# Smart School Schedule
# Final Year Project Report

**DT211**
**BSc in Computer Science (Infrastructure)**

**James Wilson**
**Supervisor: John McAuley**

School of Computing
Dublin Institute of Technology

**08/04/2016**

# Abstract

Timetable management is a difficult problem that schools and universities face. There are attempts to provide scheduling of timetables for schools but as it is an NP Hard problem, it is not an easy task to undertake.

One of the project aims is to simplify this scheduling problem, optimizing timetables using the genetic algorithm. The idea behind this project is to schedule timetables for schools and overcoming the NP hard problem.

The project will be a web application, multi-device friendly so it can be used on any device using PHP as the server side language, and HTML, JavaScript, CSS, twitter bootstrap to develop the frontend of the application. Java will also be used to implement the system.

# Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

___________________________

James Wilson

C12527253

08/04/2016

# Acknowledgements

Firstly I would like to thank DIT for allowing me to be able to participate in this course. I would also like to thank my supervisor, John McAuley, for his patience, guidance and advice throughout the project. His feedback and suggestions helped me immensely and helped shape the project.

I would most like to thank my long term girlfriend Shauna and my 11 month son Caiden for their continued love and support throughout this project, their support gave me the motivation to work even when times were tough, this is for you.

I would also to thank my Mam, Dad, two sisters and two nieces for the support they gave me throughout my project.

Thank you all very much. I could not have done it without your assistance.

## Contents

# 1. Introduction

## 1.1 Project Statement

The aim of this project is to create an online secondary school timetable and system. Online school systems allow the administrators, to schedule, view and delete the school timetable.

This project is a Web Application Secondary School System which allows the easy use for administrators to handle timetables which are automated. This project works by detecting classes that are clashing, the teachers cannot teach more than one class at the same time, the students cannot be at more than one class at the same time.

The system will be a three tier system which consists of the presentation tier, the second tier is the domain logic tier and also a data storage tier. The presentation tier would consist of the client side ( the user interface). The logic tier will consist of all the manipulation in the unique secondary school database (all of the commands). Then finally the data storage tier, where the database contents and the user are linked by the server which is how the web application is hosted on the Internet.

The system will aim to address the NP-Hard problem of scheduling a timetable using the genetic algorithm as a proven approach to solving this problem.

## 1.2 Goals & Objectives

The goal of this project is to develop an online Secondary School System application. This section outlines and describes the aims of this project and the objectives that are used to achieve them.

- This system aims to allow the administrators  to be able to automatically create their school timetable with little effort and no worry of clashing of teacher's and student's classes and subjects.
- This system also allows the admin to be able to see all of the teachers, classroom and student classes there are in the system

To complete these system goals, the objectives within the system are very important:

1. A database has to be developed. The database will be created using MYSQL, the database will be manipulated by the Admin and also the teacher. In the data, there will be information on the relevant classes, the teachers, the teachers hours, the students, the subjects and the students grades.

2. A server must also be set up. This server will host the Web App, which will enable the users to be able to connect via the Internet. It is developed using a server-side language which will be PHP and it will be able to connect to client side (front end) to the database.

3. A front end/ client-side must also be created when developing this system. This is the actual visual aspect of the system, it is essential not be difficult to navigate and additionally the system should capture the users eye.

## 1.3 System Requirements

### 1.3.1 Functional & Non Functional Requirement

| Functional | Non-Function |
|---|---|
| 1. View/ Access Timetable | 1. Reliability |
| 2 Schedule Timetable | 2. Platform Compatible |
| 3. Add/ Delete/ View Teacher | 3. Cost |
| 4. Add/ delete/ View Subject | 4. Accessibility |
| 5. Add/ delete/ View class | 5. Security |
| 6. Add/ delete/ View Student | 6. Documentation |
| 7. Add/ delete/ View class | |

### 1.3.2 Requirements Matrix

| Requirement ID | Name of Requirement | Description | Priority | User Contact | Use Case Related Requirement |
|---|---|---|---|---|---|
| 1.1 | Access Timetable | All users must be able to view the timetable in the system | High | Admin | View/ Access Timetable |
| 1.2 | Generate Timetable | Admin generates the student timetable, has final decision on the scheduled timetable | High | Admin | Generate Timetable |
| 1.3 | Publish Timetable | Admin is able to publish the | High | Admin | Publish Timetable |
| 1.4 | Add/ delete view teachers | Teacher is able to publish homework for the student | High | Admin | CRUD operations on the teachers |
| 1.5 | Add/ delete/ view subjects | Teacher is able to publish grades for the student | High | Admin | CRUD operations on the subjects |
| 1.6 | Add/ delete / view classrooms | The teacher & student are able to view the homepage published | High | Admin | CRUD operations on the classrooms |
| 1.7 | Log in/ Log out | All users must be able to log in and out of the system | High | Admin | Log in/ Log out |

## 1.4 Project Challenges

### 8.1.1 Automated School Timetable System (NP-Hard Problem)

Automating a secondary school system using an algorithm is going to be a very daunting task. The school time tabling system has always a problem, and it is going to be a hard algorithm is crack as this timetabling system is an NP-Hard problem
Especially that timetable management is an NP- hard problem, overcoming this problem will be a difficult task. The Genetic algorithm will be the solution to solving these NP hard problem.

The Genetic algorithm used for scheduling that is required to implement the timetable schedule requires a lot of constraints. These constraints are classified into hard and soft constraints. The soft constraints consists of one teacher cannot be teacher more than one class at a time, a class cannot be in two classes at the same time, the same class cannot be sitting two different subjects at the one time. The hard constraints must also be identified, these constraints are the teacher has their preferred room to teach and the preferred time they teach.

### 8.1.2 Lack of familiarity with the Genetic Algorithm

Developing the application with no prior knowledge with the algorithm will be a hurdle which needs to be dealt with and to overcome. Creating  a complex algorithm to implement the solution is quiet a daunting task. With not much background on machine learning this will be difficult to overcome and come up with a solution.

## 1.4 Paper Structure

The paper structure covers the main points and objectives of the chapters in this section

**Chapter 1 -  Introduction**
This chapter covers the project statement,  goals and objectives of the system as well as functional and non functional requirements

**Chapter 2 -Background Research**
This chapter covers the background research that was conducted. Research about timetable management, the scheduling algorithm and also the NP hard problem will be discussed in this chapter.

**Chapter 3 - Similar Systems**
This chapter will cover research conducted in the similar systems. The system that have been chosen will be analysed using Nielsen's Heuristics, in order to grade the overall performance of the systems.

**Chapter 4 - Technology Review**
This chapter covers all the different technologies that could be used in relation to this project. It also discusses the technologies that were chosen and best suits the project.

**Chapter 5 - Design**
This chapter outlines and discusses the system's design and the design methodologies. Unified modelling language (UML) diagrams will be used to familiarize the reader with what actors are in the system and what are their purpose in the system. The database tables will be

presented using the Entity-relationship diagram, which will show the structure of the database.

### Chapter 6 - Implementation
The implementation chapter will cover all of the aspects of the coding and implementation of the project.

### Chapter 7 - System Testing & Evaluation
The system testing and evaluation chapter will discuss the testing methods. It also talks about the testing that has been implemented into the project.

### Chapter 8 - Project Plan
This chapter will provide information about the functionality that was not implemented in the final version of the system. It discusses why the functionality was not met, and what approach would be taken if starting the project over.

### Chapter 9 - Project Evaluation & Conclusion
The final chapter in the paper is the project evaluation and conclusion, it will provide an overview of the previous chapters. It will also indicate the key learning obtained in relation to the project. It also covers the future plans of the project and what can be done, as well highlighting the personal reflection.

## 1.7 Project Timeline

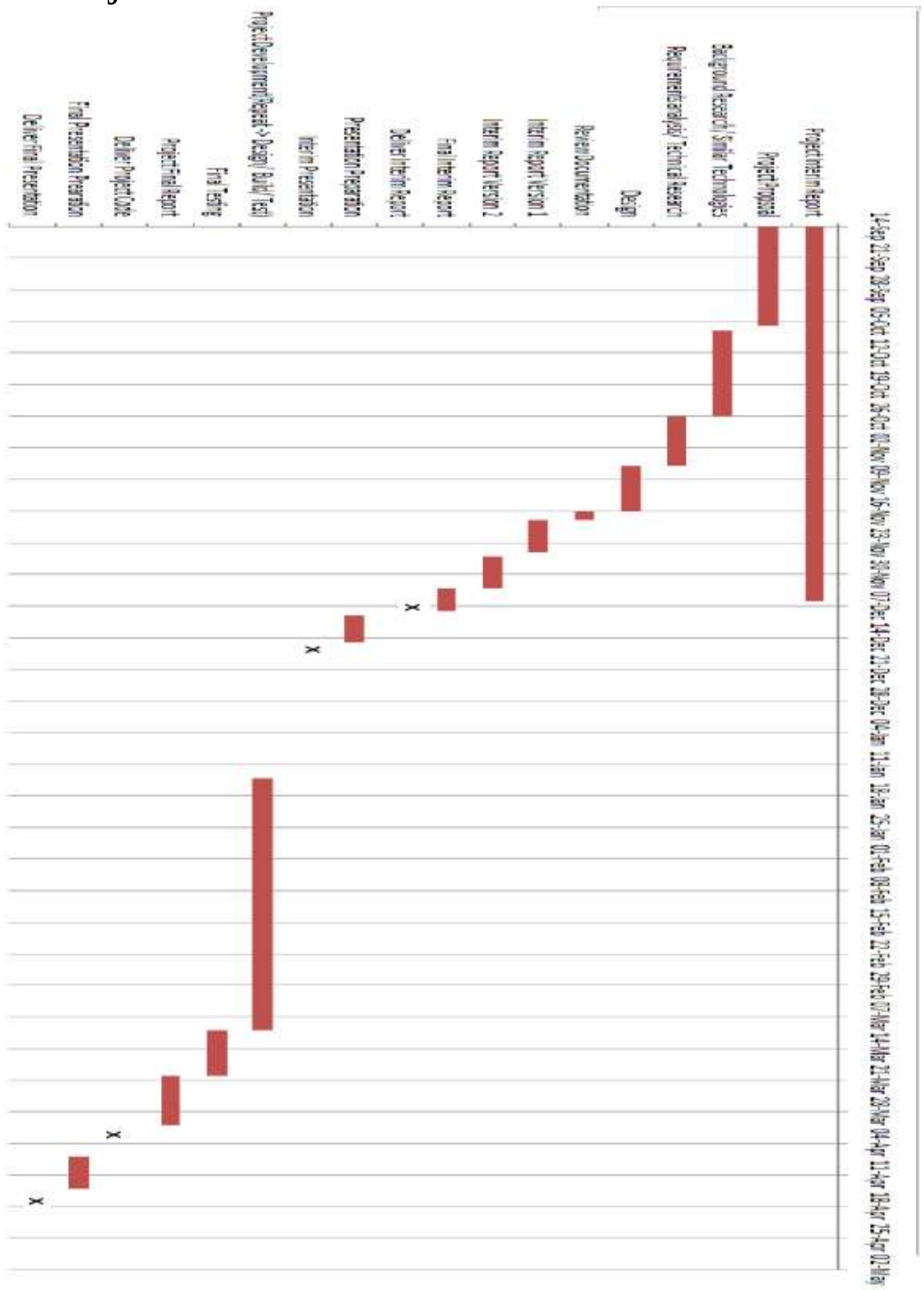


**Figure 1.1 Plan and Future Work**

### 1.7.1 Phase 1 - Research
Phase 1 involves carrying out detailed research on the project and making an analysis based on that research. The research that was conducted was background research, the genetic algorithm and also similar systems and technologies. The aim was to have all this research finished by the first week in November.

### 1.7.2 Phase 2 Design
Phase 2 involves designing the some of the web the pages in the system, identifying the system architecture that will be used and also creating a basic prototype for the system. This phase has to be met by the middle of November.

### 1.7.3 Phase 3 Interim Report Versions
This phase consists of documenting different versions of the interim report that is for reviewal and proof reading. This phase has to be completed by the start of December.

### 1.7.4 Phase 4 Preparing Interim Presentation
This phase involves preparing for the presentation thus providing the key aspects of the project for the presenting. This phase should be completed by the second week in December.

### 1.7.5 Phase 5 Project Development
This phase involves developing the project derived from the design phase. This phase should be completed by the middle of March.

### 1.7.6 Phase 6 Testing & Evaluation
Time allocation to testing is very important , as developing the project there will be testing and coding throughout the implementation phase but it is necessary to complete the final testing phase by the mid to end March.

### 1.7.7 Phase 7 Final Project Document
Phase 7 will involve completing the final project manual. This phase should be completed by the end of March and delivered in the start of April.

## 1.8 Conclusion
This chapter briefly outlines the aim of the project, in terms of the requirements both functional and non functional. This chapter also provides an overview of what the project should provide in terms of the goals and objectives that are outlined. Details of the challenges associated with the project is outlined and briefly discussed. This introduction chapter provides details about the layout of the document and what will be outlined in each chapter. And lastly, the project timeline was presented, this describes the timeline for the interim phase such as the project proposal, the background research, the similar systems and technologies, the interim report version as well as the development phase, the testing and evaluation phase, the final project document and the final presentation. The following chapter discusses timetable management, the NP hard problem and the Genetic algorithm.

# 2. Background Research

## 2.1 Introduction

The following chapter discusses the background to educational systems and the difficulties of a secondary school creating a timetable for their school. A brief insight into the problems with making a school timetable and also history on the genetic algorithm used to solve this problem.

## 2.2 How technology is changing Education

In the 21st century, technology has changed the ways in which people communicate and go about their lives and it has had a major impact when it comes to secondary schools.
Schools have purchased huge amounts in terms of technology both hardware and software. Children and young adults are very enthusiastic about technology nowadays and are becoming increasingly skilled in using technology as they spend a lot of time using digital technology for entertainment, socializing etc. All of the technology developments were strategically linked to the school improvement plan and utilized to improve educational outcomes, as teachers described how the use of technology has helped pupils to develop their skills and become more inquisitive [1]. Technologies in schools allows for instant feedback from the teachers which enable the students to improve their work. Nowadays there are interactive textbooks which often web based sites including assessments, videos, animations and other materials to support the learning of new content. The use of eBooks are on the steady rise as many schools adapting with the advancements of e-readers and tablet computers, so in a few years big heavy bags will not be an issue [2].
In the medieval times, books were rare and only few had access to education, now with the use of technology there are massive amounts of information such as books audio images or videos are available at one's click of a button via the web, there are numerous opportunities for learning online such as Khan Academy, MOOCs, online degree programs and many more. Technology is a very powerful tool that supports and transforms education, from making it easier for teachers to create instructional materials that enables new ways for people to learn and work together [3].
Technology is offering exciting opportunities for schools and its way of teaching and learning for teachers and students. It is shaping education and giving students to take part in the future of innovation.

## 2.3 Timetable Management

Timetabling and timetabling systems seems to be a very time consuming to set up for secondary schools alike. A school timetable is setup to accommodate students, teachers and the classrooms. These timetables are very important for the secondary school administration, as it gives the teachers their working hours, what classes they are teaching and it also gives the students where and when to be in for their appropriate classes.

## 2.4 NP Hard Problem

The NP-Hard problem is the complexity of decision problems that are harder than those that can be solved by a nondeterministic Turing machine in polynomial time. When the decision version of the optimization problem is proved to belong to the class of NP-complete problems, then the optimization version is NP-Hard.
The timetable management project is an NP- Hard problem project. This means that there is no known way to locate a solution in the first place, there is no fast known solution to this problem. The reason why scheduling a timetable system is NP- Hard is because of all the constraints. The more constraints within the timetable the harder it is to produce the timetable,

even without using a computer program, it is impossible to solve the scheduling of a school timetable. The more constraints in the timetable the more complex the timetable gets.

## 2.5 The Genetic Algorithm

Genetic Algorithms provide a proven approach to addressing this NP-Hard problem which exists in timetable management. Normal heuristic search algorithms solve the problem of the timetabling management, but only for simple cases. The genetic algorithm comes into play when the inputs and requirements are more complex.
In the 1950's and 1960's several computer scientists studied the evolutionary systems with the intentions that evolution could be used an as optimization tool for engineering problems. The idea of all these systems was to evolve a population of candidate solutions to a given problem, using operators which are inspired by natural genetic variation and natural selection. The Genetic Algorithm was first invented in the 1960's by John Holland. His goal was not to design an algorithm to solve problems, but his goal was to study the phenomenon of adaptation as it occurs in nature and to develop the mechanisms of natural adaptation and how it might be imported into computer systems [4].
The Genetic Algorithm can be used for solving this scheduling problem. It is capable of solving the most complex of search  spaces providing optimal solutions. The Genetic Algorithm is a heuristic search mimicking the process of natural selection, this heuristic search (Genetic Algorithm) creates a solution for the optimization problem for the timetabling system.
The genetic algorithm in its simplest form is divided up into three types of operators known as

- Selection
- Crossover
- mutation

When implementing the genetic algorithm the first operator that is used is the selection operator. This operator select the chromosomes in the population for reproduction. The fittest chromosome is selected to reproduce. The crossover operator randomly chooses a locus and exchanges the subsequence's before and after that locus between two chromosomes to create two offspring  [4]. The mutation operator maintains the genetic diversity from one generation of a population to the next.
There are other different ways to solve this NP-Hard problem such as dynamic programming, brute force and backtracking but these methods are not as efficient as the Genetic Algorithm as the Genetic Algorithm is able to achieve a good quality solution for the scheduling problem.

## 2.6  Conclusion

This chapter has provided an insight to what the project idea is and what the outcome of the project should be. Outlining a brief background to secondary school systems, timetable management and also to the genetic algorithm and the reasoning behind its selection.

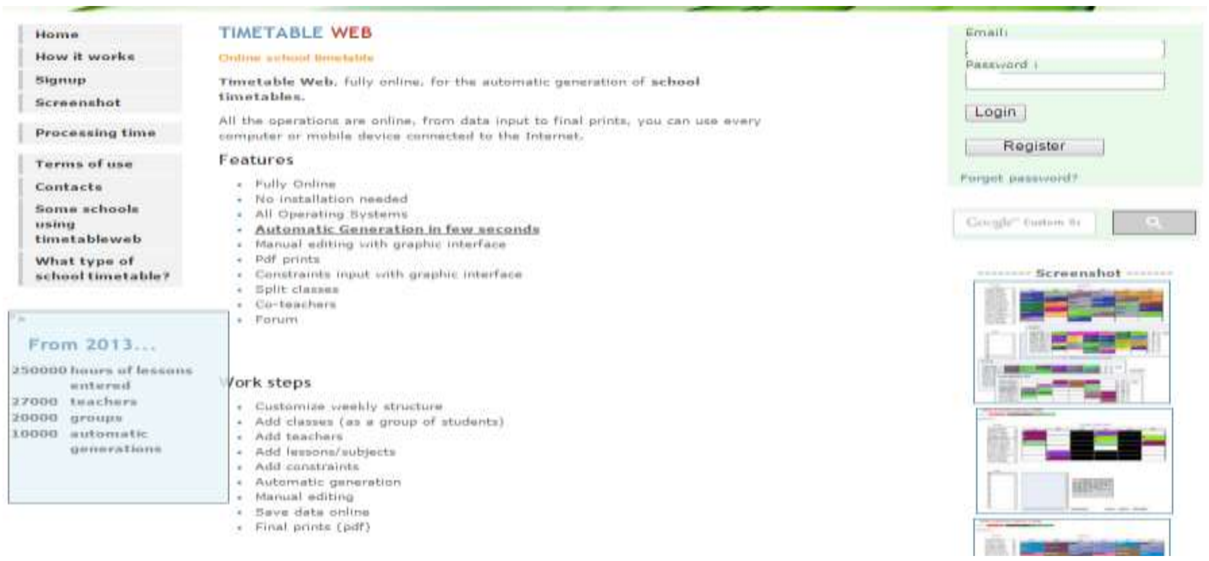
# 3. Similar Systems

## 3.1 Introduction

An evaluation of three similar systems will be identified to determine and identify the user requirements for the proposed system. Three similar systems that I will be evaluating is timetableweb, schedulebuilder and also gradetracker. These three systems will allow me to examine how the user interacts with the systems, what the user requirements are, and how the system will be designed and also how the system can be implemented. The evaluation method that will be used will be Jakob Nielsen's Heuristics for User Interface Design.
There are ten principles for the interaction design are(5):
**(1) Visibility of system status**: The users using the system must be informed about what is going on, this is done through appropriate feedback within reasonable time.
**(2) Match between system and the real world:** The system must be able to speak the language of the user, the wording and phrases must be at an understandable level. The information must appear in a natural and logical order.
**(3) User control and freedom:** Users can of the choose system functions by mistake and will need a clearly marked exit that would be able to leave an unwanted state.
**(4) Consistency and standards:** Users should not get confused if that they think an action or situation means the same thing.
**(5) Error Prevention:** Good error messages prevents a problem for happening in the first place with the user. Either present the user with a confirmation message before they
**(6) Recognition rather than recall:** Minimise the users memory load by making objects, options and actions visible. The user should not have to remember information from one dialog to another. The system instructions should be visible.
**(7) Flexibility and efficiency of use:** Accelerators can sometimes speed up the interaction for the experienced user such that the system is able to cater both the inexperienced and experienced users.
**(8) Aesthetic and minimalist design:** The system dialogues should not contain information which is rarely need or somewhat irrelevant.
**(9) Help users recognize, diagnose, and recover from errors:** Error messages should be expressed in plain language, to indicate the problem and suggest a solution
**(10) Help and documentation:** Necessary to provide help and documentation to the user. This information should be easy to search, focused on the tasks of the user, list concrete steps that have to be carried out and it should not be too large.

## 3.2 Timetable Web

The first system that will be discussed is the timetable web system. It is an online system were the user is able to create a time table online. It can viewed at www.timetableweb.com/index.php. The interface design looks relatively simple from a visual perspective, it is not the most pleasing to the user as the text is very small and also difficult to navigate.



TIMETABLE WEB

Online school timetable

**Timetable Web**, fully online, for the automatic generation of **school timetables**.

All the operations are online, from data input to final prints, you can use every computer or mobile device connected to the Internet.

Features

- Fully Online
- No installation needed
- All Operating Systems
- Automatic Generation in few seconds
- Manual editing with graphic interface
- Pdf prints
- Constraints input with graphic interface
- Split classes
- Co-teachers
- Forum

From 2013...

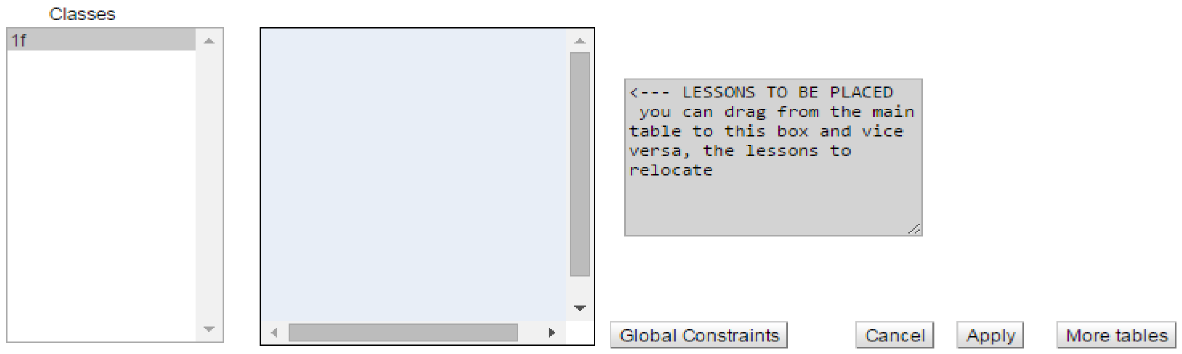
| | |
|---|---|
| 250000 | hours of lessons entered |
| 27000 | teachers |
| 20000 | groups |
| 10000 | automatic generations |

Work steps

- Customize weekly structure
- Add classes (as a group of students)
- Add teachers
- Add lessons/subjects
- Add constraints
- Automatic generation
- Manual editing
- Save data online
- Final prints (pdf)

**Figure 3.1 Screenshot of timetable web**

## Visibility of system status



<--- LESSONS TO BE PLACED you can drag from the main table to this box and vice versa, the lessons to relocate

**Figure 3.2 Visibility of system status**

The system gives the user a clear indication of where the lessons can be placed, that the user can easily just drag from the main table to the box indicated. (See figure 3.2) **Score 9/ 10.**

## Match between system and the real world

The language used in the website is easy to understand. The system gives the user clear instructions and the phrases are easy to understand for example to add a teacher is very easy as the user just has to go the tab to add a teacher and simply click a button. **Score 9/ 10.**

## User control and freedom

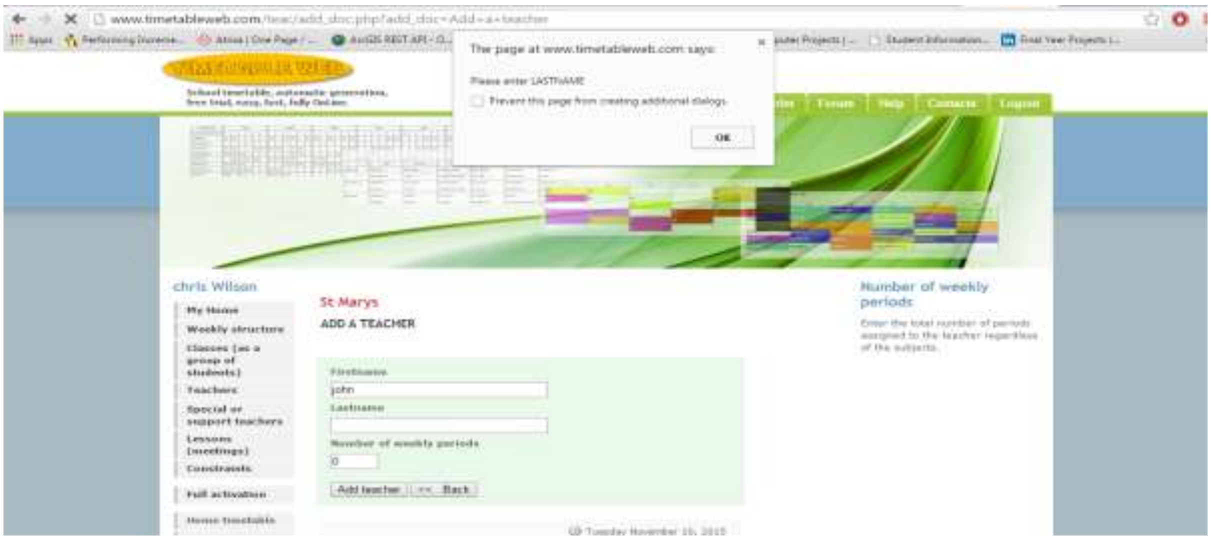
The system tries to make sure the user know exactly what he/ she is doing, for example the home screen is very clear and shows the user their summary of their timetable, i.e. how many teachers there is, how many classes and if their timetable is generated or not. **Score 8/ 10.**

## Consistency and standards

The system indictors are consistent, the user is always aware where they are within the system and the tabs allow for quick navigation for the user. **Score 6/10**

## Error Prevention

The system's error message are very simple for the user, and it is very to clear and to the point.


The page at www.timetableweb.com says:

Please enter LASTNAME

Prevent this page from creating additional dialogs

OK

chris Wilson

St Marys

ADD A TEACHER

Firstname: john

Lastname:

Number of weekly periods: 0

Number of weekly periods

Enter the total number of periods assigned to the teacher regardless of the subjects.

**Figure 3.3 Error Prevention**

As seen in figure 3.3 the use of error prevention is incorporated into the system and is very visible to the user and it is easy to understand what error was inputted. Here the system is telling the user that the he/ she should enter in the last name and it cannot be blank. But there is a massive flaw within the system error messaging and prevention, once the user checks the checkbox in the alert box the error messaging becomes non-existent. This is the reason for its very low scoring. **Score 3/10**

**Recognition rather than recall**

The system has clearly labelled functions, again it is clear and easy to understand for the user. **Score 7/10.**

**Flexibility and efficiency of use**

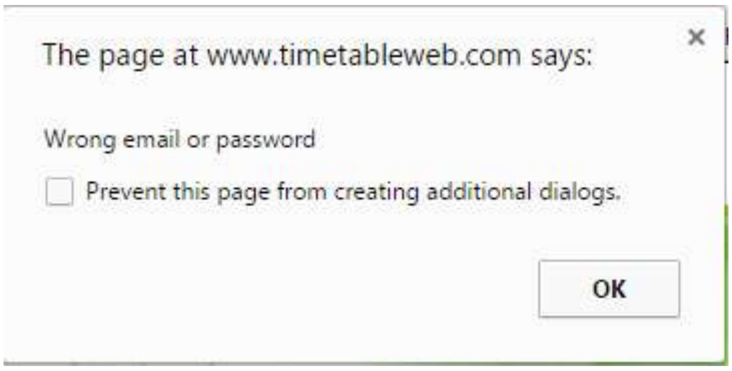
The system can be confusing at first to a new user. But once the user gets familiar with the system it becomes easier to use. It is still user friendly but at times the interface can be difficult to use. **Score 4/10.**

**Aesthetic and minimalist design**

The system dialogues rarely contain any information that is not needed, the information provided is very much relevant to what the system is. The system's important information is displayed on the screen with no need of scrolling up or down to visit any important additional information. **Score 7/10.**

**Help users recognize, diagnose, and recover from errors**

The system error messages used are in plaintext and it is very clear to the user. As you can see in Figure 3.4 the user has entered in an incorrect username or password, but there is a great downfall in this system that these error messages can be avoided by the user checking the checkbox, but defeats all the error messages in the system and it becomes non error existent. **Score 4/10**



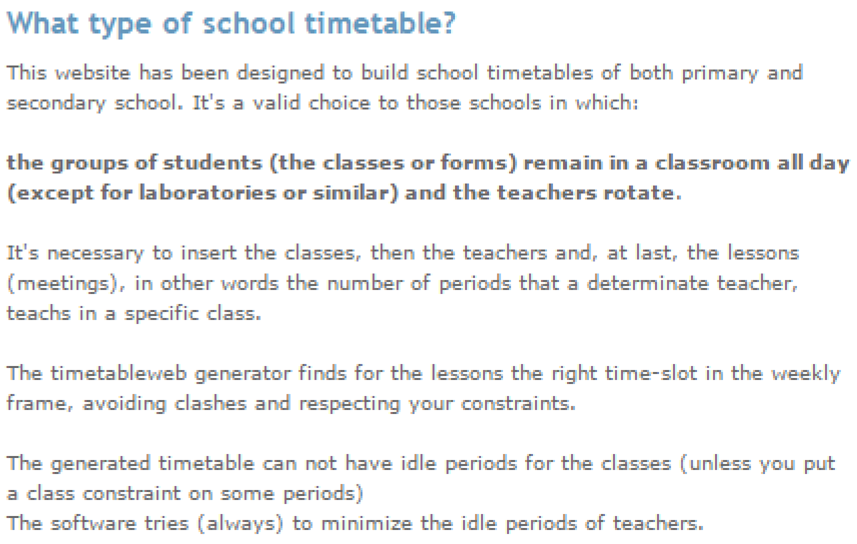
The page at www.timetableweb.com says:

Wrong email or password

☐ Prevent this page from creating additional dialogs.

