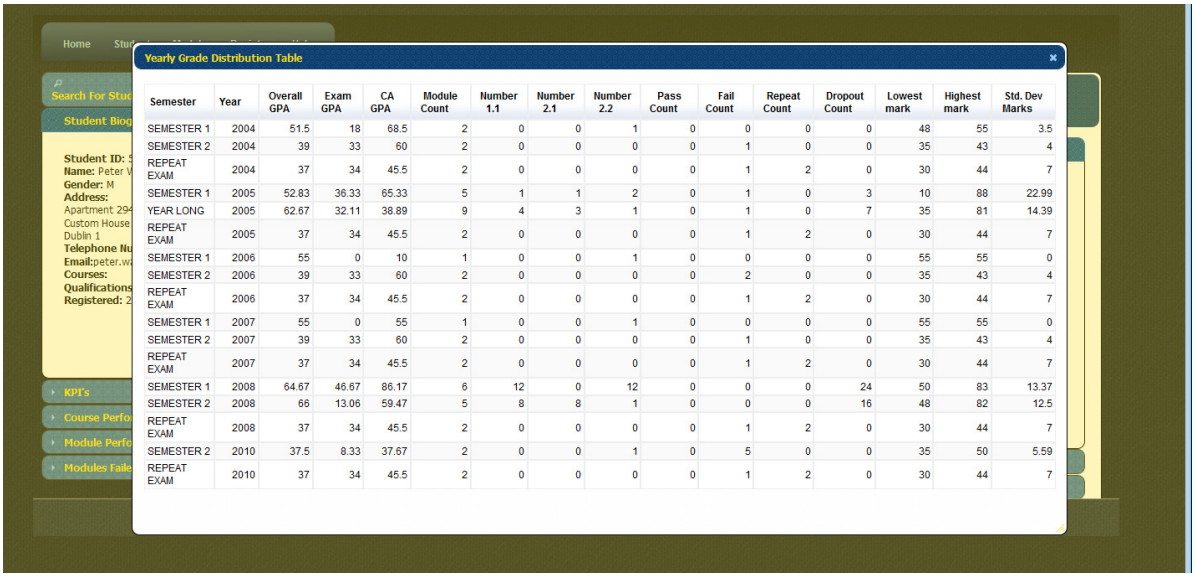


Figure 75 Student Analysis Yearly Grade deviation dialog button menu

In the grade distribution analysis tab an accordion menu is used to hold a Google Data Table, Motion Chart and Drastic Table which plot grade deviation on a per semester basis. Many other useful visualisations can be used to bring the data table to life but too many accordion items mean that the content size of the accordion needs to decrease to enable the additional menu buttons to be seen and therefore clickable. The use of dialog buttons makes it easy for the user to open full screen views of the data in question; in the case of Figure 76 they have chosen to view a full screen version of the data table.



Semester	Year	Overall GPA	Exam GPA	CA GPA	Module Count	Number 1.1	Number 2.1	Number 2.2	Pass Count	Fail Count	Repeat Count	Dropout Count	Lowest mark	Highest mark	Std. Dev Marks
SEMESTER 1	2004	51.5	18	68.5	2	0	0	1	0	0	0	0	48	55	3.5
SEMESTER 2	2004	39	33	60	2	0	0	0	0	1	0	0	35	43	4
REPEAT EXAM	2004	37	34	45.5	2	0	0	0	0	1	2	0	30	44	7
SEMESTER 1	2005	52.83	36.33	65.33	5	1	1	2	0	1	0	3	10	88	22.99
YEAR LONG	2005	62.67	32.11	38.89	9	4	3	1	0	1	0	7	35	81	14.39
REPEAT EXAM	2005	37	34	45.5	2	0	0	0	0	1	2	0	30	44	7
SEMESTER 1	2006	55	0	10	1	0	0	1	0	0	0	0	55	55	0
SEMESTER 2	2006	39	33	60	2	0	0	0	0	2	0	0	35	43	4
REPEAT EXAM	2006	37	34	45.5	2	0	0	0	0	1	2	0	30	44	7
SEMESTER 1	2007	55	0	55	1	0	0	1	0	0	0	0	55	55	0
SEMESTER 2	2007	39	33	60	2	0	0	0	0	1	0	0	35	43	4
REPEAT EXAM	2007	37	34	45.5	2	0	0	0	0	1	2	0	30	44	7
SEMESTER 1	2008	64.67	46.67	86.17	6	12	0	12	0	0	0	24	50	83	13.37
SEMESTER 2	2008	66	13.06	59.47	5	8	8	1	0	0	0	16	48	82	12.5
REPEAT EXAM	2008	37	34	45.5	2	0	0	0	0	1	2	0	30	44	7
SEMESTER 2	2010	37.5	8.33	37.67	2	0	0	1	0	5	0	0	35	50	5.59
REPEAT EXAM	2010	37	34	45.5	2	0	0	0	0	1	2	0	30	44	7

Figure 76 Dialog Full Screen Yearly Grade Distribution Table

The raw data displayed in this table can be easily exported simply by selecting the text and copying to the clipboard then pasting to a text editor to remove unwanted formatting and the data can then again be pasted into to other applications such as Microsoft Excel and used for other reasons. This would allow users to use the data retrieved by the application to plot their own visualisations that aren't currently available as part of the application.

The use of additional button dialogs was a late addition to the application design and proved to be one of the most useful yet as it allows for an almost unlimited amount of visualisations to be neatly grouped and hidden away in an unobtrusive manner until requested by the user. There are some flaws spotted in that sometimes High Charts do not render correctly after the dialog is initially called closed and re-opened.

Another problem is that the Motion Chart or Drastic map or any other mainly flash based visualisations lock the user in a modal dialog once opened which cannot be exited without refreshing the page. That is why these items are included in the accordion sub menus instead. When the user attempts to close the dialog using the close button it tries to close it can only manage to reload the visualisation.

4.3 Visualisations used

There are plenty of different types of visualisation types available and an important part of the design process was deciding what ones would best convey the data available for representation. Various charts and graphs were considered and it was important to ensure that the visualisations were meaningful and could be used to aid student performance analysis. A visualisation is merely a way of representing data and the design perspective needs to be targeted towards inferring predictions using the graphs rather than merely displaying data in a visually appealing but useless manner. A discussion of the core visualisations chosen for use in the application is given in the following sections. Visualisations are categorised into charts and graphs and data tables for organisational purposes.

4.3.1 Charts and Graphs

4.3.1.1 Google motion chart:

The Google Visualisation API Motion Chart (3) is the best animated web visualisation tool currently available so it was vital to include it in this project. It allows the user to interactively view the changes in values of metrics over a times series using scatter bubble plots, moving columns or an overall line chart. It is included each analysis section in the application as much as possible as it can bring any data source to life. It allows for a high level of interactivity and can be populated with a vast amount of data so it makes sense to use it to display each of the metrics used in a single stand alone chart.

An example of one of its uses in the student analysis section is illustrated in Figure 77. Here one chart is used to plot the distribution of student grade and count of achievements e.g. first class honours throughout their entire history at the university and endless additional parameters can be added in for more data comparisons. It is an extremely powerful tool and trail lines can be turned on to follow a particular metric overtime. Data is also classified into profiles of the individual student, all students, all students the same gender as the student and all students taking the same modules as the student.

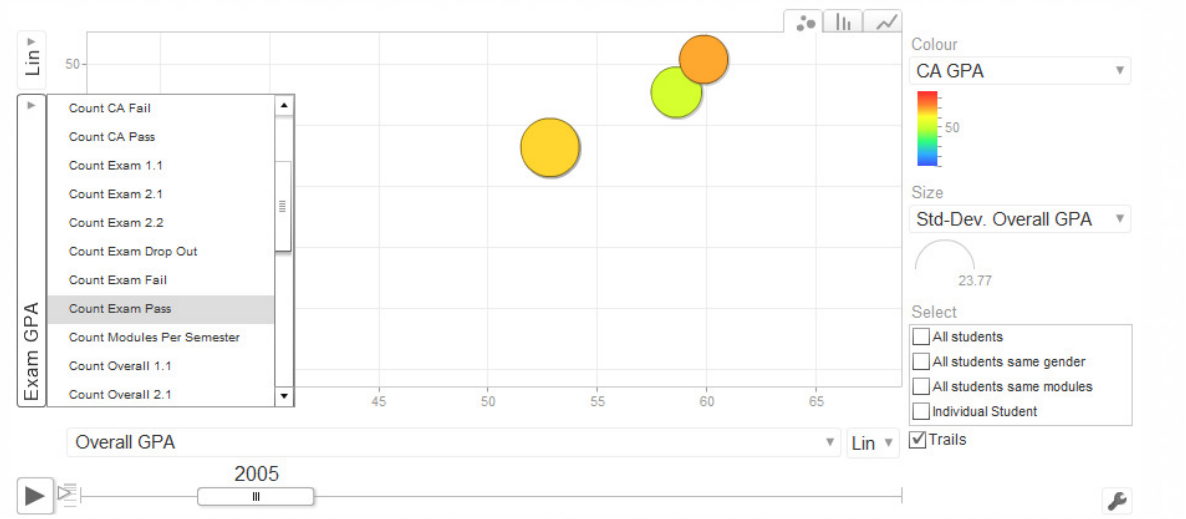


Figure 77 Student Analysis Motion Chart Bubble Chart

Student profiles can be highlighted and filtered on and off if one needs to have more attention than the others and the scale of representation can even be switched between linear and logarithmic scales. The level of data manipulation performed purely on the client side is stunning and motion charts offers numerous different ways of analysing the same data. The colouring of items can also be filtered according to any input data parameter required as can the bubble sizes in the bubble chart.

With the simple switch of a tab an interactive moving column chart can be viewed to focus in more closely on individual aspects. This tab is illustrated in Figure 78 Student Analysis Motion Chart Moving Columns. The Y-axis parameter defines the height of the column and the width is adjusted according to the value of the X-axis parameter at that point in time.

Figure 78 Student Analysis Motion Chart Moving Columns

The final tab of the motion chart is a line chart showing the metrics across a fixed time axis to illustrate the peaks and dips over time and an example of this is shown in Figure 79 Student Analysis Motion Chart Line.

Figure 79 Student Analysis Motion Chart Line

The benefits of the motion chart are clear to see and it is in fact a data visualisation library in itself pretty much. Any type of numerical data can be added to the JSON input series to plot any number of different metrics against each other, all of which can interact with the user on the client side once the chart is initially loaded. An initial objective of this project was to allow users to plot their own visualisations but the motion chart allows them to do this in a far better manner than would have been achievable given the time constraints of this project and is a better substitute to a form based graphing system as all data to be formatted by the user is already downloaded to the client in the motion chart which makes the interactivity a lot faster and smoother. The same performance and style wouldn't be possible using HTML input forms without a lot of work and use of AJAX data requests.

4.3.1.2 Drastic map:

Drastic Tree map (71) by Drastic Data is a Flash based tree map visualisation that provide an open source version of their software that can be run using the Google Visualisation API. It allows for grouping of variables using colour to separate variables into groups and displays their size according to its percentage value compared to that of the other variables in its own group. It also allows for filtering of up to three metrics using a side menu and was chosen for use in this project as it's a quite useful tool when analysing differences between various metrics of student performance. Figure 80 Student Overview Drastic Tree Map illustrates its use in the application in the student analysis section.

Figure 80 Student Overview Drastic Tree Map

The result of all of the students modules are grouped according to the year taken and colour coded in this manner. Individual modules are sized inside of these groups according to the weight of percentage the mark achieved by the student compares to the other modules. The colour coded years are also re-sized in this manner so the years when the students overall GPA average for all modules is larger will display in a larger size than years when the overall module GPA was lower. The various module colour blocks display the student grade percentage of overall grade for that year and the grade achieved when hovered over and further interaction is offered by the menu at to the top right of Figure 80.

This side menu enables users to switch between up to three metrics of analysis and is an excellent quick way of visualising the differences between Overall GPA, Exam GPA and CA GPA of students throughout the history of their attendance at the university. Drastic Tree Maps were chosen for use in the final application as they are unique in what they offer n terms of graphical representation and the closest alternative currently available is a much poorer version created by Google themselves which offers much poorer quality and is shown in Figure 81.

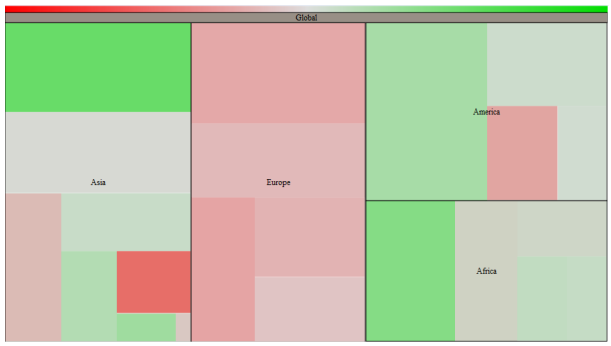


Figure 81 - Google Tree Map (72)

Drastic Tree Map is free for use but requires the download of `DrasticTreemapGApi.js` from the Drastic Data website in order to use it in your application. It also requires the inclusion of a flash object file kept locally on the web server to render the chart in the browse. It is a good choice for use in this application as it offers the look and feel of a RIA widget but can be populated using pure JSON and avoiding overcomplicating things by getting involved with Flex, Flash and Actionscript. The only real drawback at the moment is its currently limited capability of only allowing a maximum number of three data filtering columns and the fact that it needs a client side flash plug-in in the user's browser to display.

4.3.1.3 Regression Scatter plots:

Regression scatter plots were an important part of this application as much of the research covered in the area of academic performance analysis used some kind of linear regression analysis. Most visualisation packages offer some form of scatter chart with a line incorporated and it was difficult to make a decision on which technology would be best to work with. High Charts is chosen due to its extremely easy data population syntax and unlimited graph upon graph functionality meant that any number of scatter points for different metrics and their corresponding regression lines can be displayed in one graph at the same time. High Charts also implement simple but effective and extremely useful user interactions to allow users to filter data series on and off with the graph axes re-adjusting accordingly if not set to a fixed height. The following screenshot shows the use of High Charts regression scatter plots in the student analysis section of the application

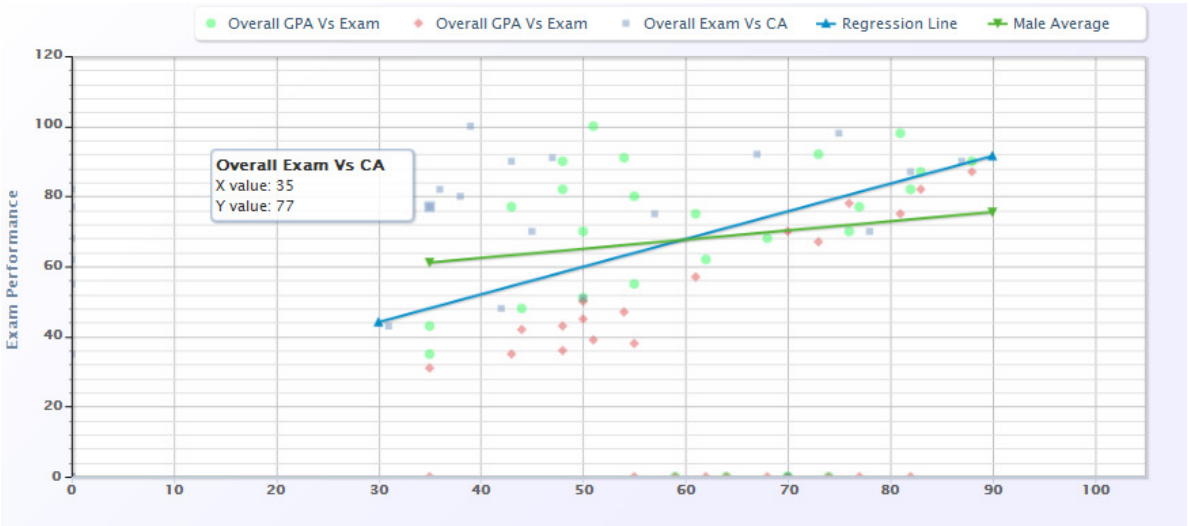


Figure 82 Student Overview Linear Regression Analysis

Figure 82 shows a the regression analysis of a single students grade however the power of HighCharts allows multiple additional data series to be plotted on the same canvas using colour coding to distinguish between them. This allows for far more complex multiple regression analysis to be plotted and the performance of one metric can easily be compared to that of another with slightly different circumstances. This is best explained by Figure 83 where regression analysis of continuous assessment grade versus overall mark is plotted for

- an individual student,
- other students the same year,
- other students all years,
- other students of the same gender as the student being analysed.

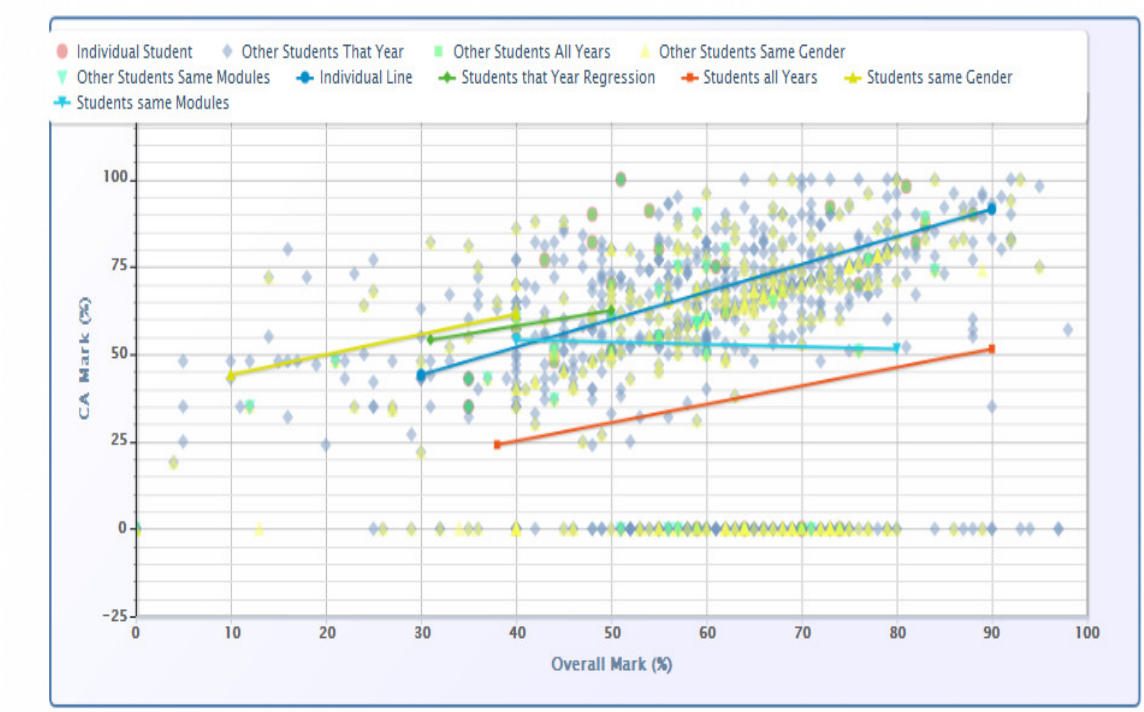


Figure 83 Colour Regression Scatter

4.3.1.4 Grade frequency distribution column charts:

A useful source of information on the topic of frequency distributions of is a book entitled "*Carrying out Investigations in Psychology - Methods and Statistics*" (73) which describes carrying out statistical analysis investigation of psychological data and provided many useful examples of how to look beyond data for patterns and infer meaning from them. It introduced the idea of a bar chart to show average grade per year which is in fact a frequency distribution of grades.

Frequency distribution of grades is plotted the application through the use if HighCharts column charts. HighCharts allow multiple chart types to be drawn together and this is the main reason why this was chosen here. This allows a frequency distribution to be plotted along with lines to highlight the trends as shown in Figure 84 where student yearly grade distribution is plotted using the maximum and minimum exam and continuous assessment grades as columns in the frequency distribution. Trend lines are also included for maximum and minimum overall grade and the standard deviation of overall marks.

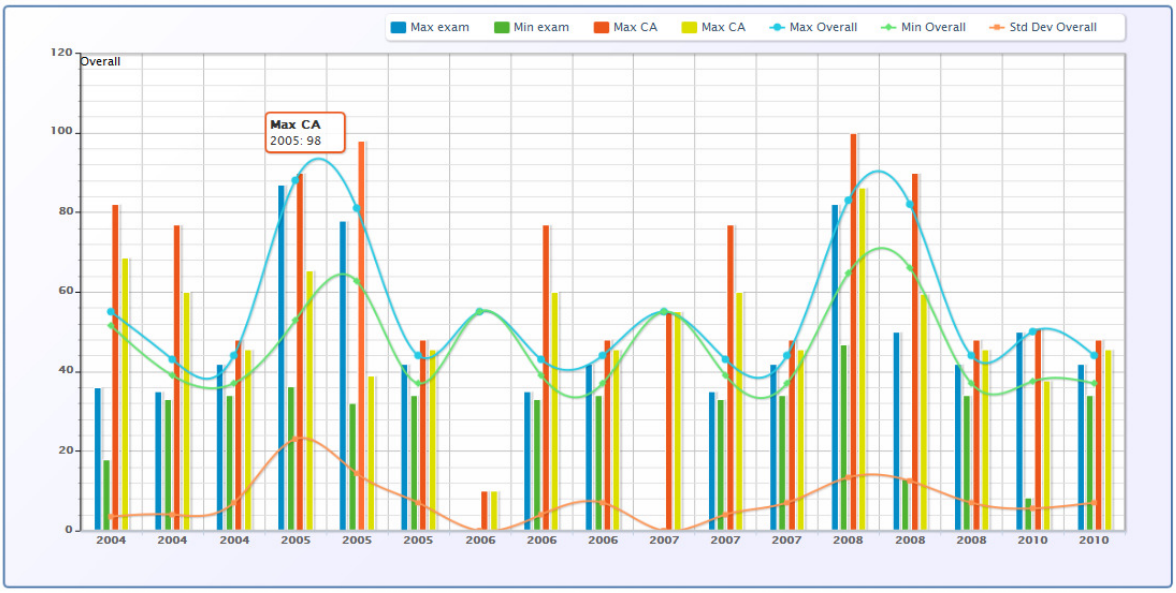


Figure 84 Student Overview Yearly Grade Deviation Combination Chart

These types of combination charts were chosen for use in the application as they are an excellent way of conveying various data sources to the user in an easy to understand and interactive manner. They also support the use of multiple axes which proved useful when plotting counts against percentages, for example when analysing the number of modules failed per semester against the overall semester GPA an additional scale is needed for each as the maximum values for each are extremely different. This makes it difficult to infer any meaning from the data so a dual axis approach such as that in Figure 85 is the best approach to take.