**Figure 3.4 Help users recognize, diagnose, and recover from errors**

**Help and documentation**
Necessary to provide help and documentation to the user. This information should be easy to search, focused on the tasks of the user, list concrete steps that have to be carried out and it should not be too large.
The help and documentation is easy to search for, for the user they simply can just click the tab 'What type of school timetable' for example, and it clearly provides the information of what the user is looking for, see figure 3.5. **Score 9/10**



What type of school timetable?

This website has been designed to build school timetables of both primary and secondary school. It's a valid choice to those schools in which:

**the groups of students (the classes or forms) remain in a classroom all day (except for laboratories or similar) and the teachers rotate.**

It's necessary to insert the classes, then the teachers and, at last, the lessons (meetings), in other words the number of periods that a determinate teacher, teachs in a specific class.

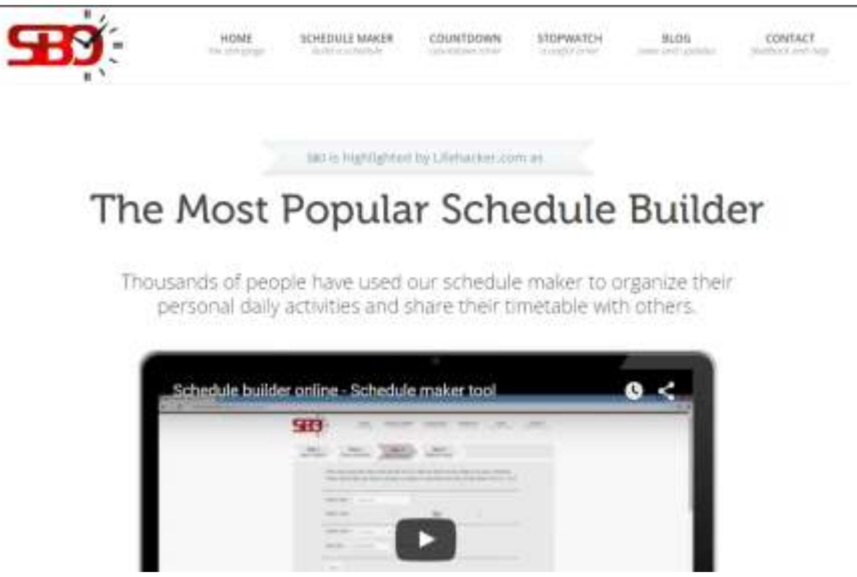
The timetableweb generator finds for the lessons the right time-slot in the weekly frame, avoiding clashes and respecting your constraints.

The generated timetable can not have idle periods for the classes (unless you put a class constraint on some periods)
The software tries (always) to minimize the idle periods of teachers.

**Figure 3.5 Help and documentation**

## 3.3 Schedule Builder

Schedule Builder can be found at (http://schedulebuilder.org) . Schedule Builder is an online timetabling system whereby the user can create an online timetable. The homepage of the system is very pleasing to the user's eye and very attractive as seen in figure 3.6. The use of images and the fonts chosen greatly enhances the appearance of the system.



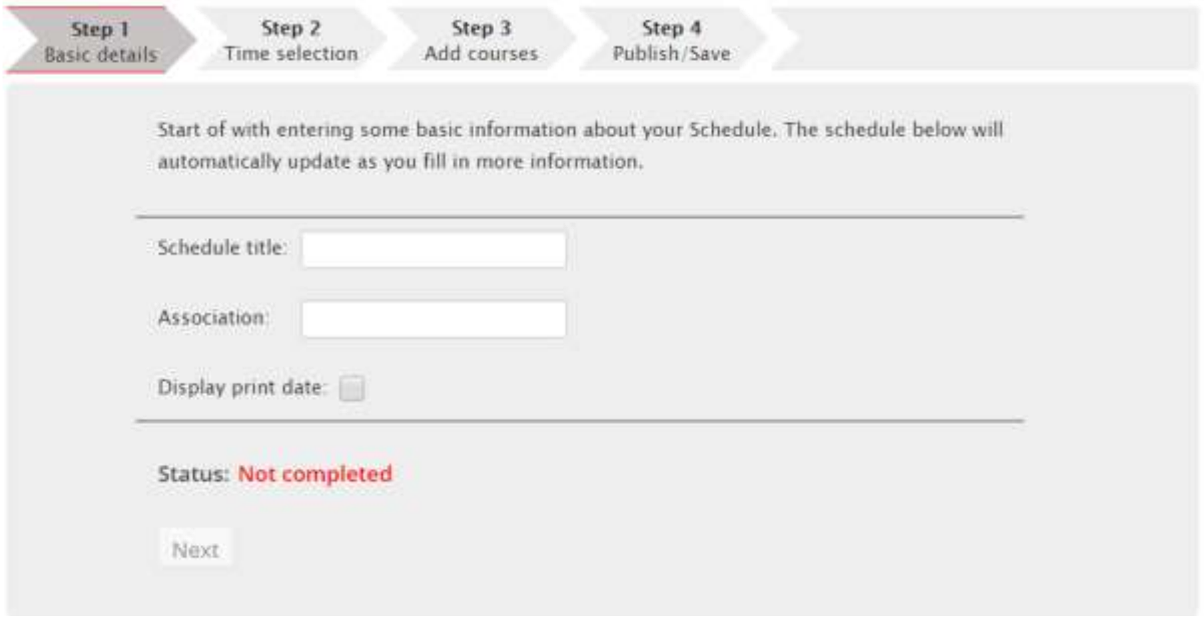
SBO is highlighted by Lifehacker.com as

The Most Popular Schedule Builder

Thousands of people have used our schedule maker to organize their personal daily activities and share their timetable with others.

**Figure 3.6 Schedule Builder Homepage**

**Visibility of system status**
This system tells the user of what exactly is going on and where they stand. The system clearly informs the user that in the process of making their timetable that a step is not completed. So the user knows that they have to fill in the all the fields until the status has changed to complete so they can move on to the next step. (Figure 3.7 below) **Score 9/10**



Start of with entering some basic information about your Schedule. The schedule below will automatically update as you fill in more information.

Schedule title:

Association:

Display print date:

Status: Not completed

**Figure 3.7 Visibility of system status**

**Match between system and the real world**
The language the system uses is very easy for the user to understand. Each function in the system can be clearly identified by the user. For example when the user clicks on 'Create a Schedule Now' it bring the user to the creation screen of the timetable/ schedule they are creating. (See figure 3.8) **Score 9/10**

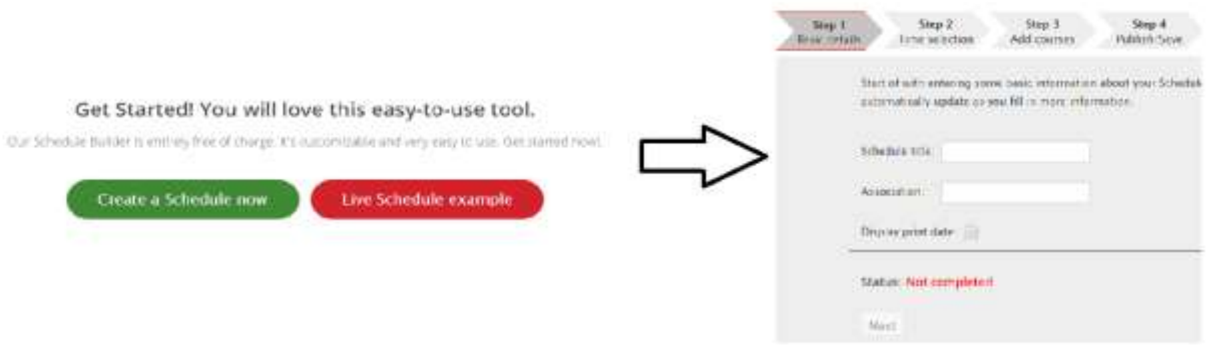


**Figure 3.8 Match between system and the real world**

**User control and freedom**
The system doesn't have too many clearly marked exits, if the user makes a mistake in step 1 of building a timetable the system has no redo or back button the user has to click the step 1 in order to go back from step 2. **Score 5/10**

**Consistency and standards**
The terms that are used throughout the system are consistent. This makes it easy for the user not to get lost within the system. The navigation tabs are clearly labelled and identified. **Score 10/10**

**Error Prevention**
The use of error prevention used within the system are very good. The is identified if they had made an error within the system, and the system also notifies the user if what needs to be completed before continuing to the next step. **Score 10/10**

**Recognition rather than recall**
All the required functionality is stored within the system, it is easily understandable for the user. **Score 9/10**

**Flexibility and efficiency of use**
This system allows for new time users with its basic steps and clear information. **Score 8/10**

**Aesthetic and minimalist design**
There is an issue with the homepage of the system, relating to creating a timetable and being able to see the live preview. The user has to scroll down to see these buttons, these buttons should be more striking to the user and it should be in the middle of the page when the user visits the page (See below Figure 3.9) **Score 6/10**

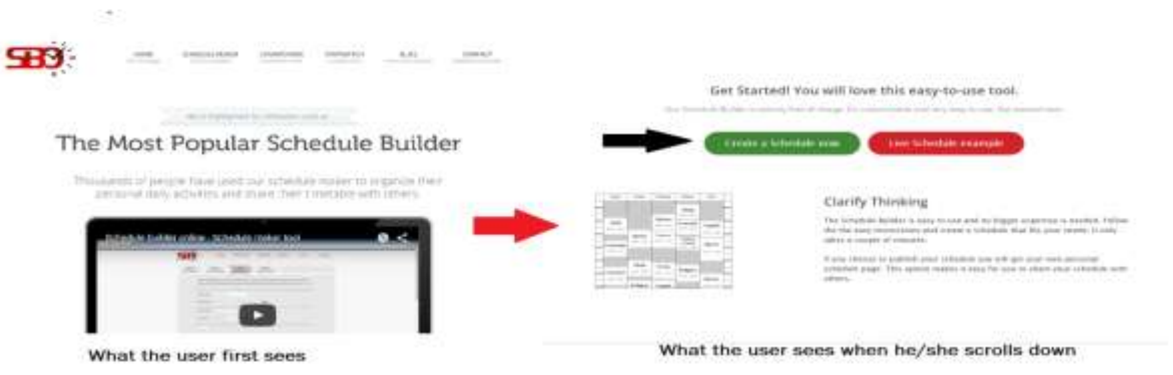

The Most Popular Schedule Builder
What the user first sees
Get Started! You will love this easy-to-use tool.
Clarify Thinking
What the user sees when he/she scrolls down

**Figure 3.9 Aesthetic and minimalist design**

**Help users recognize, diagnose, and recover from errors**
This system performs well in this area, this system is designed to help users to diagnose from their errors. The error messages that are displayed by the system is easy to understand and also it helps users to recognize these errors as it easy to see the error messages (see figure 3.10). **Score 7/10**

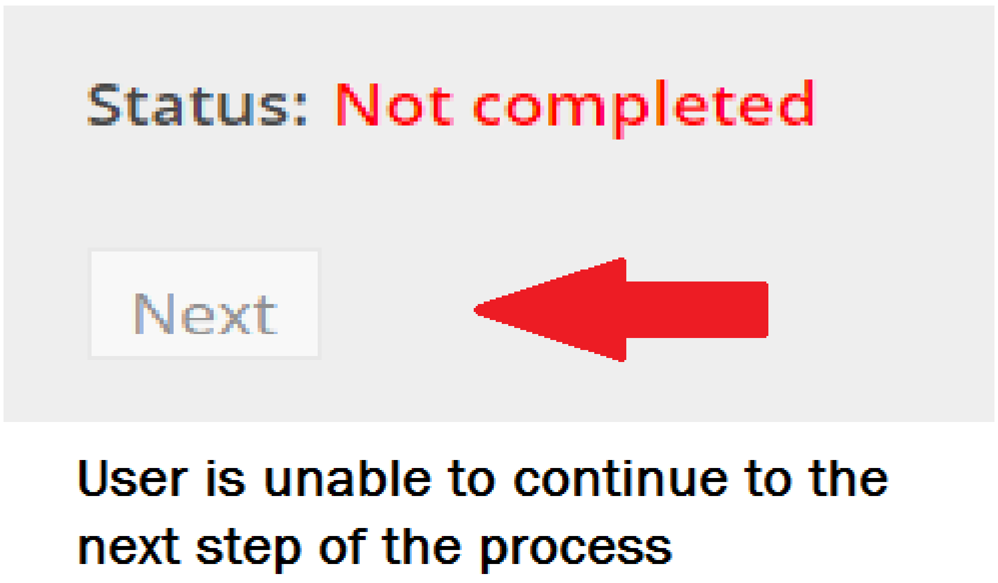


**Figure 3.10 Help users recognize, diagnose, and recover from errors**

**Help and Documentation**
The system does not have that much help and documentation, but it does have step guidance to ensure the user is entering in the correct information into the system (See figure 3.11). **Score 5/10**

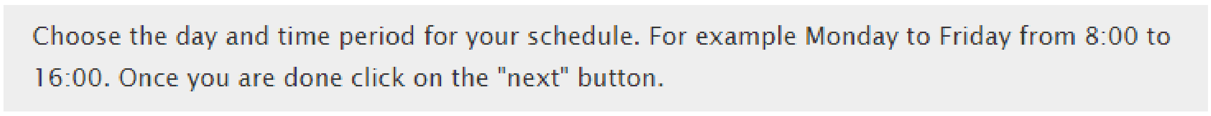
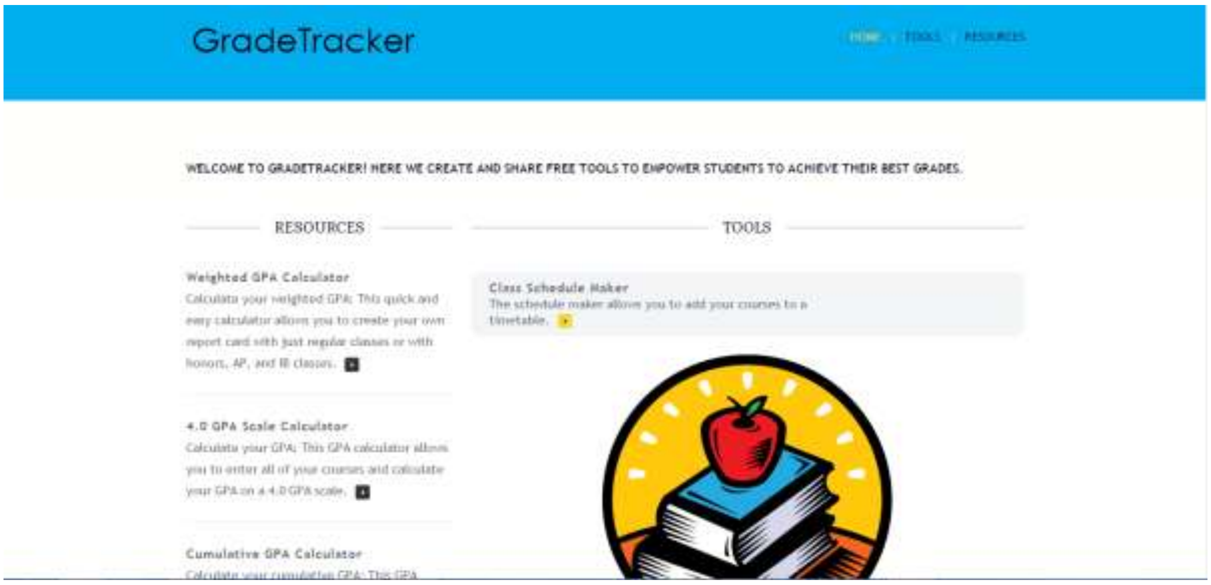

Choose the day and time period for your schedule. For example Monday to Friday from 8:00 to 16:00. Once you are done click on the "next" button.

**Figure 3.11 Help & Documentation**

## 3.4 Grade Tracker

Grade tracker (http://gradetracker.com/) is a responsive web application where students are able to keep track of their grades using specific tools in the web application, they are also able to create their own timetable. The web application doesn't require any authentication, anyone from the public is able to access the tools that are available at  gradetracker.com. The student/ user is able to keep track of their grades and also personalize their timetable. The homepage of the system is simple and there is no amazing special effects (see figure 3.12)



GradeTracker

HOME / TOOLS / RESOURCES

WELCOME TO GRADETRACKER! HERE WE CREATE AND SHARE FREE TOOLS TO EMPOWER STUDENTS TO ACHIEVE THEIR BEST GRADES.

RESOURCES

Weighted GPA Calculator
Calculate your weighted GPA: This quick and easy calculator allows you to create your own report card with just regular classes or with honors, AP, and IB classes.

4.0 GPA Scale Calculator
Calculate your GPA: This GPA calculator allows you to enter all of your courses and calculate your GPA on a 4.0 GPA scale.

Cumulative GPA Calculator
Calculate your cumulative GPA: This GPA

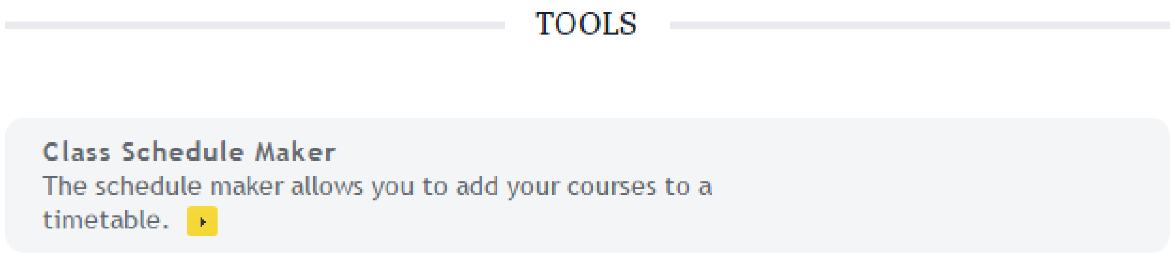
TOOLS

Class Schedule Maker
The schedule maker allows you to add your courses to a timetable.

**Figure 3.12 Grade Tracker Homepage**

**Visibility of system status**
The system keeps the user of what is going on throughout the system, it keeps the users informed on what is going on. **Score 7/10**

**Match Between system and real world**
The system uses easy understandable language that allows the user to easily understand where to go as each function is clearly defined, if the user wishes to create a timetable schedule they would simply click on Class Schedule Maker (see figure 3.13). **Score 8/10**

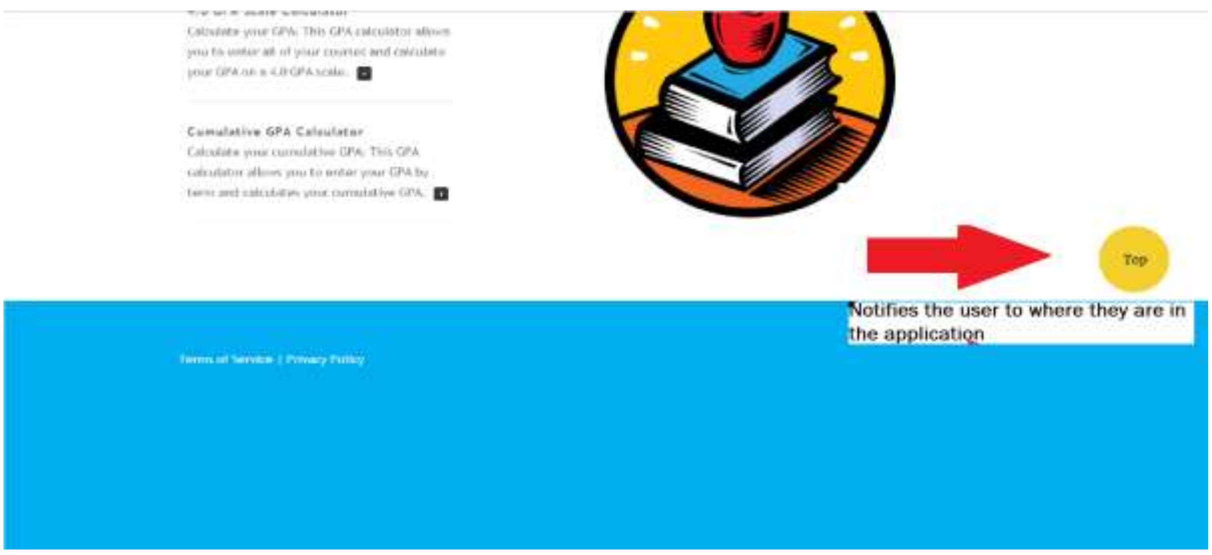


TOOLS

Class Schedule Maker
The schedule maker allows you to add your courses to a timetable.

**Figure 3.13 Class Schedule Maker**

**User control and freedom**
The system uses control and freedom for the user, for instance if the user scrolls down, the system displays a top bottom so that the user can easily go back to the top of the application instead of scrolling up themselves it also lets the user where they are in the system (see figure 3.14). There is also some flaws in the systems user control, if the user adds a new subject to their timetable but click the back button the subject is delete and is not updated this is a major issue in the system to solve this problem a confirmation message should be implemented to notify the user to ask the user are they sure to go back without committing changes. **Score 7/10**



Calculate your GPA. This GPA calculator allows you to enter all of your courses and calculate your GPA on a 4.0 GPA scale.

**Cumulative GPA Calculator**

Calculate your cumulative GPA. This GPA calculator allows you to enter your GPA by term and calculates your cumulative GPA.

Top
Notifies the user to where they are in the application
Terms of Service | Privacy Policy

**Figure 3.14 User Control & Freedom**

### Consistency and standards

The terms that are used throughout the system are consistent maintaining the same structure such as tabs for quick location change at the top of the page. **Score 9/10**

### Error prevention

There is not much error prevention in this system. If the user makes a mistake within the system which they didn't intend to do they are not notified of their error. **Score 4/10**

### Recognition rather than recall

The functions with the system is easily understandable for the user, the functions are clearly indicated for the user. **Score 8/10**

### Flexibility and efficiency of use

The system is simple to understand, it caters for inexperienced users as the functionality is not complex, it gives the user step by step instructions to help them understand certain concepts of the system. **Score 6/10**

### Aesthetic and minimalist design

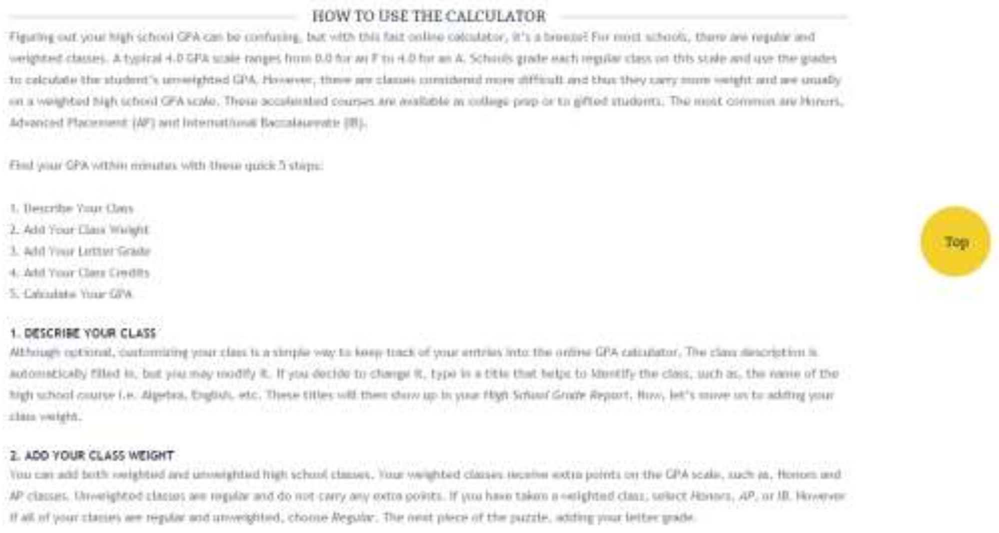
This system scores well in this area. The application is not clustered and it a simplistic design. The system clearly displays all the functionality of the system on the homepage and there is no irrelevant information throughout the application. **Score 9/10**

### Helps users recognize, diagnose and recover from errors

The system does not perform well in this area for example the user is able to add an empty slot for their timetable schedule where the system should notify the user that they must enter in a subject. **Score 4/10**

### Help and documentation

This system performs well in this area. The system clearly explains every functionality in the system for example the systems tells the user how to use the average grade calculator (see figure 3.15). This gives  the user a step by step guidance of what needs to be done in order to complete their intended function, this allows users of any level to complete tasks efficiently without making mistakes. **Score 10/10**


HOW TO USE THE CALCULATOR

Figuring out your high school GPA can be confusing, but with this fast online calculator, it's a breeze! For most schools, there are regular and weighted classes. A typical 4.0 GPA scale ranges from 0.0 for an F to 4.0 for an A. Schools grade each regular class on this scale and use the grades to calculate the student's unweighted GPA. However, there are classes considered more difficult and thus they carry more weight and are usually on a weighted high school GPA scale. These accelerated courses are available as college prep or to gifted students. The most common are Honors, Advanced Placement (AP) and International Baccalaureate (IB).

Find your GPA within minutes with these quick 5 steps:

1. Describe Your Class
2. Add Your Class Weight
3. Add Your Letter Grade
4. Add Your Class Credits
5. Calculate Your GPA

1. DESCRIBE YOUR CLASS

Although optional, customizing your class is a simple way to keep track of your entries into the online GPA calculator. The class description is automatically filled in, but you may modify it. If you decide to change it, type in a title that helps to identify the class, such as, the name of the high school course i.e. Algebra, English, etc. These titles will then show up in your *High School Grade Report*. Now, let's move on to adding your class weight.

2. ADD YOUR CLASS WEIGHT

You can add both weighted and unweighted high school classes. Your weighted classes receive extra points on the GPA scale, such as, Honors and AP classes. Unweighted classes are regular and do not carry any extra points. If you have taken a weighted class, select *Honors*, *AP*, or *IB*. However if all of your classes are regular and unweighted, choose *Regular*. The next piece of the puzzle, adding your letter grade.

**Figure 3.15 Help and documentation**

## 3.5 Conclusion

| **Heuristic** | **System 1** | **System 2** | **System 3** |
|---|---|---|---|
| | **Score /10** | **Score /10** | **Score /10** |
| Visibility of system status | 9 | 9 | 7 |
| Match between system and the real world | 9 | 9 | 8 |
| User control and freedom | 8 | 5 | 7 |
| Consistency and standards | 6 | 10 | 9 |
| Error prevention | 3 | 9 | 3 |
| Recognition rather than recall | 7 | 9 | 8 |
| Flexibility and efficiency of use | 4 | 8 | 6 |
| Aesthetic and minimalist design | 7 | 6 | 9 |
| Help users recognize, diagnose, and recover from errors | 4 | 7 | 3 |
| Help & Documentation | 9 | 5 | 10 |
| **Total Score** | **66** | **77** | **70** |

Having evaluated the three chosen systems using Nielsen's heuristics, we are able to see what the user would want from the online application and what the user would not want. System 2 scored highly with good use of error preventions and consistency and standards compared to the other two systems. The systems all scored well in ' Aesthetic and minimalist design' and this is very important when it comes to the user and his/ her interaction. The systems also scored well in the ' Match between system and the real world' which is a very important heuristic to take into account as the language used in the system must be very easy to understand for the user.
By taking into account the information gathered by Nielsen's heuristics it is easy to gather that the user requires an interface which is simple, uncluttered and pleasing to the user's eye. The design of the system will look to address in relation to user input, error detection, error recovery which is weak in some of the similar systems. The menus should follow a set standard in the system so it does not confuse the user, also the system should also be aware of the current status of the system.

Having gathered information on the user requirements and what to accomplish to meet their demands the next section will talk about the technologies that have been researched and the technologies used to develop and deliver the system.

# 4. Technology Review

## 4.1 Introduction

In this section I will be reviewing the technologies that I am going to be implementing into my system and also the similar technologies that I have not chosen for this project but serve the same purpose. The technologies I will be reviewing will both be server side programming languages and client side programming languages as I am developing a web application. Also I will be discussing the different server of which I could of used in developing this system.

## 4.2 Server-side Languages

In recent years server-side languages where used to reduce the client-side processing of web data by moving some of the pre-processing to the web server [6].
The server side language that I have chosen is PHP, I will also talk about three other server-side languages that could of been implemented into this system.