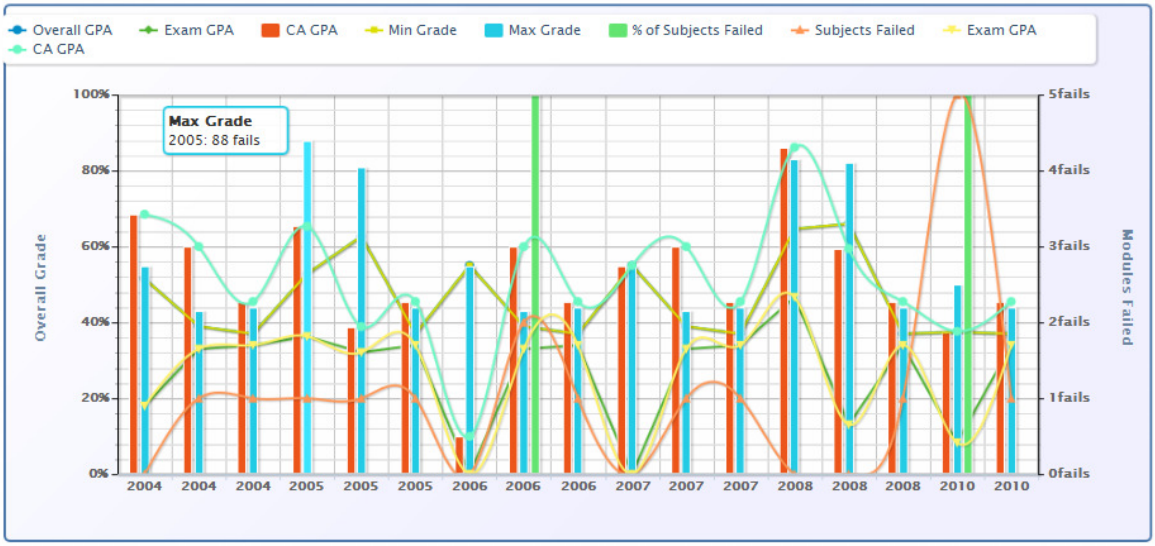


Figure 85 Dual Axis Student GPA Vs Modules Failed

4.3.1.5 Google Parallel Co-Ordinates chart:

Google Parallel Co-Ordinates charts are a relatively new bespoke visualisation provided with the API and offer a very low level alternative to emerging high level visualisation techniques such as the Textile Plot which was initially considered in the planning stages of this project however ruled out due to the complexities that would be involved in designing such a visualisation from the ground up. Textile plots are very high dimensional data visualisations using modified parallel co-ordinate plots and are suitable for mapping any data type (74).

A commonly referenced sample implementation of textile plot analysis is a visual representation of the differences in attributes of flowers. Petal length, species, petal width, sepal length and sepal width are compared to try and spot major differences in the species. An example visualisation of this is shown in Figure 86 (74).

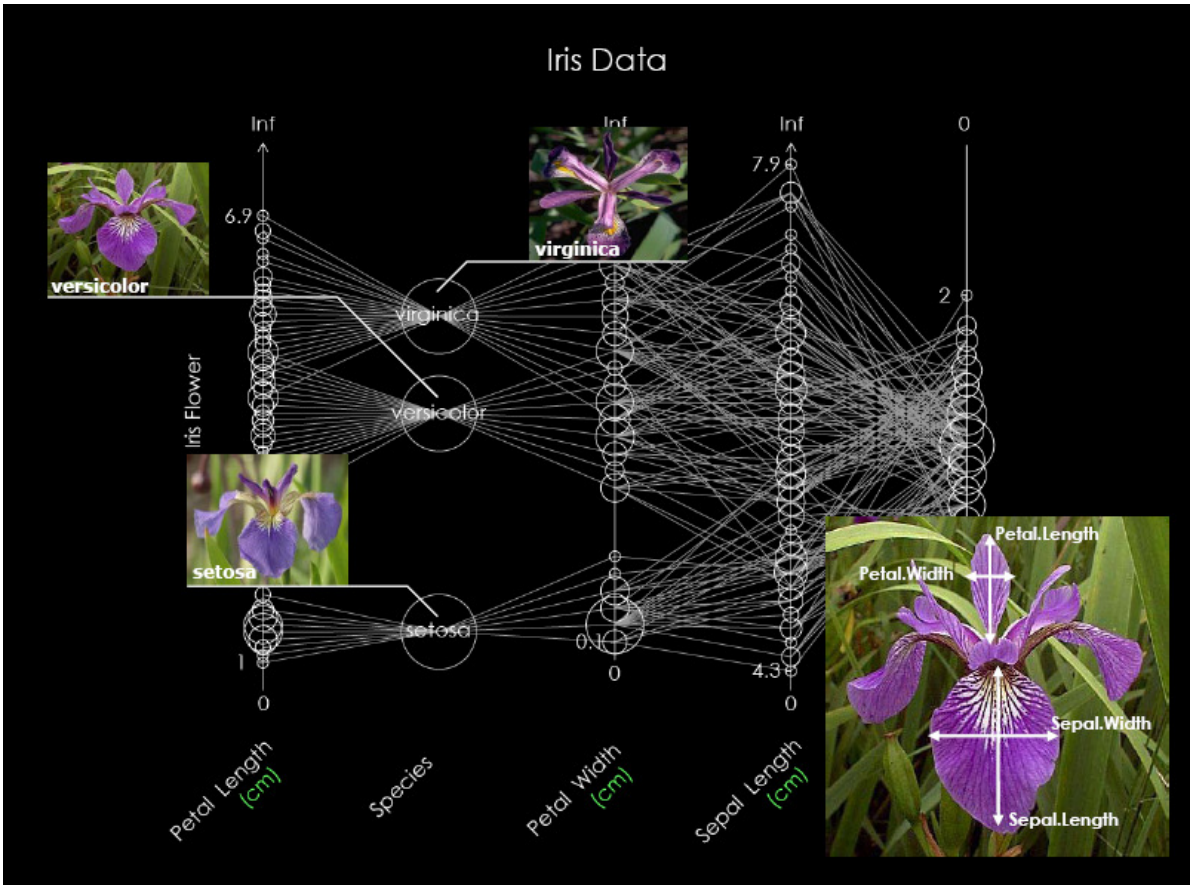


Figure 86 Textile Plot - Iris Data Example (74)

If variables are closely related in textile plots then the lines will follow as straight a line as possible from left to right. Axes are totally independent and can be inverted so the first column could be a scale of 2.5 to 9.7 and the second could be a scale of 9.7 to 2.5 for example. Textile plots offer a new way of visualising data however it is thought to be over-complicated for the purposes of this project. It would be possible to plot it using HTML5 canvas however it was not worth it in the case of this particular project due to the time constraints in place.

Parallel Co-Ordinates charts are a much simpler version of these and are included as a simple example to indicate how this type of analysis could also be performed on the student data set. Figure 87 shows an example of how they are applied to the student data set.

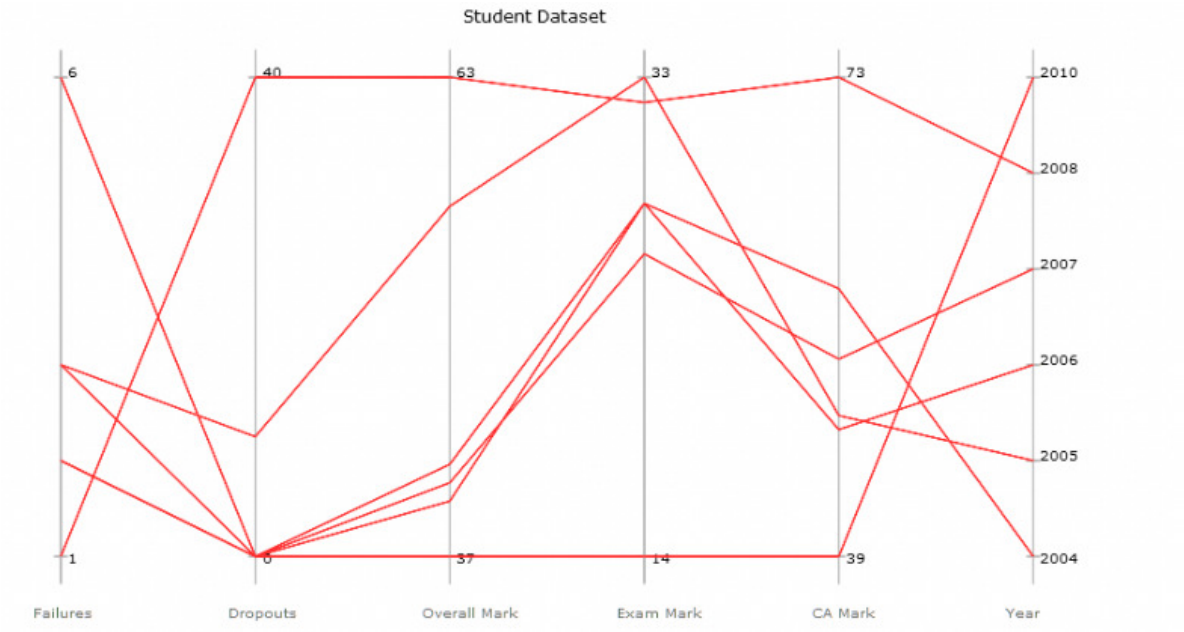


Figure 87 Student Data set Parallel Co-Ordinates Chart

Yearly performance lines are drawn from left to right according to several metrics of student performance including the students module failure count, dropout count, average examination grade, average CA grade and overall average grade. This is not one of the best visualisations used in the project and is merely used as a sample of how a textile plot style visualisation could be achieved using the Google Visualisation API. The chart is written in pure JavaScript and doesn't require Adobe Flash like most of the other Google Visualisations.

4.3.1.6 Google Gauge Chart:

The Google Gauge Chart was chosen to display an instant overview of performance using a speedometer style display. Figure 88 shows an example from the student analysis section where it is used to display student GPA averages in continuous assessment, exams and overall. A key feature of the gauge visualisation is the possibility to allocate different colours to different range sections.

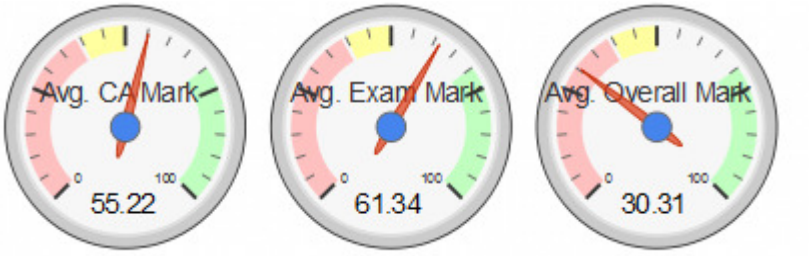


Figure 88 Student Overview Google Gauge Chart

In Figure 88 the scale is divided up according to the class of mark obtained so grades between 0 and 40% are coloured in red to indicate the student is failing and grades between 40 and 50% are highlighted in yellow to indicate that the student is borderline between pass and fail. Grades between 70 and 100% are coloured green to indicate that the student is doing well and is achieving first class honours. Second class honour grades between 50 and 70% are left without colour as only 3 types of colours can be used in a single gauge.

This visualisation can provide an instant analysis of how a metric is performing in respect of a performance range and can be applied to many other purposes as long a percentage rather than numbers are considered as the scale needs to be from 0 to 100. This visualisation can be used to analyse modules by showing the average module mark in the same fashion as in Figure 88 or by showing for example the percentage of students who fail the module using a scale of acceptable fail percentage. If the average failure rate for all modules in the university is 2% an inverse scale could be used to move towards 100% failures.

4.3.1.7 Visifire Charts:

Due to the vast amount of initial research into Visifire charts before High Charts were discovered several graphs were prepared as trial visualisations and some of these were used in the final application where appropriate. Visifire charts are useful in certain circumstances and can render easy to interpret graphs at smaller sizes than the other technologies so they are useful when creating visualisation dash boards and quick overview pages. Visifire Column, Funnel, Area and Pie charts are all included in the main accordion menu of the landing page of each analysis area. This helps the user quickly flick through the overall patterns of the student, module or course at a glance before delving further into more detailed analysis. Figure 89 gives an overview of their use in the student analysis area.

Figure 89 Student Analysis Visifire Visualisations

4.3.2 Data Tables

Data tables are a powerful way of conveying statistical information to the end user and there are numerous powerful data table visualisation technologies currently available. After extensive research the top three were decided upon and are discussed in further details in the sections below.

4.3.2.1 Google Data Tables:

As part of the Google Visualisation API a data table visualisation is available which enables adding formatters to certain fields and table manipulation by the end user allowing columns to be assorted according the numeric or alphabetical value. Data tables take the same JSON input as Google visualisations and should be used as the first point of call when developing complex visualisations such as the motion chart. Once a Google data table is populated a motion chart can easily be created by adding the exact same data source JSON string to a motion chart constructor or even simply initialising the motion chart in the same constructor as the data table.

Google data tables are used extensively in all analysis sections both in the accordion side menus as key performance indicators to give a quick view of performance and in the main tabbed area to provide more in depth data analysis.

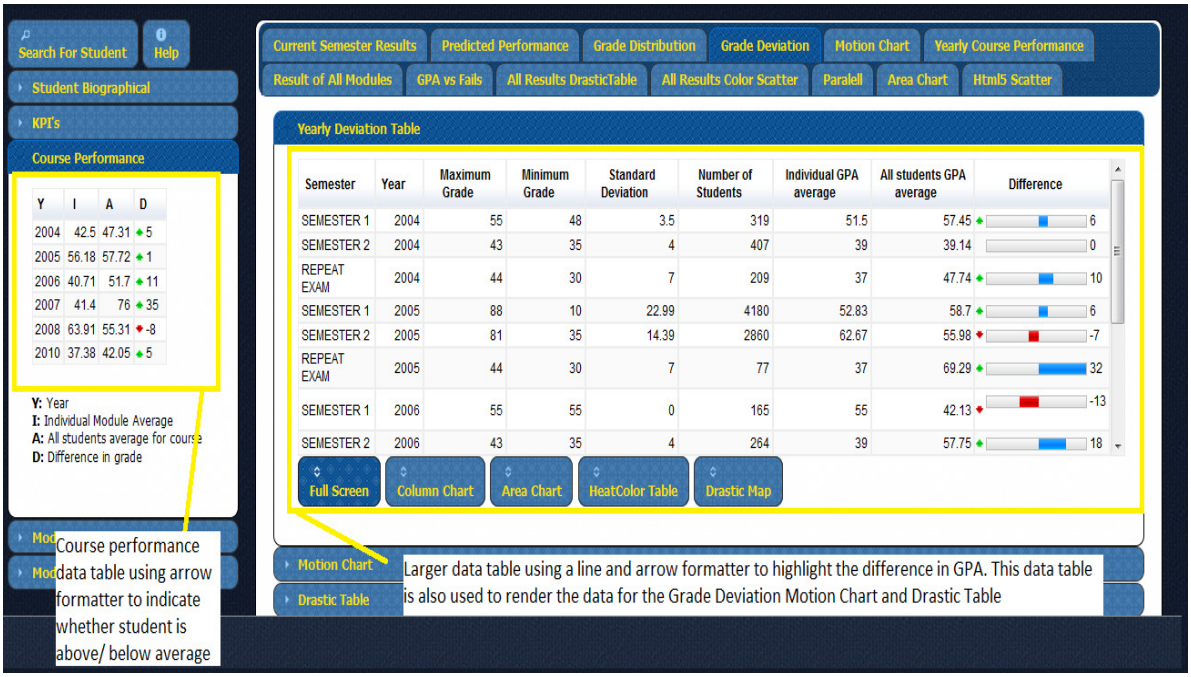


Figure 90 Student Analysis Google Data Tables

Google data tables are good to use as a clean and crisp way of displaying data however a lot of development has been done in this area and there are many other better solutions available to solve different problems. Various other alternatives are also far easier to populate as they can be generated purely by including the relevant java script source code and then creating an ordinary HTML table using `<tr>` and `<td>` elements rather than needing to generate a JSON string. As most of the user interface is built using jQuery it made sense to research into data tables using jQuery in a bit to find new and better ways of representing tabular data.

4.3.2.2 jQuery Data Tables plug-in:

The DataTables (75) plug-in for jQuery provides an excellent means of displaying data in client side HTML tables and uses AJAX technologies to allow for a high level of interaction which is all performed on the client side. New rows can be dynamically added to tables without refreshing the page and there is an extremely useful built in search function which allows the user to filter through the results in the table. The tables are also sort able by numeric value or in alphabetical order for strings. Colour coding can be attached to the sorting mechanism to highlight rows in colour based on certain values and draw the users attention to them.

These tables are extremely useful when wanting to display a table with a lot of results ina small space as they have a built in paging method so it is possible to define the maximum number of rows to be displayed in one page of the table and subsequent rows are then grouped into separate pages. Data table pages can be browsed using the built in navigation and a really beneficial aspect of these tables is that all rows can be sorted ascending and descending even though they may be currently spread out over different pages. Figure 91 shows an example usage in the student analysis area where a table is used to display the result of all modules the student has ever taken.

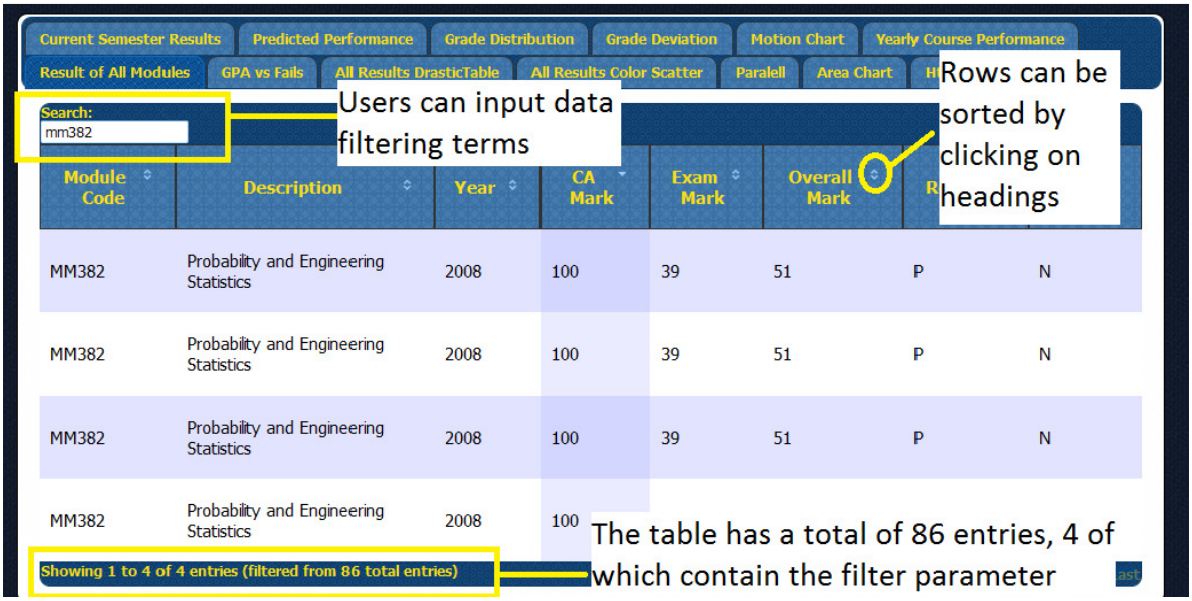


Figure 91 jQuery Data Tables plug-in

4.3.2.3 jQuery Heat Colour Tables:

Colour can be an extremely useful when visualising and sorting data in interactive tables and the jQuery HeatColor (76) plug-in is a great tool for doing this. It automatically colour codes rows when sorting to highlight similar rows and help the user quickly draw similarities between rows. This is best illustrated by Figure 92 which is a heat colour table used in the student analysis section. The results are sorted based on the standard deviation or variance of GPA to highlight which year saw the most inconsistent results for the student.

Semester	Year	GPA	E-GPA	CA-GPA	Module Count	1.1 Count	2.1 Count	2.2 Count	Pass Count	Fail Count	Repeat Count	Drop-Out Count	MIN GPA	MAX GPA	STD. DEV GPA
SEMESTER 1	2004	48.67	12.0	64.67	3	1	0	0	0	2	0	0	14.0	92.0	32.43
SEMESTER 2	2008	49.78	49.78	39.13	5	0	14	14	0	4	0	23	0.0	66.0	23.13
YEAR LONG	2004	39.0	34.75	54.75	4	0	0	1	0	3	0	0	31.0	50.0	7.11
REPEAT EXAM	2004	35.0	34.0	45.5	2	0	0	0	0	2	2	0	30.0	40.0	5.0
SEMESTER 1	2008	70.16	70.16	40.42	4	9	15	0	0	0	0	19	67.0	77.0	3.7
SEMESTER 2	2010	5.0	0.0	35.0	1	0	0	0	0	1	0	0	5.0	5.0	0.0
REPEAT EXAM	2010	50.0	0.0	50.0	1	0	0	1	0	0	1	0	50.0	50.0	0.0

Figure 92 Yearly Grade Distribution HeatColor Table

The colour coded sorting of columns makes it easy to distinguish between semesters with closely related GPA deviations at a glance and it makes it easier for the user to view the data and follow the full length of the row without becoming lost in between data cells. The styling of HeatColor tables isn't as futuristic looking as Google data tables nor as interactive as jQuery DataTables but they offer better functionality to help the user wade through masses of data than either of the others.

4.3.3 Other visualisations

4.3.3.1 HTML5 Canvas:

Another type of regression scatter plot is also included in the application which was coded from scratch using JavaScript and HTML5 canvas. Work on this had been completed before HighCharts were encountered but it is included as an example of how powerful the canvas tag is and how that with some work the possibilities for drawing on the web are endless. It would take a lot of work to develop graphs that are as interactive as High Charts so for the purpose of this project it is better to use something already completed.

It can be seen in Figure 93 below that it is possible to achieve a similar level of graphic representation using purely JavaScript and HTML5 and without the use of any external libraries. Some excellent resources on getting started with coding our own HTML 5 graphs at the Think Vitamin drawing with HTML5 canvas tutorial page (77) and in another tutorial on the Dive Into HTML5 website. (78) The full source code for the example in Figure 93 can be found in Appendix 4 (d): HTML5 Canvas plot code:.

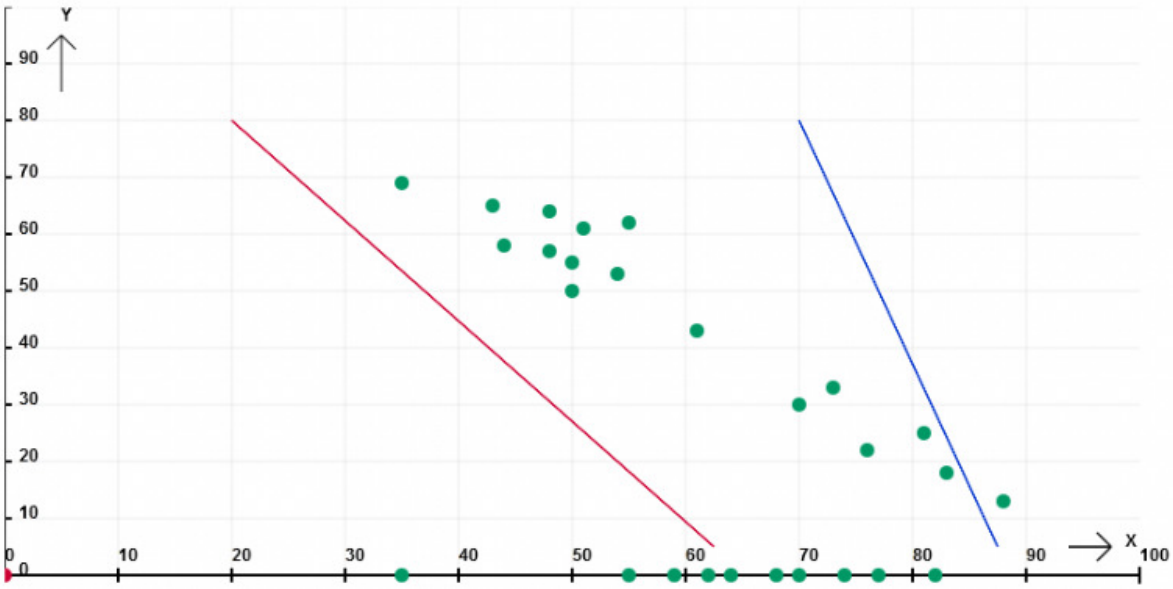


Figure 93 HTML 5 Regression Scatter

4.3.3.2 Google Annotated Timeline:

The Google annotated timeline (Figure 2) is a useful analysis tool that had been intended for use in this project however problems were found with its integration with the jQuery UI framework and the charts would not render inside of jQuery UI components. This is the only type of visualisation that exhibited these symptoms and even code verified as correct in the Google Code Playground (8) and directly inserted to any jQuery UI component would not render anything. A bug was raised with the Google code team to alert them to this issue and a response is still pending so it was decided to omit it from this project.

It had been intended to use it to display a complete GPA history for both courses and modules which would have been quite useful however the whole application wasn't going to be re-designed to suit one particular visualisation. It could have been included in separate page in the application on its own without any jQuery UI components however this would defeat the whole purpose of the RIA experience that is being attempted. There isn't much merit in designing the whole application so the user can do anything they wish on one page without reloading it if you then force them to travel to another page and back again just to view a single visualisation.

4.4 Application logic

4.4.1 Java Server Pages

Initially ordinary HTML files were used as the front end to the servlets however after some analysis and coding it was discovered that this would not work at all when plotting Java Script based visualisations. When the browser loads a HTML document it first process the Java Script elements, and then the server side includes and then renders the page in the browser. The application works by using servlets to dynamically decide on data to be drawn using an input parameter and to add this data to the chart rendering java script code.

A lot of issues were encountered when attempting to plot graphs using HTML files as the Java Script was being processed by the browser before the servlet data values were inserted. The graphs were then rendered and displayed without data and the data was inserted to the relevant place after the graph was drawn. This meant that the graph was not displayed but when the HTML source code was viewed the correct data values were present in the correct place which made de-bugging a nightmare as even the tried and trusted Firebug Firefox web development plug-in (79) couldn't find any issues with the code.

When the pages were being rendered as plain HTML pages the JavaScript was being interpreted first upon page load followed by server-side includes and this led to the graphs rendering without data in the browser but the source code working fine when copy and pasted to a new plain HTML file. The problem was that the data was placed inside the JavaScript after it had already been rendered to the HTML tag containing the visual so the graph has no data to display but when the final source code is viewed from the browser the data exists in the right place.

The use of JSP's eliminates this issue as the entire page content is automatically compiled into a single raw HTML file at the web server and sent to the browser with all includes processed. JSP's proved to be far more useful than raw HTML files with server side includes in the latter side of the development as they enable java source code to be written in line with HTML similarly to the operation of PHP. This meant that when the number of visualisations grew and a more intelligent way of providing data needed to be found.

By switching from plain HTML files to Java Server Pages (JSP's) the problem was eliminated. With JSP's all data on the page is sent to the server and compiled before rendering the output in the browser. This means that all data from servlets to be plotted is inserted into the correct location in Java Script before the page is rendered in the browser and the graphs are drawn correctly.

The performance of JSP's are a lot better than that of plain HTML files due to the way the documents are processed. Plain HTML files take the same amount of time to load every time they are loaded regardless of what has changed on the page. The first time a JSP page is loaded in the browser can take some time however upon subsequent re-loads of the page only information that has been changed is reloaded, the rest is taken from the browser cache. So in the case of this application only the new data provided to the graphs changes so this is all that needs to be loaded.

The use of JSP's vastly improved this application and it is a big regret they weren't seen as a viable option sooner. Using servlets alone to plot the graphs was a very cumbersome task as using plain HTML server side includes are not permissible directly inside Java Script code. So to plot a JavaScript chart with a servlet successfully the entire script needs to be in the servlet in the form of `out.println("")` commands. Using JSP's the servlet output can be directly included inside the JavaScript code in the JSP page so all of the graphs can be written normally inside the JSP HTML and only the output taken from a servlet.

4.4.2 Java Servlets

Java servlets are used to perform the business logic in the application and to connect to the database and retrieve and format information for display in visualisations. The servlet structure was planned and a separate package was created for database connectivity and each specific area of analysis so one package for student analysis, one for module analysis and another for course and faculty analysis.

This setup is mirrored in the structure of the JSP pages in the web root directory to make it as easy as possible for another developer to take the project further and improve on the application. A key source of information on servlets used as a reference point was the EE557 Server Side Development wiki page (80) which provides a detailed description of

how J2EE applications work along with multiple example servlets and JSP's to use as a starting point.

The servlets in this application are designed to connect to the database and call stored procedures which return a result set to the servlet for processing. This data is then manipulated for use in visualisations or data tables. The JSON data format is used as much as possible and both HighCharts and the Google Visualisation API can generate data using this format. The aim of the servlets is to connect to the database and perform set of operations and then transform the results sets into JSON strings to be written to session variables for use throughout the application.

Once the servlet has created a session variable with the data it is available globally for the course of that session and the string variable can be included within the data input tags of any a visualisation inside the JSP front end pages. The design and implementation of this took a lot of thought as the most intelligent solution with the least amount of repetition was sought. HighCharts graphs stood head and shoulders above the competition again here as all of the data series input follow the same simple format and are a JSON string of comma separated values. Each data set has its own distinct string and only values belonging to that data set exist in that string. This means that once a string has been created once it can be re-used in several charts with no need to change the format.

Google visualisations are not as good in this aspect as there can be several subtle differences in the input format required from visualisation to visualisation. Some require specific fields in a certain order for example a string followed by a date so if you want to cut as many corners as possible when rendering the charts it is worth considering all of the possible charts you want to draw using the data will display using the same input format.

Some of the other technologies used do not support JSON and a less re-usable format of population is needed. In these cases servlets are used either to output a set of table rows to be included inside a JSP file or in the case of some the Visifire charts used to plot the whole visualisations using java print statements. Servlets are a powerful and efficient way of providing information for this application and their inner workings are discussed in forther detail in the implementation section.

4.4.3 Database:

Initially the database used was an Oracle database as this is where the DCU student academic data currently resides and seemed the best option to begin with as then the application could integrate with the existing source extremely easily, simply by changing the database URL, username and password in the application. This worked well for initial development and a remote Oracle database hosted by the university was used.

This proved to cause problems after some time as the application working depended on the availability of this database which is controlled externally. It also required internet access to connect to this remote database and there was limited access to the database itself. A decision was made that a local database would be best to use to eliminate these issues however some more issues were encountered here. Oracle is not free software and requires the purchase of a license to set up locally. There is a free-ware version available known as Oracle Express (81) however it does not work on Microsoft Vista or on Windows 7 which were the host operating systems for the development of this project.

To eliminate all of these issues a decision was made to move to a mySQL database hosted locally. mySQL is an open source alternative to Oracle that offers all the features needed in this project. There is also a limited amount of work to be done to integrate it to the real-time Oracle student database. The SQL queries used in the application are identical and all that would be needed to move to the live database would be to change the connection details and use a different Java Database Connect (JDBC) jar file.

The WAMP web-server package (82) was used as this incorporates mySQL and includes the phpmyadmin (83) interface which makes creating and maintaining the database a lot easier through a browser GUI. The installation was far easier compared to oracle and very little configuration is required. The web server can be run in tandem with Tomcat as long as they are running on a different port so it was perfect for this application. The only major issue encountered was that Skype (84) used the same default port initially so if it was running and the WAMP server was started it could not correctly start the database. Once the default port was changed this issue was eliminated and there were no further problems.

WAMP is installed as a stand-alone application and to improve the portability of this project the database is included on the attached CD (see Appendix 5 – Contents of accompanying

CD) hosted on another more portable web server known as 'USB Web server'. Browse to the USBWebServer folder and launch UsbWebserver.exe. This will start an instance of Apache running on port 8088. This portable distribution contains a mySQL database running on port 3306 populated with the studentdata database and all stored procedures used as part of this project. The ports both services are configurable in the easy to use menu and have been set so as to avoid conflict with the Tomcat port used during development - 8080. For more information about USB web server see (85).

One disadvantage of using mySQL is that Oracle has a lot of built in functionality that would have been useful in performing complex statistical analysis however similar techniques are possible in mySQL with some work. Another excellent feature of Oracle is the ability to return queries as a JSON string which would have been very useful when plotting visualisations and would eliminate the need to use JavaScript to re-format the values for visualising.

4.4.4 Database interactivity

Gathering the data required to plot the various visualisations used in the application can be quite cumbersome as many variables need to be calculated. To try and reduce the load being sent between server and client stored procedures are used to speed things up. It is a lot quicker for the database to call a stored procedure being called from an application than running a raw SQL query as with raw queries the entire query and the input parameter must be sent to the server and then the database. If stored procedures are used only the input parameters need to be sent to the server and this greatly reduces traffic on the network.

As more complex queries were being performed and more stored procedures were being created the phpmyadmin web user interface was ditched first for raw command line SQL and eventually for MySQL Workbench (86) which is quite similar to Oracle SQL developer (87). The decision to do this was because a lot of problems were encountered when trying to create stored procedures in phpmyadmin.

The main issue due to the fact that when creating a stored procedure the SQL delimiter element needs to be set to another value, typically '$$' so that the usual delimiter ';' can exist inside the procedure without ending it. The phpmyadmin interface doe not handle

changing the delimiter well and proved to be quite frustrating to use. MySQL Workbench is free to download and highly recommended. Oracle SQL developer is also compatible after the installation of a MySQL plug-in however this proved to be a difficult set-up to get right and MySQL workbench did just as good a job.

4.4.5 Conclusion

By spending a lot of time during the design phase the application implementation phase was a lot easier and fewer issues were encountered than if the development has started before a clear objective was established. The user interface and what visualisations were chosen to be displayed was a key part of organising the layout and focusing on the relevance of data being displayed. The style and graphics of graphs is of secondary importance to their use in making analysis of student information and careful design was required to make the end product as useful as possible.

With a clear design structure in place and a plan of the steps needed to be takent o achieve this the project could move on to the implementation phase. This is covered in the next section Chapter 5 – Implementation of the application which goes into further detail on the actual implementation of the application and gives a detailed explanation of the behind the scenes workings.

Chapter 5 – Implementation of the application

After the design of the application was finalised and the objectives solidified it was time to move on to building the application itself. The structure of the application is very simple with only 8 actual web pages and the jQuery UI is used to emulate the functionality of a much more complex site in as simple a manner as possible. The application structure follows the conventional WAR file structure and the web page files are stored in the root web directory and its subdirectories. The web pages are the:

- Project home index page
- Student visualisation homepage
- Student analysis page
- Module visualisation homepage
- Module analysis page
- Course visualisation homepage
- Course analysis page
- Help page

Separate folders are used to organise the structure of the analysis areas as even though there is only a single page displayed to the user each analysis page is split up into several JSP files to make it easier to manage the content. A separate JSP file exists for each visualisation used in the analysis area and these are then included in the index page for that particular area. This makes it easier to make changes to individual visualisations without having to delve through the source code for the entire page.

The home pages and the help page are simple and non-interactive pages used to extend the project beyond the scope of simply visualising different metrics of student performance. The inclusion of these pages turns the project into a stand-alone web application that can be used on its own as well as being integrated into existing systems.

5.1 Application styling

The style of the application is controlled using a global CSS style file located in the CSS directory. The page is divided into sections using div elements and the sections are the header, footer, left accordion menu area and main tabbed content areas. Inline CSS was initially used to alter some elements of the page however this was revised and a global style

file GlobalStyle.css was created to hold all styling in one place. It is far easier to make global configuration changes and if the configuration is centrally managed in this manner. Dashed borders were drawn around the page sections during the development stages to clearly distinguish between sections and to help optimise the design by altering the space available to each element to ensure no usable space is wasted.

5.2 jQuery UI Components:

To use jQuery UI components they need to have the jQuery UI source and the theme available locally on the web server and have instances of the various features required in the page declared in JavaScript at the head of the page. jQuery components are set to run on document load so they will be called into action as soon as the page is loaded.

An example jQuery UI dialog, accordion and tab declaration is given in the java script source code below:

```
<script type="text/javascript">
$(function(){
// Accordion menu is initialised
$("#individual_accordion").accordion({ header: "h3" });
// An instance of tabs is initialised
$('#individual_tabs').tabs();
// The help dialog is initialised
$('#help_dialog').dialog({
                              autoOpen: false,
                              modal: true,
                              show: 'clip',
                              hide: 'puff'
                        });
// The help button is intialised
$('#help_button')
            .button()
            .click(function() {
            $('#help_dialog').dialog('open');
                        });         });   </script>
```

Firstly the accordion menu is initialised with the name individual_accordion. This is the name that will be use to reference the div element containing the accordion in the HTML code. Next an instance of tabs is created in a similar fashion with the destination div tag of the tabs being given the name individual_tabs. That is all that is needed to enable the jQuery UI components once the core package is also included and the syntax is surprisingly easy considering the excellent quality interactions it can produce.

The dialog declaration section is a bit more complicated but makes perfect sense once figured out. First a dialog is initialised. This is the actual pop-up box that will appear in front of the user. In this example it's the help dialog and it has properties predefined. The autopen property indicates whether the dialog should open when the page is initially loaded. The modal property controls whether or not the user can interact with the background application once the dialog is active. The show and hide options simply specify the animate style used to show and hide the dialog.

The next declaration is extremely important and without it there wouldn't be any way of accessing the dialog, A new button is created with a click event bound to it. When the click event is activated on the button the dialog is called and displayed to the user. The HTML code for the dialog is placed inside the body tags of the page and follows a simple format, an example of which is shown below:

```
<div id="help_dialog" title="Help">
      <div class="ui-widget">     </div>
            <p>
            Several metrics of student performance are shown.
           </p>

  </div>
```

The id of the div is used to map it to the call from the button and the title is the display title in the dialog. jQuery syntax is surprisingly easy to understand considering the complexities of what it can do and is worth spending some time getting to grips with due to the benefits that can be achieved from its use. The inclusion of the previously declared tabs element is also quite straightforward and the following example shows how to call the tab in HTML with the accordion called in the first tab:

```
<div id="individual_tabs">
<ul>
    <li><a href="#tabs-1">First tab</a></li>
    <li><a href="#tabs-2">Second tab</a></li>
</ul>

<!-- The tabs  -->
<div id="tabs-1">
 <!--  First tab content here-->
            <!-- Accordion menu -->
            <div id="accordion">
                  <h3>
                        <a href="#">First header</a>
                  </h3>
                        <div> First content
                        </div>
                  <h3>
                        <a href="#">Second header</a>
```

```
                    </h3>
                    <div>Second content</div>
                    </div>
                    <!-- End accordion menu -->
</div>
<div id="tabs-2">
 <!--  Second tab content here-->
<button id="help_button”> Help</button>
 </div>
 </div>
```

An unordered list is used to store the tab Labels and anchor links are used to bind these to the tab div elements. These links can also be used to directly jump to a specific tab on another page by specifying its anchor in the URL for example www.tabs.com#tabs-2 . jQuery tabs are an excellent way of organising a user interface and offer many advantages such as the fact that they automatically scale to size so there is no need to worry about changing the size of the tabbed interface if you wanted to increase the size of a visualisation in the content area. All you need to do is resize the visualisation and the tabs will size themselves to fit perfectly.

The accordion menu also automatically sizes to shape and its configuration is similarly easy to understand. The accordion uses headings to organise the menu headings and a local anchor to the div directly beneath the heading. This means the heading is essentially a link and when followed the div element directly beneath it is called as no anchor id is specified just ‘#’. The exact heading type can be specified in the accordion initialisation but the default value of a third level heading works well. If this is increased then the heading area takes up more space and reduces the content pane space available.

Combining the tabbed interface and the accordion as explained can provide the foundations for a highly interactive application very quickly and this is one of the reasons jQuery is so appealing to web application developers. The example above includes a declaration of a help button in the second tab and if the dialog HTML code discussed previously is included in the same page as these tabs this button would call and open the dialog. This makes the possibilities for expansion almost limitless as it allows for as many pop up full screen dialogs to be used as you like. This is better than delving too deep into submenus by using more than two tiers of accordions or tabs which can cause more confusion to the layout than benefit.

One of the main benefits of using jQuery is the fact that its reliable across browsers and responds to standard browser scaling very well so can easily adjust to different screen resolutions. A huge source of frustration to web developers is the tedium of cross browser testing and often a slight change to fix a bug in one browser can trigger a host of other problems in another browser. Using a user interface library that has trusted compatibility makes a big difference and the question begins to become why wouldn't you use jQuery rather than why you should.

Another jQuery plug-in that is used is the form validate plug-in which is used to validate user input in the search forms using JavaScript client side validation. Different criteria are required for each form for example the student ID number must be 8 digits long and only consist of numeric characters. The validate plug-in provides tools to prompt the user and let them know where they are going wrong which can greatly decrease user frustration if they cannot understand why the form won't accept their input.

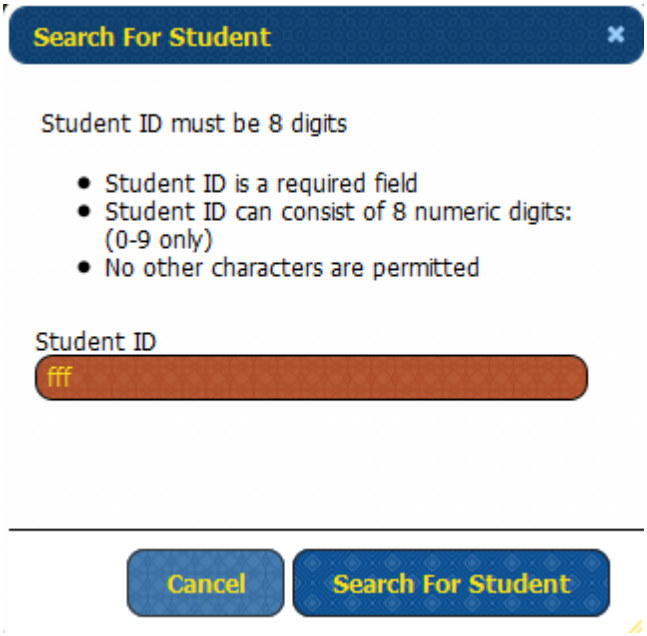


Figure 94 jQuery Form Validation

A problem observed with the search form dialog is that users were able to bypass the validation by repeatedly pressing enter on their keyboard. A workaround to this can be achieved by simply disabling the enter button as a form submitter using the following code:

```
// Disable Enter command for form submit
$("#Student-Search-form").keypress(function(e) {
                        if (e.which == 13) {
                              return false;} });
```

Once the validation is happy with the input from the form the user is redirected to the validation servlet using a JavaScript Re-direct which is illustrated by the following code:

```
// create var to store URL rewrite and redirect to servlet for validation
var redirect = "/Thesis/servlet/ValidateStudentID?studentID=" +
studentSearch_ID.val() + "&valid=1";
      $(location).attr('href',redirect);
```

5.3 Heat Colour Tables:

Heat colour tables are generated by applying a JavaScript sorting function to fields of a HTML table referenced by a div id. They are initialised in the head of the page as shown below:

```
<script type="text/javascript">
$(function() {

if (!($.browser.msie && $.browser.version == 6.0))
$("#ex3").tablesorter();

function sortwithcolor( column ) {
$("#ex3 > tbody > tr").heatcolor(
function() { return $("td:nth-child(" + column + ")", this).text(); }
);
};

$("th").click(function() {
$(this)
.siblings()
.css("background-color","#cccccc")
.end()
.css("background-color","#dd0000");
sortwithcolor( $(this).parent().children().index( this ) + 1 );
});
// this sorts on 6th column when first shown
sortwithcolor(7);


});
</script>
```

5.4 Preparing visualisation data

Preparing the student data for visualisation is divided between the database and the application layer. Initially the bulk of the work was done in the servlet however as the queries became more complex better ways of processing the data at a direct database level were sought. Sending large queries to the database for each visualisation places a lot of stress on the connection and can cause poor responsiveness in the application. Using stored procedures reduces the load being sent between client and server and improves the loading time of the application significantly.

In the following subsections a discussion of the key logic performed in the servlets and at a database level is discussed to provide a fully detailed explanation of how the application works.

5.4.1 Java Servlets:

Servlets are used to prepare data for visualisation and connect to the database using JDBC which involves the use of a database specific jar file. The mySQL JDBC connector is known as connector/J and can be downloaded from (88). Once downloaded it needs to be added to the libraries of the web application and placed on the build path also. This enables the servlet to communicate directly with the database and is the foundation for how the application works.

A separate java package is used to store the servlet used to connect to the database and this servlet is referenced in subsequent data gathering servlets by calling its connection method. This servlet is called SQL_DBUTIL.java and can be found in appendix 1. It is a simple enough servlet that tries to establish a database connection and returns an error code of 1 if successful. In other visualisation servlets the connect method is called and if its error code is returned as 1 the servlet carries out user specified operations and if not it displays and error message.

Other servlets used are organised into packages according to the section of the application they are gathering data for as this makes it easier to make changes to a specific section. The student analysis area was developed as the master page and the module and course analysis areas were cloned from this. The operation of all servlets in each section is essentially to provide the same functionality however different stored procedures are called as different metrics are being analysed and results can be required in a different format so it's better to spate things out.

The key classes to take note of in each section are the search form validation class and JSON population servlets. All JSON population was initially done in a single servlet but as more and more visualisations were added this became unmanageable and the best option was to use separate servlets for each Google visualisation, separate servlets for populating data tables a single Visifire population servlet and HighCharts JSON population servlets.

The reason for this separation is the fact that each visualisation type needs a different type of JSON format and it's easier to simply create multiple files to do this. High charts were by far the easiest visualisations to populate as all of their data source code follows the same structure of values inside square brackets for example [1, 1,1] or ['string', string']. As this convention is consistent across all charts it meant that several predefined metrics could be added to session variables in the form of JSON strings and be re-used when needed throughout the application by requesting the relevant session variable.

The basic operation of the application is that when a user successfully submits a search form they are redirected to the ValidateID.java file for their section with the search term they entered re-written to the URL. As an example the ValidateStudnetID.java servlet is provided in appendix 1. This servlet checks the validity of the search term in the database and if the term exists and is valid it moves on to the next stage which is the first JSON population servlet. If the search term is invalid they are re-directed to an error page. If the term is valid the validation servlet writes the term to a session variable which is fed into stored procedures in the subsequent data population servlets. This improves the security of the application and minimises the risk of users trying to hack their way in by attempting to insert parameters into the URL.

The StudentOverviewPopulateJSON.java servlet is also included in the appendix as appoint for reference even though it is quite a long servlet. It is the best example of how the other servlets operate as it performs all operations they do. This version is taken from the student analysis section and populates JSON strings for all of the student visualisations displayed using high charts by calling the student populate JSON stored procedure. Most of the work is performed on the database side and the servlet merely gathers the result set parameters and adds them to a JSON object. This object is then written to a session variable so it is available throughout the application.

Some calculations are performed on the returned data set to calculate the % of some metrics as shown in the sample code below where the count of student 1.1's, 2.1's and other grades are transformed into percentages to see what percentage of subjects they achieved these grades for:

```
// Calculate % of 1.1'S per semester
int semester11Count = rs.getInt("COUNTOVERALL_FIRSTCLASSHONOURS");
```

```
double Percent11Semester = (semester11Count/semesterModuleCount)*100;
// Calculate % of 2.1'S per semester
int semester21Count = rs.getInt("COUNTOVERALL_SECONDCLASSHONOURS_G1");
double Percent21Semester = (semester21Count/semesterModuleCount)*100;
// Calculate % of 2.2'S per semester
int semester22Count = rs.getInt("COUNTOVERALL_SECONDCLASSHONOURS_G2");
double Percent22Semester = (semester22Count/semesterModuleCount)*100;
// Calculate % of P per semester
int semesterPCount = rs.getInt("COUNTOVERALL_PASS");
double PercentPassSemester = (semesterPCount/semesterModuleCount)*100;
```

This servlet provides an all in one source for all HighCharts visualisations except for the scatter charts as the procedure run there is different as it also calculates regression variables. This procedure is included in appendix 3 for reference. The Student Overview Populate JSON servlet is also cloned to create all the same session variables for different student profiles. This servlet also populate the KPI session variables referenced in the accordion side menus of the application.