### 4.2.1 PHP

PHP (Hypertext Pre-processor) is a scripting language that is HTML-embedded written in 1994. PHP is mainly focused on server-side scripting such as generating dynamic page content, sending and receiving cookies, but it also has the ability to do much more such as command line scripting (making PHP run without any server or browser) and also writing desktop applications [7].

### 4.2.2 Python

Python was first created by a developer known as Guido von Rossum in 1990. The name 'Python' was named after the *Brit-com Monty Pythons Flying Circus*[8]
Python can be a server-side scripting language. The syntax is extremely different is that of PHP. Python is able to output HTML just like other languages can, but Python is more commonly used as a module rather than being intertwined like PHP[9].

### 4.2.3 Ruby on Rails

Ruby like Java and C is a general purpose language even though it is best known for web programming, it was first created 20 years ago by Yukihiro "Matz" Matsumoto [10]. Rails is a library that extends the Ruby programming language and it was first created by a developer known as David Heinemeier Hansson in 2003, from there it has been further implemented and extended by the Rail core team (over 4,000 contributors).
Ruby on Rails is able to run on most Web Servers that support CGI. The rails framework supports POSTgreSQL, MYSQL, SQLite, SQL Server, DB2, and Oracle [11].

## 4.3 Client-side Languages

Client side scripting is the front end development is everything is visible to the human eye, it's the interaction between the user and the system. In this section, some client side technologies will be reviewed and the client side technologies that were chosen

### 4.3.1 JavaScript

JavaScript is a script/ programming language from Netscape. JavaScript can be embedded in HTML pages and it is interpreted by the client [12]. JavaScript was the most widely used programming language in 2015, as this language is capable of several functions such as editing the content within on the HTML page and also being able to control the browser as well as being compatible on all browsers.
The main features of JavaScript is that it is structured, meaning that is it a highly structured language with proper and planned syntax that came from C, it is dynamic as it enables the developer to test the type of an object in many different ways and it is functional as all functions

### 4.3.2 VB Script

VB script, its full name Microsoft Visual Basic Scripting Edition language is a simplified version of the Visual Basic and Visual Basic for applications family of programming languages and it is considered to be very closely related to the BASIC programming language [13]. VBScript is the default language that is used in Active Server Pages (ASP).
VB script can be compared to different scripting languages that are used on the web such as:

- Sun Microsystems Tcl
- IBM's REXX
- Netscape's JavaScript

[14].
There is a very good platform of coverage when developing using VBScript, it is able to run in many environments. It can be used both for client-side and server-side scripting. Unlike object oriented programming languages such as Python and Java, VBScript is an object-based oriented language meaning that you cannot inherit from other classes which can be a disadvantage of using VBScript.

## 4.4 Data Storage

### 4.4.1 MYSQL

MYSQL is an open source relational database management system (RDBMS). It is one of the most used relational database system in the world. It is very popular with web applications, LAMP (Linux) and XAMPP(Windows) use MYSQL. The MYSQL database server is reliable, fast and also scalable, it can run comfortably on laptops and desktops. A web application that uses MYSQL, includes web pages that have some sort of access to a database. MYSQL is a client/ server system, it consists of a multi-threaded SQL Server. The server supports many different back end's and many different client programs [15].

### 4.4.2 Firebase

Firebase is a scalable is a real-time backend for an application. Developers are able to build applications without writing server side code or managing servers. It supports all the main browsers such as Google Chrome and Firefox, Safari etc.

Data from Firebase is stored as JSON, so it is NOSQL. This means that the database is not a relational database. At a high level you are able to store any type of data in the Firebase database, it ranges from game state to chat messages to images or other media files [16]. Firebase data is synchronized in real-time to all the connected clients. When you build cross-platform apps with IOS and Android, also JavaScript SDKs, all of the clients share the one database on Firebase and they are able to receive updates automatically with all of the newest data [17].

### 4.4.3 Mongo DB

MongoDB is an open source database, it is a NoSQL database similar to Firebase as it is document oriented, the data is stored as JSON instead of tables and relations as you would normally see in a relational database. The data is represented as a name - value pair something similar to columns and rows in a relational database.
Mongo supports sharding, this permits horizontal scaling by divvying up a collection of documents across a cluster of node, which makes reads faster. The Mongo database is also a schema-less database so it will store any document that is put into it [18]. These documents are grouped into collections, these collections are stored in a binary JSON format as I have discussed earlier.

## 4.5 Programming Languages

### 4.5.1 Java

Java is a programming language which is designed for use in the distributed environment of the internet, it was first introduced by Sun Microsystems in 1991. Java was originally known as OAK, and it was designed for handheld devices and set-top boxes. Oak was unsuccessful and Sun Microsystems changed its name to Java in 1995. Java is similar to C++ but it is simplified to eliminate common programming errors [26].

### 4.5.2 C++

C++ was developed by Bjarne Stroustrup, it was an extension of the C programming language. Even though C++ is an object oriented language it is also possible to code in C style. The C++ language can be considered to be an intermediate level language, as it has both low level and high level language. [27].  C++ is related to Java but it is optimized for the distribution of program objects in a network such as the internet [38].

## 4.6 Conclusion

With close assessments of the technologies that were discussed the server side language that will be used will be PHP. PHP is one of the most commonly used server side language and many web applications are developed using PHP. PHP will provide the functionality to create dynamic web applications to the clients browser. The client side language that will be used in the system will be JavaScript.
The MYSQL database will be used in the development of the project to store all of the data in a relational database, running on an apache server. XAMPP will be used as it provides a good user interface for MYSQL database. PhpMyAdmin (located in the XAMPP control panel) provides the developer with an easy to use interface to create tables within the schema and easy to view the database as a whole, which is why it will be used in the project.
The genetic algorithm is going to be developed in java, and it will send the timetable data that is scheduled to the database, this comes down to the general superior performance of Java.
The research that was conducted in this chapter is very important in relation to choosing the best technologies suitable to developing this project.

# 5. Design

The following section of the report justifies and describes the design of the system, it will examine the overall design of the project. Documents such as the system architecture diagram, use case diagram, the ERD,  will be explained in detail. The functional properties of the design are described with use case narratives and also visually described using UML diagrams.

## 5.1 Technical architecture diagram:

The system which the system will be developed is a three tier application. The diagram at Figure 2 displays the overall structure of the system. The server waits from the user requests from their workstations. The connection between the server and the database is essential as the data that is need to be returned to the user to contained in the database, this data is obtained from the server.

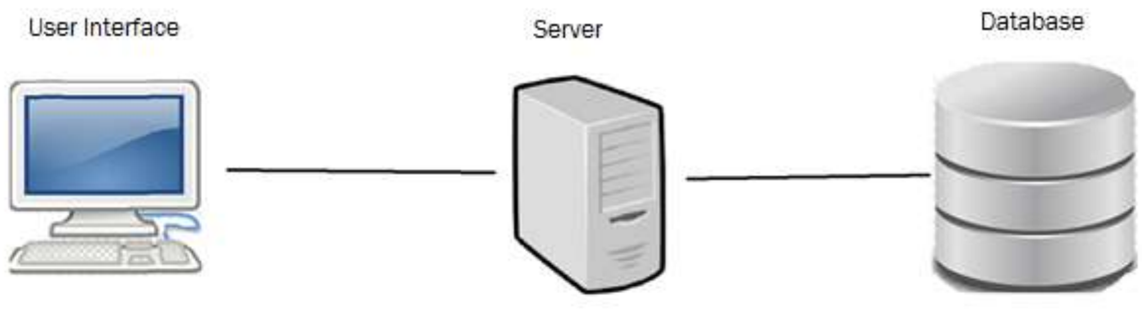


**Figure 5.1 3 Tier Application**

## 5.2 Use Case Diagram

The high-level functionality of the system is shown the diagram Figure 1. This use case diagram describes visually the overall use case for the actors in the system, and also how they interact with each other. Once the actions are clearly illustrated to the user, it is able to reveal more information about the back end of the program, so it basically highlights what the systems needs to do.
To be able to develop a use case model you have to identify the different roles that the humans in the system can play [23]. The actors in this diagram is the actual users in the system.

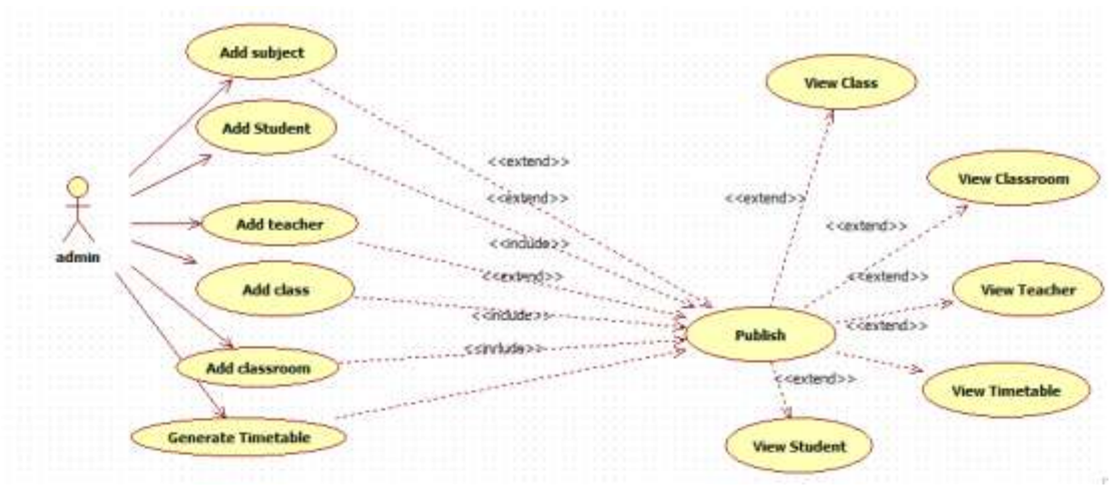


**Figure 5.2 High Level Use Case Diagram**

This use case diagram shows every action of every user in the system. Such actions include adding an exam timetable which is automated, also logging in and out of the system, all the details of the back-end of the system.

## 5.3 ERD Diagram

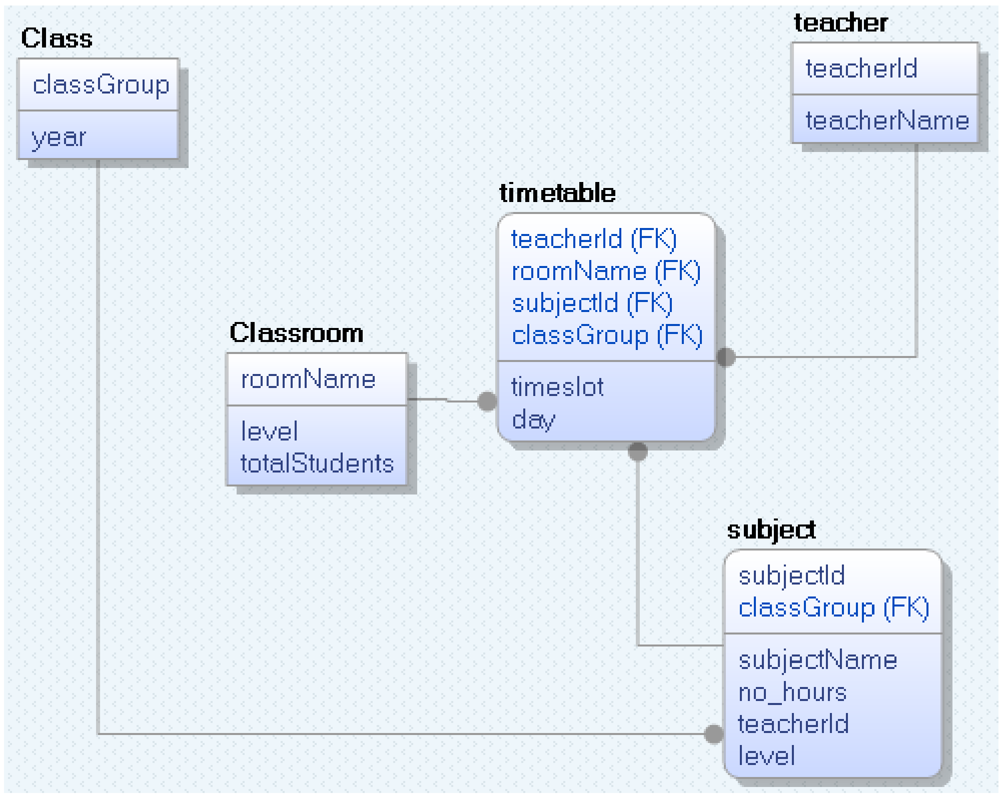


**Figure 5.3 Entity Relationship Diagram**

The database in this system consists of five tables, which consist of the the classroom table, the timetable table, the subject table, the teacher table, and also the class table . These tables are very important as they contain the information to build the database and allow for the user to interact with the system.

Class

The class table contains the class group and the class year the class name. The class group is unique in this table, and it will be used as a foreign key in the subjects table as was as the timetable table.

Teacher
The teacher table contains the teacher's id (primary key), the teachers name, that will be used for the timetable schedule.

Subjects Table
The subject table consists of the subject id (primary key), the subject name, the number of hours per week of classes, the teacher id (foreign key from the teachers table), and also the level whether is higher or lower.

Timetable Table
The timetable table will consist mainly of foreign keys in the database. The timetable consists of the timeslot, the day of the week, the teacher id (foreign key from the teachers table), the subject id (foreign key from the subjects table), the room name (foreign key from the classroom table) and finally the student group (foreign key from the class table)

Classroom
The classroom table consists of the classgroup (primary key) and the student year.

## 5.4 Design Prototype

The design prototype that has been conducted is the design of the students, teachers and administrators pages.
Home Page

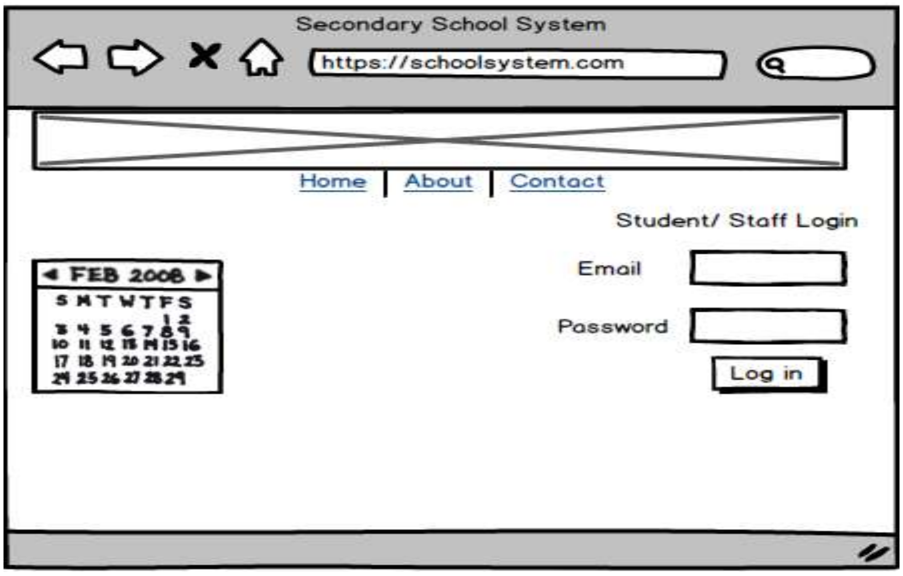
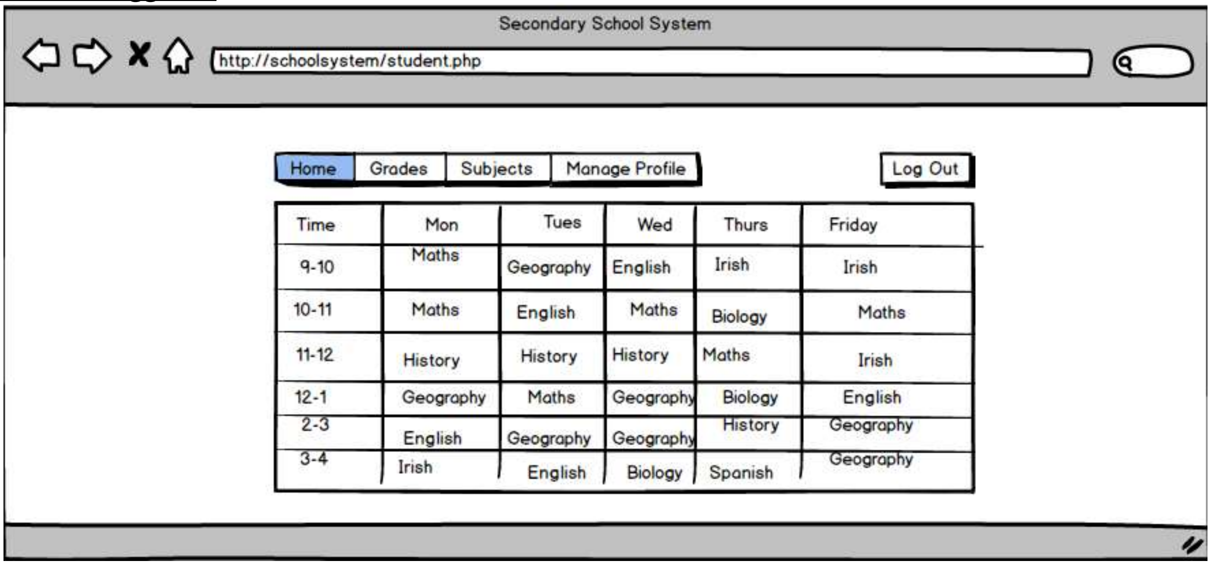


**Figure 5.4 Homepage**
The image above (figure 5.4) is a prototype of the homepage of the web application which is the initial screen of the system. The user is able to log in whether he/ she is a student, teacher or administrator. This screen also shows additional information about the system if the user clicks the about tab and if the user clicks the contact tab it brings the user to the contact page where there is a contact form.

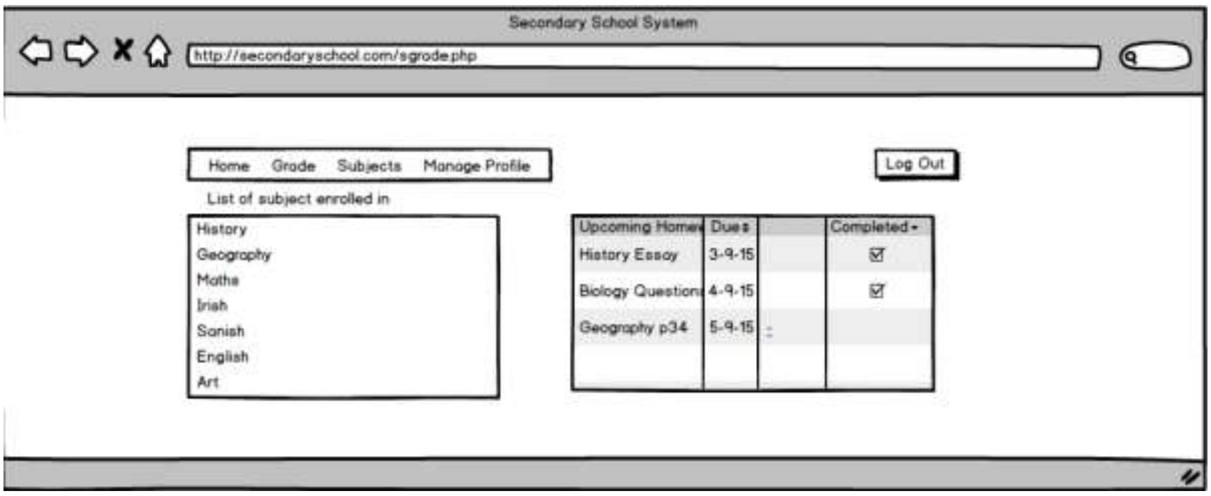
<u>Student Logged in</u>



| Time | Mon | Tues | Wed | Thurs | Friday |
|---|---|---|---|---|---|
| 9-10 | Maths | Geography | English | Irish | Irish |
| 10-11 | Maths | English | Maths | Biology | Maths |
| 11-12 | History | History | History | Maths | Irish |
| 12-1 | Geography | Maths | Geography | Biology | English |
| 2-3 | English | Geography | Geography | History | Geography |
| 3-4 | Irish | English | Biology | Spanish | Geography |

**Figure 5.5 Initial Student Page After Log in**

The above image (figure 5.5) shows the initial page of the student page once they have logged in. They are able to view the timetable and it displays all the subjects in which they are enrolled in. The page also provides a navigation bar which the student is able to view their grades and the modules that they are enrolled in.



| Upcoming Homew | Due ↕ | | Completed ▾ |
|---|---|---|---|
| History Essay | 3-9-15 | | ☑ |
| Biology Question | 4-9-15 | | ☑ |
| Geography p34 | 5-9-15 | | |

**Figure 5.6 Subjects Page**

The subjects page brings the user to where they are able to view a list of their subjects they are enrolled in as well as list of all their upcoming homework assignments (see figure 5.6). The student is able to click on any of the subjects.

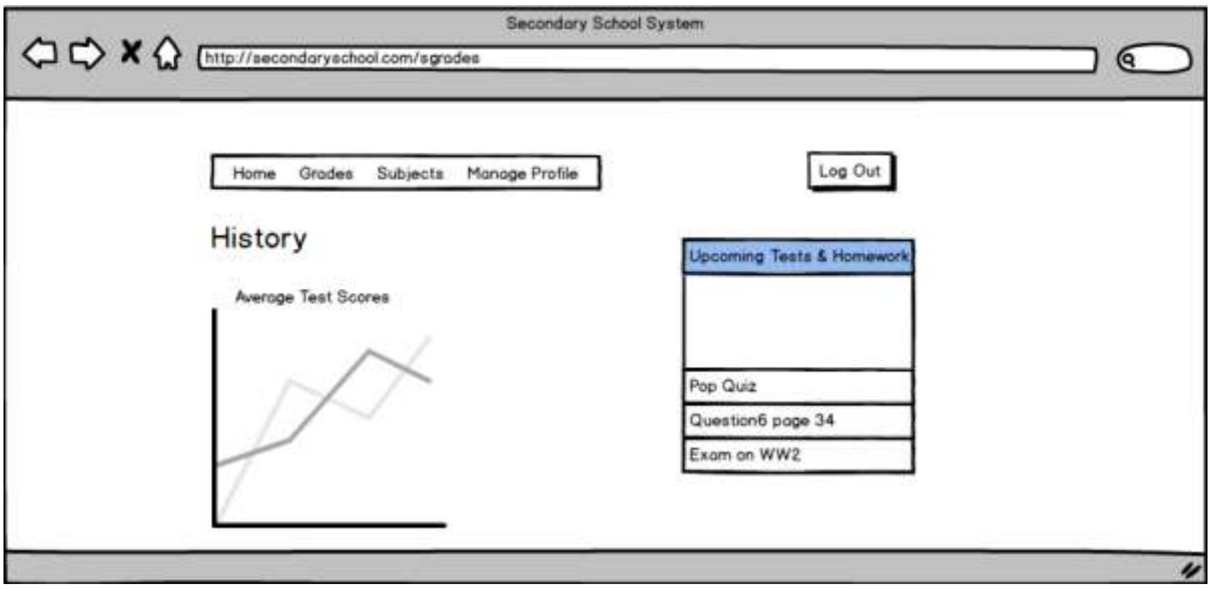
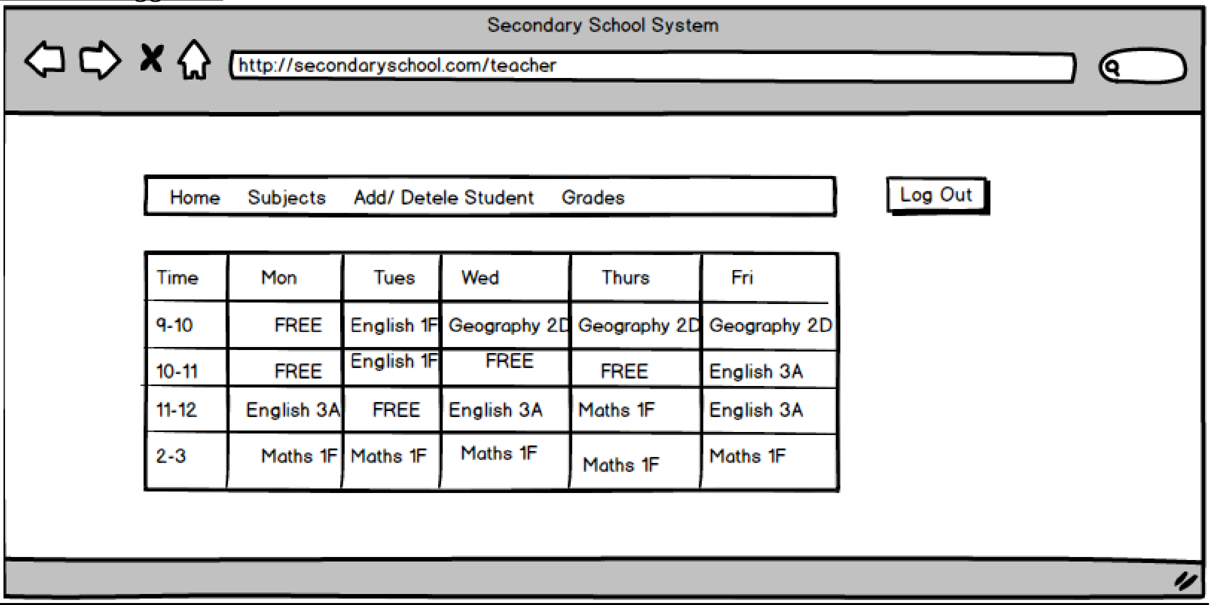


**Figure 5.7 Subject Page**

This page is accessed once the user clicks on a subject on the list of subjects page. This page brings up all the details about the subject such as the students grades and the upcoming homework and exams (see figure 5.7).

<u>Teacher Logged In</u>

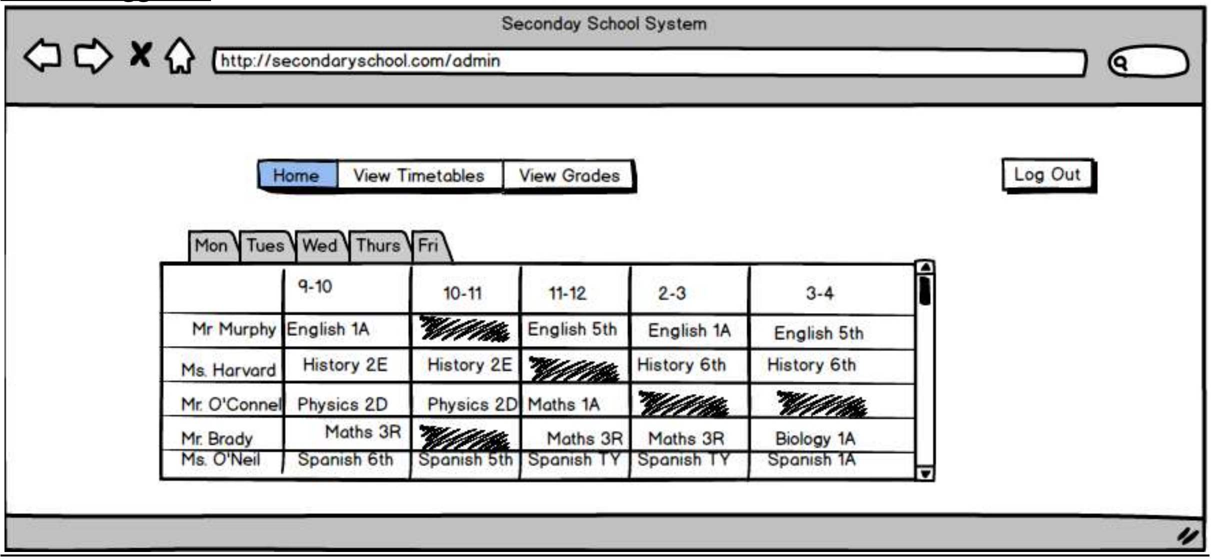


| Time | Mon | Tues | Wed | Thurs | Fri |
|---|---|---|---|---|---|
| 9-10 | FREE | English 1F | Geography 2D | Geography 2D | Geography 2D |
| 10-11 | FREE | English 1F | FREE | FREE | English 3A |
| 11-12 | English 3A | FREE | English 3A | Maths 1F | English 3A |
| 2-3 | Maths 1F | Maths 1F | Maths 1F | Maths 1F | Maths 1F |

**Figure 5.8 Teacher Logged In Page**

Once the teacher has logged into the system, the initial page shows the teacher what hours he/ she is teaching (see figure 5.8). It tells the teacher what free hours they have and what classes they have. They are able to quickly navigate through the tabs, they can view all the students grades in their class. They also have the authority to be able to add and delete students from their class.

Admin Logged In

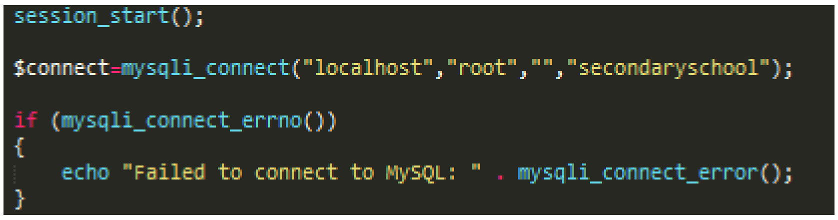


| | 9-10 | 10-11 | 11-12 | 2-3 | 3-4 |
|---|---|---|---|---|---|
| Mr Murphy | English 1A | ▨ | English 5th | English 1A | English 5th |
| Ms. Harvard | History 2E | History 2E | ▨ | History 6th | History 6th |
| Mr. O'Connel | Physics 2D | Physics 2D | Maths 1A | ▨ | ▨ |
| Mr. Brady | Maths 3R | ▨ | Maths 3R | Maths 3R | Biology 1A |
| Ms. O'Neil | Spanish 6th | Spanish 5th | Spanish TY | Spanish TY | Spanish 1A |

**Figure 5.9 Admin Login Page**

Figure 5.9 is the initial page for the admin once they log in. It shows the admin of all the teachers working on a particular day. It shows the admin when the teachers are free, and when and where they are teaching. This is very significant as the admin can quickly see what teachers are free to cover for a absent teacher. The admin is able to view all the timetables in the system and they are able to see all the grades of the students and the teachers class grades to see how well they are doing.