The servlet code is exactly the same but it calls a different stored procedure for example a JSON population procedure for all students who are the same gender as the student being analysed. This is done for the main analysis metrics such as all students who have the same modules as the student, all students in the same year as the student took their examinations and could be extended further easily simply by changing the WHERE clause in the stored procedure.

All of these servlets are run and variables added to the session and the application moves the Google Visualisation servlets and runs their population code. Some Google visuals need input data that is too long to be passed using session strings and needs to be included directly inside the visualization code using a direct reference to the servlet used to gather the data. The JSON result set is then included in the visualisation code when Tomcat compiles the JSP at page load time and displayed to the user visually.

The data table and Visifire population servlets use a simpler approach and data from the result set is formatted into the correct syntax using `out.println("")` statements which are then included inside the visualisation script.

5.4.2 Stored Procedures:

The use of stored procedures helps to protect the secrecy of the SQL queries being run as it hides them from the user as much as possible and the only reference to the data being collected is the call to the stored procedure in the SQL query. If a full query was used in the servlets the database structure could easily be exposed. Numerous stored procedures were written and tailor made to suit individual visualisations but the two included in appendix 3 (a) and (b) should provide a sufficient overview to how the others work.

A stored procedure is basically a set of predefined queries to be run n the database and multiple select statements can be used within each which makes them a powerful tool for accessing different data sets and performing calculations. The main calculations used are the traditional SQL functions to calculate average values, sums, counts, maximum values, minimum values and standard deviation of values. Values are rounded to two decimal places to make it easier to plot the values visually. This sort of precision is more than enough accuracy when analysing percentage marks of students. The database structure is provided in Appendix 2 – Database structure and as many mathematical operations as possible are performed on columns thought to be significant performance predictors.

The linear regression procedure was created using help from the MySQL Cookbook (89) and was initially thought to be a great way of calculating regression. After extensive testing problems with reliability were found and it caused a lot of wasted development time as the same procedure would return a full result set on some occasions and null values for certain things on other occasions. Too much time was spent trying to get this to work and eventually it had to be omitted and the regression lines manually plotted on the scatter charts. If there had been more time to complete this project it would have been the main focus of further research and is something that could prove extremely valuable to future researchers working on similar applications.

5.5 Plotting visualisation data

The application analysis areas are divided into three sections to distinguish between the types of metric analysis being performed. Each JSP checks for the existence of the metrics session variable upon the initial load of the page. If a valid session is not found then the JSP displays a totally different interface than would be seen in the actual analysis area. It is in fact however the same page and it is simply an intelligent JSP page that checks if a valid

session exists, and if not shows a different set of tabs than if a session does exist. When the user arrives at this page they can choose to search for a student or module to analyse or to view a default student page using a link to a URL with a hardcoded student ID number.

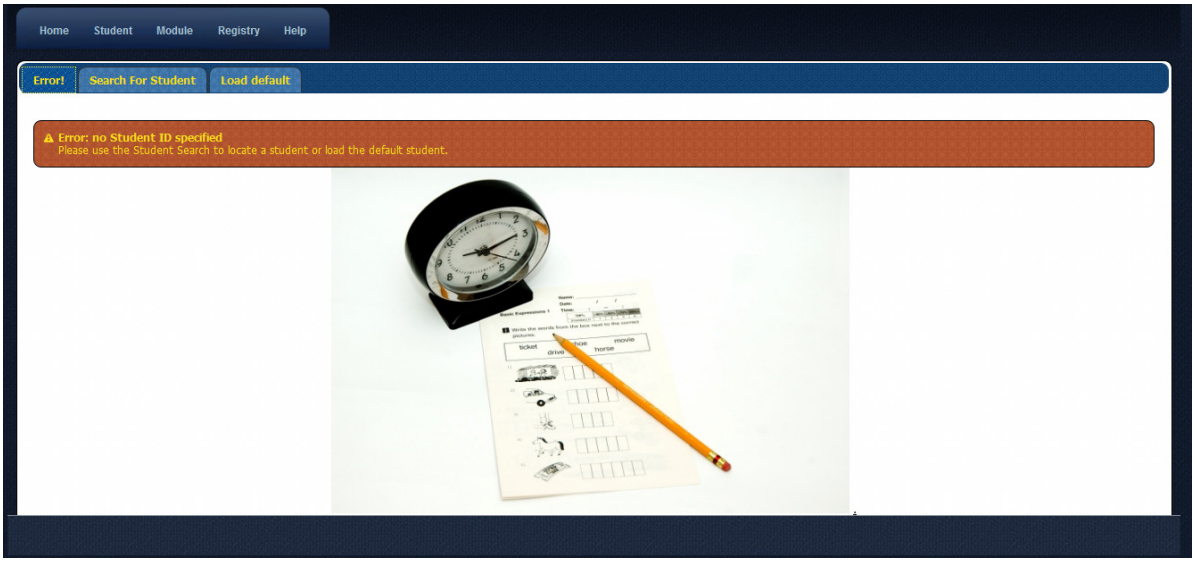


Figure 95 Error page shown when session non-existent

Each visualisation area has its own set of bespoke visuals to suit each metric and these are discussed in further detail in the following sections.

5.5.1 Student Analysis:

The student overview homepage is shown in Figure 96 below and offers several metrics of student performance analysis.

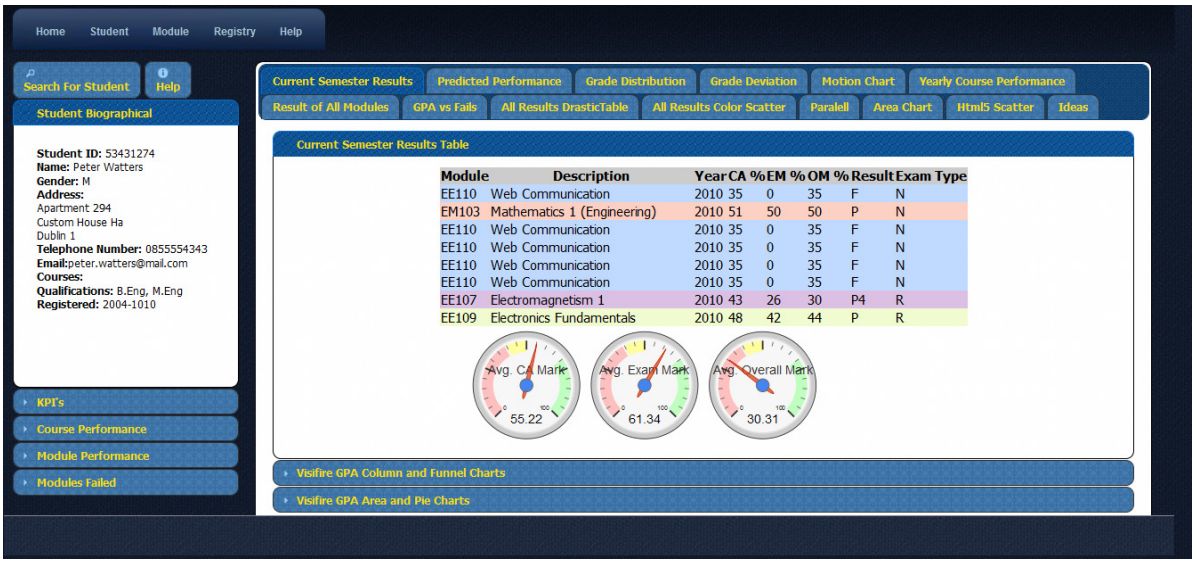


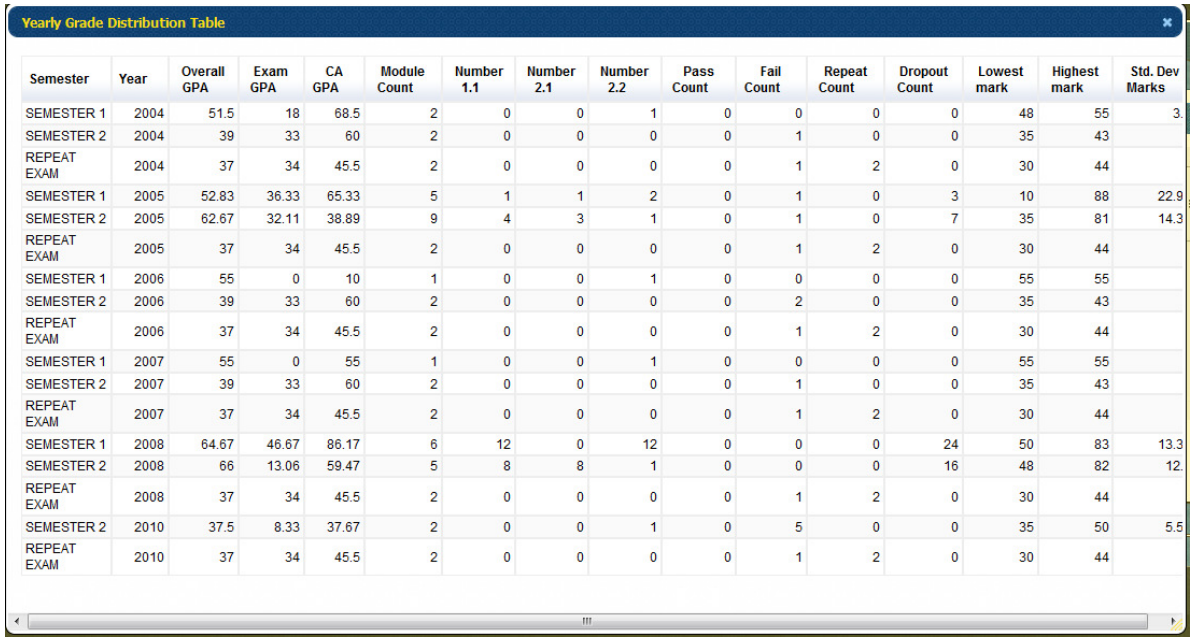
Figure 96 Student Analysis Homepage

Student grade distributions can be viewed on a semester by semester basis, all modules for the entire students' history and various other metrics are analysed. The accordion menu on the left hand side displays personal information about the student along with small Google data tables illustrating an overview of KPI's such as the students course and module performance GPA per year against that of other students in the university. The full listing of the personal KPI metrics in this section with sample values are given in the bulleted point list below:

- **Student ID:** 53431274
- **Name:** Peter Watters
- **Gender:** M
- **Address:**
  Apartment 294
  Custom House Harbour
  Dublin 1
- **Telephone Number:** 0855554343
- **Email:**peter.watters@mail.com
- **Qualifications:** B.Eng, M.Eng
- **Registered:** 2004-1010
- **Course:** DME
- **Overall GPA:** 2.1
- **Exam average:** 56%
- **CA Average:** 67%
- **Started:** 2004
- **Graduation:** 2008
- **Modules Failed:** 2

The first tab for current semester results uses an accordion menu to organise content and the first section displays a heat colour table containing the student results for the current semester along with Google Visualisation Gauges to give an indication of their overall GPA. The next tab displays a linear regression scatter of the students CA mark against exam mark and includes trend regression lines. The next two tabs contain detailed data tables to

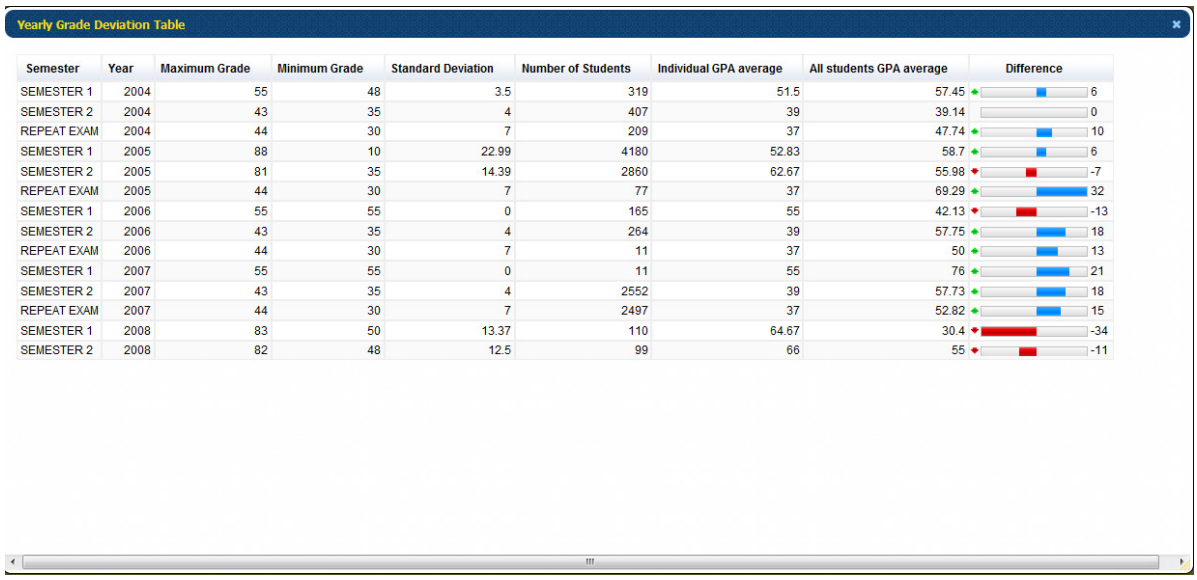
describe student yearly grade deviation and distribution and the metrics illustrated in the tables below are shown on a per semester basis.



Yearly Grade Distribution Table

Semester	Year	Overall GPA	Exam GPA	CA GPA	Module Count	Number 1.1	Number 2.1	Number 2.2	Pass Count	Fail Count	Repeat Count	Dropout Count	Lowest mark	Highest mark	Std. Dev Marks
SEMESTER 1	2004	51.5	18	68.5	2	0	0	1	0	0	0	0	48	55	3.
SEMESTER 2	2004	39	33	60	2	0	0	0	0	1	0	0	35	43	
REPEAT EXAM	2004	37	34	45.5	2	0	0	0	0	1	2	0	30	44	
SEMESTER 1	2005	52.83	36.33	65.33	5	1	1	2	0	1	0	3	10	88	22.9
SEMESTER 2	2005	62.67	32.11	38.89	9	4	3	1	0	1	0	7	35	81	14.3
REPEAT EXAM	2005	37	34	45.5	2	0	0	0	0	1	2	0	30	44	
SEMESTER 1	2006	55	0	10	1	0	0	1	0	0	0	0	55	55	
SEMESTER 2	2006	39	33	60	2	0	0	0	0	2	0	0	35	43	
REPEAT EXAM	2006	37	34	45.5	2	0	0	0	0	1	2	0	30	44	
SEMESTER 1	2007	55	0	55	1	0	0	1	0	0	0	0	55	55	
SEMESTER 2	2007	39	33	60	2	0	0	0	0	1	0	0	35	43	
REPEAT EXAM	2007	37	34	45.5	2	0	0	0	0	1	2	0	30	44	
SEMESTER 1	2008	64.67	46.67	86.17	6	12	0	12	0	0	0	24	50	83	13.3
SEMESTER 2	2008	66	13.06	59.47	5	8	8	1	0	0	0	16	48	82	12.
REPEAT EXAM	2008	37	34	45.5	2	0	0	0	0	1	2	0	30	44	
SEMESTER 2	2010	37.5	8.33	37.67	2	0	0	1	0	5	0	0	35	50	5.5
REPEAT EXAM	2010	37	34	45.5	2	0	0	0	0	1	2	0	30	44	

Figure 97 Student Analysis Grade Distribution table



Yearly Grade Deviation Table

Semester	Year	Maximum Grade	Minimum Grade	Standard Deviation	Number of Students	Individual GPA average	All students GPA average	Difference
SEMESTER 1	2004	55	48	3.5	319	51.5	57.45	6
SEMESTER 2	2004	43	35	4	407	39	39.14	0
REPEAT EXAM	2004	44	30	7	209	37	47.74	10
SEMESTER 1	2005	88	10	22.99	4180	52.83	58.7	6
SEMESTER 2	2005	81	35	14.39	2860	62.67	55.98	-7
REPEAT EXAM	2005	44	30	7	77	37	69.29	32
SEMESTER 1	2006	55	55	0	165	55	42.13	-13
SEMESTER 2	2006	43	35	4	264	39	57.75	18
REPEAT EXAM	2006	44	30	7	11	37	50	13
SEMESTER 1	2007	55	55	0	11	55	76	21
SEMESTER 2	2007	43	35	4	2552	39	57.73	18
REPEAT EXAM	2007	44	30	7	2497	37	52.82	15
SEMESTER 1	2008	83	50	13.37	110	64.67	30.4	-34
SEMESTER 2	2008	82	48	12.5	99	66	55	-11

Figure 98 Student Analysis Grade Deviation table

Multiple additional visualisations are generated using these data tables and Google Motion Charts, Drastic data tables and Heat Colour data table charts are provided using the same data set as the Google data tables to provide for different interpretations of the data. The grade distribution tab also includes several high charts that can be used to measure student performance. A stacked column chart is used to show the percentage of 1.1, 2.1, 2.2, pass and fail grades achieved by the student on a per semester basis and is shown in Figure 99 .

The inspiration for this chart was provided by “A Comparative Analysis of Techniques for Predicting Academic Performance” (48) and in particular Figure 48 in this document.

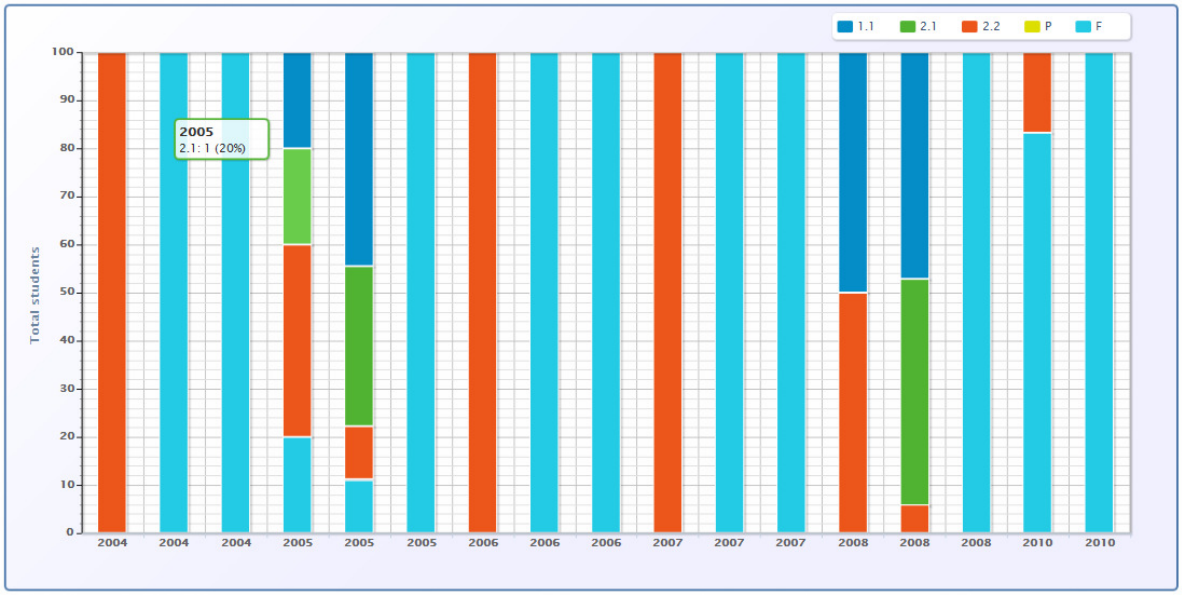


Figure 99 Student Analysis Grade Distribution Stacked Column Chart

A dual axis High Charts chart is also available to view the grade distribution of students on a semester to semester basis with trend spine lines for average overall mark, CA mark and exam mark for modules taken that semester and this is illustrated in Figure 100 .

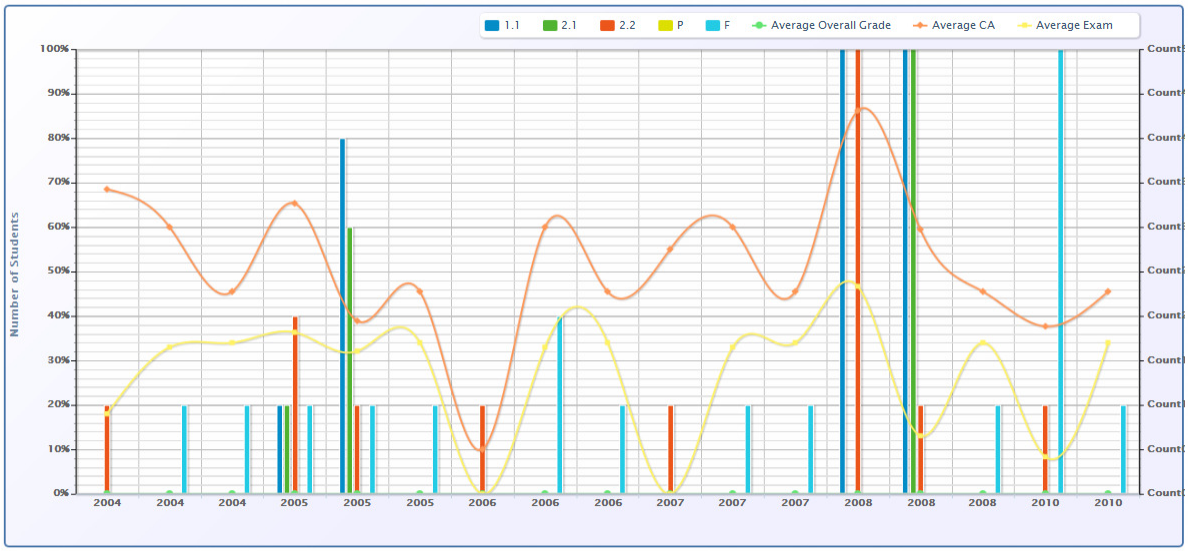


Figure 100 Student Overview Dual Axis Grade Combination Chart

This kind of combination plot built on the techniques discovered in “Academic Performance at Simon Fraser University” (36) in particular Figure 43 and developed them to the next level. Two axes are provided, one axis is for percentage values and the other is for failure count.

Using two axis here makes it easier to compare both data sets as it is very difficult to compare counts to percentages on the same axis.

The final chart provided in this section is a yearly grade distribution area chart that uses colour coding to separate the percentage Y axis into sections to represent each qualification level. This is better explained by Figure 101 below.