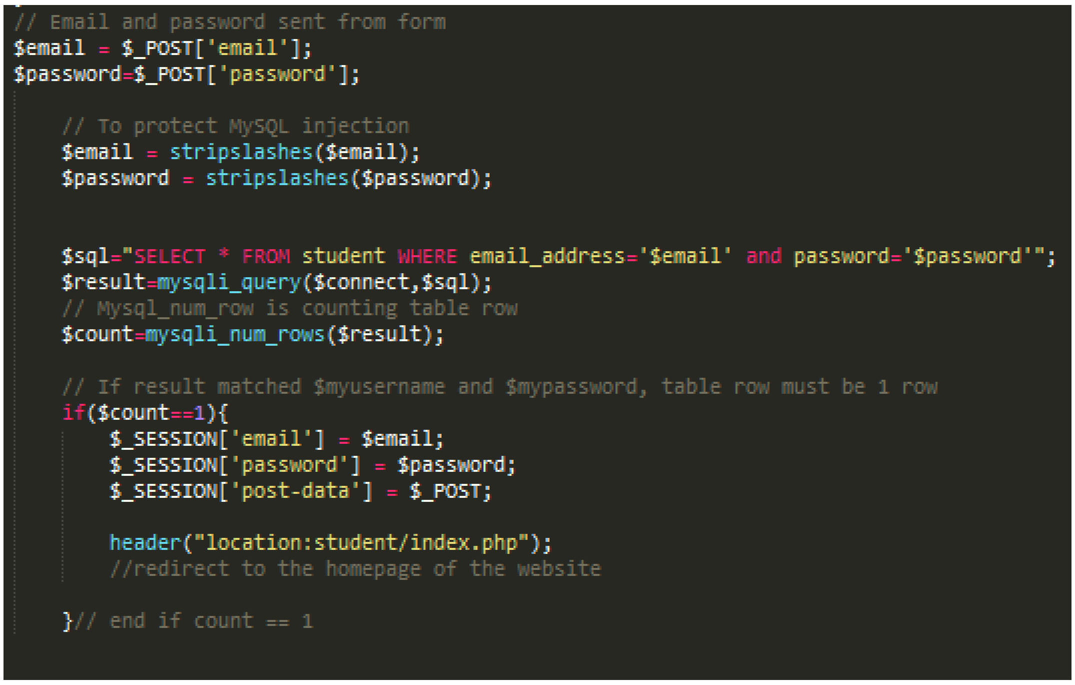
## 5.5 Development Prototype



```
session_start();

$connect=mysqli_connect("localhost","root","","secondaryschool");

if (mysqli_connect_errno())
{
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}
```

**Figure 5.10 Connecting to the Database**

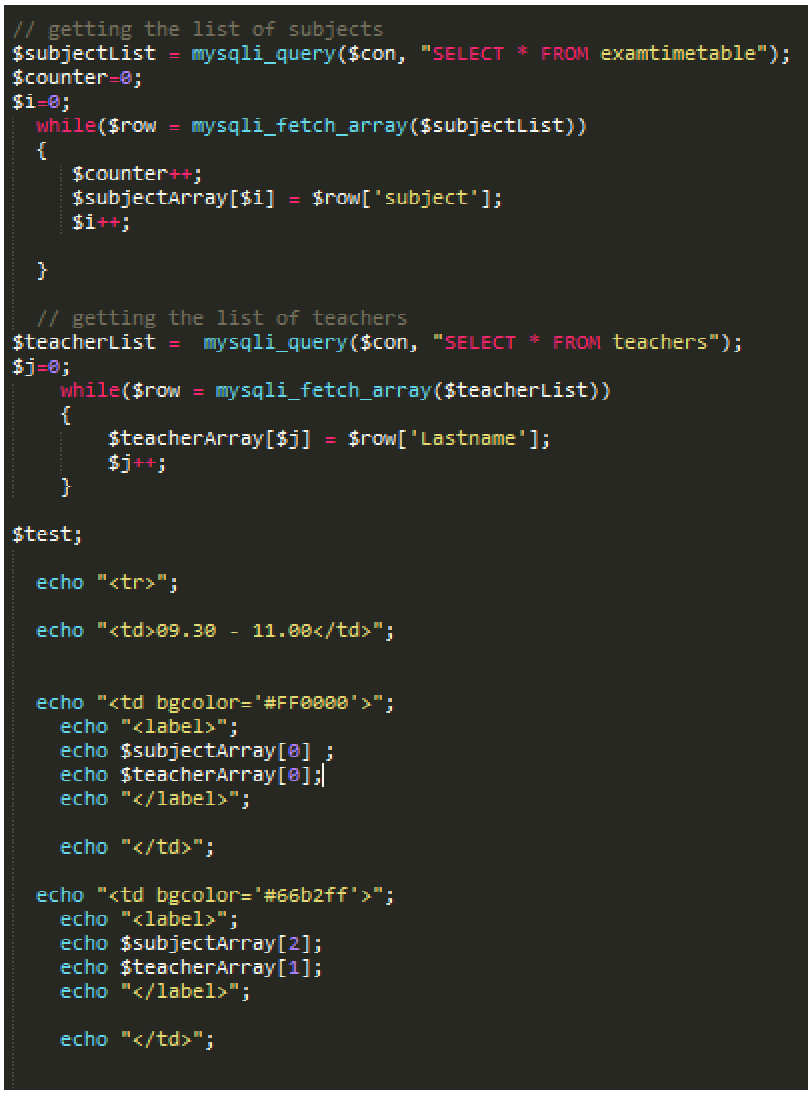
**Figure 5.10 above** is an example on how to connect to the loca database in PHP. The correct credentials have to be entered the database should in this case the database name is called "secondaryschool".



```
// Email and password sent from form
$email = $_POST['email'];
$password=$_POST['password'];

    // To protect MySQL injection
    $email = stripslashes($email);
    $password = stripslashes($password);


    $sql="SELECT * FROM student WHERE email_address='$email' and password='$password'";
    $result=mysqli_query($connect,$sql);
    // Mysql_num_row is counting table row
    $count=mysqli_num_rows($result);

    // If result matched $myusername and $mypassword, table row must be 1 row
    if($count==1){
        $_SESSION['email'] = $email;
        $_SESSION['password'] = $password;
        $_SESSION['post-data'] = $_POST;

        header("location:student/index.php");
        //redirect to the homepage of the website

    }// end if count == 1
```

**Figure 5.11 Log In**

**Figure 5.11 above** shows a log-in in PHP, the email and password is sent by the POST method. The striplashes method protects the system from SQL injection which is a common security threat in terms of web applications. The checkstudent variable is checking to see if there is a student with the correct credentials entered in from the log in form, a select query is performed to see if the student logging in is in the system. The query is then executed  using the mysqli_query function. Once the student has logged in, their email and password has started in the session which will be used to store the information across multiple web pages.



```
// getting the list of subjects
$subjectList = mysqli_query($con, "SELECT * FROM examtimetable");
$counter=0;
$i=0;
  while($row = mysqli_fetch_array($subjectList))
  {
     $counter++;
     $subjectArray[$i] = $row['subject'];
     $i++;

  }

  // getting the list of teachers
$teacherList =  mysqli_query($con, "SELECT * FROM teachers");
$j=0;
    while($row = mysqli_fetch_array($teacherList))
    {
        $teacherArray[$j] = $row['Lastname'];
        $j++;
    }

$test;

  echo "<tr>";

  echo "<td>09.30 - 11.00</td>";


  echo "<td bgcolor='#FF0000'>";
    echo "<label>";
    echo $subjectArray[0] ;
    echo $teacherArray[0];
    echo "</label>";

    echo "</td>";

  echo "<td bgcolor='#66b2ff'>";
    echo "<label>";
    echo $subjectArray[2];
    echo $teacherArray[1];
    echo "</label>";

    echo "</td>";
```

**Figure 5.12 List Of Subject & Teachers for Timetable**

Figure 5.12 is the prototype of the timetable. It displays the first version of the timetable with the subject and the teachers name in an array in displayed in a table. This prototype allows for a very basic version of how the timetable would be displayed, its final version will be automatically generated.

## 5.6 Approach and Methodology

### 5.6.1 Waterfall Methodology

"The idea of development process or 'steps' or 'stages' emerged in the 1950's and Royce's formulation of the waterfall model, was a particularly influential refinement of the earlier thinking that explicitly recognized the presence of 'feedback loops' between the various stages of development"[19].

The waterfall methodology has eight stages in the development process and each phase has to be completed before moving onto the next phase. You cannot go back to a previous stage and make changes if the step has been completed. This can be a costly disadvantage as the customer cannot make changes in their decision to the final product. "With the waterfall methodology, the client knows what to expect"[ 20]. This can be an advantage in the sense that you have an idea of the timeline and the cost of the project.

### 5.6.2 Agile Methodology

The agile methodology is a iterative approach to software delivery that builds software incrementally from the very beginning of a project, instead of delivering the project all at the end [21]. It breaks the project into smaller parts of the user functionality known as user stories. It supports the constant revision of previous stages of the development process. The agile process emphasises production of on time and on budget software. Developing focuses on keeping the code fairly simple, testing the code frequently and delivering parts of the application when they are ready when using the agile methodology this allows for evolving requirements within the system.

### 5.6.3 Spiral Model

The spiral methodology also known as the spiral lifecycle model, is a systems development lifecycle model which is used in the area of IT (Information Technology). The spiral model is a combination of the features of the waterfall model and the prototyping model [http://searchsoftwarequality.techtarget.com/definition/spiral-model].
The spiral model has four main stages, they are known as the:

- identification phase
- Design phase
- Engineering phase (also known as build phase)
- Evaluation and Risk Analysis

The spiral model is very flexible model for project managing and monitoring. The development phases of the system can be determined by the project manager, according to its complexity within the project [22].
There are some problems with using the Spiral model as a design methodology for the system. The cost can normally be relatively high, and it can also be very complicated and there is a lot of documentation required within the intermediate stages, it can lead to a very complex affair[22].

### 5.6.4 Conclusion
In conclusion and examining all of the methodologies, the project will be developed using the agile methodology. The reason why the agile methodology will be used is because using it is most suitable for projects that are of medium size compared to the Waterfall and Spiral methodology, they are used more in bigger projects.  Also with frequent releases of the product means that there is the possibility to implement new features. For this project, the requirements specification will be subject to change and the agile methodology is best known for frequent development and testing. The agile methodology allows for quality software realises and as there isn't a huge time for development the agile methodology is the perfect methodology.

## 5.7 Conclusion

In conclusion of examining the design of the system, the Unified Modelling Language (UML) diagram gave a great insight into the actors that are in the system and their key functionalities within the system. The overall system architecture was also discussed in this chapter, it describes clearly that the system will be a three tier system and shows the interaction between the client, server and the database. The ERD gave a clear indication in how the database will be implemented, it also gave an insight into the complexity of the database with its relationship between all the tables.

# 6. Implementation

## 6.1 Introduction

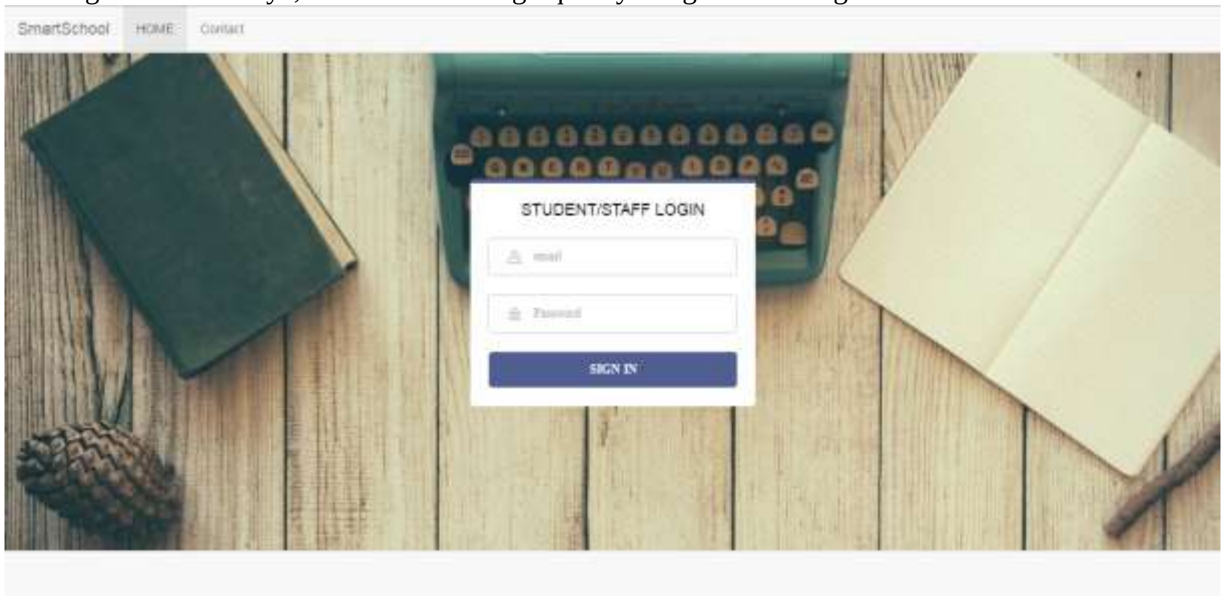
This chapter offers a complete overview of the development of the smart school site. This chapter explains all of the development and design of the final version of the web application. It describes the development in detail and describes the iteration process of the development.

## 6.2 System Overview

This section details the development of all the different sections of the Smart School application. The covered sections include how the timetable is scheduled, the web development (frontend and backend) and the Java development, integrating two different programming language, connecting to phpMyAdmin from Java.

### 6.2.1 Homepage

The first screen that the user sees is the login screen, where the staff/ students are able to log into their account. The background image and the responsive navigation bar allows for the application to be used on any device. The log in page is very straight forward and it is catching to the user's eye, as it contains a high quality image and the log-in box.



**Homepage (figure 6.1)**

As you can see from the homepage (figure 6.1) the log-in box is very user friendly and easy to use. For first time users it is very hard to get lost from the homepage. From here the user is able to log into their account.

If the user logs in incorrectly they are prompted with an error message stating that the email or password that they have entered is incorrect. Also all fields are required for the log in, and if one field is not entered in the user is notified.

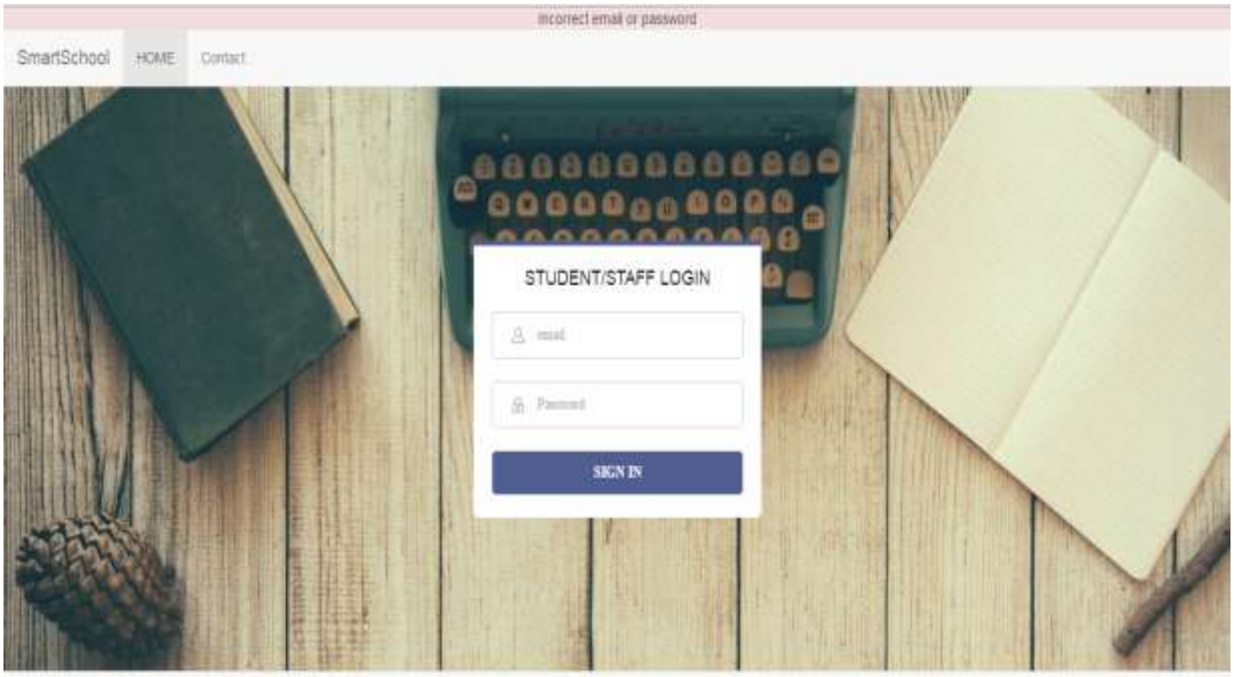
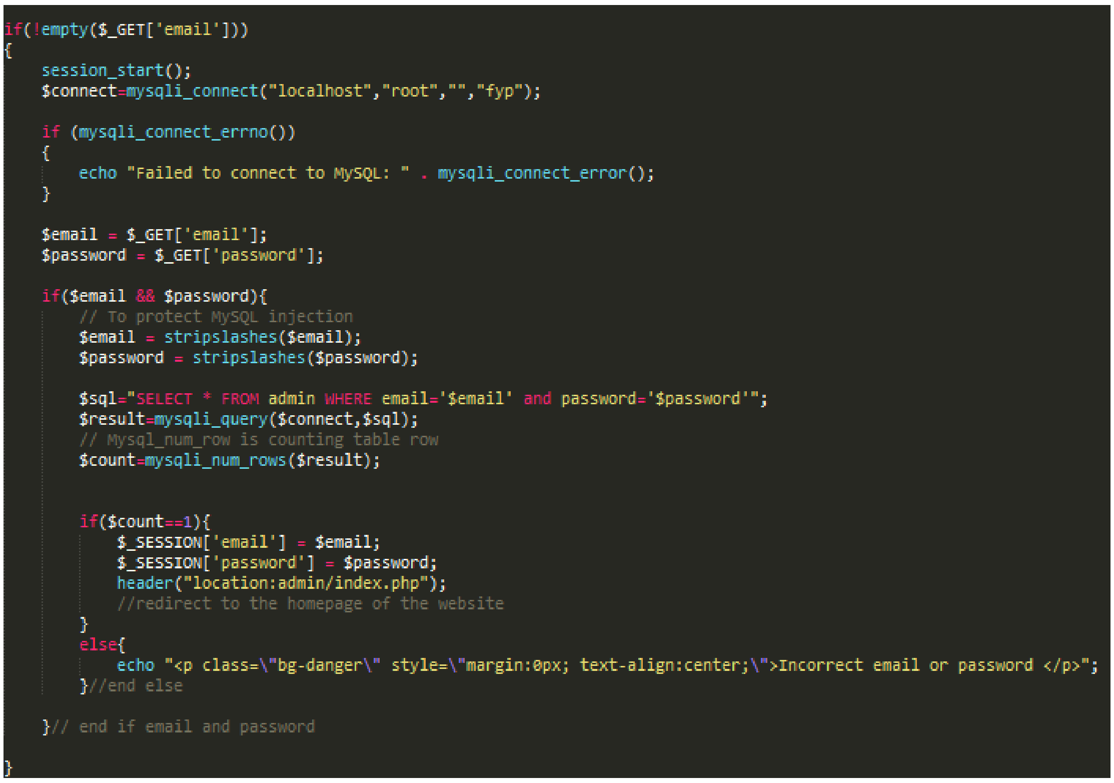


Figure 6.2 Incorrect Login

Figure 6.2 above gives the user a clear indication that the login credentials were incorrect.



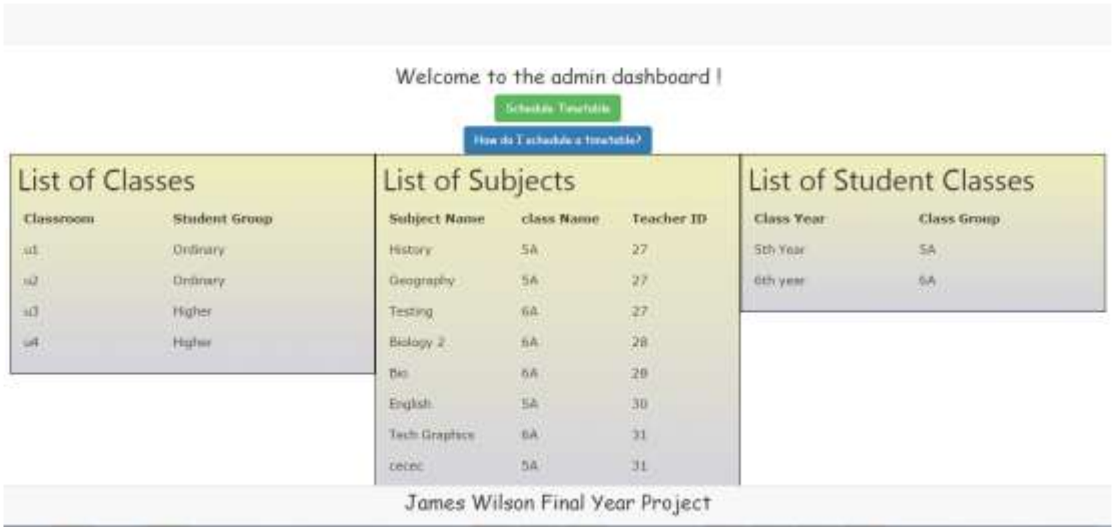
```php
if(!empty($_GET['email']))
{
    session_start();
    $connect=mysqli_connect("localhost","root","","fyp");

    if (mysqli_connect_errno())
    {
        echo "Failed to connect to MySQL: " . mysqli_connect_error();
    }

    $email = $_GET['email'];
    $password = $_GET['password'];

    if($email && $password){
        // To protect MySQL injection
        $email = stripslashes($email);
        $password = stripslashes($password);

        $sql="SELECT * FROM admin WHERE email='$email' and password='$password'";
        $result=mysqli_query($connect,$sql);
        // Mysql_num_row is counting table row
        $count=mysqli_num_rows($result);


        if($count==1){
            $_SESSION['email'] = $email;
            $_SESSION['password'] = $password;
            header("location:admin/index.php");
            //redirect to the homepage of the website
        }
        else{
            echo "<p class=\"bg-danger\" style=\"margin:0px; text-align:center;\">Incorrect email or password </p>";
        }//end else

    }// end if email and password

}
```

**Figure 6.3 Login PHP**

See figure 6.3 that contains the code to validate the log in for the web application. It is getting the email and password that was entered in from the login form, and creating a session for the email will be very important as it will allow the system to know how is currently logged into the system.

## 6.2.2 Admin Dashboard

When the admin logs in they are in the dashboard, which contains quick and precise information about all the data that is in the database. They are able to see what subjects, what teachers, the classes and the classrooms that are in there which are needed to schedule a timetable. If there is a timetable already scheduled they are able to view the timetable once they click on the view timetable link (which is scheduled for each class). The homepage for the admin consists of a simple neat design that allows for the user to know what is going on in their system. The buttons that are aligned to the centre of the screen means that the user cannot get lost in trying to find where to schedule their timetable and also how to schedule a timetable as this is very important, in order of the user to understand how scheduling a timetable for their school works.


Welcome to the admin dashboard !
Schedule Timetable
How do I schedule a timetable?
James Wilson Final Year Project

List of Classes

| Classroom | Student Group |
|---|---|
| u1 | Ordinary |
| u2 | Ordinary |
| u3 | Higher |
| u4 | Higher |

List of Subjects

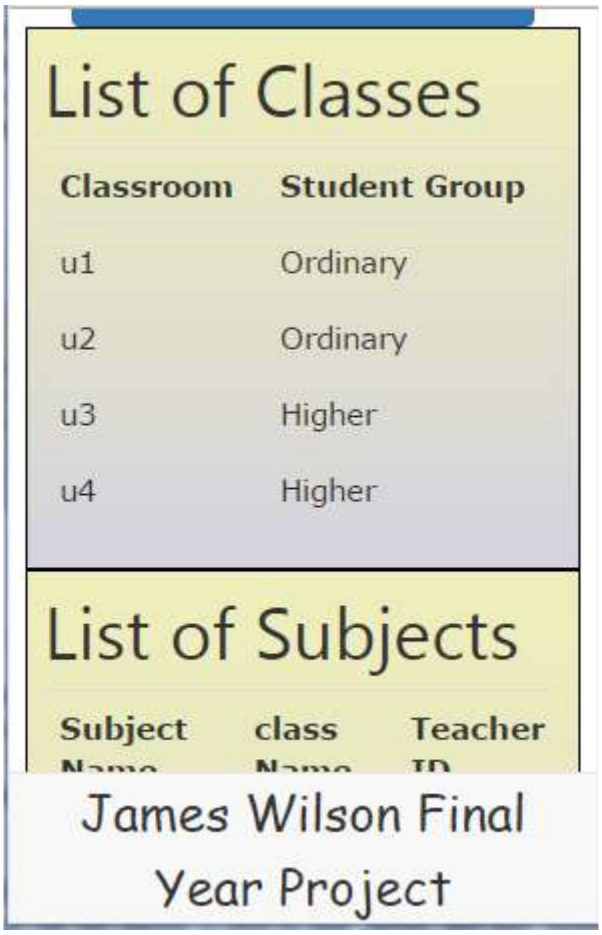
| Subject Name | class Name | Teacher ID |
|---|---|---|
| History | 5A | 27 |
| Geography | 5A | 27 |
| Testing | 6A | 27 |
| Biology 2 | 6A | 28 |
| Bio | 6A | 28 |
| English | 5A | 30 |
| Tech Graphics | 6A | 31 |
| cecec | 5A | 31 |

List of Student Classes

| Class Year | Class Group |
|---|---|
| 5th Year | 5A |
| 6th year | 6A |

**Figure 6.4 admin dashboard**

The design is also a responsive design, it can be able to be used on any device. W3schools allows for a responsive design which is used for the div tags, by including *class="w3-third"*. In the figure below it is visible that the design is responsive and easy to use on mobiles and tablets (*See figure 6.4*).



## List of Classes

| Classroom | Student Group |
|---|---|
| u1 | Ordinary |
| u2 | Ordinary |
| u3 | Higher |
| u4 | Higher |

## List of Subjects

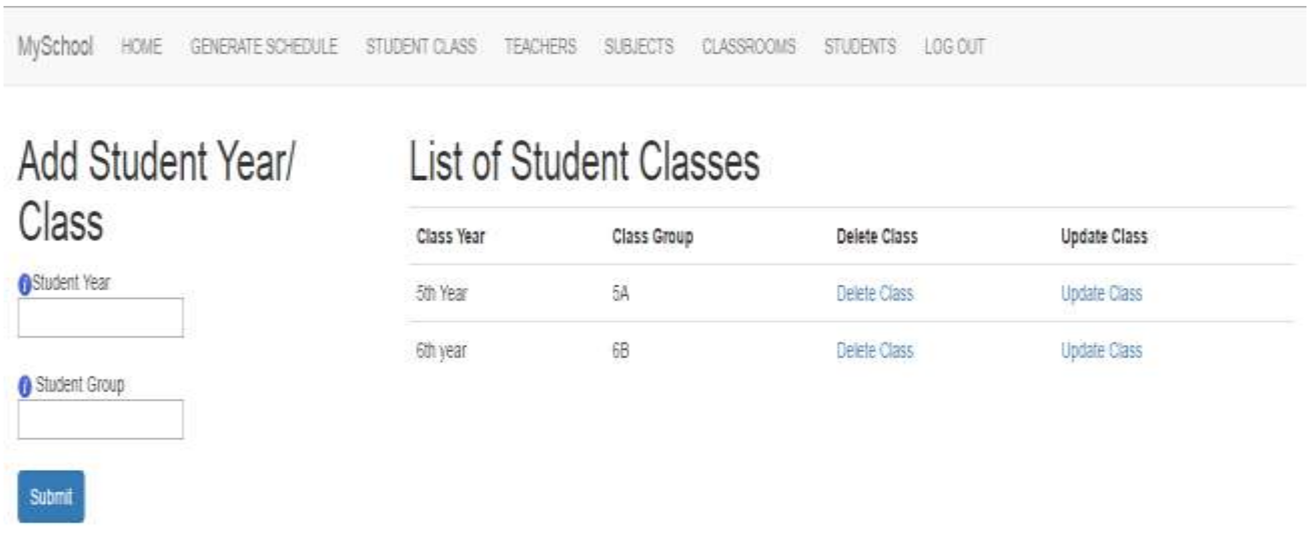
| Subject Name | class Name | Teacher ID |
|---|---|---|

James Wilson Final Year Project

**Figure 6.5 Responsive Design**

### 6.2.3 Student Class Page

In the student class tab the admin is able to add, delete and also update the student class that is currently in the database. The admin is not able to add two of the same class groups for example in figure 6.6 the class groups that are in the database are '5A' and '6A', if the admin tries to add another class with the value '5A' or '6A' it will not insert as the class group is a unique key.



Add Student Year/ Class

Student Year

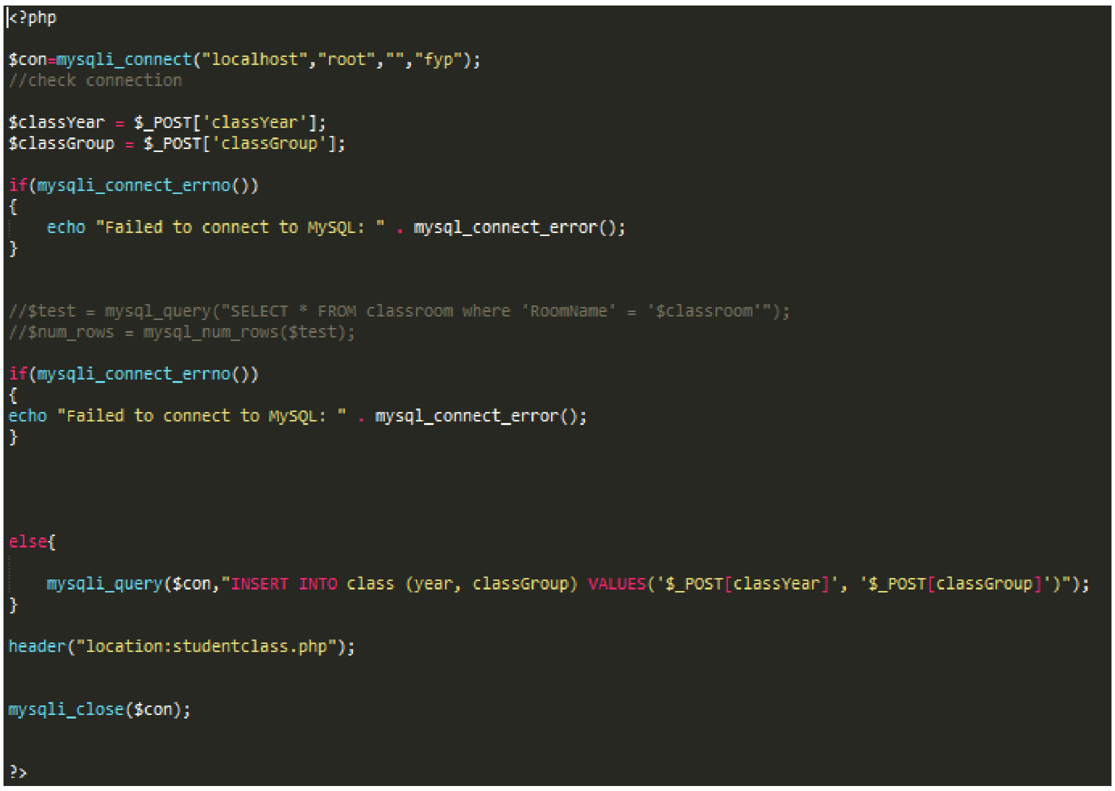
Student Group

List of Student Classes

| Class Year | Class Group | Delete Class | Update Class |
|---|---|---|---|
| 5th Year | 5A | Delete Class | Update Class |
| 6th year | 6B | Delete Class | Update Class |

**Figure 6.6 Student Class Screen**

As you can see from *figure 6.6* there is an information icon beside every input box, the admin is able to hover over this icon and it will display information about what they should enter into the input box. In *figure 6.7* there is an example of the inserting a class code. The information that is entered in from the user is stored in variables ( $classYear, $classGroup).



```php
<?php

$con=mysqli_connect("localhost","root","","fyp");
//check connection

$classYear = $_POST['classYear'];
$classGroup = $_POST['classGroup'];

if(mysqli_connect_errno())
{
    echo "Failed to connect to MySQL: " . mysql_connect_error();
}


//$test = mysql_query("SELECT * FROM classroom where 'RoomName' = '$classroom'");
//$num_rows = mysql_num_rows($test);

if(mysqli_connect_errno())
{
echo "Failed to connect to MySQL: " . mysql_connect_error();
}




else{

    mysqli_query($con,"INSERT INTO class (year, classGroup) VALUES('$_POST[classYear]', '$_POST[classGroup]')");
}

header("location:studentclass.php");


mysqli_close($con);


?>
```

**Figure 6.7 Example Creating a new Class code**

If the admin deletes the class, for example from figure 6.6 if the admin deletes the first row in the table (the row containing the '5A') it will set off a trigger in the database that will delete the row '5A' in the subject table and the classroom table. This delete trigger is necessary because if the admin decides to automatically schedule the timetable but they have deleted a row(s) from the 'class' table the timetable would not be able to be generated as it does not have the valid information to do that, as the class group 5A would not exist. See figure 6.8 for the delete trigger

```
CREATE TRIGGER `class_delete` BEFORE DELETE ON `class`
 FOR EACH ROW BEGIN
    DELETE FROM subject WHERE subject.class_name=old.classGroup;
    DELETE FROM classroom WHERE classroom.studGroup=old.classGroup;
END
```

**Figure 6.8 Delete trigger for class table**

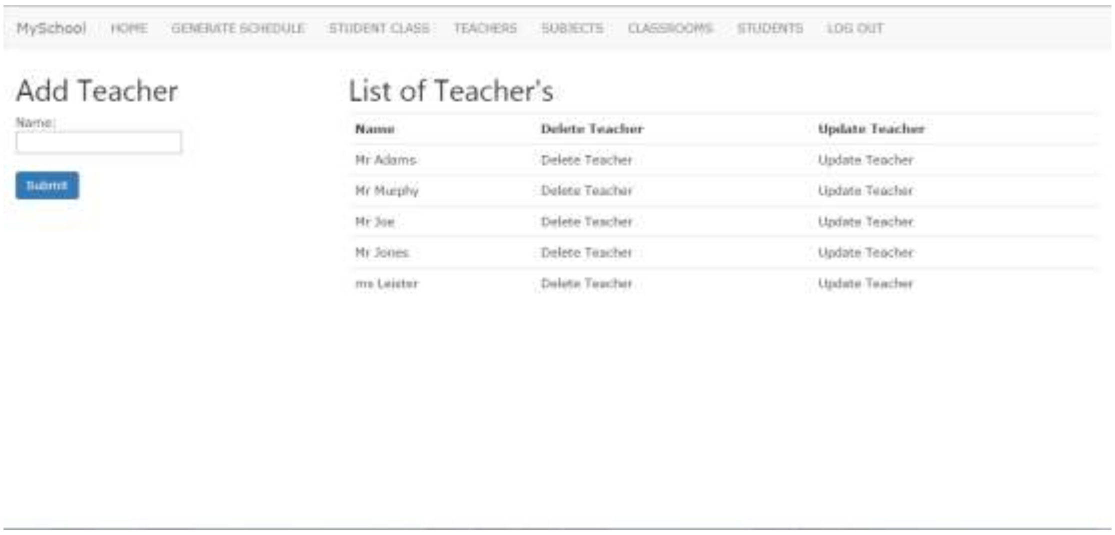
Also if the admin decides to update one of the class groups, it should also be updated automatically in the classroom table and the subjects table so it doesn't disrupt the data that is needed to schedule the timetable. So if the admin changes the class group from '5A' to '5B' it should also be updated the in the class room table as well as the subject table  see (*figure 6.9)* for the update trigger in the 'class' table.

```
CREATE TRIGGER `au_class` AFTER UPDATE ON `class`
 FOR EACH ROW BEGIN
    UPDATE classroom
        SET classroom.StudGroup=new.classGroup
        WHERE classroom.StudGroup=old.classGroup;

    UPDATE subject
        SET subject.class_name=new.classGroup
        WHERE subject.class_name=old.classGroup;
end
```

**6.9 auto update 'class' trigger**

### 6.2.4 Teacher Page

In the teachers page the admin is able to add/delete/update teachers from the database. The teachers that are added to the database are used in the subjects table where they are allocated subjects by the admin. (*See figure 6.10*)



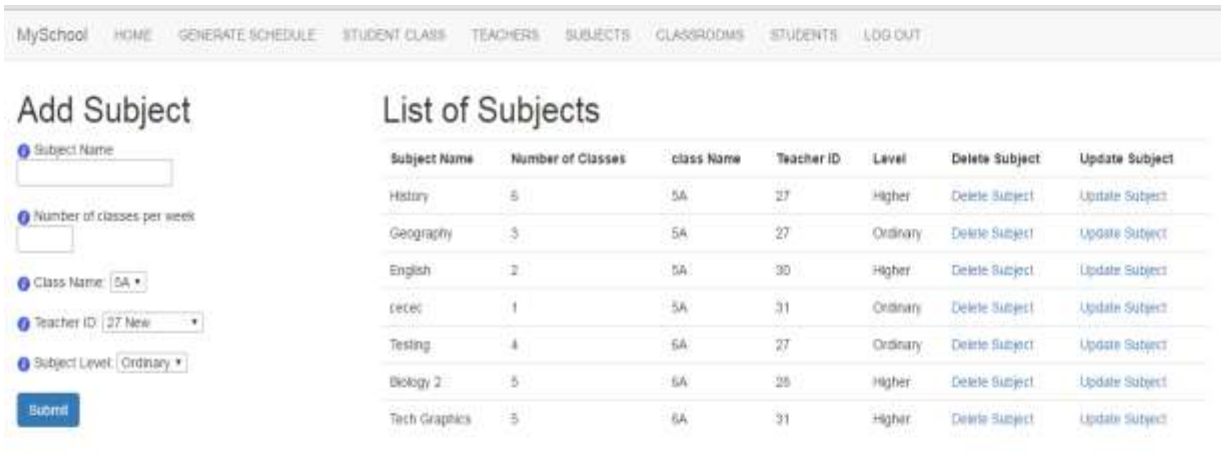
List of Teacher's

| Name | Delete Teacher | Update Teacher |
|---|---|---|
| Mr Adams | Delete Teacher | Update Teacher |
| Mr Murphy | Delete Teacher | Update Teacher |
| Mr Joe | Delete Teacher | Update Teacher |
| Mr Jones | Delete Teacher | Update Teacher |
| ms Leister | Delete Teacher | Update Teacher |

**Figure 6.10 Teachers Page**

### 6.2.5 Subjects Page

In the subjects page (*see figure 6.11*) the admin is able to see all of the subjects that are in the database.



Add Subject

Subject Name

Number of classes per week

Class Name: 5A

Teacher ID: 27 New

Subject Level: Ordinary

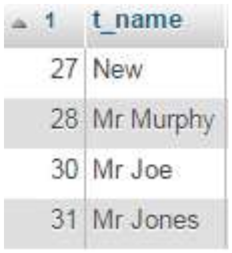
Submit

List of Subjects

| Subject Name | Number of Classes | class Name | Teacher ID | Level | Delete Subject | Update Subject |
|---|---|---|---|---|---|---|
| History | 6 | 5A | 27 | Higher | Delete Subject | Update Subject |
| Geography | 3 | 5A | 27 | Ordinary | Delete Subject | Update Subject |
| English | 2 | 5A | 30 | Higher | Delete Subject | Update Subject |
| cecec | 1 | 5A | 31 | Ordinary | Delete Subject | Update Subject |
| Testing | 4 | 6A | 27 | Ordinary | Delete Subject | Update Subject |
| Biology 2 | 5 | 6A | 28 | Higher | Delete Subject | Update Subject |
| Tech Graphics | 5 | 6A | 31 | Higher | Delete Subject | Update Subject |

**Figure 6.11 Subjects Page**

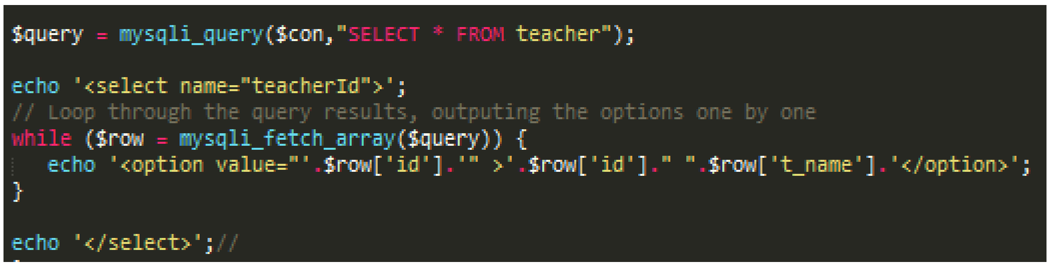
The admin is able to see what teachers are teaching what subject which is identified by the id of the teacher. When adding a subject to the database the admin can select a name for the subject and also the number of classes per week. In the class name and teacher ID input box the admin is only able to update from what is the database.



| 1 | t_name |
|---|---|
| 27 | New |
| 28 | Mr Murphy |
| 30 | Mr Joe |
| 31 | Mr Jones |

**Figure 6.12 Teacher Database**

This is what's in the teacher's table, so when adding a subject the admin should only be able to add these teachers. This is done by implementing a drop down input box, selecting all of the data from the teacher table (*See figure 6.12*)



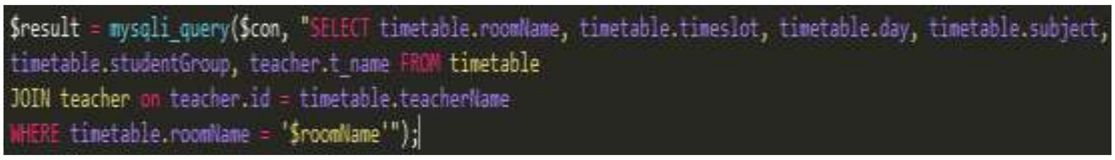
```
$query = mysqli_query($con,"SELECT * FROM teacher");

echo '<select name="teacherId">';
// Loop through the query results, outputing the options one by one
while ($row = mysqli_fetch_array($query)) {
   echo '<option value="'.$row['id'].'" >'.$row['id']." ".$row['t_name'].'</option>';
}

echo '</select>';//
```

**Figure 6.13 Code for drop down containing data for the database**

From *figure 6.13* the drop down list will only contain the data that is from the database, this is very important as the admin is not able to add a teacher that does not exist.

## 6.2.6 Classrooms Page

In the classrooms page the admin is able to see the list of classrooms that are in the database. They are able to add/delete/update these classrooms, and they are also able to view the schedule for a specific class. Once the admin clicks the view schedule link, it gets the room name and selects from the timetable table where the room name of the classroom matches the room name that is in the timetable table (*See figure 6.14*).



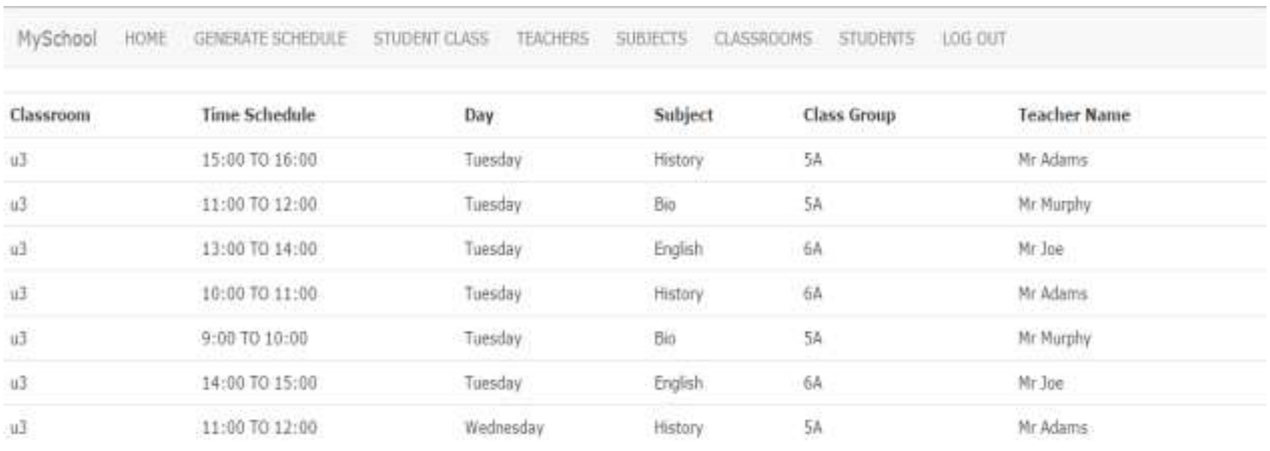
```
$result = mysqli_query($con, "SELECT timetable.roomName, timetable.timeslot, timetable.day, timetable.subject,
timetable.studentGroup, teacher.t_name FROM timetable
JOIN teacher on teacher.id = timetable.teacherName
WHERE timetable.roomName = '$roomName'");
```

**Figure 6.14 Example select statement to display timetable**

The $roomName is given to the room name that was passed when the admin clicked the link: " *$roomName = mysql_real_escape_string($_GET['RoomName']);*"

This allows the admin to be able to view any of the classes that are scheduled for a specific room, so that they can see when the room is free at any given time, and when they are being used. (*See figure 7.10*). So when the admin clicks on a room they want to view the schedule, the room that was clicked is passed through and using the GET function in PHP, the system gets the room name and displays the schedule for the room , see example of the SQL code below:

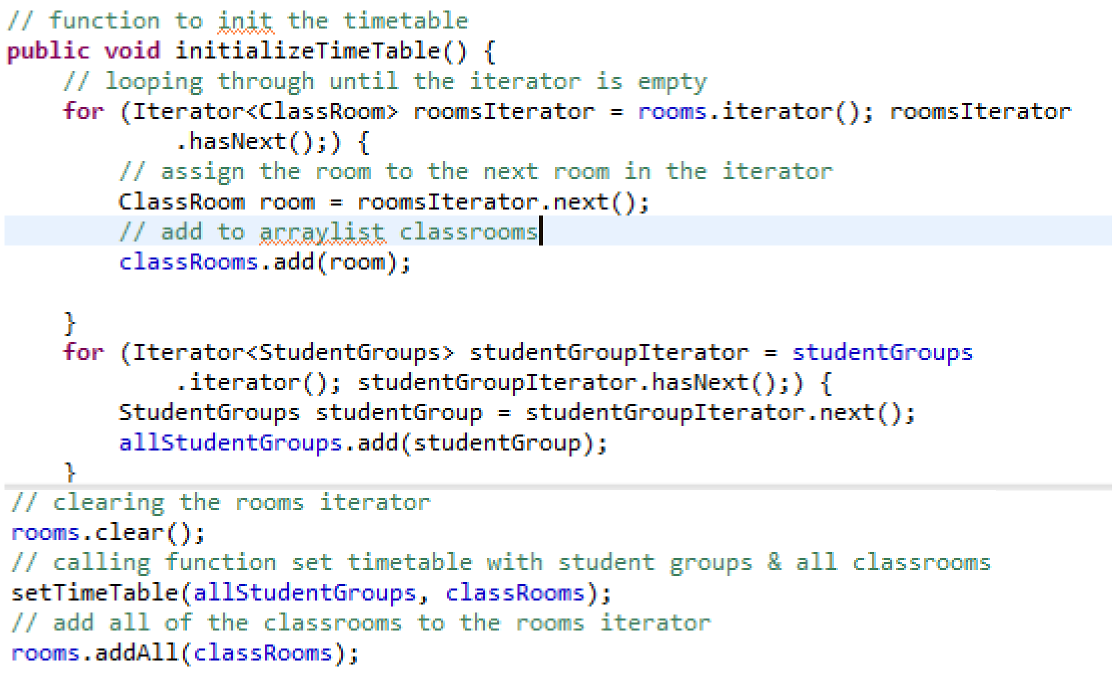
```
"$result = mysqli_query($con, "SELECT timetable.roomName, timetable.timeslot,
timetable.day, timetable.subject,
        timetable.studentGroup, teacher.t_name FROM timetable
        JOIN teacher on teacher.id = timetable.teacherName
        WHERE timetable.roomName = '$roomName' ORDER BY timetable.day"); "
```



| Classroom | Time Schedule | Day | Subject | Class Group | Teacher Name |
|---|---|---|---|---|---|
| u3 | 15:00 TO 16:00 | Tuesday | History | 5A | Mr Adams |
| u3 | 11:00 TO 12:00 | Tuesday | Bio | 5A | Mr Murphy |
| u3 | 13:00 TO 14:00 | Tuesday | English | 6A | Mr Joe |
| u3 | 10:00 TO 11:00 | Tuesday | History | 6A | Mr Adams |
| u3 | 9:00 TO 10:00 | Tuesday | Bio | 5A | Mr Murphy |
| u3 | 14:00 TO 15:00 | Tuesday | English | 6A | Mr Joe |
| u3 | 11:00 TO 12:00 | Wednesday | History | 5A | Mr Adams |

**Figure 6.15 Schedule**

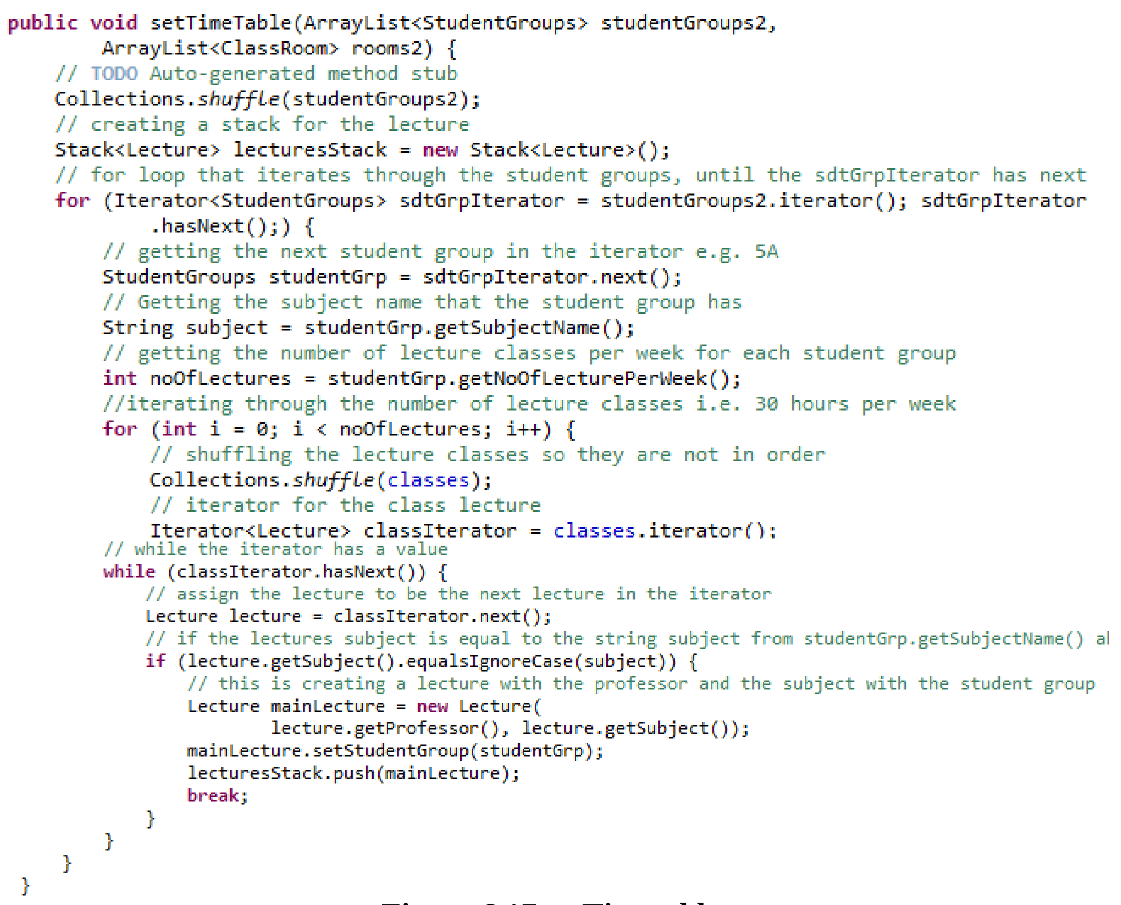
## 6.2.7 Timetable Class - Java



```
// function to init the timetable
public void initializeTimeTable() {
    // looping through until the iterator is empty
    for (Iterator<ClassRoom> roomsIterator = rooms.iterator(); roomsIterator
            .hasNext();) {
        // assign the room to the next room in the iterator
        ClassRoom room = roomsIterator.next();
        // add to arraylist classrooms
        classRooms.add(room);

    }
    for (Iterator<StudentGroups> studentGroupIterator = studentGroups
            .iterator(); studentGroupIterator.hasNext();) {
        StudentGroups studentGroup = studentGroupIterator.next();
        allStudentGroups.add(studentGroup);
    }
// clearing the rooms iterator
rooms.clear();
// calling function set timetable with student groups & all classrooms
setTimeTable(allStudentGroups, classRooms);
// add all of the classrooms to the rooms iterator
rooms.addAll(classRooms);
```

**Figure 6.16 Initializing the timetable**

This function (*see figure 6.16*) is in the timetable class, and what it does is it initializes the timetable. From the figure above the function is simply adding the rooms and the student groups and it is calling the function setTimeTable passing through all of the rooms and the student groups stored in the arraylists.

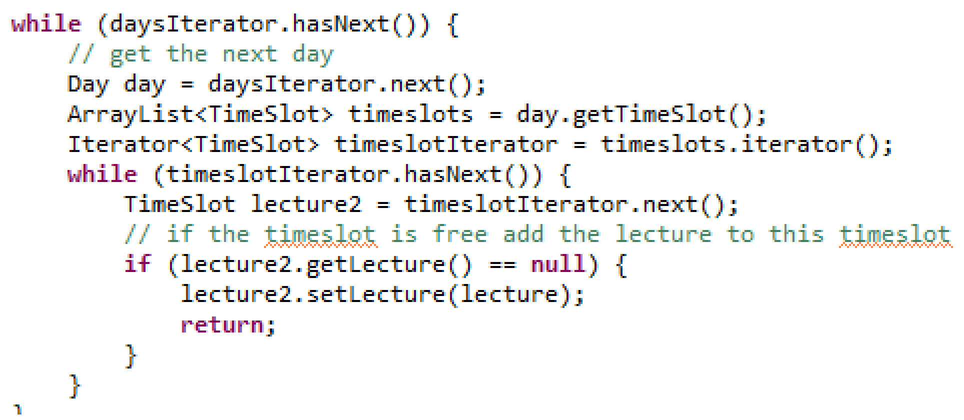


```java
public void setTimeTable(ArrayList<StudentGroups> studentGroups2,
        ArrayList<ClassRoom> rooms2) {
    // TODO Auto-generated method stub
    Collections.shuffle(studentGroups2);
    // creating a stack for the lecture
    Stack<Lecture> lecturesStack = new Stack<Lecture>();
    // for loop that iterates through the student groups, until the sdtGrpIterator has next
    for (Iterator<StudentGroups> sdtGrpIterator = studentGroups2.iterator(); sdtGrpIterator
            .hasNext();) {
        // getting the next student group in the iterator e.g. 5A
        StudentGroups studentGrp = sdtGrpIterator.next();
        // Getting the subject name that the student group has
        String subject = studentGrp.getSubjectName();
        // getting the number of lecture classes per week for each student group
        int noOfLectures = studentGrp.getNoOfLecturePerWeek();
        //iterating through the number of lecture classes i.e. 30 hours per week
        for (int i = 0; i < noOfLectures; i++) {
            // shuffling the lecture classes so they are not in order
            Collections.shuffle(classes);
            // iterator for the class lecture
            Iterator<Lecture> classIterator = classes.iterator();
        // while the iterator has a value
        while (classIterator.hasNext()) {
            // assign the lecture to be the next lecture in the iterator
            Lecture lecture = classIterator.next();
            // if the lectures subject is equal to the string subject from studentGrp.getSubjectName() al
            if (lecture.getSubject().equalsIgnoreCase(subject)) {
                // this is creating a lecture with the professor and the subject with the student group
                Lecture mainLecture = new Lecture(
                        lecture.getProfessor(), lecture.getSubject());
                mainLecture.setStudentGroup(studentGrp);
                lecturesStack.push(mainLecture);
                break;
            }
        }
    }
}
```

**Figure 6.17 setTimetable**

In the figure above (*Figure 6.17*), it is visible to see what is going on by reading the comments. A stack is created for the lecture as it will be used to pop the lecture at the top of the list further down the function. The for loop iterates through the student groups, within the for loop it is clear that the subject for each student group is got. It is also getting the number of lectures the student group has per week, and iterating through the number of lectures and shuffling the classes (which is an arraylist of all the lectures). Within the while loop the lecture is created creating the professor, the subject and also the student group.