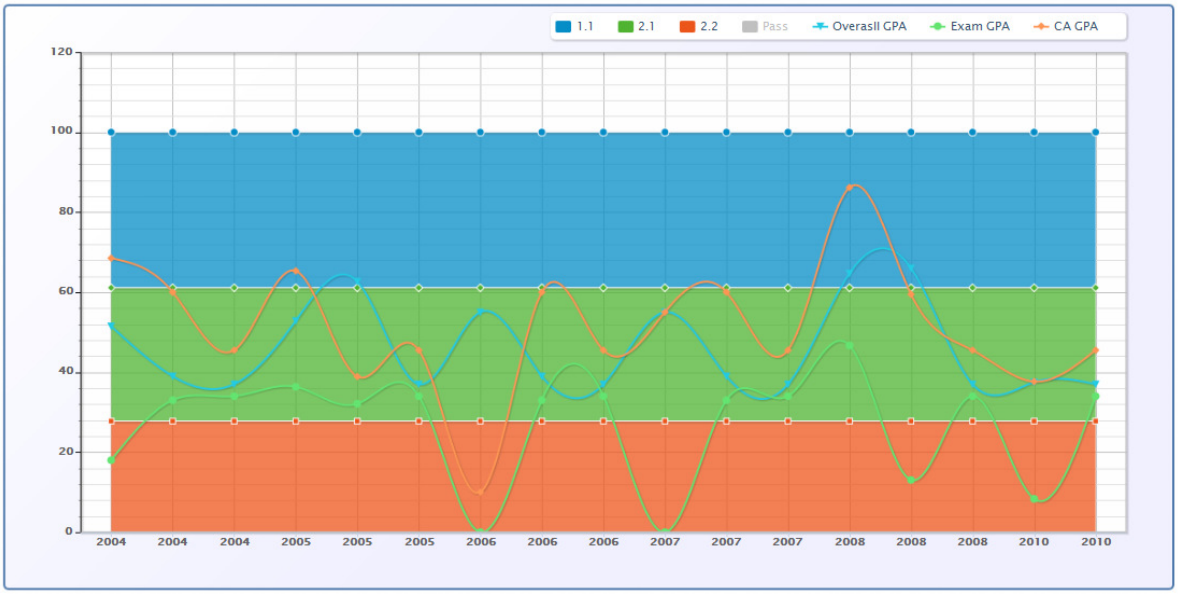


Figure 101 Student Overview Yearly Grade distribution Area Chart

The colour coding is based on the university grading scale and the blue section at the top represents what would be required to achieve a first class honours. The green section below this indicates what is required to obtain a second class honours qualification and the red section at the bottom indicates a failing grade.

By dividing the visualisation into these sections an using trend lines to plot GPA averages on top a clear student progress pattern can be observed and it is easy to see if their grades are steadily declining. The idea for this sort of classification was provided by the classification done in ” Determination of factors influencing the achievement of the first-year university students using data mining methods” (49).and the main inspiration in this paper was Figure 49 in this document. The grade deviation tab of the student analysis page re-uses these visualisations to view its own data in just as an effective manner.

The motion chart tab on the student analysis page provides a large screen motion chart that uses every metric calculated for display in other graphs. Motion charts have a huge number

of filtering option and by providing this chart with all the data available you are enabling the user to perform pretty much any analysis that's possible using the dataset.

The yearly course performance tab provides a large sized combination chart that plots student GPA averages and percentage modules failed using columns and an overall GPA average trend line to show the general distribution of grades. It also incorporates a small pie chart which gives average grade per calendar year rather than by semester which is how the columns are drawn. This visualisation is shown in Figure 102.

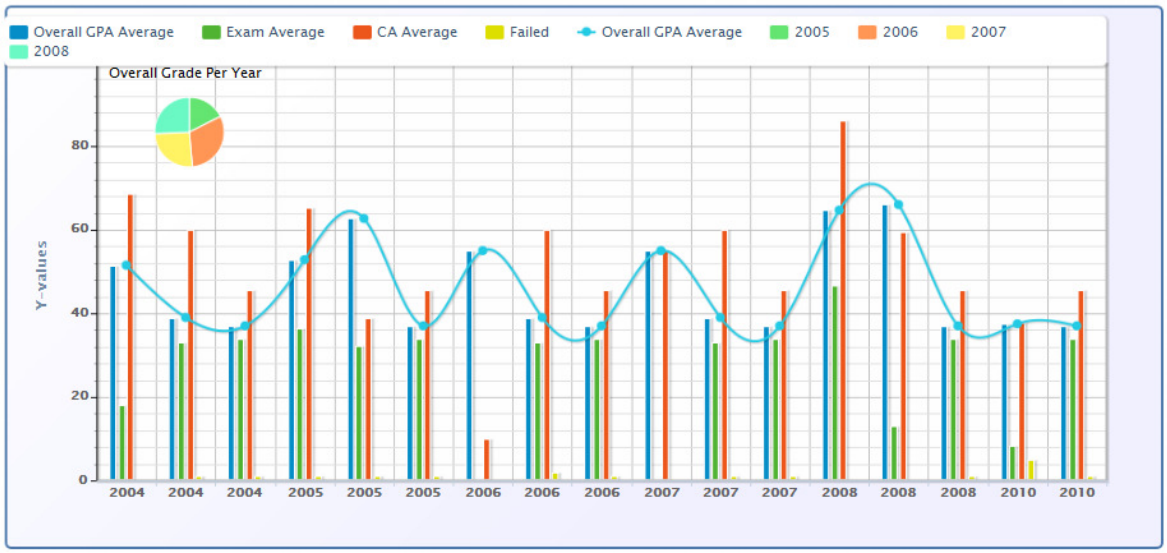


Figure 102 Student Yearly Performance Combination Chart

Another frequency distribution column chart is used in the GPA Vs. Fails tab (Figure 103) to show the relationship between the count of modules failed per semester and overall GPA average.

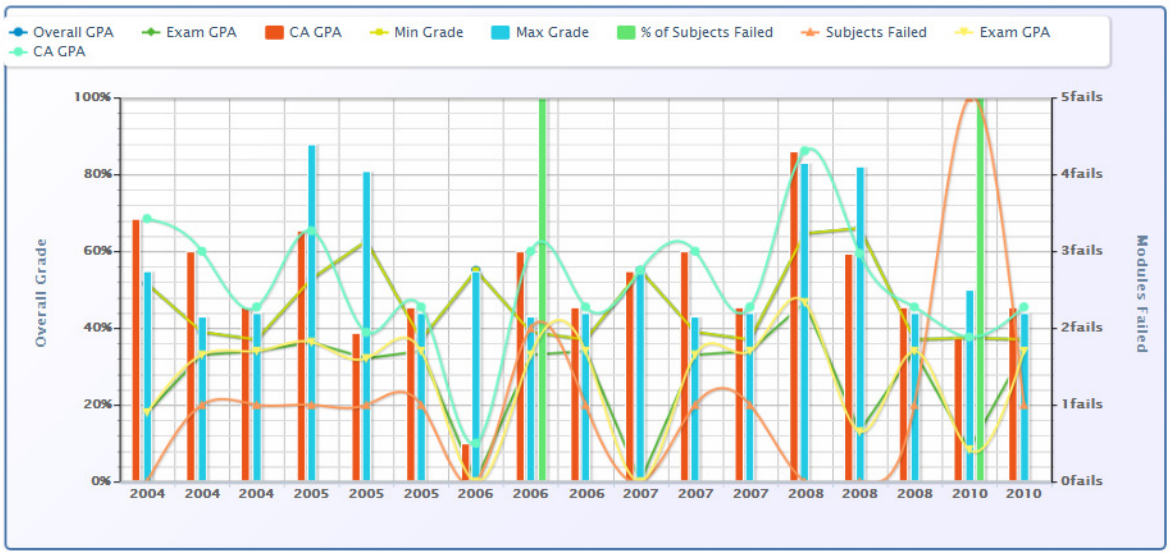


Figure 103 Student Analysis GPA Vs Fails

A useful way of viewing the student's entire academic history in a single tab is provided using the jQuery Data Table plug-in in the Result of All Modules Tab. Every module the student has ever taken is listed here along with the module code, description CA mark, exam mark, overall mark, result and exam type so as much information about eah module is provided along with the grade results.

Another visualisation provided in the student analysis areas is a standalone Drastic Tree Map that displays all of the students' module results grouped by year and filterable by Overall GPA, Exam GPA and CA GPA and the chart in Figure 80 is the same as what is used here. The regression colour scatter available in the student analysis area was also previously discussed and a screen shot can be viewed in Figure 83. The other visualisation metrics available on this page are the Visifire Area Chart , Google Parallel co-ordinate chart and the HTML5 canvas scatter line, all of which have been discussed in depth in previous sections.

5.5.2 Module Analysis:

The module analysis section uses an almost identical layout to that of the student analysis area and the only major differences are the data being returned to the visualisations. The first tab here also displays up to date information about the current semester and the heat colour table here displays the results of all students who have taken the module in the most recent semester and this is illustrated in Figure 104.

Student ID	Gender	Year	CA %	EM %	AM %	Result	Exam Type
53431274	M	2010	35	0	35	F	N
53431275	M	2010	35	0	12	F	N
53431276	F	2010	35	0	5	F	N
53431278	M	2010	35	0	25	F	N
53431280	M	2010	35	0	35	F	N
53431282	F	2010	35	0	35	F	N
53431283	M	2010	35	0	35	F	N
53431284	M	2010	35	0	90	F	N
53431285	F	2010	35	0	31	F	N
53431286	M	2010	35	0	11	F	N

Figure 104 Module Analysis current semester results

The KPI listing used in the accordion side-bar menu of the module analysis page contains the following KPI's:

Module Code: EE110

Started: 2010

Average Grade: 40.58 %

Average Exam Mark: 40.58 %

Average CA Grade: 40.63 %

Average Drop Outs Per year: 50

Count of 1.1:10%

Count of 2.1: 15%

Count of 2.2: 15%

Count of Pass: 40%

Count of Failure: 20%

Count students who have ever taken module:24

Worst ever exam mark:0%

Worst ever CA mark: 35%

Worst ever overall mark: 5%

Best ever exam mark: 0%

Best ever CA mark: 50%

Best ever overall mark: 90%

These KPI's can provide a quick overview to see how a module is performing basing on these key analysis metrics.

The regression analysis plot performed here is the exact same as that performed in the student analysis area except the GPA Vs CA averages across the entire module are used to calculate the regression equation. The same can be said of the grade distribution and deviation tabs which analyse the same type of data but on the basis of the particular module. The GPA Vs Dropout and the Colour Scatter also have exactly the sae format but use module data rather than individual student data. The Drastic Tree Map is the most totally bespoke visualisation in this section and it provides a view of grade distribution amongst individual students for the module in the current semester.(Figure 105)

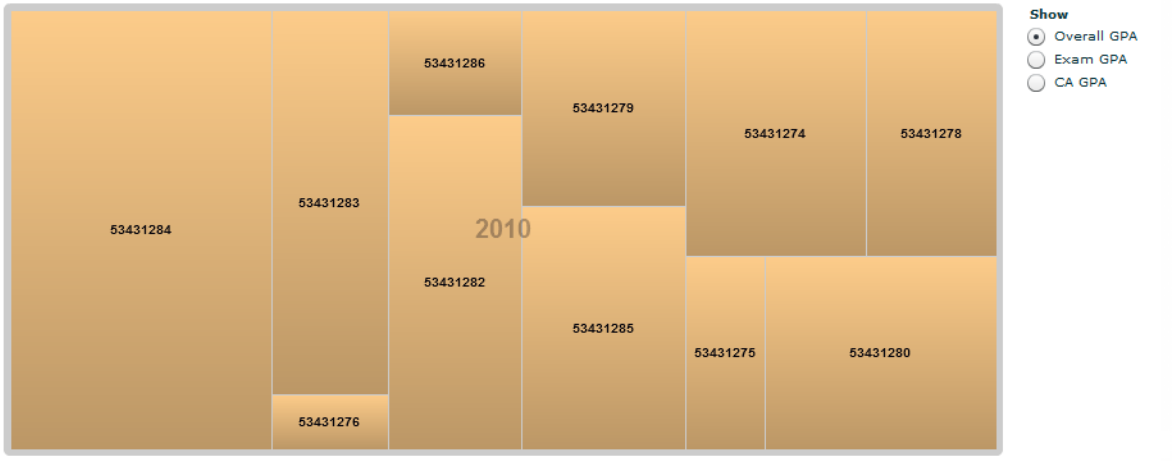


Figure 105 Module Analysis Drastic Tree Map

More ideas for additional bespoke visualisations in this section were planned but there was insufficient time to implement them and individual academic analysis was the research priority so more focus was paid to that area.

5.5.3 Course analysis:

The course analysis section again follows the same template as both the module and student visualisation areas and offers the same visualisations but tailored to a specific course. The main differences are in the metrics displayed in the KPI side menus, a full listing of these are provided in bullet points below with sample value outputs:

- **Course:** DME
- **Started:** 2007
- **Average Grade:** 55%
- **Average Exam Mark:**60%
- **Average CA Grade:** 70%
- **Average Drop Outs Per year:** 6
- **Average % of 1.1:** 10%
- **Average % of 2.1:** 40%
- **Average % of 2.2:** 20%
- **Average % of Pass:** 10%
- **Average % of Failure:** 20%

5.6 Problems encountered during the development phase

Several issues were encountered during the development phase and are discussed in detail in the sections below in the hope this may be of aid to others continuing research in this area.

5.6.1 Issues with servlet load time:

Issues were encountered with the initial loading time of the servlet due to the way the application is designed and the repeated connections to the database. This places a big load on the database when a call is initially made to one of the analysis pages and it can take some time to generate the page. This can cause issues with the display and causes some jQuery dialog div tag contents to become visible before the application has been loaded as shown in Figure 106

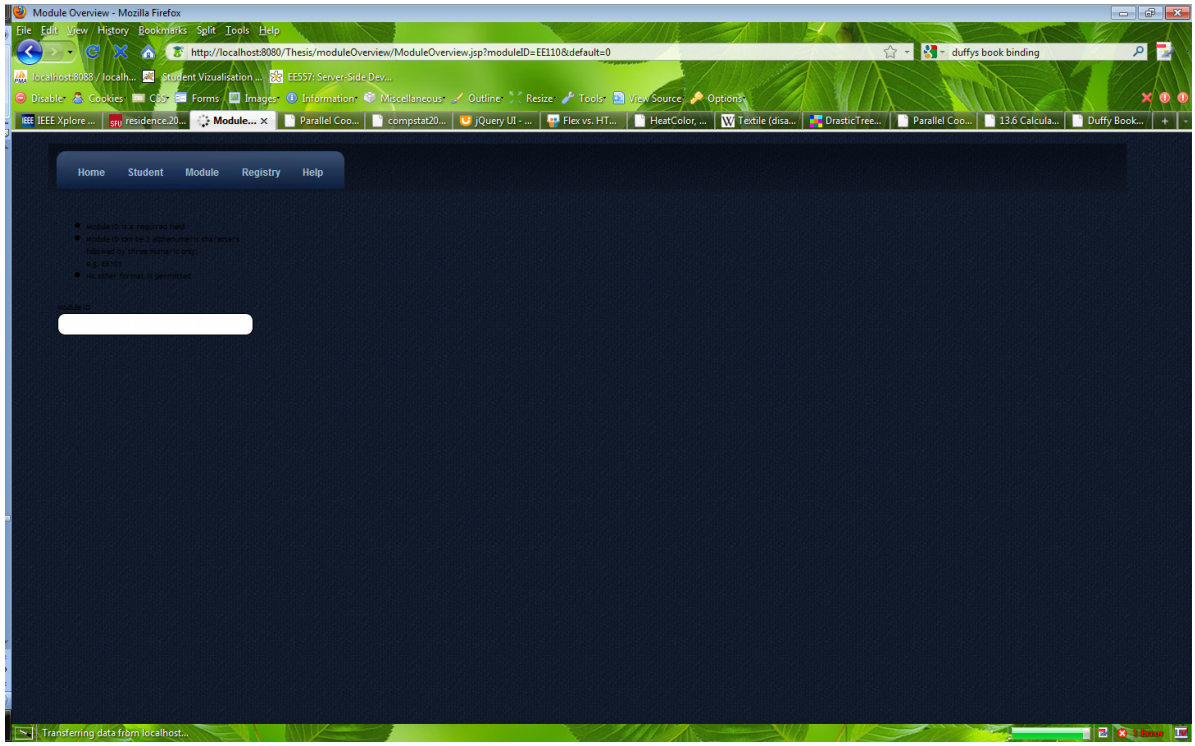


Figure 106 Problems caused by slow database performance

A lot of research was done into ways of improving the load time by using pooled connections however as the live student database is an Oracle database it shouldn't experience as many problems and should be able to handle multiple connections better. None the less some excellent background reading on mySQL connection pooling can he found on the MySQL website (90) and some information on improving the performance of servlets by re-using database objects can be found in the Java Servlet Programming manual (91).

5.6.2 Issues with jQuery dialog displaying some charts:

Some visualisations had unexpected behaviour when embedded in jQuery dialogs and the motion chart or Drastic Tree Map are not suitable for use in dialogs as once opened they cannot be closed without refreshing the page. Another problem found with the jQuery dialog is that if there are existing Visifire charts in the foreground of the page the dialog will not be able to open properly and will be hidden behind the chart.

5.6.3 Problems with bizarre behaviour of jQuery:

jQuery can be quite sensitive and although the syntax is easy to understand if a mistake is made it can be very difficult to find the problem. If a div tag is accidentally deleted for one accordion or tab the whole page will have problems and the layout will be all over the place. Problems are also caused if charts are called in the application but their JavaScript file is not referenced. Rather than display a blank graph with the rest of the application intact the whole application can seem corrupted and it can be very difficult to get to the root of the issue.

5.6.4 Over-reliance of Google Visualisations on the internet:

Google visualisations rely on an internet connection to operate as access to the google code repository is needed however this can be quite frustrating if the internet speed in your area is particularly poor and it takes some time to load visualisations. If you have a couple of visualisations in your page and a slow connection speed you have little chance of getting your application to load as usually the flash initialise for the chart will time out before all of the data is loaded.

5.6.5Issues encountered trying to plot visualisations using raw html files:

A lot of issues were encountered initially with servlets trying to produce JavaScript using sever side includes and ordinary HTML files. Websites work by loading and running all JavaScript, then producing HTML based on that. The main issue was that when trying to include a servlet producing JavaScript a servlet was needed to produce nearly a whole HTML page for it to work. This was as it would load the page, run the JavaScript, find the

SSI and run it, adding the resulting JavaScript AFTER the whole page was loaded. So the visuals wouldn't work because the data wasn't loaded on time. This problem was eliminated by using JSP's to produce the application pages as opposed to raw HTML files and this resolved the problem immediately. There were still some issues though as the Eclipse IDE highlights any other code tags written inside of JavaScript as erroneous and underlines them to indicate a syntax error even though the code itself will actually work perfectly.

5.7 Conclusion

This concludes the application section which has uncovered the inner workings of the application and discussed the various existing bugs and flaws. Further source code examples are available in the appendices as a point of reference and the attached CD contains a fully functional sample web application in the form of a WAR file and a mySQL database. The WAR file should be deployed to a J2EE Apache Tomcat web server and the mySQL database can be launched from the CD using the included USB web server package.

After development was completed it was clear that there was a lot of room for improvements in the application and these are discussed in further detail in Chapter 6 - Testing and strengthening of the application. Various issues were encountered and they were documented along with their solutions where solved to increase the universal merit of the project. This is to provide help to others trying to implement visualisation technologies along with information to others interested in academic data analysis

Chapter 6 - Testing and strengthening of the application

The application was tested on two fronts to ensure both portability and to verify security. Cross platform independence was examined thoroughly as a key part of any application and particularly a web application such as this is to ensure interoperability across platforms and browsers. There is no way of knowing which browser or platform will be chosen to run the application in future so it is best to ensure as much platform independence as possible.

The security of this application was also examined in detail due to the sensitivity of the information being analysed. As it stands the application is designed to be integrated into an existing application which is already well secured so it was not a major concern of this project. It is however always a good idea to ensure an application is as secure as possible so this area was examined to ensure thoroughness of testing.

6.1 Cross platform testing

Once the application was built the web front end was tested in a variety of browsers including:

- Microsoft Internet Explorer IE,
- Firefox,
- Chrome,
- Opera,
- Safari.

The application was tested on 4 different operating systems including Windows XP, Windows Vista, Windows 7 and Ubuntu. It is extremely important to ensure platform dependence for any application especially with the constantly changing technological world. If there had been more time it would also have been tested on an iPhone, Apple Macintosh and other portable devices. In the case of this application itself this is not necessary as people won't be using mobile devices to analyse student devices. However with the advent of apples new iPad this could happen and it could be used as a portable way of viewing real time student information in the class room.

jQuery and the jQuery UI are fully cross browser compatible and this is tested and verified by the jQuery development team themselves. This takes a lot of stress and work out of web application development as the developer can be sure interfaces built using the jQuery UI will not experience any cross browser issues. When building these types of interfaces from scratch ensuring cross browser compatibility can prove a frustrating and tedious task and the fact this can be avoided by using the jQuery UI is another huge advantage to be obtained from going this route.

Cross platform issues:

The jQuery top menu used only works in Firefox and this is a big flaw in the application. It is an excellent navigation system when used within Firefox however it is not acceptable for use commercially if it does not work in other browsers. Other menu's were trialled during the development phases and none were found to be as high quality and overall browser compatible as this one so the decision was made to use it anyway even if it didn't work correctly in one out of 5 browsers tested. The focus of this project was from a research aspect rather than a development aspect and to move forward with the project as a commercial enterprise this menu would have to be replaced.

As of yet IE does not support the HTML5 canvas tag so High Charts chart's or HTML5 canvas graphs built from scratch will not work at all in IE. A work around for this is available with the inclusion of a java-script canvas emulator file – excanvas.js. If the browser is detected as a flavour of IE this script is used to plot the canvas to draw the charts on and the issue is avoided. This issue is likely to be eliminated soon as due to the lack of support for new web technology within internet explorer it's seeing a dramatic drop in usage figures. The vast majority of home computer users use a windows system which comes with internet explorer bundled as a default so the drop in usage is even more telling.

Users are going out of their way to download another web browser due to their frustration with internet explorer and to amend this Microsoft will have to enable canvas support along with various other technologies specified by the W3C web standards consortium. The application developed as part of this project is built to web consortium standards rather than specific configurations of current browsers as this is what will really indicate the future of web development and browsers will be forced to conform to standards or see a decrease in market share.

6.2 Security issues

The security of the application could be improved by using the java cryptography extension to encrypt and decrypt the variables used for session data and for URL re-writing. There is currently minimal security as regards the data being sent to and from the server. The application only performs select queries on the database so to further increase security a special database user should be set up with read only database privileges.

The search forms are the only place user input is allowed and with the validation on these both client and server side to except only alphanumeric characters the risk of SQL injection through these forms is minimised. SQL injection can occur when a hacker inserts code or database queries into submission forms. Protection against this should be three fold and validation should occur in the client browser, at the web server and in the database itself to ensure it's as difficult as possible for an intruder to break through.

6.2 Conclusion

Issues were discovered during the testing of this application and had more time been available and more focus been placed on the application rather academic analysis research suitable alternatives could have been sought. As it stands the application has some flaws but when viewed using a web standards compliant browser such as Mozilla Firefox it runs very well and as most browsers will be forced to meet the standards sooner rather than later it suffices for this project.

The security of the application could also be improved a lot but the same could be said of any application. There is never too much security for a web application or no room for further strengthening and this is something that would need to be researched in further detail if this application is to be developed further into a commercial product. From the perspective of integrating the application into a pre-existing, secured system the security as is would suffice for these purposes.

Chapter 7- Conclusion and Discussion of results

There were several major challenges faced during the course of this project and there were many positive outcomes of the research done. A lot of useful techniques for analysing academic performance of students were discovered with the stand out prediction method being the user of Linear Regression analysis. Linear Regression analysis allows for prediction of any type of data value based on the state of another value. It is used extensively in many areas of research and can be of just as much use in an academic analysis setting as anywhere else.

A lot of benefit was gathered from the research performed into new web technologies and the corner stone of the web application for this project became the High Charts visualisation library due to its high level of interactivity and ease of use. Another key benefit of High Charts is that it uses the HTML 5 canvas tag to display the graphs and no flash is required. The charts are cross browser compatible and offer far more customizability than Google Visualisation charts. The research into user interface technologies also proved to be extremely useful and jQuery really played a big part in the development of this project. It is a technology that is not as well known as it should be and is something that will be used almost everywhere on the web in the near future.

The research aspect of this aspect of this project also provided several useful ideas of initiatives that could be set up in DCU to help aid students and reduce the drop-out rates. If a remedial program was set up to give additional programming lessons to first year engineering student it could result in a marked improvement in their performance at first year exam level. Similarly if the entry requirements to a course were changed the impact of this could be evaluated using this application to see if or not it decreased the overall average GPA or standard deviation of grades between classmates.

If this application was integrated further into other data sources such as a module change database it would enable the analysis of the university from a business perspective. This could aid prioritisation of resources and profit overall helping to increase the profile of the university and the quality of students produced. This kind of detailed analysis is common in most businesses now and can play a vital role in the key decisions made. Learning more

about students and how they perform and why can be of great help to the university and needs to be seen as high priority.

A lot more analysis could have been drawn from if more biographical and personal details about students were available and it would be a good idea for DCU to begin to start gathering this type of information for future use. With more data available for analysis about students the accuracy of predicted results increases and more patterns in behaviour and performance can be spotted as there is more data to correlate together.

After research was performed into other student academic data analysis techniques it was found that while there are applications available to do this currently the technology they use is outdated and there is room in the market for a student academic performance analysis using new web technologies and this project could have the potential to be developed into a software application to be sold to secondary schools and universities.

A lot was learned during this project both from a research point of view and a development point of view and a very comprehensive review of student data performance analysis literature and techniques was performed that could be taken further by other researchers in future. A lot of mistakes were made but documenting them will hopefully prevent others from making the same ones in future and the mistakes made were undoubtedly key to the overall learning throughout the project.

References

1. Google Visualisation API Code Source. [Online] [Cited: 18 08 2010.] http://www.google.com/jsapi.

2. Visualization: Annotated Time Line. *Google Code.* [Online] http://code.google.com/apis/visualization/documentation/gallery/annotatedtimeline.html.

3. **Google.** Visualization: Motion Chart. *Visualization: Motion Chart.* [Online] [Cited: 18 08 2010.] http://code.google.com/apis/visualization/documentation/gallery/motionchart.html.

4. **Minder, Gap.** Gap Minder Google Motion Chart. *Gap Minder.* [Online] [Cited: 18 08 2010.] http://www.gapminder.org/.

5. **Google.** Google Chart. *Google Chart API's.* [Online] [Cited: 18 08 2010.] http://www.wait-till-i.com/2008/01/08/generating-charts-from-accessible-data-tables-using-the-google-charts-api/ .

6. **Google** Visualization: Area Chart. *Google.* [Online] [Cited: 18 08 2010.] http://code.google.com/apis/visualization/documentation/gallery/areachart.html.

7. **Google** Visualization: Bar Chart. *Google.* [Online] 18 08 2010. [Cited: 18 08 2010.] http://code.google.com/apis/visualization/documentation/gallery/barchart.html.

8. **Google** Google Code Playground. *Google Code Playground.* [Online] [Cited: 18 08 2010.] http://code.google.com/apis/ajax/playground/.

9. **MooTools .** MooTools - a compact JavaScript framework. *MooTools* . [Online] http://mootools.net/.

10. **Murray R. Speigel, John Schiller, R. Alu Srinivasan.** *Prabability and Statistics, Third Edition.* s.l. : Mc Graw Hill, Schaum's Outlines Series. ISBN 978-0-07-154425-2.

11. **Rumsey, Deborah.** *Statistics For Dummies.* s.l. : Rumsey. ISBN - 978-0-7645-5423-0.

12. *Statistical data: The underestimated tool for higher education Management.* **Nakabo-Ssewanyana, Sarah.** Kampala, Uganda : Kluwer Academic Publishers, 1999, Vols. 37: 259–279.

13. **Cook, Graham Upton and Ian.** *The Oxford Dictionary Of Statistics.* Oxford : Oxford University Press, 2008. ISBN 978-0-19-954145-4.

14. O'Reilly Transact SQL Cookbook. [Online] [Cited: 18 08 2010.] http://oreilly.com/catalog/transqlcook/chapter/ch08.html.

15. **Larry D. Schroeder, David L. Sjoquist, Paula E. Stephan.** *Understanding Regression Analysis - An introductory Guide.* Newbury Park, London : Sage Publications, 1996.

16. **Analysis, Discriminant.** Applications of Discriminant Analysis. *Wikipedia.* [Online] [Cited: 18 08 2010.] http://en.wikipedia.org/wiki/Discriminant_analysis#Applications.

17. Probability Theory. *Wikipedia.* [Online] [Cited: 18 08 2010.] http://en.wikipedia.org/wiki/Probability_theory.

18. **Wilkins, Britney.** 50 Google Charts Tricks for Your Next Classroom Presentation. *College@Home.* [Online] [Cited: 18 08 2010.] http://www.collegeathome.com/blog/2008/06/05/50-cool-things-you-can-do-with-google-charts-api/ .

19. **Chart, Google Chart API Embedded.** Google Chart API Embedded Chart. *Google Chart API.* [Online] [Cited: 18 08 2010.] http://code.google.com/apis/chart/docs/chart_params.html#embedded_charts.

20. **Visualisations, Visifire.** Visifire Visualisations - Silverlight & WPF Charts. *Visifire Visualisations.* [Online] 2010. [Cited: 18 August 2010.] www.visifire.com.

21. High Charts home Page. *High Charts.* [Online] [Cited: 18 08 2010.] www.highcharts.com.

22. amCharts. [Online] [Cited: 18 August 2010.] http://www.amcharts.com/.

23. aXiis Open Source Data Visualisation. [Online] aXiis. [Cited: 18 August 2010.] http://www.axiis.org/.

24. **Axiis.** Axiis Examples. *Axiis.org.* [Online] [Cited: 18 08 2010.] http://axiis.org/examples/.

25. **R-Graph.** RGraph: HTML5 canvas graph library based on the HTML5 canvas tag. *R-Graph.* [Online] [Cited: 18 08 2010.] http://www.rgraph.net/.

26. **Neuberg, Brad.** *Introduction to HTML 5.* s.l. : Developer Programs, Google, 2009.

27. **MugTug.** MugTug Sketch Pad. *MugTug .* [Online] [Cited: 18 08 2010.] http://mugtug.com/sketchpad.

28. ColorJack Blog Page. *Color Jack.* [Online] [Cited: 18 08 2010.] http://www.colorjack.com/blog.

29. HTML 5 Presentation. *Scribd.* [Online] [Cited: 18 08 2010.] http://www.scribd.com/doc/19111526/HTML5-Presentation.

30. **Sevarac, Zoran.** *Neuro Fuzzy Reasoner for Student Modeling.* University of Belgrade, Serbia and Montenegro : Department of Information Systems, School of Business Administration FON,, 1999.

31. *THE LEAVING CERTIFICATE AND FIRST YEAR UNIVERSITY.* **M. A. Moran, M. J. Crowley.** Part 1,197 8/7 9, pp 231-266., Cork : Journal of the Statistical and Social Inquiry

Society of Ireland, Department of Statistics, University College, Cork, 1979, Vols. Vol. XXIV,.

32. *An assessment of the extent to which subject variation between the Arts and Sciences in relation to the award of a First Class degree can explain the 'gender gap' in UK universities.* **Woodfield, Ruth and Earl-Novell, Sarah.** 3, 355 — 372, s.l. : British Journal of Sociology of Education, July 2006,, Vols. 27, .

33. *Research on Integrated Performance Assessment at the Post-Secondary Level: Student Performance Across the Modes of Communication.* **Adair-Hauck, Eileen W. Glisan Daniel Uribe Bonnie.** 1 (September/septembre), 39–68, s.l. : The Canadian Modern Language Review/La Revue canadienne des langues vivantes, 2007, Vols. 64, .

34. *The Association of Student Examination Performance with Faculty and Resident Ratings Using a Modified RIME Process.* **Charles H. Griffith III, MD, MSPH1,2 and John F. Wilson, PhD1.** University of Kentucky, Lexington, KY, USA : Society of General Internal Medicine, 2008.

35. *Student Performance Online Vs. Onground:A Statistical Analysis of IS Courses.* **Ury, McDonald, McDonald, and Dorn Sat.** Proc ISECON 2005, v22 (Columbus OH): §3162 (refereed) c 2005 EDSIG, Atlanta, GA : Georgia Institute of Technology, 2005.

36. **Joanne Heslop, Senior Analyst Jessica Tilley,.** Academic Performance at Simon Fraser University. [Online] 2006. [Cited: 18 08 2010.]
http://www.sfu.ca/irp/special_reports/documents/residence.2006.pdf.

37. Google Visualization: Map. *Google Visualization: Map.* [Online] Google.
[Cited: 18 08 2010.]
http://code.google.com/apis/visualization/documentation/gallery/map.html.

38. **Smith, Jeremy, Naylor, Robin.** Determinants of Degree Performance in UK Universities: A Statistical Analysis of the 1993 Student Cohort. [Online] Article provided by Department of Economics, University of Oxford in its journal Oxford Bulletin of Economics & Statistics., February 2001. [Cited: 18 08 2010.]
http://ideas.repec.org/a/bla/obuest/v63y2001i1p29-60.html.

39. **Rosovsky, Henry.** *The University An Owners Manual.* New York : W.W. Norton & Company Inc., 1990. ISBN 0-393-30783-2.

40. **Jens Henrik Haahr, Thomas Kibak Nielsen, Martin Eggert Hansen, Søren Teglgaard Jakobsen.** Explaining Student Performance - Evidence from the international PISA, TIMSS and PIRLS surveys. *European Commission - Education and Training.* [Online] November 2005. [Cited: 18 08 2010.]

http://ec.europa.eu/education/pdf/doc282_en.pdf.

41. **Thompson, Franklin T.** Student Achievement, Selected Environmental Characteristics and Neighborhood Type . *Springer Link.* [Online] 01 09 2002. [Cited: 18 08 2010.] http://www.springerlink.com/content/t4854w17121142q6/.

42. Statistical Analysis Of Performance Of Learning Disabled Students. *http://etd.lib.ttu.edu/theses/available/.* [Online] December 2006. [Cited: 18 08 2010.] http://etd.lib.ttu.edu/theses/available/etd-11272006-134251/unrestricted/Jain_Sonal_Thesis.pdf.

43. **Blaženka Divjak, Dijana Oreški.** Prediction of Academic Performance Using Discriminant Analysis. *IEEE Xplore Digital Library.* [Online] 22-25 June 2009 . [Cited: 18 08 2010.] http://ieeexplore.ieee.org/Xplore/login.jsp?url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F5174538%2F5196028%2F05196084.pdf%3Ftp%3D%26arnumber%3D5196084%26punumber%3D5174538&authDecision=-203.

44. **Olani, Aboma.** Predicting First Year Students' Academic Success. [Online] 2008. [Cited: 18 August 2010.] http://www.investigacion-psicopedagogica.org/revista/articulos/19/english/Art_19_376.pdf.

45. **Esteve, Edgar Breso.** *Well-being and Performance in Academic Settings. The Predicting Role of Self-efficacy.* s.l. : Spanish Ministry of Science and Technology , 2008.

46. **Gallacher, Marcos.** Marcos Gallacher. *Predicting Academic Performance.* [Online] [Cited: 18 August 2010.] http://www.aaep.org.ar/espa/anales/works05/gallacher.pdf.

47. **Luna, Julie.** Predicting Student Retention and Academic Success at New Mexico Tech. [Online] August 2000. [Cited: 18 August 2010.] http://euler.nmt.edu/~brian/students/julie.pdf.

48. **Nguyen Thai Nghe, Paul Janecek, and Peter Haddawy.** A Comparative Analysis of Techniques for Predicting Academic Performance. [Online] 10-13 October 2007. [Cited: 18 August 2010.] http://www.iist.unu.edu/www/homepage/haddawy/haddawy-pub-57.pdf.

49. **J.F. Superby, J.-P. Vandamme, N. Meskens.** Determination of factors influencing the achievement of the first-year university students using data mining methods. [Online] 2005. [Cited: 18 August 2010.] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.101.4916&rep=rep1&type=pdf.

50. **Cantwell, Robert, Archer, Jennifer and Bourke, Sid.** A Comparison of the Academic Experiences and Achievement of University Students Entering by Traditional and Non-Traditional Means. [Online] 2001. [Cited: 18 August 2010.]

http://www.eric.ed.gov/ERICWebPortal/search/detailmini.jsp?_nfpb=true&_&ERICExtSearch_SearchValue_0=EJ629709&ERICExtSearch_SearchType_0=no&accno=EJ629709.

51. **McCormick, Nawarut Charupatanapong and William C.** Predicting Academic Performance of Pharmacy Students: Demographic Comparisons. [Online] Fall 1994. [Cited: 18 August 2010.] http://www.ajpe.org/legacy/pdfs/aj5803262.pdf.

52. *Effects of Peer Tutoring, Attitude and Personality on Academic Performance of First Year Introductory Programming Students.* **Golding, Lisa Facey-Shaw and Paul.** Kingston, Jamaica : 35th ASEE/IEEE Frontiers in Education Conference, 2005.

53. **Smyth, Delma Byrne and Emer.** *NO WAY BACK? - The Dynamics of Early School Leaving.* Dublin : The Liffey Press in association withThe Economic and Social Research Institute, 2010. ISBN 978-1-905785-80-3.

54. **O'Leary, John.** *Good University Guide 2010.* London : Times Books, 2010. ISBN 978-0-00-731348-8.

55. **Miron, G. & Applegate, B.** Review of Multiple Choice: Charter School Performance in 16 States. [Online] 24 June 2009. [Cited: 18 August 2010.] http://epicpolicy.org/thinktank/review-multiple-choice.

56. **AlmaLaurea.** Academic performance at the reformed university. [Online] 18 August 2010. [Cited: 18 August 2010.] http://www.almalaurea.it/en/universita/profilo/profilo2006/pdf/chapter8.pdf (for other chapters replace 8 wht the chapter number).

57. **Duff, Angus Professor.** Understanding academic performance and progression of first-year accounting and business economics undergraduates: the role of approaches to learning and prior academic achievement. *InformanWorld.* [Online] 2004. [Cited: 18 August 2010.] http://www.informaworld.com/smpp/content~db=all~content=a713949890~frm=titlelink. DOI: 10.1080/0963928042000306800 .

58. **ePortal.** ePortal. [Online] 18 August 2010. [Cited: 18 August 2010.] http://eportal.sourceforge.net/eng_index.html.

59. **ERSAN, DERVIS Z. DENIZ and IBRAHIM.** An Academic Decision-Support System Based on Academic Performance Evaluation for Student and Program Assessment. [Online] 2002. [Cited: 18 August 2010.] http://www.ijee.dit.ie/articles/Vol18-2/IJEE1274.pdf.

60. Grange Community School. [Online] [Cited: 18 08 2010.]
http://www.google.ie/url?sa=t&source=web&cd=1&ved=0CBMQFjAA&url=http%3A%2F%2Fwww.grangecc.ie%2F&ei=dF5tTODsEMLc4AbW4Oy9Cw&usg=AFQjCNEgvJHfGWt9URi9vfhSXacLYKVHdw&sig2=cK5FSx5EnYELLqGv_BWpbA.

61. **Ann M, Urbinu, Mario R. Ledn de Ea Barra, Guiliermo E. Len de la Barra.** A Remedial Program For First Year Engineering Students. *IEEE Xplore.* [Online] 10-13 November 1999. [Cited: 14 May 2010.] http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=839165.

62. **Fox, Pamela.** Flex Vs. HTML5 for RIAs. *www.slideshare.net.* [Online] 2009. [Cited: 18 08 2010.] http://www.slideshare.net/wuzziwug/flex-vs-html5-for-rias-presentation.

63. **jQuery.** jQuery - The Write Less Do More JavaScript Library. *jQuery Home.* [Online] [Cited: 18 08 2010.] www.jquery.org.

64. —. jQuery UI ThemeRoller. *jQuery UI.* [Online] [Cited: 19 08 2010.] http://jqueryui.com/themeroller/.

65. **Google.** Google Code Libraries. *Google Code Libraries.* [Online] [Cited: 18 08 2010.] http://code.google.com/apis/libraries/.

66. **jQuery.** jQuery UI Demo Page. *jQuery UI.* [Online] [Cited: 18 08 2010.] http://jqueryui.com/demos/ .

67. **Sharkie, Earle Castledine & Craig.** SitePoint. *jQuery: Novice To Ninja.* [Online] [Cited: 08 August 2010.] http://www.sitepoint.com/books/jquery1/.

68. **The Now Factory.** [Online] [Cited: 18 08 2010.] http://www.thenowfactory.com/.

69. **Krug, Steve.** *Don't Make Me Think.* [ed.] Second Edition. Berkeley, CA : New Riders Publishing, 2006.

70. **Apycom.** jQuery CSS Menu. *apycom menus.* [Online] [Cited: 18 08 2010.] http://apycom.com/menus/6-midnight-blue.html.

71. Drastic Tree Map. *drasticdata.nl.* [Online] Drastic Data, 2010. [Cited: 18 August 2010.] http://www.drasticdata.nl/DrasticTreemapGApi/index.html.

72. **Google.** Google Tree Map. [Online] [Cited: 18 August 2010.] http://code.google.com/apis/visualization/documentation/gallery/treemap.html.

73. **Jeremy J. Foster, Ian Parker.** Carrying out Investigations in Psychology - Methods and Statistics . *Carrying out Investigations in Psychology - Methods and Statistics* . s.l. : The British Psychological Society, 1995, pp. 136-164.

74. **Natsuhiko KUMASAKA, Ritei SHIBATA.** Implementation of Textile Plot. [Online] 2006. [Cited: 18 August 2010.]
http://www.stat.math.keio.ac.jp/kumasaka/slides/compstat2006.pdf.
75. jQuery DataTables Plugin. [Online] [Cited: 18 August 2010.]
http://www.datatables.net/.
76. HeatColor, a jQuery plugin. *jQuery Heat Color Home Page.* [Online]
[Cited: 18 August 2010.]
http://www.jnathanson.com/blog/client/jquery/heatcolor/index.cfm.
77. **Newman, Jamie.** How to Draw with HTML5 Canvas. *http://thinkvitamin.com.* [Online] [Cited: 18 08 2010.]
http://thinkvitamin.com/dev/html-5-dev/how-to-draw-with-html-5-canvas/.
78. **http://diveintohtml5.org/.** Let's Call It a Draw(ing Surface) - Dive Into HTML 5. *http://diveintohtml5.org/.* [Online] [Cited: 19 08 2010.]
http://diveintohtml5.org/canvas.html.
79. Firebug Web Development Plugin. [Online] [Cited: 18 08 2010.]
http://getfirebug.com/.
80. **Molloy, David.** EE557: Server-Side Development. *EE557: Server-Side Development* . [Online] [Cited: 18 08 2010.] http://wiki.eeng.dcu.ie/ee557/396-EE.html.
81. Oracle Express Edition. [Online] [Cited: 18 August 2010.]
http://www.oracle.com/technology/products/database/xe/index.html.
82. WAMP Web server. [Online] [Cited: 18 August 2010.]
http://www.wampserver.com/en.
83. phpMyAdmin. [Online] [Cited: 18 August 2010.]
http://www.phpmyadmin.net/home_page/index.php.
84. Skype. [Online] www.skype.com.
85. USB Webserver. *USB Webserver.* [Online]
[Cited: 19 August 2010.] http://www.usbwebserver.nl/.
86. MySQL Work Bench. [Online] [Cited: 18 August 2010.] http://wb.mysql.com/.
87. **Oracle.** Oracle SQL Developer. [Online]
[Cited: 18 August 2010.] http://www.oracle.com/technetwork/developer-tools/sql-developer/overview/index.html.
88. Connector/ J mySQL JDBC library. [Online]
[Cited: 18 August 2010.] http://dev.mysql.com/downloads/connector/j/5.0.html.
89. MySQL Cookbook. [Online]

[Cited: 18 08 2010.] http://www.freeopenbook.com/mysqlcookbook/mysqlckbk-chp-13-sect-6.html.

90. Connection pooling with MySQL Connector/J. *MySQL.* [Online]
[Cited: 18 August 2010.]
http://dev.mysql.com/tech-resources/articles/connection_pooling_with_connectorj.html.

91. O'Reily Java Servlet Programming. [Online] [Cited: 18 August 2010.] http://docstore.mik.ua/orelly/java-ent/servlet/ch09_03.htm.