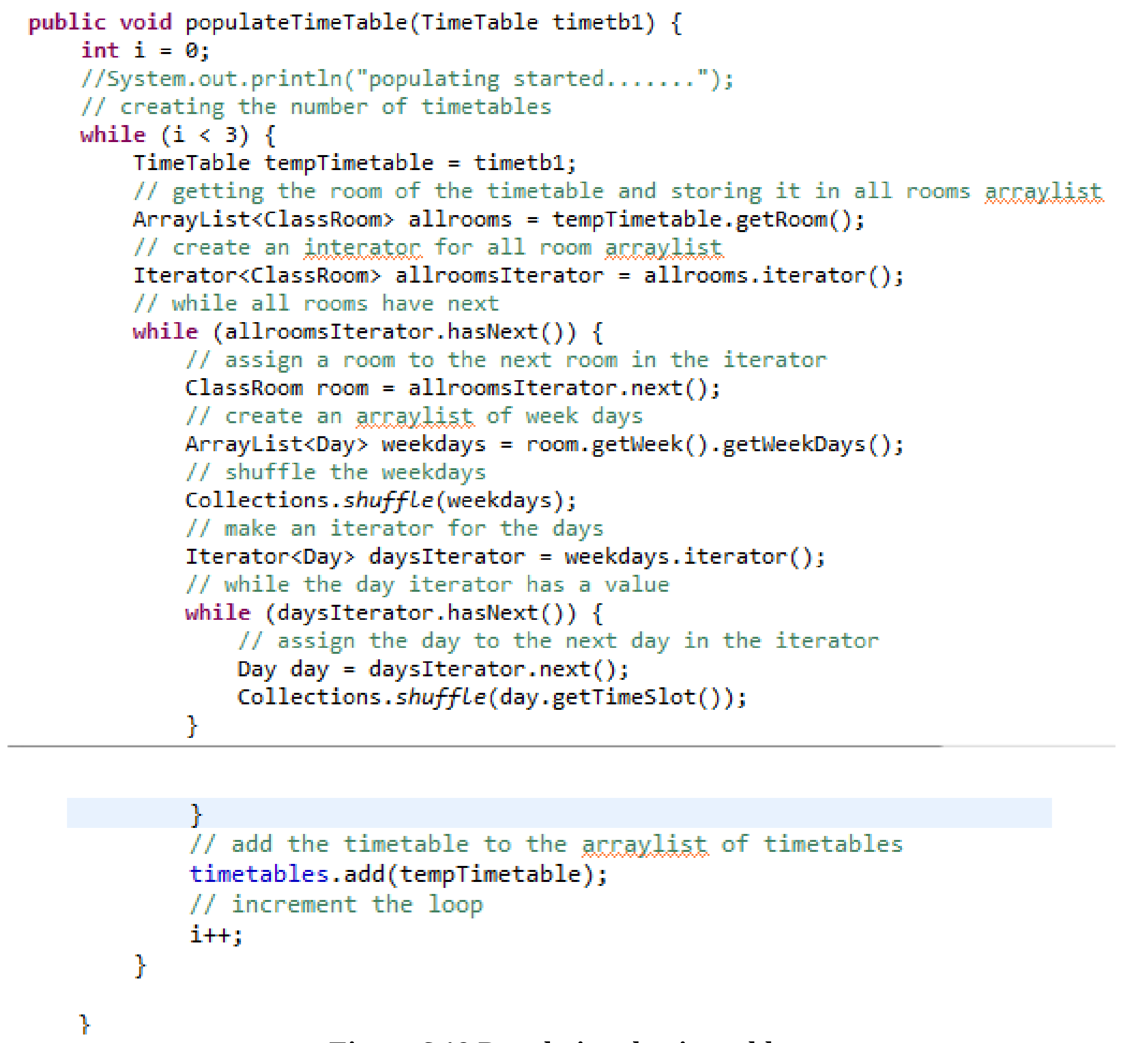
In the figure below (*6.18*), it is placing a lecture in a timeslot where there is no lecture placed. So it checks if the timeslot to put a lecture is free (checking if the lecture in that timeslot is equal to null), and if the lecture is free the lecture can then be assigned to that timeslot.



```java
while (daysIterator.hasNext()) {
    // get the next day
    Day day = daysIterator.next();
    ArrayList<TimeSlot> timeslots = day.getTimeSlot();
    Iterator<TimeSlot> timeslotIterator = timeslots.iterator();
    while (timeslotIterator.hasNext()) {
        TimeSlot lecture2 = timeslotIterator.next();
        // if the timeslot is free add the lecture to this timeslot
        if (lecture2.getLecture() == null) {
            lecture2.setLecture(lecture);
            return;
        }
    }
}
```

**Figure 6.18 Placing a lecture if the timeslot if free**

Once the timetable is then populated in the initialization class. In the populate function, the timetable that was passed through the initialize timetable is then passed through the populate timetable function and it is populated four times, randomly shuffling the timeslots and adding to an arraylist of timetables.



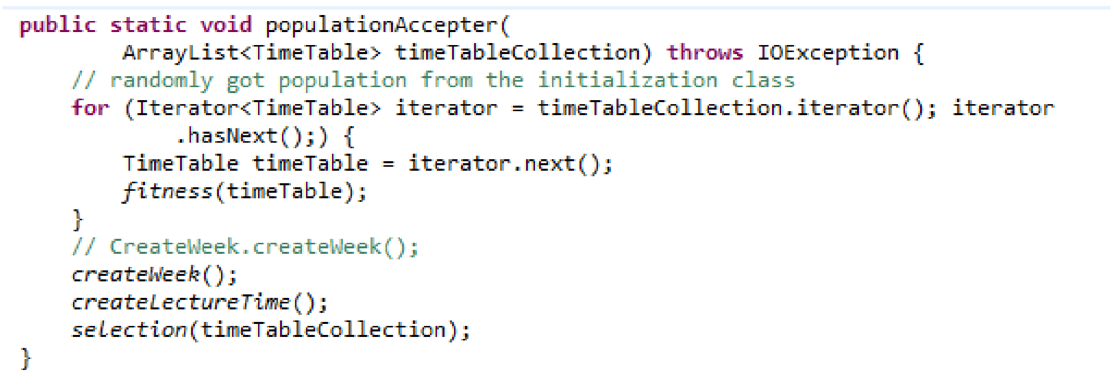
```java
public void populateTimeTable(TimeTable timetb1) {
    int i = 0;
    //System.out.println("populating started.......");
    // creating the number of timetables
    while (i < 3) {
        TimeTable tempTimetable = timetb1;
        // getting the room of the timetable and storing it in all rooms arraylist
        ArrayList<ClassRoom> allrooms = tempTimetable.getRoom();
        // create an interator for all room arraylist
        Iterator<ClassRoom> allroomsIterator = allrooms.iterator();
        // while all rooms have next
        while (allroomsIterator.hasNext()) {
            // assign a room to the next room in the iterator
            ClassRoom room = allroomsIterator.next();
            // create an arraylist of week days
            ArrayList<Day> weekdays = room.getWeek().getWeekDays();
            // shuffle the weekdays
            Collections.shuffle(weekdays);
            // make an iterator for the days
            Iterator<Day> daysIterator = weekdays.iterator();
            // while the day iterator has a value
            while (daysIterator.hasNext()) {
                // assign the day to the next day in the iterator
                Day day = daysIterator.next();
                Collections.shuffle(day.getTimeSlot());
            }

        }
        // add the timetable to the arraylist of timetables
        timetables.add(tempTimetable);
        // increment the loop
        i++;
    }

}
```

**Figure 6.19 Populating the timetable**

Once the timetables have been populated it is then passed to the genetic algorithm class: " ge.populationAccepter(timetables);"

## 6.2.8 Genetic Algorithm and how it works

The genetic algorithm has four important functions. They are known as selection, the fitness function, the mutation function and the crossover function. In the genetic algorithm class there is a function which takes in an arraylist of timetables. In the population accepter function it takes in all the timetables that were passed through from the initialization class. It then iterates through the timetable in the for loop getting the next timetable (*See figure 6.20*).

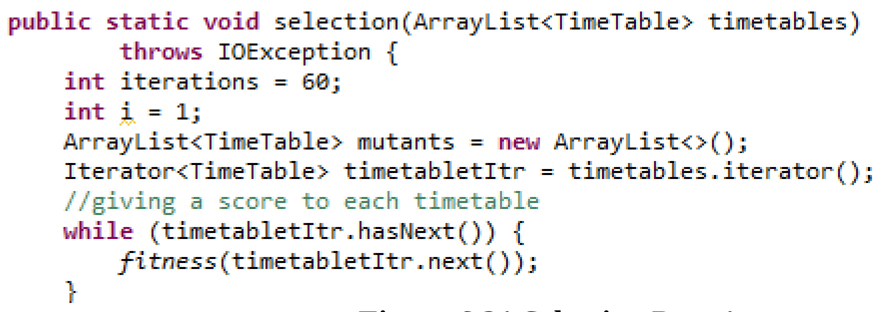


```
public static void populationAccepter(
        ArrayList<TimeTable> timeTableCollection) throws IOException {
    // randomly got population from the initialization class
    for (Iterator<TimeTable> iterator = timeTableCollection.iterator(); iterator
            .hasNext();) {
        TimeTable timeTable = iterator.next();
        fitness(timeTable);
    }
    // CreateWeek.createWeek();
    createWeek();
    createLectureTime();
    selection(timeTableCollection);
}
```

**Figure 6.20 population accepter**

In the above figure the fitness function is called for the next timetable in the arraylist, which is passed from the initialization class. As seen from the figure above it can be seen that the fitness function has been called. The function will be explained later in the paper, but it is where the timetables are given scores. The first function that will be explained will the Selection function.

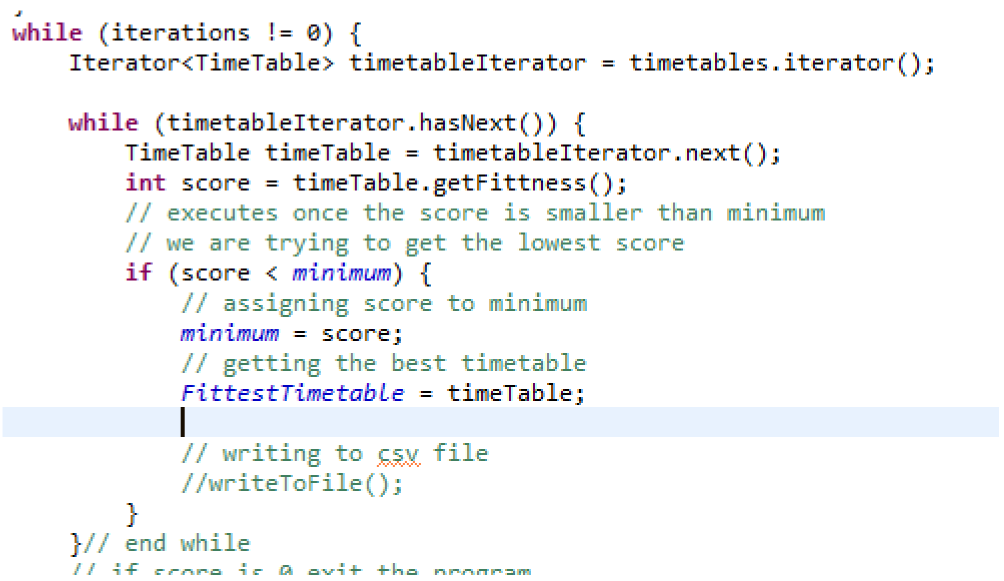
## Selection



```
public static void selection(ArrayList<TimeTable> timetables)
        throws IOException {
    int iterations = 60;
    int i = 1;
    ArrayList<TimeTable> mutants = new ArrayList<>();
    Iterator<TimeTable> timetabletItr = timetables.iterator();
    //giving a score to each timetable
    while (timetabletItr.hasNext()) {
        fitness(timetabletItr.next());
    }
```

**Figure 6.21 Selection Part 1**

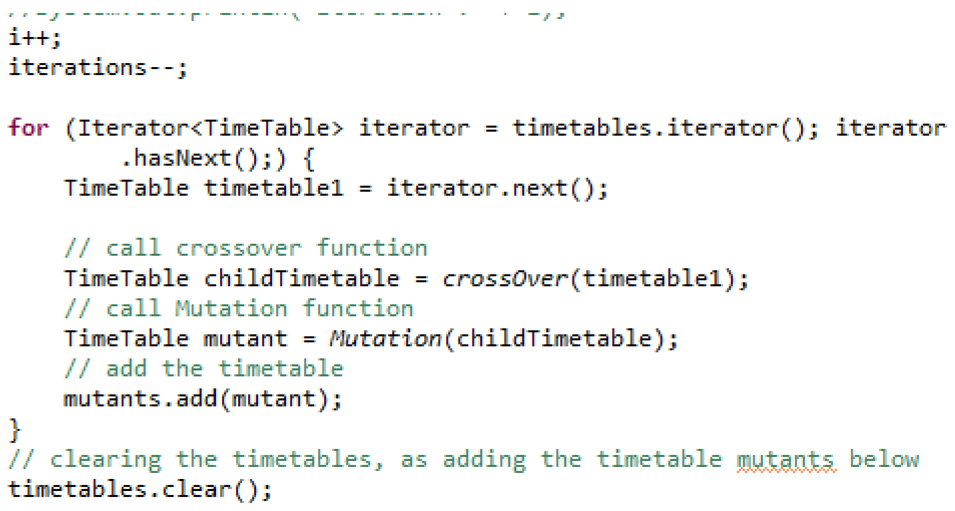
As seen from the figure above (*figure 6.21*) the selection passes an arraylist of timetables from the population accepter function (the timetable collection is passed from the initialization class). An arraylist is initialized for the timetables called mutants. Then an iterator is declared for the timetables. In the while loop the fitness function is called for one timetable at a time (the next timetable).



```
while (iterations != 0) {
    Iterator<TimeTable> timetableIterator = timetables.iterator();

    while (timetableIterator.hasNext()) {
        TimeTable timeTable = timetableIterator.next();
        int score = timeTable.getFittness();
        // executes once the score is smaller than minimum
        // we are trying to get the lowest score
        if (score < minimum) {
            // assigning score to minimum
            minimum = score;
            // getting the best timetable
            FittestTimetable = timeTable;

            // writing to csv file
            //writeToFile();
        }
    }// end while
    // if score is 0 exit the program
```

**Figure 6.22 Selection Part 2**

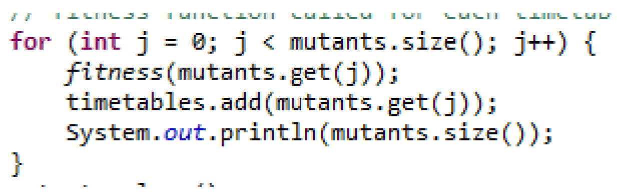
The next while loop includes the iterations (which was initialized earlier to be 60 iterations. The more iterations declared the more accurate the outcome of the fittest timetable will be, but it will take longer to execute. In the figure above an iterator of the timetables is then re-initialized (so that the timetable starts from the first timetable). Then a score of type integer is declared and gets the score of every timetable in the timetable iterator. Then the if statement is executed as it checks if the score is lower than the minimum score of a previous timetable (this allows for the timetable with the lowest score to be passed through). If it is lower than the minimum, it is assigned as the fittest timetable.



```
[illegible]
i++;
iterations--;

for (Iterator<TimeTable> iterator = timetables.iterator(); iterator
        .hasNext();) {
    TimeTable timetable1 = iterator.next();

    // call crossover function
    TimeTable childTimetable = crossOver(timetable1);
    // call Mutation function
    TimeTable mutant = Mutation(childTimetable);
    // add the timetable
    mutants.add(mutant);
}
// clearing the timetables, as adding the timetable mutants below
timetables.clear();
```

**Figure 6.23 Selection Part 3**

After the while loop is executed from *figure 6.23*, the iterations is then decremented. A for loop is then executed containing the timetable iterator until it has no value. Then a timetable is assigned from the .next() function in the iterator, this timetable is then passed into the Crossover function, (see crossover section for functionality). Once then childTimetable gets the value of the timetable that was passed through the Crossover function, it is then passed through the mutation function, and the timetable object (mutant) is then the value of "Mutation(childTimetable)". Then the mutant timetable is then added to the timetable arraylist called mutants.



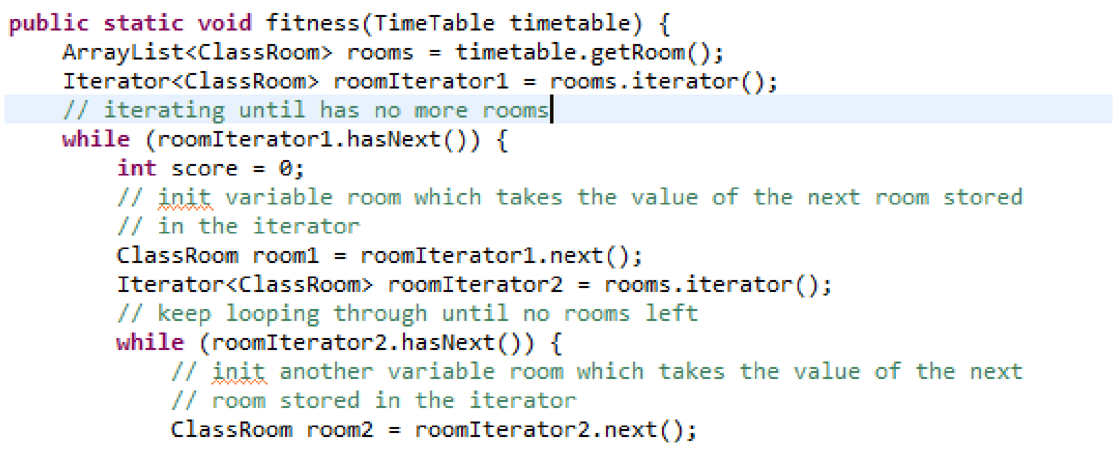
```
for (int j = 0; j < mutants.size(); j++) {
    fitness(mutants.get(j));
    timetables.add(mutants.get(j));
    System.out.println(mutants.size());
}
```

**Figure 6.24 Selection Part 4**

In the figure above (*figure 6.24*), containing the for loop, it iterates through the size of the timetables arraylist (mutants). It passed through every mutant timetable one at a time to the fitness function where it is given a score.

**Fitness Function**

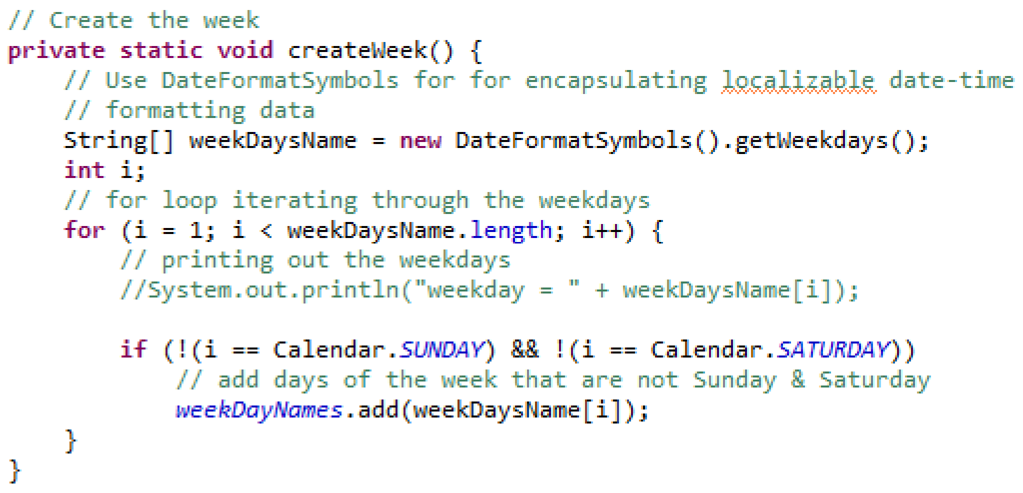
The fitness function is where the timetable is allocated a score. In *figure 6.25*, there is an arraylist for the rooms which is stored in timetable.getRoom(), and then an iterator for the rooms is initialized, and the while loop is executed until the first room iterator contains no more rooms, the first classroom is given the value of the next room in that specific iterator (roomIterator1), also initializing the score to be 0. Once inside this while loop another iterator is initialized called roomIterator2 which will be used to compare against the first room iterator.



```
public static void fitness(TimeTable timetable) {
    ArrayList<ClassRoom> rooms = timetable.getRoom();
    Iterator<ClassRoom> roomIterator1 = rooms.iterator();
    // iterating until has no more rooms
    while (roomIterator1.hasNext()) {
        int score = 0;
        // init variable room which takes the value of the next room stored
        // in the iterator
        ClassRoom room1 = roomIterator1.next();
        Iterator<ClassRoom> roomIterator2 = rooms.iterator();
        // keep looping through until no rooms left
        while (roomIterator2.hasNext()) {
            // init another variable room which takes the value of the next
            // room stored in the iterator
            ClassRoom room2 = roomIterator2.next();
```

**Figure 6.25 Fitness function iterating through rooms**

As you can see from *figure if rooms do not match* the weekdays of each of the rooms are initialized (Monday-> Friday, this is from the function which creates the weekdays in the timetable (*see figure 6.26*). The DateFormatSymbols public class is used in Java "for encapsulating localizable date-time formatting data, such as the names of the months, the names of the days of the week, and the time zone data" [29]. Within the for loop it is iterating through the length of the String array (being the length of 7 Monday -> Sunday). In the timetable there is not going to be any schedule for the days Saturday and Sunday using the calendar class in Java. If the i is not the value of Saturday or Sunday. In the if statement, it is adding the week day name to the string array weekDayNames.

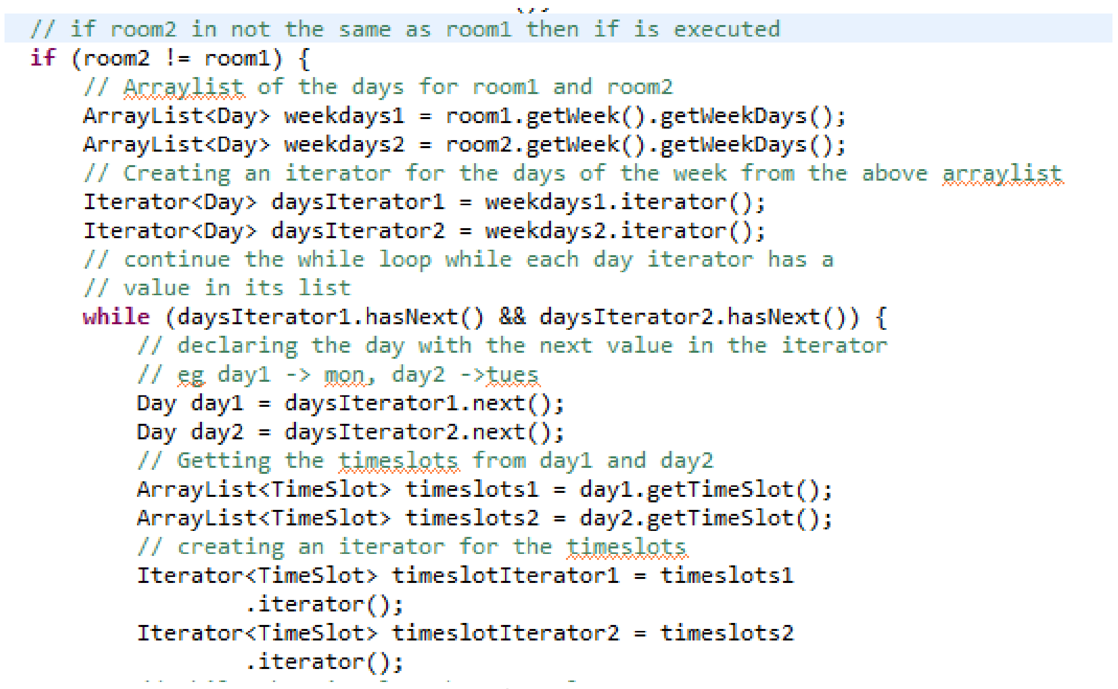


```
// Create the week
private static void createWeek() {
    // Use DateFormatSymbols for for encapsulating localizable date-time
    // formatting data
    String[] weekDaysName = new DateFormatSymbols().getWeekdays();
    int i;
    // for loop iterating through the weekdays
    for (i = 1; i < weekDaysName.length; i++) {
        // printing out the weekdays
        //System.out.println("weekday = " + weekDaysName[i]);

        if (!(i == Calendar.SUNDAY) && !(i == Calendar.SATURDAY))
            // add days of the week that are not Sunday & Saturday
            weekDayNames.add(weekDaysName[i]);
    }
}
```

**Figure 6.26 Create Week**

From *figure 6.27* below you can see that once the arraylist of the weekdays for room1 and room2 there is an iterator initialized for the days (so it can be used in the while loop). In the while loop once they both have next, the days are then initialized to be the next day in the iterator (the first time they are iterated it is going to be a Monday for example).

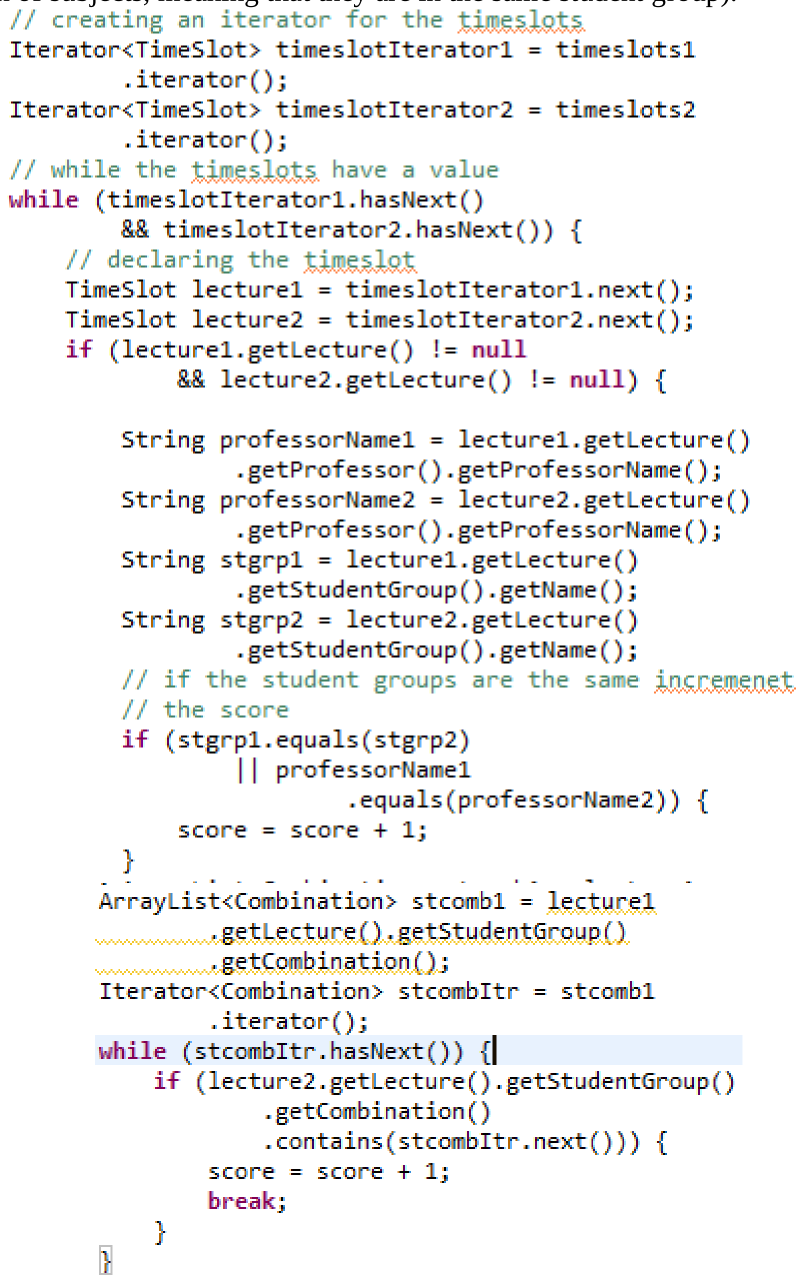
Once we have day1 and day2 we then get the timeslots for those days, the getTimeSlot is an object in the Day class. Once the time slot is stored in the time slot object arraylist, then create an iterator for that arraylist.



```
// if room2 in not the same as room1 then if is executed
if (room2 != room1) {
    // Arraylist of the days for room1 and room2
    ArrayList<Day> weekdays1 = room1.getWeek().getWeekDays();
    ArrayList<Day> weekdays2 = room2.getWeek().getWeekDays();
    // Creating an iterator for the days of the week from the above arraylist
    Iterator<Day> daysIterator1 = weekdays1.iterator();
    Iterator<Day> daysIterator2 = weekdays2.iterator();
    // continue the while loop while each day iterator has a
    // value in its list
    while (daysIterator1.hasNext() && daysIterator2.hasNext()) {
        // declaring the day with the next value in the iterator
        // eg day1 -> mon, day2 ->tues
        Day day1 = daysIterator1.next();
        Day day2 = daysIterator2.next();
        // Getting the timeslots from day1 and day2
        ArrayList<TimeSlot> timeslots1 = day1.getTimeSlot();
        ArrayList<TimeSlot> timeslots2 = day2.getTimeSlot();
        // creating an iterator for the timeslots
        Iterator<TimeSlot> timeslotIterator1 = timeslots1
                .iterator();
        Iterator<TimeSlot> timeslotIterator2 = timeslots2
                .iterator();
```

**Figure 6.27 if rooms do not match**

Once there is iterators for the timeslots for each day for both rooms, there can now be a comparison between the timeslots for the different rooms. In the screenshot below *(figure 6.28)* it checks if the professors and the student groups are clashing, so for example if the student group of lecture1 is equal to the student group of lecture2 then the score increments. Also the fitness score is incremented if the student groups both have the same combination of subjects (i.e. Student group 1 subjects -> "maths/English/Tesing", Student group 2 subjects "maths/English/Testing, this example means it is the both the student groups have the same combination of subjects, meaning that they are in the same student group).

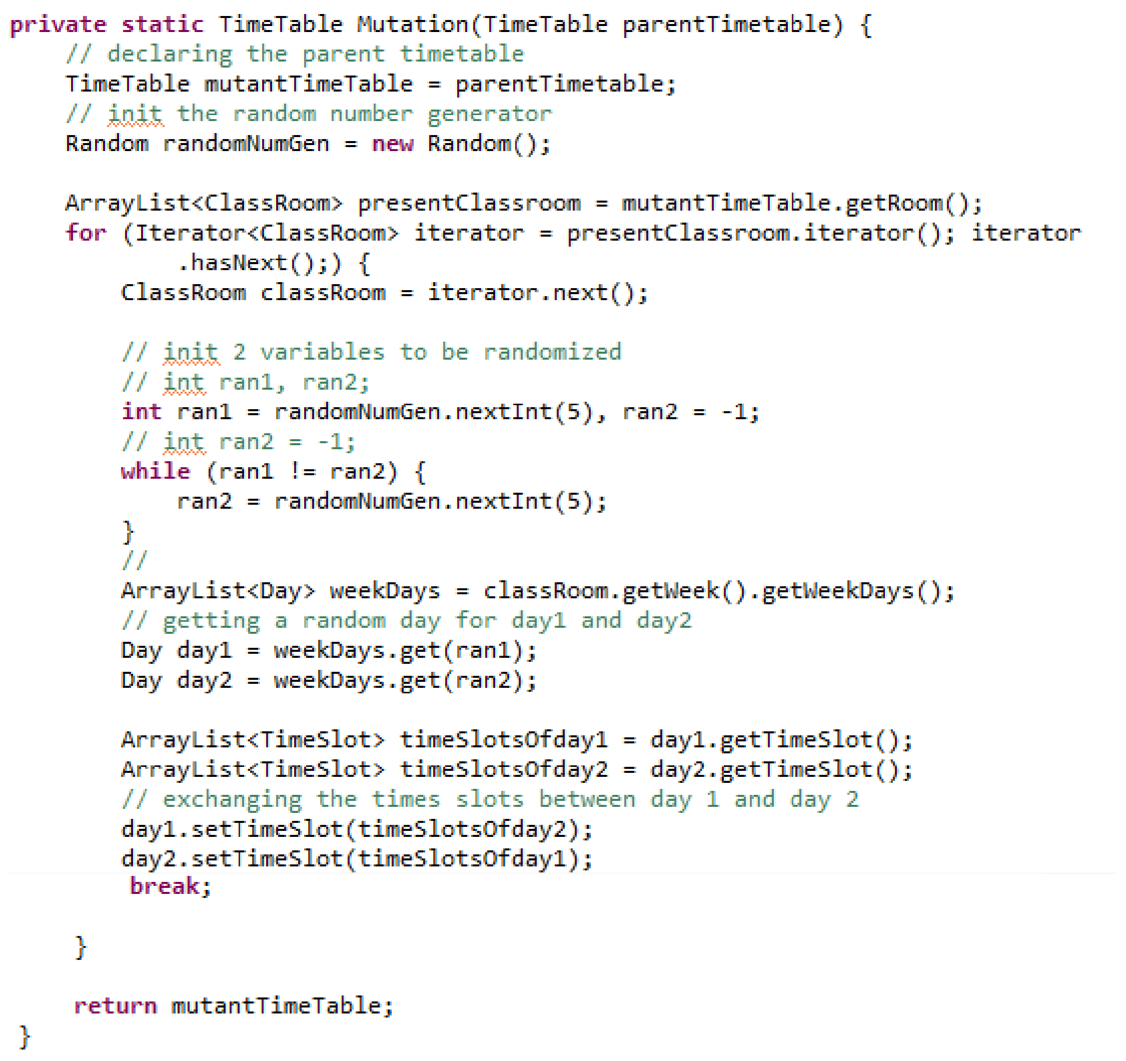


```
// creating an iterator for the timeslots
Iterator<TimeSlot> timeslotIterator1 = timeslots1
        .iterator();
Iterator<TimeSlot> timeslotIterator2 = timeslots2
        .iterator();
// while the timeslots have a value
while (timeslotIterator1.hasNext()
        && timeslotIterator2.hasNext()) {
    // declaring the timeslot
    TimeSlot lecture1 = timeslotIterator1.next();
    TimeSlot lecture2 = timeslotIterator2.next();
    if (lecture1.getLecture() != null
            && lecture2.getLecture() != null) {

        String professorName1 = lecture1.getLecture()
                .getProfessor().getProfessorName();
        String professorName2 = lecture2.getLecture()
                .getProfessor().getProfessorName();
        String stgrp1 = lecture1.getLecture()
                .getStudentGroup().getName();
        String stgrp2 = lecture2.getLecture()
                .getStudentGroup().getName();
        // if the student groups are the same incremenet
        // the score
        if (stgrp1.equals(stgrp2)
                || professorName1
                        .equals(professorName2)) {
            score = score + 1;
        }
      ArrayList<Combination> stcomb1 = lecture1
              .getLecture().getStudentGroup()
              .getCombination();
      Iterator<Combination> stcombItr = stcomb1
              .iterator();
      while (stcombItr.hasNext()) {
          if (lecture2.getLecture().getStudentGroup()
                  .getCombination()
                  .contains(stcombItr.next())) {
              score = score + 1;
              break;
          }
      }
```

**Figure 6.28 Incrementing fitness score**

**Mutation**

In the mutation function the timeslots between the days are shuffled between one another. From the figure below (*figure 6.29)* it can be seen that two days at a time are chosen at random ( between 1 and 5 representing the days of the week). The timeslots are swapped between them so the timeslots of day1 become the timeslots of day2 (*see figure 6.30).*



```
private static TimeTable Mutation(TimeTable parentTimetable) {
    // declaring the parent timetable
    TimeTable mutantTimeTable = parentTimetable;
    // init the random number generator
    Random randomNumGen = new Random();

    ArrayList<ClassRoom> presentClassroom = mutantTimeTable.getRoom();
    for (Iterator<ClassRoom> iterator = presentClassroom.iterator(); iterator
            .hasNext();) {
        ClassRoom classRoom = iterator.next();

        // init 2 variables to be randomized
        // int ran1, ran2;
        int ran1 = randomNumGen.nextInt(5), ran2 = -1;
        // int ran2 = -1;
        while (ran1 != ran2) {
            ran2 = randomNumGen.nextInt(5);
        }
        //
        ArrayList<Day> weekDays = classRoom.getWeek().getWeekDays();
        // getting a random day for day1 and day2
        Day day1 = weekDays.get(ran1);
        Day day2 = weekDays.get(ran2);

        ArrayList<TimeSlot> timeSlotsOfday1 = day1.getTimeSlot();
        ArrayList<TimeSlot> timeSlotsOfday2 = day2.getTimeSlot();
        // exchanging the times slots between day 1 and day 2
        day1.setTimeSlot(timeSlotsOfday2);
        day2.setTimeSlot(timeSlotsOfday1);
         break;

    }

    return mutantTimeTable;
}
```

**Figure 6.29 Mutation**

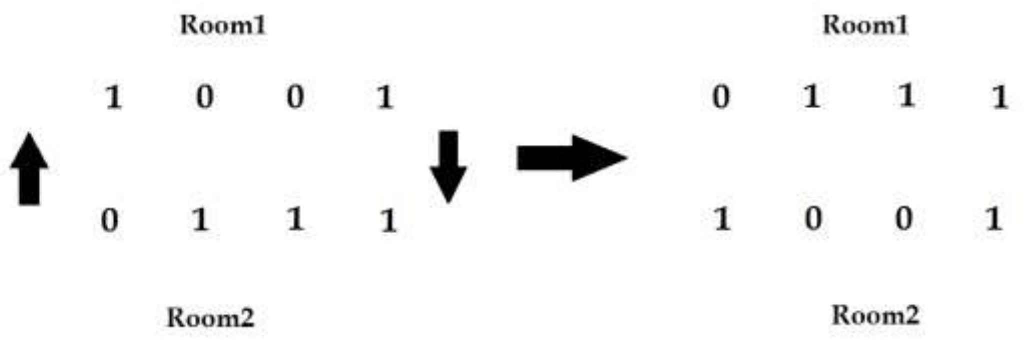
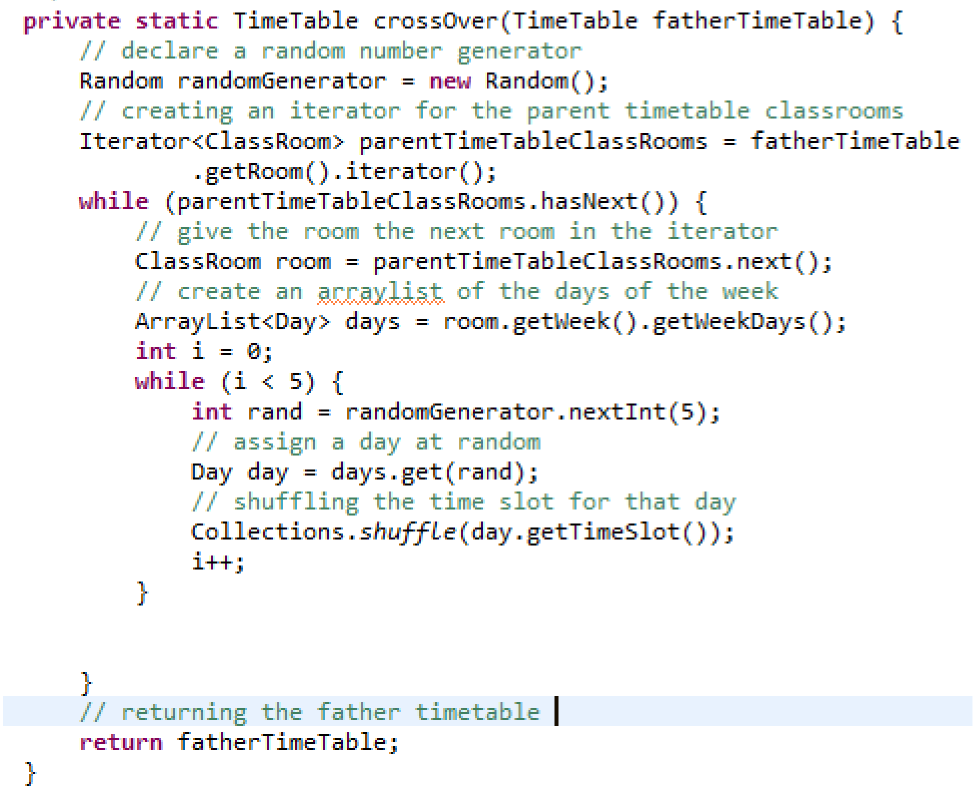


**Figure 6.30 Example of Mutation (binary encoding)**

**Crossover**
The crossover function shuffles all of the timeslots for the day, which is picked at random of 5 days. In the figure below, the father timetable is passed through which came from the selection function. A randomGenerator is then initialized, using the random function in Java. An iterator for the classrooms that are in the father timetable are stored in parentTimeTableClassRooms iterator. In the while loop it is getting the days of the week, and within the second while (iterated 5 times mirroring the days of the week) a number is then chosen at random (from 1 to 5). Then the day object is equal to the days.get(rand) any day from Monday to Friday. Once the day is got at random then the timeslots for that day is then shuffled.



```
private static TimeTable crossOver(TimeTable fatherTimeTable) {
    // declare a random number generator
    Random randomGenerator = new Random();
    // creating an iterator for the parent timetable classrooms
    Iterator<ClassRoom> parentTimeTableClassRooms = fatherTimeTable
            .getRoom().iterator();
    while (parentTimeTableClassRooms.hasNext()) {
        // give the room the next room in the iterator
        ClassRoom room = parentTimeTableClassRooms.next();
        // create an arraylist of the days of the week
        ArrayList<Day> days = room.getWeek().getWeekDays();
        int i = 0;
        while (i < 5) {
            int rand = randomGenerator.nextInt(5);
            // assign a day at random
            Day day = days.get(rand);
            // shuffling the time slot for that day
            Collections.shuffle(day.getTimeSlot());
            i++;
        }


    }
    // returning the father timetable
    return fatherTimeTable;
}
```

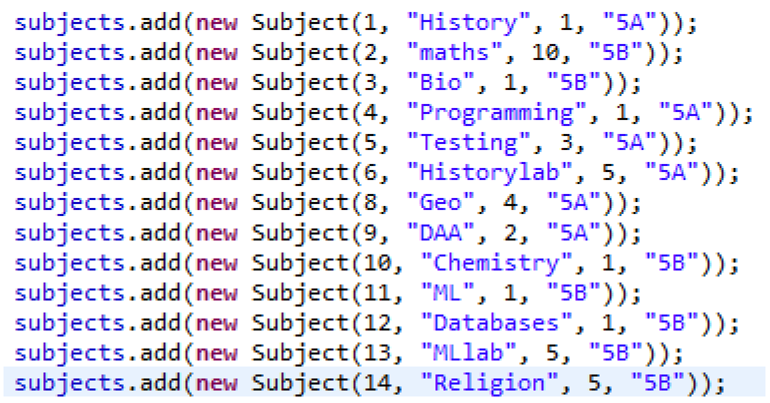
**Figure 6.31 Crossover**

## 6.2.9 Iterations of inserting data into Java

There was three iterations of data manipulation when developing in Java. The first iteration was adding to the arraylist manually, the second iteration was reading and writing CSV files and the third iteration was reading and

### *6.2.9.1 Iteration 1 - Manually Inserting Data to Arraylist*

The first iteration was inserting data into the arraylist via hardcode. As you can see from *figure 6.32* the subjects were added line by line.

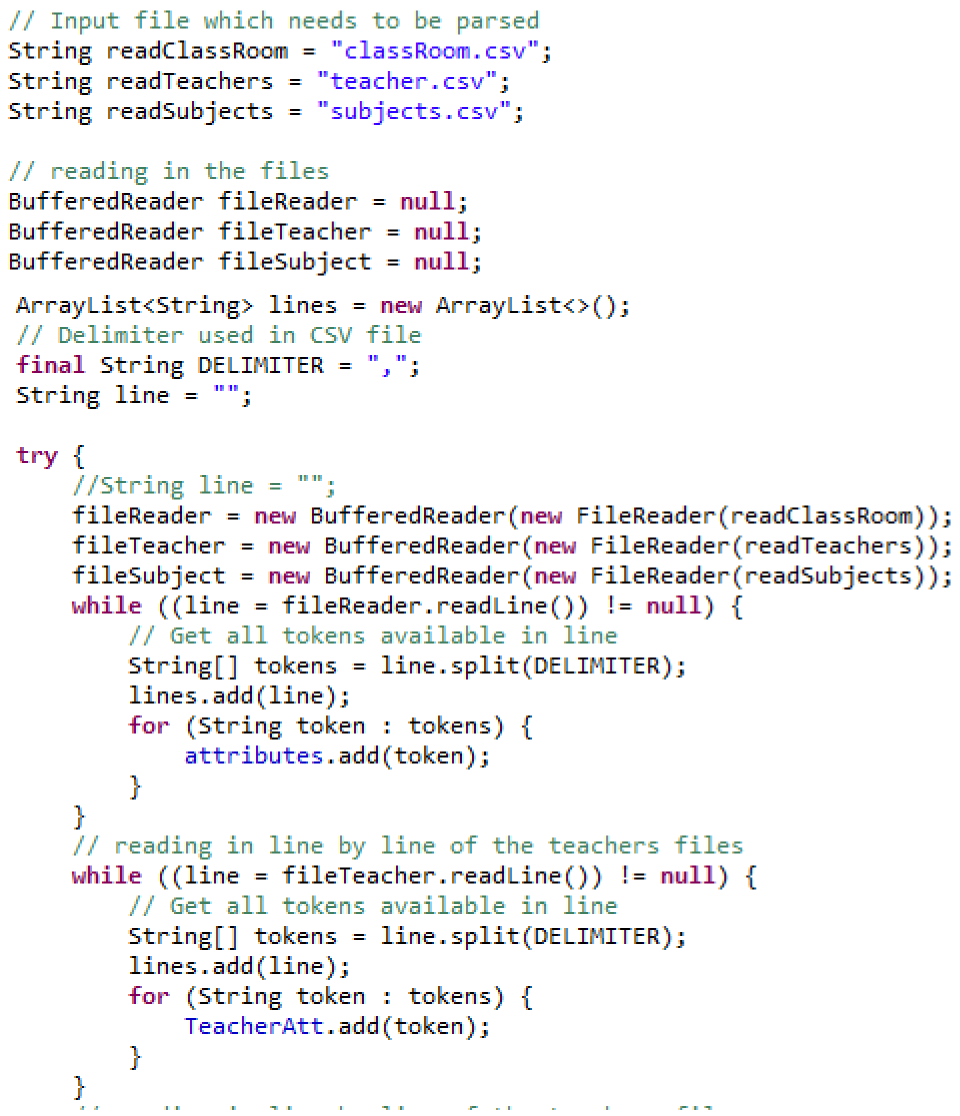


```java
subjects.add(new Subject(1, "History", 1, "5A"));
subjects.add(new Subject(2, "maths", 10, "5B"));
subjects.add(new Subject(3, "Bio", 1, "5B"));
subjects.add(new Subject(4, "Programming", 1, "5A"));
subjects.add(new Subject(5, "Testing", 3, "5A"));
subjects.add(new Subject(6, "Historylab", 5, "5A"));
subjects.add(new Subject(8, "Geo", 4, "5A"));
subjects.add(new Subject(9, "DAA", 2, "5A"));
subjects.add(new Subject(10, "Chemistry", 1, "5B"));
subjects.add(new Subject(11, "ML", 1, "5B"));
subjects.add(new Subject(12, "Databases", 1, "5B"));
subjects.add(new Subject(13, "MLlab", 5, "5B"));
subjects.add(new Subject(14, "Religion", 5, "5B"));
```

**Figure 6.32 Example of first Iteration of adding Data**

### *6.2.9.2 Iteration 2 - Reading and writing CSV files*

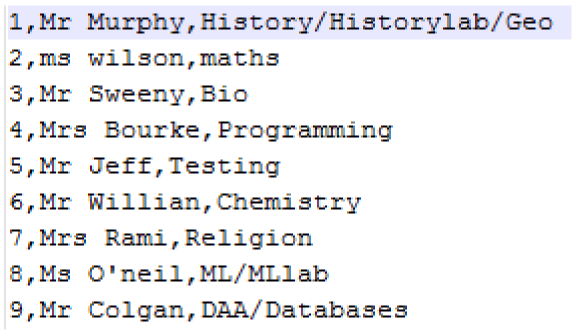
The second iteration of reading and writing the data was reading and writing CSV files. There was a teacher CSV file, a class CSV file and a subjects CSV file (*see figure 6.33*).



```java
// Input file which needs to be parsed
String readClassRoom = "classRoom.csv";
String readTeachers = "teacher.csv";
String readSubjects = "subjects.csv";

// reading in the files
BufferedReader fileReader = null;
BufferedReader fileTeacher = null;
BufferedReader fileSubject = null;

ArrayList<String> lines = new ArrayList<>();
// Delimiter used in CSV file
final String DELIMITER = ",";
String line = "";

try {
    //String line = "";
    fileReader = new BufferedReader(new FileReader(readClassRoom));
    fileTeacher = new BufferedReader(new FileReader(readTeachers));
    fileSubject = new BufferedReader(new FileReader(readSubjects));
    while ((line = fileReader.readLine()) != null) {
        // Get all tokens available in line
        String[] tokens = line.split(DELIMITER);
        lines.add(line);
        for (String token : tokens) {
            attributes.add(token);
        }
    }
    // reading in line by line of the teachers files
    while ((line = fileTeacher.readLine()) != null) {
        // Get all tokens available in line
        String[] tokens = line.split(DELIMITER);
        lines.add(line);
        for (String token : tokens) {
            TeacherAtt.add(token);
        }
    }
```

**Figure 6.33 Reading CSV files**

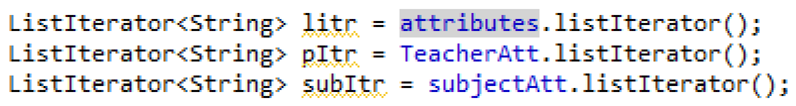
From *figure 6.32* you are able to see that here the Java program is reading in the class, subject and teacher CSV file which contains all of the necessary data (*See figure 6.34*). The String 'line' is used in the while loop until there is no more lines in the CSV to be read. Within the while loop there is a string array which is split by the ', ' delimiter. There is a for loop which loops through all of the attributes in the tokens string array and added to attributes (a string arraylist).



```
1,Mr Murphy,History/Historylab/Geo
2,ms wilson,maths
3,Mr Sweeny,Bio
4,Mrs Bourke,Programming
5,Mr Jeff,Testing
6,Mr Willian,Chemistry
7,Mrs Rami,Religion
8,Ms O'neil,ML/MLlab
9,Mr Colgan,DAA/Databases
```

**Figure 6.34 Example of teacher data in CSV file**

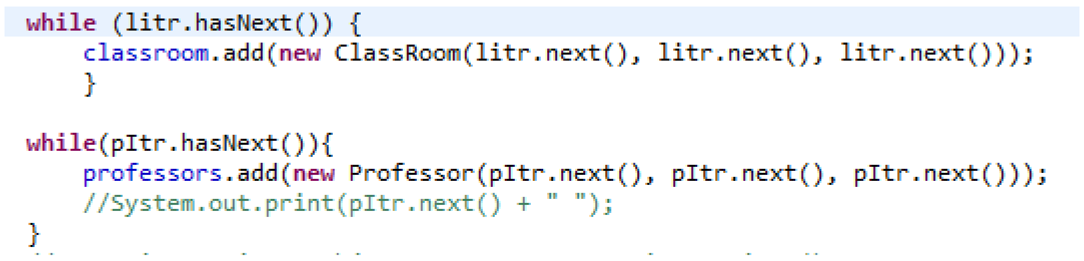
Once the attributes are added to the Arraylist they are then assigned to a ListIterator (*see figure 6.35*).



```
ListIterator<String> litr = attributes.listIterator();
ListIterator<String> pItr = TeacherAtt.listIterator();
ListIterator<String> subItr = subjectAtt.listIterator();
```

**Figure 6.35 Example of ListIterators**

These list iterators are then used to add to the arraylists (*See figure 6.36*).



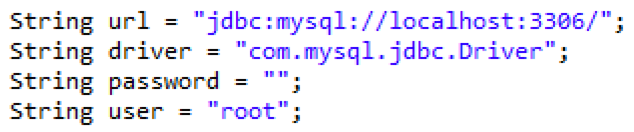
```
while (litr.hasNext()) {
    classroom.add(new ClassRoom(litr.next(), litr.next(), litr.next()));
    }

while(pItr.hasNext()){
    professors.add(new Professor(pItr.next(), pItr.next(), pItr.next()));
    //System.out.print(pItr.next() + " ");
}
```

**Figure 6.36 Example of ListIterators used to add to Arraylist**

### *6.2.9.3 - Iteration 3 - Reading and writing from MYSQL Database*

The final iteration of reading and writing data was reading and writing to and from the MYSQL database. This was the last iteration as the web application had to be developed.

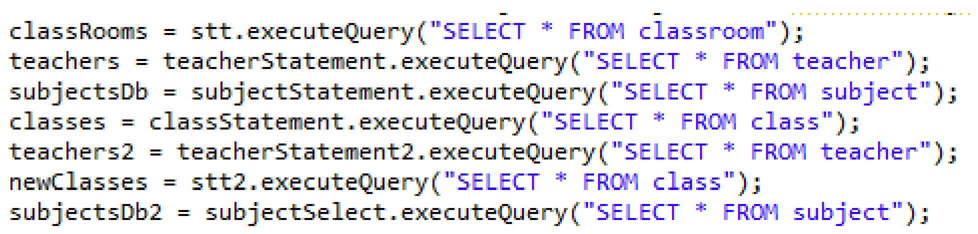
The database server that is used is an apache server, and to connect to the server from the Java. Firstly, to use Java with the Apache MYSQL server you need to download the mysql JDBC driver that will allow you to be able to connect to the MYSQL database. You then would include this jar file within the project structure. The properties that are needed to connect to the database on phpmyadmin is shown in *figure 6.37*. The port that is used is the default local port on XAMPP (port 3306) with the username and password being the credentials that is appropriate to login to phpmyadmin.



```
String url = "jdbc:mysql://localhost:3306/";
String driver = "com.mysql.jdbc.Driver";
String password = "";
String user = "root";
```

**Figure 6.37 Database Properties file**

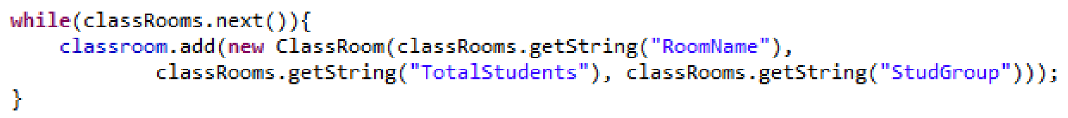
Once the connection has been successful, it is now possible to select from the database. The execute query command in java is used to execute queries from the database. These variables are known as resultsets and they store all of the data that is in the specified table. So for example the classRooms resultset will contain the class id and the class group.



```
classRooms = stt.executeQuery("SELECT * FROM classroom");
teachers = teacherStatement.executeQuery("SELECT * FROM teacher");
subjectsDb = subjectStatement.executeQuery("SELECT * FROM subject");
classes = classStatement.executeQuery("SELECT * FROM class");
teachers2 = teacherStatement2.executeQuery("SELECT * FROM teacher");
newClasses = stt2.executeQuery("SELECT * FROM class");
subjectsDb2 = subjectSelect.executeQuery("SELECT * FROM subject");
```

**Figure 6.38 Selecting from the database**

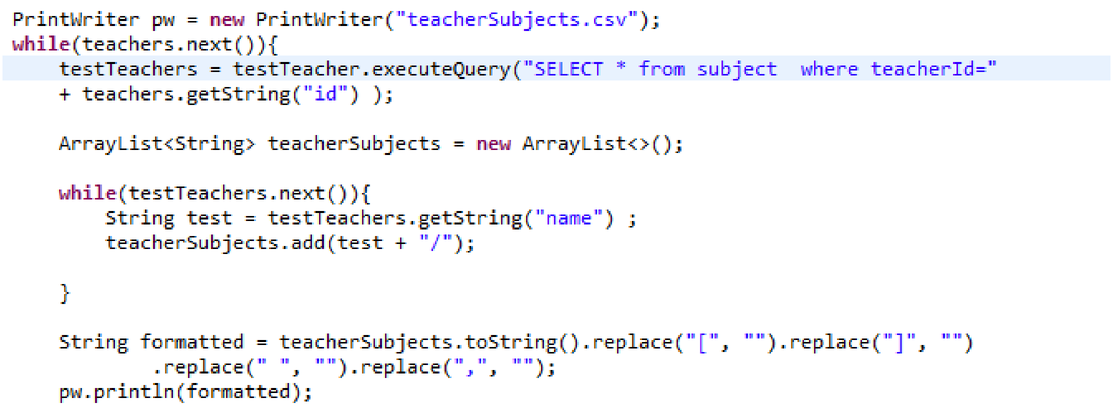
Each of these resultsets will be stored in an arraylist in Java. These arraylists mirror the structure of the database on phpmyadmin. In figure 6.39 the classRooms resultset is iterating through each row in the database and it is adding to the arraylist row by row. The while loop will be terminated when there is no more rows in the resultset (classRooms). The arraylist is an object arraylist with the object being 'ClassRoom'. The arraylist gets the room name, the total number of students and the student group from the database, until the while loop is terminated.



```
while(classRooms.next()){
    classroom.add(new ClassRoom(classRooms.getString("RoomName"),
            classRooms.getString("TotalStudents"), classRooms.getString("StudGroup")));
}
```

**Figure 6.39 Example putting data from resultset into an arraylist**

Next is to add the teachers to an arraylist. Firstly there is a CSV file created, again while the teachers.next() from the resultset has a value, there is a select statement which gets which teacher teachers which subject for example:
-> teacher with the ID of 27 (Mr Murphy) can teach more than one subject
-> Maths & English
So then an arraylist of type string is created clalled teacherSubjects and this will store all the subjects that the teacher teaches again (Mr Murphy -> English & Maths). Then the select is iterated (selecting where the teacher ID in the teacher table exists in the subjects table (*see figure 6.40*).

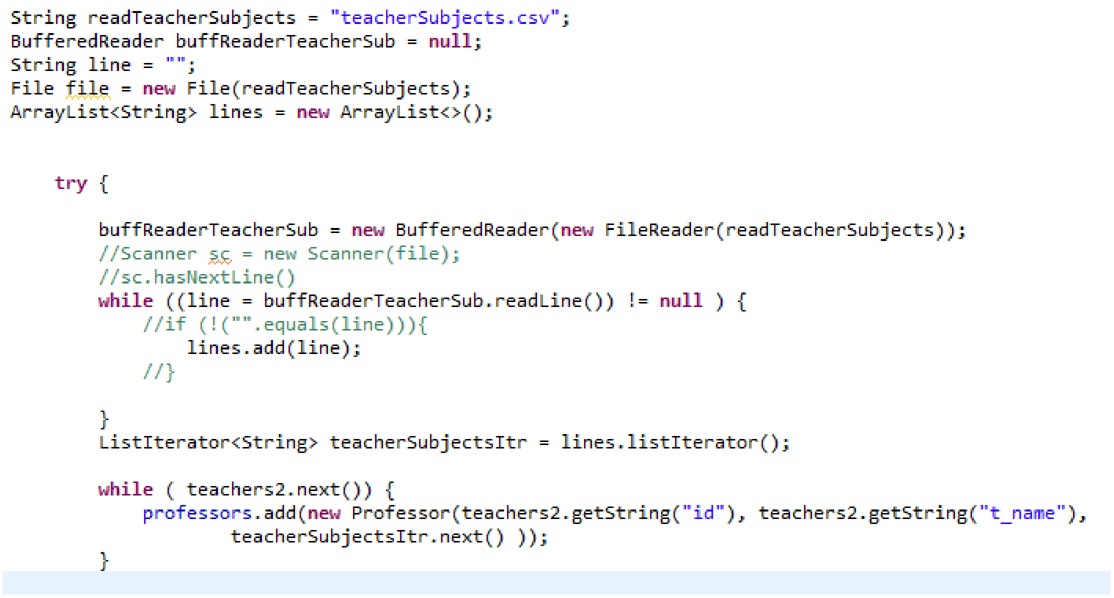


```
PrintWriter pw = new PrintWriter("teacherSubjects.csv");
while(teachers.next()){
    testTeachers = testTeacher.executeQuery("SELECT * from subject  where teacherId="
    + teachers.getString("id") );

    ArrayList<String> teacherSubjects = new ArrayList<>();

    while(testTeachers.next()){
        String test = testTeachers.getString("name") ;
        teacherSubjects.add(test + "/");

    }

    String formatted = teacherSubjects.toString().replace("[", "").replace("]", "")
            .replace(" ", "").replace(",", "");
    pw.println(formatted);
```

**Figure 6.40 Teacher Subjects to CSV file**

As you can see from *figure 6.40* the testTeachers (resultset) contains all the data from the select statement. In the while loop the teacherSubjects arraylist adds each of the subject names and appends a '/' which will separate each subject to be for example -> Maths/English/ The formatted string formats the arralist (teacherSubjects) and adds it to the CSV file using printwriter. Each line in the CSV file separates each different teacher that is in the database.