Appendix 1- Servlet Examples

1. (a): Servlet which connects to mySQL: SQL_DBUtil.java:

```
// package declaration
package mysql;
// import relevant classes and drivers
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.servlet.http.HttpSession;

public class SQL_DBUtil {
// initialise connection and statement
Connection con = null;
Statement stmt = null;
// URL of SQL database
String JDBCUrl = "jdbc:mysql://localhost/studentdata";
// DB Username and password
String username = "root";
String password = "";
// initialise results set
String visResultSet = null;

// Connection class
public int connect() {
try {
      // Point to mySQL JDBC driver
      Class.forName("com.mysql.jdbc.Driver").newInstance();
      // Connect using parameters specified
      con = DriverManager.getConnection(JDBCUrl, username, password);
      System.out.println ("Database connection established");
     }

catch( Exception e ) {
       // if connection class fails
       System.out.println ("Failed to connect");
       // return error code of 0
      return 0;
       }
// returns error code of 1 to indicate successful connection
return 1;
       }

// SQL Query class
public ResultSet query( String SQL ) {
               try {
               stmt = con.createStatement();
               visResultSet = stmt.executeQuery( SQL );
               } catch ( Exception e ) {
                       return null;
               }
               return visResultSet;
       }
```

1. (b): Servlet which validates student ID from search form: ValidateStudentID.java:

```
package studentOverview;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;


public class ValidateStudentID extends HttpServlet {
        private static final long serialVersionUID = 1L;

        mysql.SQL_DBUtil db = null;

    public ValidateStudentID() {
         super();
         db = new mysql.SQL_DBUtil();
    }

protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

int errorCode = -1;
PrintWriter out = response.getWriter();
errorCode = db.connect();

if( errorCode == 1 ) {
        //we are connected to the db
        String SQL_ValidateID =
        "SELECT * FROM STUDENT_BIOGRAPHICAL WHERE BIOSTUDNO = '"
        + request.getParameter("studentID") + "' GROUP BY BIOSTUDNO";


        java.sql.ResultSet rs = db.query( SQL_ValidateID );

        if( rs!= null ) {
                try {
                        int rowCtr = 0;
                        while( rs.next() ) {
                        rowCtr++;

                //set student ID from DB to be Java Session ID
                String S_ID =  rs.getString("BIOSTUDNO");
                HttpSession session = request.getSession(true);
                session.setAttribute("sStudentID", S_ID );
                session.setAttribute("sFIRSTNAME", rs.getString("BIOFIRSTNAME"));
                session.setAttribute("sSURNAME", rs.getString("BIOSURNAME"));
                session.setAttribute("sSEX", rs.getString("BIOSEX"));
                session.setAttribute("sADDR1", rs.getString("BIOADDR1"));
                session.setAttribute("sADDR2", rs.getString("BIOADDR2"));
                session.setAttribute("sADDR3", rs.getString("BIOADDR3"));
                session.setAttribute("sTELNO", rs.getString("BIOTELNO"));
                session.setAttribute("sEMAIL", rs.getString("BIOEMAIL"));
                session.setAttribute("sUSERNAME", rs.getString("BIOUSERNAME"));

                response.sendRedirect("/Thesis/StudentOverviewPopulateJSON?studentID=" +
                rs.getString("BIOSTUDNO") + "&default=0" );
                        }
                        // Validation - if student doesn't exist
                        if (rowCtr == 0){
                                out.println( "Problem executing statement" );
                                out.println( "Student Not Found" );
                                        }
                        // Validation - duplicate records found
                        if (rowCtr > 1){
                                out.println( "Problem executing statement" );
                                out.println( "Duplicate Records Found" );
                        }


                }
```

```
        // if no records found to match specified student number
        catch(SQLException e_SQL){
                                        out.println( "Problem executing statement" );
                                        out.println( "Student Not Found" );
                                        System.out.println("1");
                                }
                        }

                } else {
                        //display error
                        out.println( "Cannot connect to database" );
                }
        }

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        // Servlet exception class
        }
}
```

1. (c): Servlet to populate JSON strings and add to session:

StudentOverviewPopulateJSON.java:

```
package studentOverview;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
// import JSON classes - downloadable from www.json.org
import net.sf.json.JSONArray;


public class StudentOverviewPopulateJSON extends HttpServlet {
private static final long serialVersionUID = 1L;

mysql.SQL_DBUtil db = null;

public StudentOverviewPopulateJSON() {
super();
db = new mysql.SQL_DBUtil();
}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
int errorCode = -1;

PrintWriter out = response.getWriter();
errorCode = db.connect();
if( errorCode == 1 ) {


String Call_Student_Proc = "call studentdata.populateJSON('" +
request.getParameter("studentID") + "')";
String SQL_INDIVIDUAL_STUDENT = Call_Student_Proc;

java.sql.ResultSet rs = db.query( SQL_INDIVIDUAL_STUDENT );

if( rs!= null) {
try {
int rowCtr = 0;
//High Charts JSON array declarations
JSONArray arrayObj_I_Year=new JSONArray();
JSONArray arrayObj_I_AVG_OverallMark=new JSONArray();
JSONArray arrayObj_I_AVG_ExamMark=new JSONArray();
JSONArray arrayObj_I_AVG_CAMark=new JSONArray();
JSONArray arrayObj_I_STDDEV_OverallMark=new JSONArray();
JSONArray arrayObj_I_STDDEV_ExamMark=new JSONArray();
JSONArray arrayObj_I_STDDEV_CAMark=new JSONArray();
JSONArray arrayObj_I_MAX_OverallMark=new JSONArray();
JSONArray arrayObj_I_MAX_ExamMark=new JSONArray();
JSONArray arrayObj_I_MAX_CAMark=new JSONArray();
JSONArray arrayObj_I_MIN_OverallMark=new JSONArray();
JSONArray arrayObj_I_MIN_ExamMark=new JSONArray();
JSONArray arrayObj_I_MIN_CAMark=new JSONArray();
JSONArray arrayObj_I_COUNTOVERALL_FIRSTCLASSHONOURS=new JSONArray();
JSONArray arrayObj_I_COUNTOVERALL_SECONDCLASSHONOURS_G1=new JSONArray();
JSONArray arrayObj_I_COUNTOVERALL_SECONDCLASSHONOURS_G2=new JSONArray();
JSONArray arrayObj_I_COUNTOVERALL_PASS=new JSONArray();
JSONArray arrayObj_I_COUNTOVERALL_FAIL=new JSONArray();
JSONArray arrayObj_I_COUNTEXAM_FIRSTCLASSHONOURS=new JSONArray();
JSONArray arrayObj_I_COUNTEXAM_SECONDCLASSHONOURS_G1=new JSONArray();
JSONArray arrayObj_I_COUNTEXAM_SECONDCLASSHONOURS_G2=new JSONArray();
JSONArray arrayObj_I_COUNTEXAM_PASS=new JSONArray();
JSONArray arrayObj_I_COUNTEXAM_FAIL=new JSONArray();
JSONArray arrayObj_I_COUNTEXAM_DROPOUT=new JSONArray();
JSONArray arrayObj_I_COUNTCA_FIRSTCLASSHONOURS=new JSONArray();
JSONArray arrayObj_I_COUNTCA_SECONDCLASSHONOURS_G1=new JSONArray();
JSONArray arrayObj_I_COUNTCA_SECONDCLASSHONOURS_G2=new JSONArray();
JSONArray arrayObj_I_COUNTCA_PASS=new JSONArray();
JSONArray arrayObj_I_COUNTCA_FAIL=new JSONArray();
```

```
JSONArray arrayObj_I_COUNTCA_DROPOUT=new JSONArray();
JSONArray arrayObj_I_COUNT_MODULES_PER_SEMESTER=new JSONArray();
// Percentage calculation arrays
JSONArray arrayObj_I_Percent_FIRSTCLASSHONOURS=new JSONArray();
JSONArray arrayObj_I_Percent_SECONDCLASSHONOURS_G1=new JSONArray();
JSONArray arrayObj_I_Percent_SECONDCLASSHONOURS_G2=new JSONArray();
JSONArray arrayObj_I_Percent_PASS=new JSONArray();
JSONArray arrayObj_I_Percent_FAIL=new JSONArray();
JSONArray arrayObj_I_Percent_DROPOUT=new JSONArray();
// Visifire Area chart Arrays
JSONArray arrayObj_I_Area_FIRSTCLASSHONOURS=new JSONArray();
JSONArray arrayObj_I_Area_SECONDCLASSHONOURS_G1=new JSONArray();
JSONArray arrayObj_I_Area_SECONDCLASSHONOURS_G2=new JSONArray();
JSONArray arrayObj_I_Area_PASS=new JSONArray();
JSONArray arrayObj_I__Area_FAIL=new JSONArray();


// While the results set of the query has rows, for the next row
while( rs.next()) {

// Populate the JSON arrays with query results
// HIGHCHARTS JSON ARRAYS
// YEAR JSON
arrayObj_I_Year.add(rs.getString( "YEAR" ));
// AVG Overall GPA JSON
arrayObj_I_AVG_OverallMark.add(rs.getDouble( "AVG_OVERALL_MARK" ));
// AVG EXAM GPA JSON
arrayObj_I_AVG_ExamMark.add(rs.getDouble( "AVG_EXAM_MARK" ));
// AVG CA GPA JSON
arrayObj_I_AVG_CAMark.add(rs.getDouble( "AVG_CA_MARK" ));
// STDDEV Overall GPA JSON
arrayObj_I_STDDEV_OverallMark.add(rs.getDouble( "STD_DEV_OVERALL_MARK" ));
// STDDEV EXAM GPA JSON
arrayObj_I_STDDEV_ExamMark.add(rs.getDouble( "STD_DEV_EXAM_MARK" ));
// STDDEV CA GPA JSON
arrayObj_I_STDDEV_CAMark.add(rs.getDouble( "STD_DEV_CA_MARK" ));
// MAX Overall GPA JSON
arrayObj_I_MAX_OverallMark.add(rs.getDouble( "MAX_OVERALL_MARK" ));
// MAX EXAM GPA JSON
arrayObj_I_MAX_ExamMark.add(rs.getDouble( "MAX_EXAM_MARK" ));
// MAX CA GPA JSON
arrayObj_I_MAX_CAMark.add(rs.getDouble( "MAX_CA_MARK" ));
// MIN Overall GPA JSON
arrayObj_I_MIN_OverallMark.add(rs.getDouble( "MIN_OVERALL_MARK" ));
// MIN EXAM GPA JSON
arrayObj_I_MIN_ExamMark.add(rs.getDouble( "MIN_EXAM_MARK" ));
// MIN CA GPA JSON
arrayObj_I_MIN_CAMark.add(rs.getDouble( "MIN_CA_MARK" ));
// Count arrays
arrayObj_I_COUNTOVERALL_FIRSTCLASSHONOURS.add(rs.getDouble("COUNTOVERALL_FIRSTCLASSHONOURS")
);
arrayObj_I_COUNTOVERALL_SECONDCLASSHONOURS_G1.add(rs.getDouble("COUNTOVERALL_SECONDCLASSHONO
URS_G1"));
arrayObj_I_COUNTOVERALL_SECONDCLASSHONOURS_G2.add(rs.getDouble("COUNTOVERALL_SECONDCLASSHONO
URS_G2"));
arrayObj_I_COUNTOVERALL_PASS.add(rs.getDouble("COUNTOVERALL_PASS"));
arrayObj_I_COUNTOVERALL_FAIL.add(rs.getDouble("COUNTOVERALL_FAIL"));
arrayObj_I_COUNTEXAM_FIRSTCLASSHONOURS.add(rs.getDouble("COUNTEXAM_FIRSTCLASSHONOURS"));
arrayObj_I_COUNTEXAM_SECONDCLASSHONOURS_G1.add(rs.getDouble("COUNTEXAM_SECONDCLASSHONOURS_G1
"));
arrayObj_I_COUNTEXAM_SECONDCLASSHONOURS_G2.add(rs.getDouble("COUNTEXAM_SECONDCLASSHONOURS_G2
"));
arrayObj_I_COUNTEXAM_PASS.add(rs.getDouble("COUNTEXAM_PASS"));
arrayObj_I_COUNTEXAM_FAIL.add(rs.getDouble("COUNTEXAM_FAIL"));
arrayObj_I_COUNTEXAM_DROPOUT.add(rs.getDouble("COUNTEXAM_DROPOUT"));
arrayObj_I_COUNTCA_FIRSTCLASSHONOURS.add(rs.getDouble("COUNTCA_FIRSTCLASSHONOURS"));
arrayObj_I_COUNTCA_SECONDCLASSHONOURS_G1.add(rs.getDouble("COUNTCA_SECONDCLASSHONOURS_G1"));
arrayObj_I_COUNTCA_SECONDCLASSHONOURS_G2.add(rs.getDouble("COUNTCA_SECONDCLASSHONOURS_G2"));
arrayObj_I_COUNTCA_PASS.add(rs.getDouble("COUNTCA_PASS"));
arrayObj_I_COUNTCA_FAIL.add(rs.getDouble("COUNTCA_FAIL"));
arrayObj_I_COUNTCA_DROPOUT.add(rs.getDouble("COUNTCA_DROPOUT"));
arrayObj_I_COUNT_MODULES_PER_SEMESTER.add(rs.getDouble("COUNT_MODULES_PER_SEMESTER"));

// SOME CODE TO CALCULATE %'S
int semesterModuleCount = rs.getInt("COUNT_MODULES_PER_SEMESTER");
// Calculate % of 1.1'S per semester
int semester11Count = rs.getInt("COUNTOVERALL_FIRSTCLASSHONOURS");
```

```
double Percent11Semester = (semester11Count/semesterModuleCount)*100;
// Calculate % of 2.1'S per semester
int semester21Count = rs.getInt("COUNTOVERALL_SECONDCLASSHONOURS_G1");
double Percent21Semester = (semester21Count/semesterModuleCount)*100;
// Calculate % of 2.2'S per semester
int semester22Count = rs.getInt("COUNTOVERALL_SECONDCLASSHONOURS_G2");
double Percent22Semester = (semester22Count/semesterModuleCount)*100;
// Calculate % of P per semester
int semesterPCount = rs.getInt("COUNTOVERALL_PASS");
double PercentPassSemester = (semesterPCount/semesterModuleCount)*100;
// Calculate % of modules failed per semester
int semesterFailCount = rs.getInt("COUNTOVERALL_FAIL");
double PercentFailedSemester = (semesterFailCount/semesterModuleCount)*100;
int DropOut = rs.getInt("COUNTEXAM_DROPOUT");
double PercentDropOutSemester = (DropOut/semesterModuleCount)*100;
// Add % calculations to their own JSON

// Populate percentage arrays
arrayObj_I_Percent_FIRSTCLASSHONOURS.add(Percent11Semester);
arrayObj_I_Percent_SECONDCLASSHONOURS_G1.add(Percent21Semester);
arrayObj_I_Percent_SECONDCLASSHONOURS_G2.add(Percent22Semester);
arrayObj_I_Percent_PASS.add(PercentPassSemester);
arrayObj_I_Percent_FAIL.add(PercentFailedSemester);
arrayObj_I_Percent_DROPOUT.add(PercentDropOutSemester);
// Populate Visifire Area Chart arrays
arrayObj_I_Area_FIRSTCLASSHONOURS.add(70);
arrayObj_I_Area_SECONDCLASSHONOURS_G1.add(60);
arrayObj_I_Area_SECONDCLASSHONOURS_G2.add(50);
arrayObj_I_Area_PASS.add(40);

}


// Add arrays to session strings
HttpSession session = request.getSession(true);
// year
session.setAttribute("Json_I_AVG_Year",arrayObj_I_Year );
// averages
session.setAttribute("Json_I_AVG_OverallMark",arrayObj_I_AVG_OverallMark );
session.setAttribute("Json_I_AVG_ExamMark",arrayObj_I_AVG_ExamMark );
session.setAttribute("Json_I_AVG_CAMark",arrayObj_I_AVG_CAMark );
//std dev
session.setAttribute("Json_I_STDDEV_OverallMark",arrayObj_I_STDDEV_OverallMark );
session.setAttribute("Json_I_STDDEV_ExamMark",arrayObj_I_STDDEV_ExamMark );
session.setAttribute("Json_I_STDDEV_CAMark",arrayObj_I_STDDEV_CAMark );
// max
session.setAttribute("Json_I_MAX_OverallMark",arrayObj_I_MAX_OverallMark );
session.setAttribute("Json_I_MAX_ExamMark",arrayObj_I_MAX_ExamMark );
session.setAttribute("Json_I_MAX_CAMark",arrayObj_I_MAX_CAMark );
//min
session.setAttribute("Json_I_MIN_OverallMark",arrayObj_I_AVG_OverallMark );
session.setAttribute("Json_I_MIN_ExamMark",arrayObj_I_AVG_ExamMark );
session.setAttribute("Json_I_MIN_CAMark",arrayObj_I_AVG_CAMark );
// counts
session.setAttribute("JSON_COUNTOVERALL_FIRSTCLASSHONOURS",arrayObj_I_COUNTOVERALL_FIRSTCLAS
SHONOURS       );
session.setAttribute("JSON_COUNTOVERALL_SECONDCLASSHONOURS_G1",arrayObj_I_COUNTOVERALL_SECON
DCLASSHONOURS_G1);
session.setAttribute("JSON_COUNTOVERALL_SECONDCLASSHONOURS_G2",arrayObj_I_COUNTOVERALL_SECON
DCLASSHONOURS_G2);
session.setAttribute("JSON_COUNTOVERALL_PASS",arrayObj_I_COUNTOVERALL_PASS);
session.setAttribute("JSON_COUNTOVERALL_FAIL",arrayObj_I_COUNTOVERALL_FAIL);
session.setAttribute("JSON_COUNTEXAM_FIRSTCLASSHONOURS",arrayObj_I_COUNTEXAM_FIRSTCLASSHONOU
RS);
session.setAttribute("JSON_COUNTEXAM_SECONDCLASSHONOURS_G1",arrayObj_I_COUNTEXAM_SECONDCLASS
HONOURS_G1);
session.setAttribute("JSON_COUNTEXAM_SECONDCLASSHONOURS_G2",arrayObj_I_COUNTEXAM_SECONDCLASS
HONOURS_G2);
session.setAttribute("JSON_COUNTEXAM_PASS",arrayObj_I_COUNTEXAM_PASS);
session.setAttribute("JSON_COUNTEXAM_FAIL",arrayObj_I_COUNTEXAM_FAIL);
session.setAttribute("JSON_COUNTEXAM_DROPOUT",arrayObj_I_COUNTEXAM_DROPOUT);
session.setAttribute("JSON_COUNTCA_FIRSTCLASSHONOURS",arrayObj_I_COUNTCA_FIRSTCLASSHONOURS);
session.setAttribute("JSON_COUNTCA_SECONDCLASSHONOURS_G1",arrayObj_I_COUNTCA_SECONDCLASSHONO
URS_G1);
session.setAttribute("JSON_COUNTCA_SECONDCLASSHONOURS_G2",arrayObj_I_COUNTCA_SECONDCLASSHONO
URS_G2);
session.setAttribute("JSON_COUNTCA_PASS",arrayObj_I_COUNTCA_PASS);
```

```
session.setAttribute("JSON_COUNTCA_FAIL",arrayObj_I_COUNTCA_FAIL);
session.setAttribute("JSON_COUNTCA_DROPOUT",arrayObj_I_COUNTCA_DROPOUT);
session.setAttribute("JSON_COUNT_MODULES_PER_SEMESTER",arrayObj_I_COUNT_MODULES_PER_SEMESTER
);
// Percentage values
session.setAttribute("JSON_Percent_FIRSTCLASSHONOURS",arrayObj_I_Percent_FIRSTCLASSHONOURS);
session.setAttribute("JSON_Percent_SECONDCLASSHONOURS_G1",arrayObj_I_Percent_SECONDCLASSHONO
URS_G1);
session.setAttribute("JSON_Percent_SECONDCLASSHONOURS_G2",arrayObj_I_Percent_SECONDCLASSHONO
URS_G2);
session.setAttribute("JSON_Percent_PASS",arrayObj_I_Percent_PASS);
session.setAttribute("JSON_Percent_FAIL",arrayObj_I_Percent_FAIL);
session.setAttribute("JSON_Percent_DROPOUT",arrayObj_I_Percent_DROPOUT);
session.setAttribute("JSON_Area_FIRSTCLASSHONOURS",arrayObj_I_Area_FIRSTCLASSHONOURS);
session.setAttribute("JSON_Area_SECONDCLASSHONOURS_G1",arrayObj_I_Area_SECONDCLASSHONOURS_G1
);
session.setAttribute("JSON_Area_SECONDCLASSHONOURS_G2",arrayObj_I_Area_SECONDCLASSHONOURS_G2
);
session.setAttribute("JSON_Area_PASS",arrayObj_I_Area_PASS);
rowCtr++;

// Redirect user to student analysis page to view analysis using session variables populated
response.sendRedirect("/Thesis/studentOverview/StudentOverview.jsp?studentID=" +
request.getParameter("studentID") + "&default=0" );

} catch( SQLException e ) {
out.println( "Problem executing statement" );
out.println( "Error: " + e.getMessage() );
}
}

} else {
//display error
out.println( "ERROR" );
}
}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
// Servlet Post exception class
}
```

Appendix 2 – Database structure

2(a): Table structure for table student_biographical

Field	Type	Null	Default
BIOSTUDNO	int(8)	Yes	NULL
BIOFIRSTNAME	varchar(10)	Yes	NULL
BIOSURNAME	varchar(15)	Yes	NULL
BIOSEX	varchar(1)	Yes	NULL
BIOADDR1	varchar(15)	Yes	NULL
BIOADDR2	varchar(15)	Yes	NULL
BIOADDR3	varchar(15)	Yes	NULL
BIOTELNO	varchar(12)	Yes	NULL
BIOEMAIL	varchar(30)	Yes	NULL
BIOUSERNAME	varchar(10)	Yes	NULL
BIOPASS	varchar(8)	Yes	NULL

2(b): Table structure for table student_eng_modules

Field	Type	Null	Default
MODSTUDNO	int(8)	Yes	NULL
MODYEAR	int(4)	Yes	NULL
MODCODE	varchar(6)	Yes	NULL
MODDESC	varchar(40)	Yes	NULL
MODOFTYPE	int(2)	Yes	NULL
MODOTDESC	varchar(10)	Yes	NULL
MODBLOCKCODE	int(2)	Yes	NULL
MODBCDESC	varchar(20)	Yes	NULL
MODQUALNUMBER	int(1)	Yes	NULL
MODATTENDTYPE	varchar(1)	Yes	NULL
MODCANCELDATE	varchar(15)	Yes	NULL
MODEXEMPT	varchar(1)	Yes	NULL

2(c): Table structure for table student_eng_qualifications

Field	Type	Null	Default
QUALSTUDNO	int(8)	Yes	NULL
QUALCODE	varchar(4)	Yes	NULL
QUALDESC	varchar(40)	Yes	NULL
QUALYEAR	int(4)	Yes	NULL
QUALOFFTYPE	int(2)	Yes	NULL
QUALOTDESC	varchar(15)	Yes	NULL
QUALBLOCKCODE	int(2)	Yes	NULL
QUALBCDESC	varchar(15)	Yes	NULL
QUALSTUDYPERIOD	varchar(2)	Yes	NULL
QUALNUMBER	int(2)	Yes	NULL
QUALCANCELDATE	varchar(15)	Yes	NULL
QUALRECORDID	varchar(100)	Yes	NULL

2(d): Table structure for table student_eng_qualresults

Field	Type	Null	Default
RESQSTUDNO	int(8)	Yes	NULL
RESQCODE	varchar(4)	Yes	NULL
RESQDESC	varchar(20)	Yes	NULL
RESQYEAR	int(4)	Yes	NULL
RESOFFTYPE	int(2)	Yes	NULL
RESQOTDESC	varchar(15)	Yes	NULL
RESQBLOCKCODE	int(2)	Yes	NULL
RESQBCDESC	varchar(15)	Yes	NULL
RESQSTUDYPERIOD	int(2)	Yes	NULL
RESQNUMBER	int(2)	Yes	NULL
RESQCANCELDATE	varchar(10)	Yes	NULL
RESQANNUALRES	varchar(2)	Yes	NULL
RESQOVERALLMARK	int(3)	Yes	NULL

2(e): Table structure for table student_mod_results

Field	Type	Null	Default
RESSTUDNO	int(8)	Yes	NULL
RESMYYEAR	int(4)	Yes	NULL
RESMCODE	varchar(5)	Yes	NULL

RESMDESC	varchar(50)	Yes	NULL
RESMOFFTYPE	int(2)	Yes	NULL
RESMOTDESC	varchar(15)	Yes	NULL
RESMBLOCKCODE	int(2)	Yes	NULL
RESMBCDESC	varchar(20)	Yes	NULL
RESQUALNUMBER	int(2)	Yes	NULL
RESMCANCELDATE	varchar(8)	Yes	NULL
RESMSONTASSESS	int(2)	Yes	NULL
RESMEXAMMARK	int(2)	Yes	NULL
RESMOVERALLMARK	int(2)	Yes	NULL
RESMRESULT	varchar(2)	Yes	NULL
RESMEXAMMONTH	int(2)	Yes	NULL
RESMEXAMTYPE	varchar(1)	Yes	NULL
RESMEXEMPT	varchar(1)	Yes	NULL

Appendix 3 – SQL Stored Procedures

3 (a): Student analysis populate JSON procedure:

```
DELIMITER $$

/* IN variable reads in student number */
CREATE DEFINER=`root`@`localhost` PROCEDURE `populateJSON`(IN studentIDnumber VARCHAR(8))
BEGIN

SELECT
B.BIOFIRSTNAME AS FIRSTNAME,
B.BIOSURNAME AS SURNAME,
B.BIOSEX AS GENDER,
R.RESMYYEAR AS YEAR,
R.RESMEXAMMONTH AS MONTH,
R.RESMBCDESC AS SEMESTER,
COUNT(R.RESMYYEAR) AS COUNT_ROWS,
R.RESMOVERALLMARK,
R.RESMRESULT,
R.RESMEXAMMARK,
R.RESMSONTASSESS,
/* ROUNDED AVERAGE GRADES PER SEMESTER*/
ROUND(AVG(R.RESMOVERALLMARK),2) AS AVG_OVERALL_MARK,
ROUND(AVG(R.RESMEXAMMARK),2) AS AVG_EXAM_MARK,
ROUND(AVG(R.RESMSONTASSESS),2) AS AVG_CA_MARK,
/* ROUNDED STD DEV GRADES PER SEMESTER*/
ROUND(STD(R.RESMOVERALLMARK),2) AS STD_DEV_OVERALL_MARK,
ROUND(STD(R.RESMEXAMMARK),2) AS STD_DEV_EXAM_MARK,
ROUND(STD(R.RESMSONTASSESS),2) AS STD_DEV_CA_MARK,
/* ROUNDED MAXIMUM GRADES PER SEMESTER*/
ROUND(MAX(R.RESMOVERALLMARK),2) AS MAX_OVERALL_MARK,
ROUND(MAX(R.RESMEXAMMARK),2) AS MAX_EXAM_MARK,
ROUND(MAX(R.RESMSONTASSESS),2) AS MAX_CA_MARK,
/* ROUNDED MINIMUM GRADES PER SEMESTER*/
ROUND(MIN(R.RESMOVERALLMARK),2) AS MIN_OVERALL_MARK,
ROUND(MIN(R.RESMEXAMMARK),2) AS MIN_EXAM_MARK,
ROUND(MIN(R.RESMSONTASSESS),2) AS MIN_CA_MARK,
/*COUNT OF OVERALL FAIL PASS 1.1 2.1 PER SEMESTER*/
/*CASE/WHEN COMMANDS USED TO DEFINE GRADE RANGES */
COUNT(CASE WHEN R.RESMOVERALLMARK >= '70'
           THEN R.RESMOVERALLMARK END) AS COUNTOVERALL_FIRSTCLASSHONOURS,
COUNT(CASE WHEN R.RESMOVERALLMARK >= '60'  AND R.RESMOVERALLMARK <= '70'
           THEN R.RESMOVERALLMARK END) AS COUNTOVERALL_SECONDCLASSHONOURS_G1,
COUNT(CASE WHEN R.RESMOVERALLMARK >= '50'  AND R.RESMOVERALLMARK <= '60'
           THEN R.RESMOVERALLMARK END) AS COUNTOVERALL_SECONDCLASSHONOURS_G2,
COUNT(CASE WHEN R.RESMOVERALLMARK >= '50'  AND R.RESMOVERALLMARK <= '40'
           THEN R.RESMOVERALLMARK END) AS COUNTOVERALL_PASS,
COUNT(CASE WHEN R.RESMOVERALLMARK <= '40'
           THEN R.RESMOVERALLMARK END) AS COUNTOVERALL_FAIL,
COUNT(CASE WHEN R.RESMEXAMTYPE = 'R'
           THEN R.RESMEXAMTYPE END) AS COUNTOVERALL_REPEAT,
/*COUNT OF EXAM FAIL PASS 1.1 2.1 PER SEMESTER*/
COUNT(CASE WHEN R.RESMEXAMMARK >= '70'
           THEN R.RESMEXAMMARK END) AS COUNTEXAM_FIRSTCLASSHONOURS,
COUNT(CASE WHEN R.RESMEXAMMARK >= '60'  AND R.RESMOVERALLMARK <= '70'
           THEN R.RESMEXAMMARK END) AS COUNTEXAM_SECONDCLASSHONOURS_G1,
COUNT(CASE WHEN R.RESMEXAMMARK >= '50'  AND R.RESMOVERALLMARK <= '60'
           THEN R.RESMEXAMMARK END) AS COUNTEXAM_SECONDCLASSHONOURS_G2,
COUNT(CASE WHEN R.RESMEXAMMARK >= '50'  AND R.RESMOVERALLMARK <= '40'
           THEN R.RESMEXAMMARK END) AS COUNTEXAM_PASS,
COUNT(CASE WHEN R.RESMEXAMMARK <= '40'
           THEN R.RESMEXAMMARK END) AS COUNTEXAM_FAIL,
COUNT(CASE WHEN R.RESMCANCELDATE != '0'
           THEN R.RESMCANCELDATE END) AS COUNTEXAM_DROPOUT,
COUNT(CASE WHEN R.RESMEXAMTYPE = 'R'
           THEN R.RESMEXAMTYPE END) AS COUNTEXAM_REPEAT,
/*COUNT OF CA FAIL PASS 1.1 2.1 PER SEMESTER*/
COUNT(CASE WHEN R.RESMSONTASSESS >= '70'
           THEN R.RESMSONTASSESS END) AS COUNTCA_FIRSTCLASSHONOURS,
COUNT(CASE WHEN R.RESMSONTASSESS >= '60'  AND R.RESMOVERALLMARK <= '70'
           THEN R.RESMSONTASSESS END) AS COUNTCA_SECONDCLASSHONOURS_G1,
COUNT(CASE WHEN R.RESMSONTASSESS >= '50'  AND R.RESMOVERALLMARK <= '60'
           THEN R.RESMSONTASSESS END) AS COUNTCA_SECONDCLASSHONOURS_G2,
```

```
COUNT(CASE WHEN R.RESMSONTASSESS >= '50'  AND R.RESMOVERALLMARK <= '40'
           THEN R.RESMSONTASSESS END) AS COUNTCA_PASS,
COUNT(CASE WHEN R.RESMSONTASSESS <= '40'
           THEN R.RESMSONTASSESS END) AS COUNTCA_FAIL,
COUNT(CASE WHEN R.RESMEXAMTYPE = 'R'
           THEN R.RESMEXAMTYPE END) AS COUNTCA_REPEAT,
COUNT(CASE WHEN R.RESMCANCELDATE != '0'
           THEN R.RESMCANCELDATE END) AS COUNTCA_DROPOUT,
/*COUNT MODULES TAKEN PER SEMESTER*/
COUNT(DISTINCT(R.RESMCODE)) AS COUNT_MODULES_PER_SEMESTER,
/* Nested query to calculate average grade for all other students */
( SELECT AVG(RESMOVERALLMARK)
FROM studentdata.STUDENT_MOD_RESULTS WHERE RESSTUDNO != '53431274'
ORDER BY RESMEXAMMONTH
) AS OVERALL_MARK_OTHER_STUDENTS
FROM studentdata.STUDENT_BIOGRAPHICAL B, studentdata.STUDENT_MOD_RESULTS R
/* LIKE matches against IN variable studentIDnumber */
WHERE B.BIOSTUDNO = R.RESSTUDNO AND R.RESSTUDNO LIKE studentIDnumber
/* Group and order by year and exam month to order by year and semester*/
GROUP BY R.RESMYYEAR, R.RESMEXAMMONTH ORDER BY R.RESMYYEAR, R.RESMEXAMMONTH;

END
```

3 (b): Linear regression analysis stored procedure:

```
DELIMITER $$
/* Regression procedure Exam mark Vs Overall mark */

CREATE DEFINER=`root`@`localhost` PROCEDURE `regressionProcedureStudent`(IN studentIDnumber
VARCHAR(8))
BEGIN
SELECT
/* number of variables */
@n := COUNT( RESMOVERALLMARK ) AS N,
/* Average Exam mark */
@meanX := AVG( RESMEXAMMARK )AS "X mean",
/* Sum Exam mark */
@sumX := SUM( RESMEXAMMARK ) AS "X sum",
/* Sum Exam mark squared */
@sumXX := SUM( RESMEXAMMARK * RESMEXAMMARK ) "X sum of squares",
/* Average Overall mark */
@meanY := AVG( RESMOVERALLMARK )AS "Y mean",
/* Sum Overall mark */
@sumY := SUM( RESMOVERALLMARK ) AS "Y sum",
/* Sum Overall mark squared */
@sumYY := SUM( RESMOVERALLMARK * RESMOVERALLMARK ) "Y sum of square",
/* sum Exam by sum overall */
@sumXY := SUM( RESMEXAMMARK * RESMOVERALLMARK ) AS "X*Y sum",
/* Calculate slope of regression line */
@b := (@n*@sumXY - @sumX*@sumY) / (@n*@sumXX - @sumX*@sumX)
AS slope,
/* Calculate intercept - value of y when x = 0 */
@a := (@meanY - @b*@meanX) AS intercept,

/* Create least squares regression blob - essentially regression formula
Y = mX + b -> where m = slope and b = intercept
*/
CONCAT('Y = ',@b,'X + ',@a) AS 'least-squares regression',

/* Calculate Correlation */
 (@n*@sumXY - @sumX*@sumY)
 / SQRT((@n*@sumXX - @sumX*@sumX) * (@n*@sumYY - @sumY*@sumY))
AS correlation

FROM STUDENTDATA.student_mod_results
WHERE RESSTUDNO = studentIDnumber
GROUP BY RESSTUDNO
;
```

Appendix 4- Visualisation Code

4 (a): Google Motion Chart Source code - StudentOverviewGoogle_MotionChart.jsp :

```
<script type="text/javascript">
/* Load motion chart package */
google.load('visualization', '1', {'packages':['motionchart']});
google.setOnLoadCallback(drawChart);
function drawChart() {
var MotionChartData = new google.visualization.DataTable();
/* Add columns, first must be year, second must be date / year, the rest are variable */
MotionChartData.addColumn('string', 'Profile');
MotionChartData.addColumn('number', 'Year');
MotionChartData.addColumn('number', 'Overall GPA');
MotionChartData.addColumn('number', 'Exam GPA');
MotionChartData.addColumn('number', 'CA GPA');
MotionChartData.addColumn('number', 'Std-Dev. Overall GPA');
MotionChartData.addColumn('number', 'Std-Dev. Exam GPA');
MotionChartData.addColumn('number', 'Std-Dev. CA GPA');
MotionChartData.addColumn('number', 'MAX Overall GPA');
MotionChartData.addColumn('number', 'MAX Exam GPA');
MotionChartData.addColumn('number', 'MAX CA GPA');
MotionChartData.addColumn('number', 'MIN Overall GPA');
MotionChartData.addColumn('number', 'MIN Exam GPA');
MotionChartData.addColumn('number', 'MIN CA GPA');
MotionChartData.addColumn('number', 'Count Overall 1.1');
MotionChartData.addColumn('number', 'Count Overall 2.1');
MotionChartData.addColumn('number', 'Count Overall 2.2');
MotionChartData.addColumn('number', 'Count Overall Pass');
MotionChartData.addColumn('number', 'Count Overall Fail');
MotionChartData.addColumn('number', 'Count Exam 1.1');
MotionChartData.addColumn('number', 'Count Exam 2.1');
MotionChartData.addColumn('number', 'Count Exam 2.2');
MotionChartData.addColumn('number', 'Count Exam Pass');
MotionChartData.addColumn('number', 'Count Exam Fail');
MotionChartData.addColumn('number', 'Count Exam Drop Out');
MotionChartData.addColumn('number', 'Count CA 1.1');
MotionChartData.addColumn('number', 'Count CA 2.1');
MotionChartData.addColumn('number', 'Count CA 2.2');
MotionChartData.addColumn('number', 'Count CA Pass');
MotionChartData.addColumn('number', 'Count CA Fail');
MotionChartData.addColumn('number', 'Count CA Drop Out');
MotionChartData.addColumn('number', 'Count Modules Per Semester');

/* Add rows to the chart by including the servlet output */
MotionChartData.addRows(
<jsp:include page="/servlet/studentOverview.StudentOverviewPopulateGoogle" flush="true" />
);
/* Declare motion chart and set div reference as motionchart_div */
var MotionChart = new
google.visualization.MotionChart(document.getElementById('motionchart_div'));
/* set chart dimensions 1000x450 */
MotionChart.draw(MotionChartData, {width: 1000, height:450});
}
</script>
```

4 (b): Google Student Grade deviation data tables:

StudentOverviewGoogle_YearlyGradeDeviationTable_Large.jsp

```
<script type="text/javascript">
/* Load motionchart, table and corechart packages to allow multiple charts be drawn from
same data and to minimise repetition*/
google.load('visualization', '1', {packages: ['motionchart', 'table', 'corechart']});
</script>
<script type="text/javascript">
function drawVisualization() {
// Create and populate the data table using JSON - cols = columns, rows = rows
var GoogleYearlyGradeDeviation_tableJSONObject = {
cols: [{id: 'Semester', label: 'Semester', type: 'string'},
       {id: 'Year', label: 'Year', type: 'number'},
       {id: 'Maximum Grade', label: 'Maximum Grade', type: 'number'},
       {id: 'Minimum Grade', label: 'Minimum Grade', type: 'number'},
       {id: 'Standard Deviation', label: 'Standard Deviation', type: 'number'},
       {id: 'Number of Students', label: 'Number of Students', type: 'number'},
       {id: 'Individual GPA average', label: 'Individual GPA average', type: 'number'},
       {id: 'All students GPA average', label: 'All students GPA average', type: 'number'},
       {id: 'Difference', label: 'Difference', type: 'number'}

       ],
rows:
// Include servlet output as rows
<jsp:include page="/servlet/studentOverview.StudentOverviewGoogle_YearlyGradeDeviationTable"
flush="true" />
};

// declare new data variable containing chart data
var data = new google.visualization.DataTable(GoogleYearlyGradeDeviation_tableJSONObject,
0.5);

// Create and draw the visualization table
visualization = new
google.visualization.Table(document.getElementById('GoogleYearlyGradeDeviation_table_large')
);

// apply a bar and arrow formatter to the eigth column - difference in grade
var barformatter = new google.visualization.TableBarFormat({width: 120});
barformatter.format(data, 8); // Apply formatter to eighth column
var formatter = new google.visualization.TableArrowFormat();
formatter.format(data, 8); // Apply formatter to eighth column
// Draw visual
visualization.draw(data, {'allowHtml': true});

// Draw motion chart from data
visualizationDeviationMotionChart = new
google.visualization.MotionChart(document.getElementById('GoogleYearlyGradeDeviation_table_M
otionChart'));
visualizationDeviationMotionChart.draw(data,{width: 1000, height:350});

// Draw column chart from data
visualizationDeviationColumnChart = new
google.visualization.ColumnChart(document.getElementById('GoogleYearlyGradeDeviation_table_C
olumnChart'));
visualizationDeviationColumnChart.draw(data, {'allowHtml': true});
// Draw area chart from data
visualizationDeviationAreaChart = new
google.visualization.AreaChart(document.getElementById('GoogleYearlyGradeDeviation_table_Are
aChart'));
visualizationDeviationAreaChart.draw(data, {'allowHtml': true});
}
google.setOnLoadCallback(drawVisualization);
</script>
```

4 (c): High Charts Colour scatter chart-StudentOverviewHC_ColourScatter.jsp

```
<script type="text/javascript">
// declare chart name
var Results_colorscatter;
$(document).ready(function() {
Results_colorscatter = new Highcharts.Chart({
chart: {
        renderTo: 'Results_colorscatter',
        defaultSeriesType: 'scatter',
        zoomType: 'xy'
        },
title: {
text: ''
},
// X axis
xAxis: {
        title: {
        enabled: true,
        text: 'Overall Mark (%)'
        },
startOnTick: true,
endOnTick: true,
showLastLabel: true
},
// Y axis
yAxis: {
        title: {
        text: 'CA Mark (%)'
        }},
// tooltip to show point when hovered over
tooltip: {
formatter: function() {
return '<b>'+ this.series.name +'</b><br/>'+
this.x +', '+ this.y +'';
}},

plotOptions: {
// scatter plot
scatter: {
marker: {
radius: 5,
states: {
hover: {
enabled: true,
lineColor: 'rgb(100,100,100)'
}}},
states: {
hover: {
marker: {
enabled: false}}}}},

series: [{
name: 'Individual Student',
color: 'rgba(223, 83, 83, .5)',
data:
// Include session string JSON
<%= session.getAttribute("arrayObj_JSON_Overall_Vs_Exam") %>
}, {
name: 'Other Students That Year',
color: 'rgba(119, 152, 191, .5)',
// Include session string JSON
data: <%= session.getAttribute("arrayObj_JSON_Overall_Vs_Exam_SameYear") %>
}, {
name: 'Other Students All Years',
color: 'rgba(50, 252, 91, .5)',
// Include session string JSON
data: <%= session.getAttribute("arrayObj_JSON_Overall_Vs_Exam") %>
},
{
name: 'Other Students Same Gender',
color: 'rgba(250, 252, 91, .5)',
// Include session string JSON
data: <%= session.getAttribute("arrayObj_JSON_Overall_Vs_Exam_SameGender") %>
}
,
{
```

```
name: 'Other Students Same Modules',
color: 'rgba(50, 252, 191, .5)',
// Include session string JSON
data: <%= session.getAttribute("arrayObj_JSON_Overall_Vs_Exam_SameModules") %>
},
// regression lines data series
{
type: 'line',
name: 'Individual Line',
// Regression lines defined manually due to issues with reliability of stored procedure
data: [[30, 44.11], [90, 91.51]],
marker: {
enabled: true
},
states: {
hover: {
lineWidth: 0
}
},
enableMouseTracking: true
},{
type: 'line',
name: 'Students that Year Regression',
data: [[31, 54.11], [50, 62.51]],
marker: {
enabled: true
},
states: {
hover: {
lineWidth: 0
}},
enableMouseTracking: true
},{
type: 'line',
name: 'Students all Years',
data: [[38, 24.11], [90, 51.51]],
marker: {
enabled: true
},
states: {
hover: {
lineWidth: 0
}},
enableMouseTracking: true
},{
type: 'line',
name: 'Students same Gender',
data: [[10, 44.11], [40, 61.51]],
marker: {
enabled: true
},
states: {
hover: {
lineWidth: 0
}},
enableMouseTracking: true
},{
type: 'line',
name: 'Students same Modules',
data: [[40, 54.11], [80, 51.51]],
marker: {
enabled: true
},
states: {
hover: {
lineWidth: 0
}},
// enable moue tracking
enableMouseTracking: true
}]});});
</script>
```

4 (d): HTML5 Canvas plot code:

```
<!-- HTML5 Canvas Plot -->
// If Browser is Internet Explorer Canvas Emulator JS file is used to render plot
<!--[if IE]><script src="js/excanvas.compiled.js"></script><![endif]-->
<script type="text/javascript">
window.onload = function() {
var drawingCanvas = document.getElementById('myDrawing');

// Check the element is in the DOM and the browser supports canvas
if(drawingCanvas && drawingCanvas.getContext) {
// Initialise a 2-dimensional drawing context
var context = drawingCanvas.getContext('2d');

// Added to draw Grid
for (var x = 0; x < 800; x += 80) {
// move to a co-ordinate
context.moveTo(x, 0);
// draw line from first co-ordinate to ->
context.lineTo(x, 400);
}
for (var y = 0; y < 400; y += 40) {
context.moveTo(0, y);
context.lineTo(800, y);
}
context.strokeStyle = "#eee";
context.stroke();
// End of Grid

//draw X axis:
context.beginPath();
context.moveTo(0, 400);
context.lineTo(800, 400);

//draw scale on x axis
for (var x = 0; x < 880; x += 80) {
context.moveTo(x, 405);
context.lineTo(x, 400);
context.lineTo(x, 395);
}
for (var x = 0; x < 880; x += 80) {
var percent = x/8;
context.font = "bold 12px sans-serif";
context.fillText(percent, x,390);
context.closePath();
}

// draw Y axis
context.moveTo(0, 0);
context.lineTo(0, 400);
context.strokeStyle = "#000";
context.stroke();

//draw scale on y axis
for (var y = 0; y < 440; y += 40) {
context.moveTo(0, y);
context.lineTo(5, y);
context.strokeStyle = "#000";
context.stroke();
}
var percentLabel = 110;
for (var y = 0; y < 440; y += 40) {
if (percentLabel > 0) {
percentLabel = percentLabel -10;
}
var percent = y/4;
context.font = "bold 12px sans-serif";
context.fillText(percentLabel, 10, y);
context.closePath();
}

// Axis Labels
context.font = "bold 12px sans-serif";
context.fillText("X", 790, 380);
context.fillText("Y", 40, 10);

context.closePath();
```

```
//end Axis labels

// Axis label Arrows
// X arrow

context.moveTo(780, 380);
context.lineTo(750, 380);

context.moveTo(780, 380);
context.lineTo(770, 370);

context.moveTo(780, 380);
context.lineTo(770, 390);

context.strokeStyle = "#000";
context.stroke();

// Y arrow
context.moveTo(40, 20);
context.lineTo(40, 60);
context.moveTo(40, 20);
context.lineTo(30, 30);
context.moveTo(40, 20);
context.lineTo(50, 30);

context.strokeStyle = "#000";
context.stroke();
// origin
context.fillStyle = "#CC0033";
context.beginPath();
context.arc(0,400,5,0,Math.PI*2,true);
context.closePath();
context.fill();
// include servlet to populate points
// points need to be scaled to
<jsp:include page="/servlet/studentOverview.StudentOverviewHTML5Canvas" flush="true" />

}}
</script>
```

4 (e): Visifire Area Chart configuration:

```
<script type="text/javascript" src="Visifire2.js"></script>
<div id="AreaChart">
<script language="javascript" type="text/javascript">
var AreaChartXmlString = ''
+'<vc:Chart xmlns:vc="clr-namespace:Visifire.Charts;assembly=SLVisifire.Charts" Width="1000"
Height="450" BorderThickness="0" Theme="Theme1" View3D="True" Watermark="False"
LightingEnabled="True" ShadowEnabled="False" >'
+'<vc:Chart.Titles>'
// Chart Title Text
+'<vc:Title Text="53431274 GPA(%) Per Year vs. Continous Assessment" Enabled="True" />'
+'</vc:Chart.Titles>'
+'<vc:Chart.TrendLines>'
+'<vc:TrendLine LineColor="#FF0000" ToolTipText="Average Overall GPA %" Value="62"
Enabled="True" />'
+'</vc:Chart.TrendLines>'
+'<vc:Chart.AxesY>'
+'<vc:Axis Title="GPA %" Enabled="True" AxisType="Primary" />'
+'</vc:Chart.AxesY>'
+'<vc:Chart.Series>'
// Chart Declaration as Area for Y axis
+'<vc:DataSeries LegendText="GPA per Year (%)" RenderAs="Area" AxisYType="Primary" >'
+'<vc:DataSeries.DataPoints>'
// Y axis JSON String is read from session variable
<%=  session.getAttribute("VisifireOverallGPAByYear") %>
+'</vc:DataSeries.DataPoints>'
+'</vc:DataSeries>'
// Chart Declaration as Area for X axis
+'<vc:DataSeries LegendText="Continous Assessment" RenderAs="Area" AxisYType="Primary" >'
+'<vc:DataSeries.DataPoints>'
// X axis JSON String is read from session variable
<%=  session.getAttribute("VisifireCAGPAByYear") %>
+'</vc:DataSeries.DataPoints>'
+'</vc:DataSeries>'
+'</vc:Chart.Series>'
+'</vc:Chart>';
// Chart size is set to 1000 x 450 pixels
var AreaChart = new Visifire2("SL.Visifire.Charts.xap " , 1000 , 450 );
AreaChart.setDataXml(AreaChartXmlString);
// Chart is set to render as area chart
AreaChart.render("AreaChart");
</script>
```

Appendix 5 – Contents of accompanying CD

A CD is attached to this dissertation with the following contents:

- Web Application WAR file is in the Web Application WAR folder along with unpacked source code
- A SQL dump with table structure and stored procedures and a mySQL Database dump – with anonymised student records is in the mySQL Database folder
- A soft copy of this report is in the Report folder
- Browser plug-ins for some charts including Microsoft Silverlight, Adobe Flash and Adobe Reader are located in the BrowserPlugins folder
- A portable web server containing a fully functional and portable mySQL database used as part of this project. Browse to the USBWebServer folder and launch UsbWebserver.exe. This will start an instance of Apache running on port 8088. This portable distribution contains a mySQL database running on port 3306 populated with the studentdata database and all stored procedures used as part of this project. The ports both services are configurable in the easy to use menu. For more information see (85).