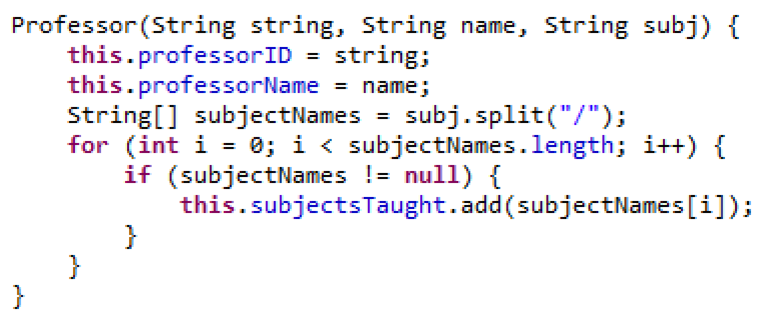
Once the subjects for each teacher is in the CSV file you then read from the CSV file using buffered reader in Java. An arraylist which will contain all of the lines in the CSV file will be created, and the list iterator will iterate through every line in the arraylist (*See figure 6.41)*.



```
String readTeacherSubjects = "teacherSubjects.csv";
BufferedReader buffReaderTeacherSub = null;
String line = "";
File file = new File(readTeacherSubjects);
ArrayList<String> lines = new ArrayList<>();


    try {

        buffReaderTeacherSub = new BufferedReader(new FileReader(readTeacherSubjects));
        //Scanner sc = new Scanner(file);
        //sc.hasNextLine()
        while ((line = buffReaderTeacherSub.readLine()) != null ) {
            //if (!("".equals(line))){
                lines.add(line);
            //}

        }
        ListIterator<String> teacherSubjectsItr = lines.listIterator();

        while ( teachers2.next()) {
            professors.add(new Professor(teachers2.getString("id"), teachers2.getString("t_name"),
                    teacherSubjectsItr.next() ));
        }
```

**6.41 Reading from CSV & adding teachers**

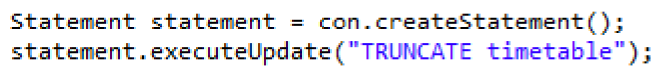
Then you add the teachers to the arraylist and the teacherSubjectsItr will contain the subjects which the teacher teaches. In *figure 6.42* you can see how the subjects that the teacher teaches is handled.



```
Professor(String string, String name, String subj) {
    this.professorID = string;
    this.professorName = name;
    String[] subjectNames = subj.split("/");
    for (int i = 0; i < subjectNames.length; i++) {
        if (subjectNames != null) {
            this.subjectsTaught.add(subjectNames[i]);
        }
    }
}
```

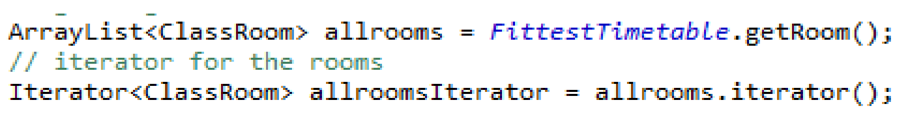
**Figure 6.42 How adding a teacher/ professor is handled**

When inserting into the database in the Genetic algorithm class, every time the timetable is scheduled the database that is in the 'timetable' table must be deleted before the data of the schedule is inserted into the database. In the function insertDatabase the 'timetable' table must be truncated (*see figure 6.43*). In the figure below is an example of how the table's data is delete/ truncated. Every time that this function is called the data that is in the table is deleted, before it inserts the new schedule.



```
Statement statement = con.createStatement();
statement.executeUpdate("TRUNCATE timetable");
```

**Figure 6.43 Truncate**

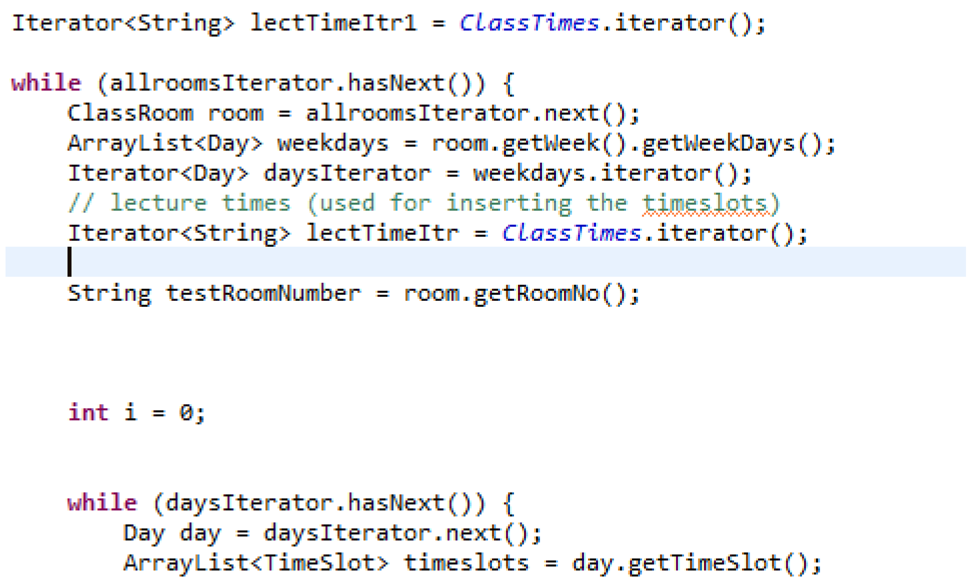


```
ArrayList<ClassRoom> allrooms = FittestTimetable.getRoom();
// iterator for the rooms
Iterator<ClassRoom> allroomsIterator = allrooms.iterator();
```

**Figure 6.44 getting fittest timetable**

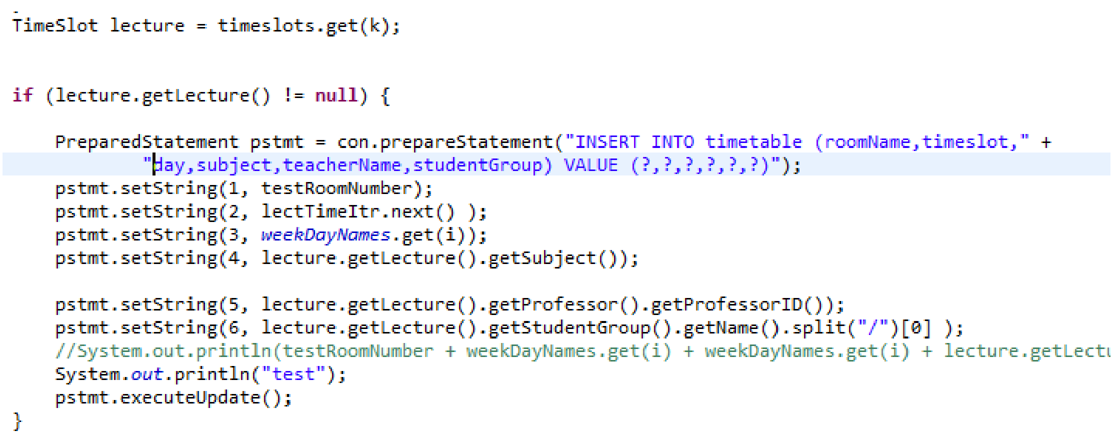
In the figure above (figure *6.44*) the allroomsIterator is an iterator from the arraylist of object classroom which gets all rooms in the fittest timetable ( the fittest timetable is again the timetable with the lowest score).

In *figure 6.45* it is getting the class times of the schedule and storing it in a lecture time iterator of type string. The while loop terminates until it has iterated through all of the rooms in the iterator. In the classroom iterator an arraylist of the days of the week are created. The testRoomNumber is the being assigned to the current room for example room number 'U1'. The second while loop iterates though all of the days in the week and within the while loop it is getting the timeslots for that specific day.



```
Iterator<String> lectTimeItr1 = ClassTimes.iterator();

while (allroomsIterator.hasNext()) {
    ClassRoom room = allroomsIterator.next();
    ArrayList<Day> weekdays = room.getWeek().getWeekDays();
    Iterator<Day> daysIterator = weekdays.iterator();
    // lecture times (used for inserting the timeslots)
    Iterator<String> lectTimeItr = ClassTimes.iterator();

    String testRoomNumber = room.getRoomNo();



    int i = 0;


    while (daysIterator.hasNext()) {
        Day day = daysIterator.next();
        ArrayList<TimeSlot> timeslots = day.getTimeSlot();
```

**Figure 6.45 Insert to Database - getting the rooms**



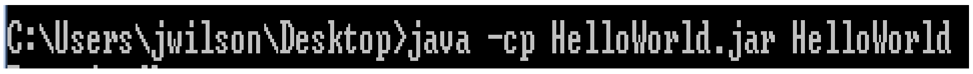
```
TimeSlot lecture = timeslots.get(k);


if (lecture.getLecture() != null) {

    PreparedStatement pstmt = con.prepareStatement("INSERT INTO timetable (roomName,timeslot," +
            "day,subject,teacherName,studentGroup) VALUE (?,?,?,?,?,?)");
    pstmt.setString(1, testRoomNumber);
    pstmt.setString(2, lectTimeItr.next() );
    pstmt.setString(3, weekDayNames.get(i));
    pstmt.setString(4, lecture.getLecture().getSubject());

    pstmt.setString(5, lecture.getLecture().getProfessor().getProfessorID());
    pstmt.setString(6, lecture.getLecture().getStudentGroup().getName().split("/")[0] );
    //System.out.println(testRoomNumber + weekDayNames.get(i) + weekDayNames.get(i) + lecture.getLectu
    System.out.println("test");
    pstmt.executeUpdate();
}
```

**Figure 6.46 Inserting to the Database using Prepared Statement**

In the figure above (*figure 6.46*), the if statement is very important as it makes sure that the current lecture in the iteration is not empty, and if it is not empty it executes. In the if, the prepared statement is used to insert into the database. Using the prepared statement projects the query from against SQL injection, the only consequence of having to parameterize the query but it makes the query a lot safer that an ordinary statement in Java. Secondly, it is good to use a prepared statement in the insert as the query is rewritten and compiled by the database server. From the image above it is inserting the current room number, the time slots so for example 9-10, 10-11 and so on, the week day names, the subject, the id of the teacher and the student group. The prepared statement is then executed using the execute update function.

## 6.2.10 Java with PHP

As the algorithm was implemented using Java, Java has to be able to work with PHP. The Java program was exported to be a jar file, and was made sure that it was able to run from command line in windows, (*see figure 6.47*).
In this example the jar was exported to the desktop and it runs the main which was a simple system print of hello world, and once that worked it was then a case of exporting the java program to XAMPP.



```
C:\Users\jwilson\Desktop>java -cp HelloWorld.jar HelloWorld
```

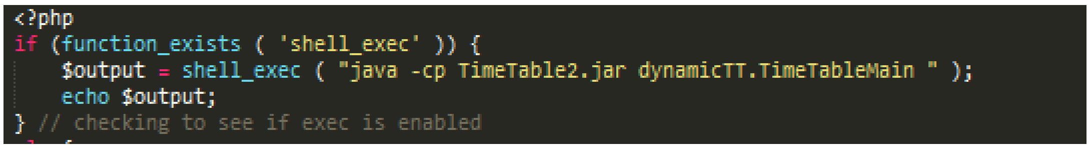
**Figure 6.47 Test compiling jar from command line**

The jar file was then exported to the admins directory in XAMPP

- /FYP/admin/timetable.jar

as it is stored on the local server with the PHP files.

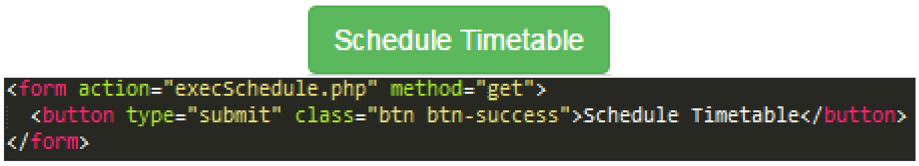
Once the jar file was exported onto the local server admin directory there is a php file called execSchedule.php which compiles the jar file mirroring the above figure, but the directory is default (so it can be used on any system) (*see Figure 6.48.)*.



```
<?php
if (function_exists ( 'shell_exec' )) {
    $output = shell_exec ( "java -cp TimeTable2.jar dynamicTT.TimeTableMain " );
    echo $output;
} // checking to see if exec is enabled
```

**Figure 6.48 PHP Shell Exec**

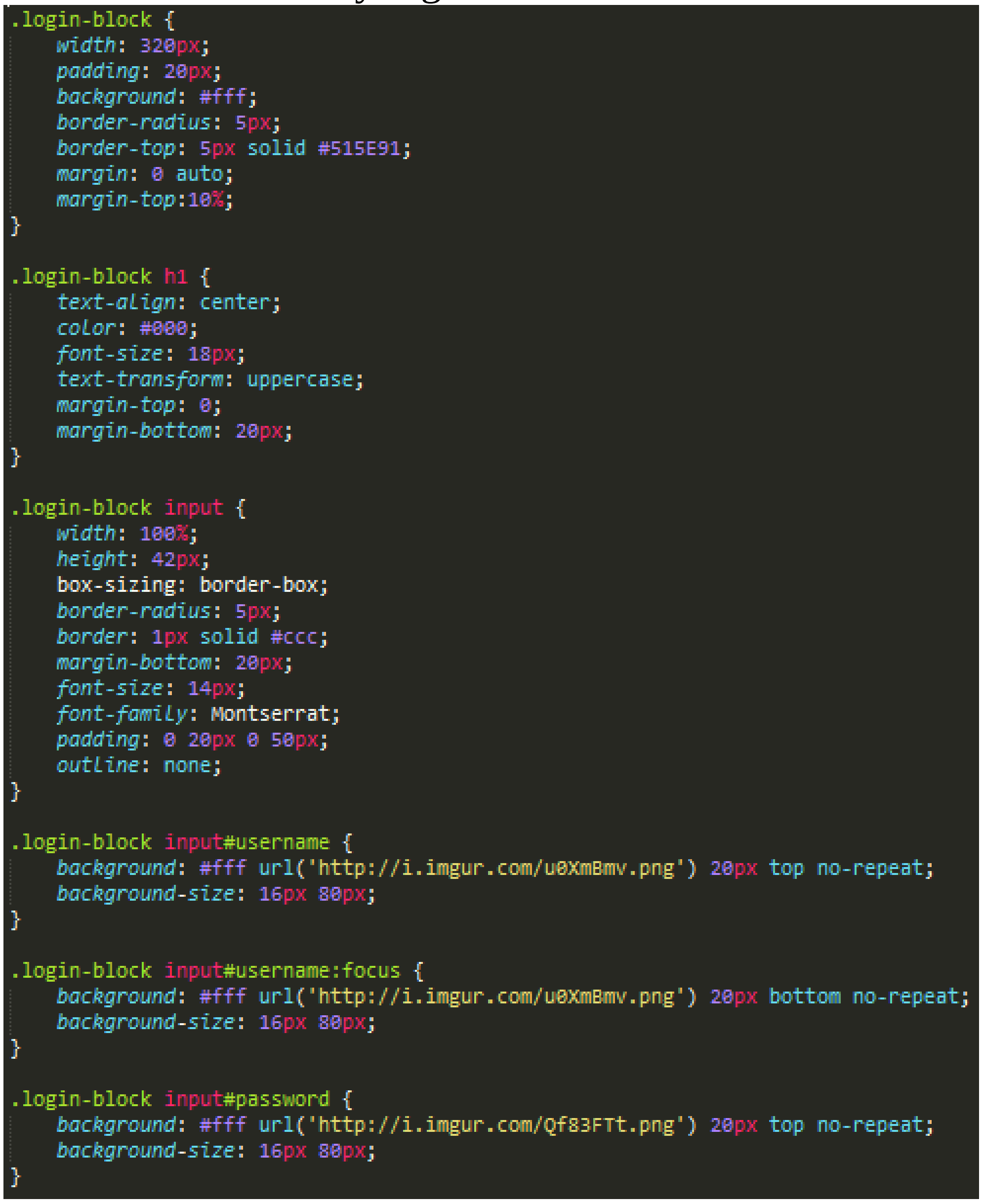
When it was possible to run the PHP script (execSchedule.php) and ran the jar file successfully a button was developed. When the button is clicked by the admin it runs the PHP file execSchedule.php (*see figure 6.98*). Once the execSchedule.php is called it is then redirected to the page where the schedule timetable exists using PHP's header location function.



```
<form action="execSchedule.php" method="get">
  <button type="submit" class="btn btn-success">Schedule Timetable</button>
</form>
```

**Figure 6.49 Button Frontend & Backend**

## 6.2.11Use of CSS and styling



```css
.login-block {
    width: 320px;
    padding: 20px;
    background: #fff;
    border-radius: 5px;
    border-top: 5px solid #515E91;
    margin: 0 auto;
    margin-top:10%;
}

.login-block h1 {
    text-align: center;
    color: #000;
    font-size: 18px;
    text-transform: uppercase;
    margin-top: 0;
    margin-bottom: 20px;
}

.login-block input {
    width: 100%;
    height: 42px;
    box-sizing: border-box;
    border-radius: 5px;
    border: 1px solid #ccc;
    margin-bottom: 20px;
    font-size: 14px;
    font-family: Montserrat;
    padding: 0 20px 0 50px;
    outline: none;
}

.login-block input#username {
    background: #fff url('http://i.imgur.com/u0XmBmv.png') 20px top no-repeat;
    background-size: 16px 80px;
}

.login-block input#username:focus {
    background: #fff url('http://i.imgur.com/u0XmBmv.png') 20px bottom no-repeat;
    background-size: 16px 80px;
}

.login-block input#password {
    background: #fff url('http://i.imgur.com/Qf83FTt.png') 20px top no-repeat;
    background-size: 16px 80px;
}
```

**Figure 6.50 Use of CSS and styling**

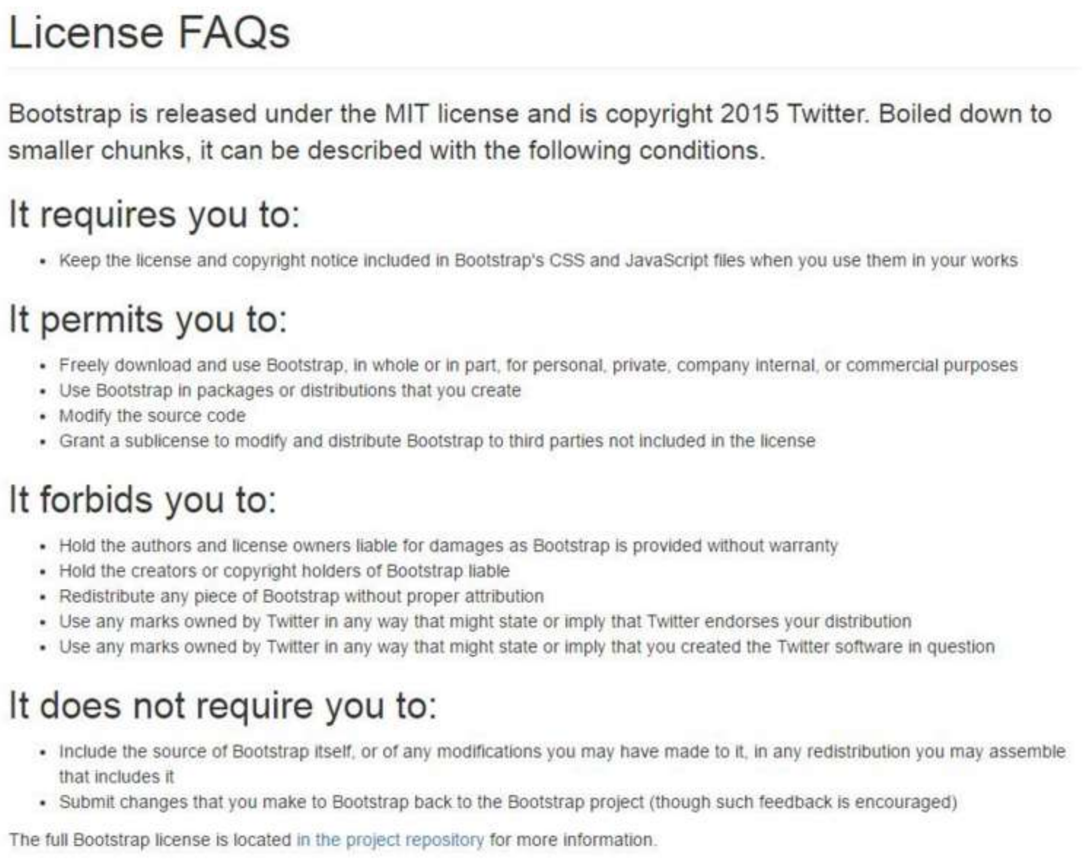
In the figure above (*Figure 6.50*), it shows how some of the CSS classes are able to be extended with the use of CSS. This CSS script has been implemented in order to further style the application.

## 6.4 Summary of third party code used

### 6.4.1 Twitter Bootstrap

Twitter bootstrap was used for the front end display of the development of the web application to display's the responsive navigation bar and footer as well as displaying the tables in the database.

**Licence information available at:**
http://getbootstrap.com/getting-started/



License FAQs

Bootstrap is released under the MIT license and is copyright 2015 Twitter. Boiled down to smaller chunks, it can be described with the following conditions.

It requires you to:

- Keep the license and copyright notice included in Bootstrap's CSS and JavaScript files when you use them in your works

It permits you to:

- Freely download and use Bootstrap, in whole or in part, for personal, private, company internal, or commercial purposes
- Use Bootstrap in packages or distributions that you create
- Modify the source code
- Grant a sublicense to modify and distribute Bootstrap to third parties not included in the license

It forbids you to:

- Hold the authors and license owners liable for damages as Bootstrap is provided without warranty
- Hold the creators or copyright holders of Bootstrap liable
- Redistribute any piece of Bootstrap without proper attribution
- Use any marks owned by Twitter in any way that might state or imply that Twitter endorses your distribution
- Use any marks owned by Twitter in any way that might state or imply that you created the Twitter software in question

It does not require you to:

- Include the source of Bootstrap itself, or of any modifications you may have made to it, in any redistribution you may assemble that includes it
- Submit changes that you make to Bootstrap back to the Bootstrap project (though such feedback is encouraged)

The full Bootstrap license is located in the project repository for more information.

**Figure 6.53 Twitter Bootstrap license**

### 6.4.2 W3 Schools

W3 schools was also used in the design of the web application. It was used so that the div tags would be responsive for any device.

**Licence information available at:**
http://www.w3schools.com/about/about_copyright.asp

## 6.4 Conclusion

This chapter covered in great detail the process of development of the smart school web application, analyzing the key development implementations and iterations. The research stage was a vital part of the implementation, being crucial to the project's success. The time that was spend understanding the algorithm and also identifying that the timetabling system was an NP-Hard problem was also key to its success. Analyzing the technologies that were chosen from chapter 3 gave the system a structure to work in.

Research and review of the similar systems allowed for a lot of key features that should be implemented when coming to the system's design, when it comes to the user's experience. Nielsen's Heuristic's was key when evaluating each of the three similar systems. Evaluating each of the systems components gave the developer a great sense of direction of what to implement and what not to implement at a very early point in the research.

The development of the application was where the majority of work was involved in the project. Developing the system was not always straight forward process, but reaching the project goals was very achieving even the simplest of tasks and overcoming various errors within the application.

The outcome of the project was a product, a first release. The smart school application was very challenging at times, and the development grew past expectations, even though not all goals were met the core complexity of the project has been met. This project is only a first release and it offers room for further development and the potential of competing with real world applications.

# 7. System Testing & Evaluation

## 7.1 Introduction

This chapter contains a brief introduction into software testing and also examines different types of software testing when implementing a software system. Testing methods such as Black-Box and White-Box testing while be examined as while the different types of testing such as Integration testing, Unit testing etc.

## 7.2 Testing Categories

When developing any software system, testing is a major part in the development process of the system. There are many different types of testing when developing such as White-box test, Black-box testing, Unit testing, Integration testing etc.

### 7.2.1 White-Box Testing

White-box testing is a testing technique preformed by software developers. The developer has to test the internal structure of the systems application. This means that a White-box tester has to have proficient level of programming in order to read and understand the code. The test that is preformed can only be accurate if the tester knows what the program is supposed to do. White-box testing focuses a lot on the strengthening of security.
White-box testing involves tracing possible execution paths through the code and then working out what input values would force the execution of those paths. White-box testing is usually linked with Unit testing, which means that all of the components code should be tested before it is released for commercial use [24].

**Advantages**

- It optimizes the code within the system
- Find errors or problems within the code
- Allowing the findings of errors that are hidden

**Disadvantages**

- It can be very costly to companies in order to employ a professional tester
- High level of programming knowledge
- Requires knowledge of the internal software

There are a number different of testing that is in the category of White-box testing, I will discuss Unit testing and Integration testing.

### 7.2.2 Black-Box Testing

Black-box is a method of testing software. It tests the functionality of an application. It is called Black-box testing because in the eyes of a software tester it is like a black box, they cannot see what's inside it [25].
Black-box testing would usual test for categories such as errors within the interface, errors in accessing the databases externally, missing or incorrect functions within the system etc. Black-box testers don't need to have any knowledge of the systems code, they only see and test the interface of the system.

There is a lot of different types of testing in the Black-box testing category, I will discuss the following

## 7.4 Implementation of Testing

### 7.4.1 Black-Box Testing

Black-box testing is essential for this system, as there is a lot of user front end functionality within the web application. Feedback generated by the end test users was essential as it provided bugs, flaws and a feel for how the system should be designed. The end users testing the application have no software development experience, and each user having different levels of computer knowledge. The first page was the login to the web application, with a number of objectives that were set out in place in order to use the application and to schedule a timetable.

INPUT
SYSTEM
OUTPUT

**Figure 7.1 black box testing**

#### *7.4.1.1 Test case narrative*

**Test Case 1 - Incorrect login**

| |
|---|
| Test Case Name: Test user login is incorrect |
| Intent: The client should be prompted with an error message |
| Precondition: Enter in an email or password that is incorrect |
| Example: "adminde4@gmail.com" |
| Dialog:<br>Step 1: User visits the homepage<br>Step 2 : User enters in email address and password<br>Step 3: User clicks log in button |
| Expected Results: The user should be prompted with an error message stating that the email or password was incorrect in the homepage |
| Test Case termination: User clicking button to log in |

**Test Case 2 - Login - Not entering in all fields**

| |
|---|
| Test Case Name: Test user login when all fields are not entered |
| Intent: The user should be prompted with a message stating that this field is required |
| Precondition: User does not enter in an email or password or does not enter either |
| Example: email: (no value) Password : abc<br>Email: a123@gmail.com Password: (no value)<br>Email: (No value) Password: (No value) |
| Dialog:<br>Step 1: User visits the homepage<br>Step 2 : User does not enter in an email or password, or both<br>Step 3: User clicks log in button |
| Expected Results: The user should be prompted with a message in the form stating that the input is required |
| Test Case termination: User clicking button to log in |

**Test Case 3 - Login - Must be an email**

| |
|---|
| Test Case Name: Test if user has entered in an email |
| Intent: The user should be prompted with a message stating that an email must be entered |
| Precondition: User does not enter in an email or password or does not enter either |
| Example: testingdbuebw.com |
| Dialog:<br>Step 1: User visits the homepage<br>Step 2 : User does not enter a valid email in the email box<br>Step 3: User clicks log in button |
| Expected Results: The user should be notified that an @ should be entered in the email box |
| Test Case termination: User clicking button to log in |

**Test case 4 - Adding a Student Class**

| |
|---|
| Test Case Name: Adding student class to the database |
| Intent: The tester should be able to successfully |
| Precondition: Tester enters in all of the required inputs |
| Example: 5th year, 5A |
| Dialog:<br>Step 1: Tester enters in valid information<br>Step 2 : Tester then clicks the submit button |
| Expected Results: There should be a student class visible in the table |
| Test Case termination: Tester clicking submit button |

**Test case 5 - Adding a Student Class (Not all fields entered)**

| |
|---|
| Test Case Name: Adding student class without all fields entered |
| Intent: The tester should be given a message stating that the field that is empty is required |
| Precondition: Tester does not enter in all input fields |
| Example: input1: 5th year, input2: (empty)<br>Input1: (empty), Input2: 5A<br>Input1: (empty), Input2: (empty) |
| Dialog:<br>Step 1: Tester does not enter in all of the required fields<br>Step 2 : Tester then clicks the submit button |
| Expected Results: A message should appear stating that the input field not entered is required |
| Test Case termination: Tester clicking submit button |

**Test case 6 - Adding a teacher**

| |
|---|
| Test Case Name: Adding teacher to the database |
| Intent: The tester should be able to see that teacher being added to the system |
| Precondition: Tester enters in all of the required fields |
| Example: Input1: Mr Murphy |
| Dialog:<br>Step 1: Tester enters in all of the input fields to add a teacher<br>Step 2 : Tester then clicks the submit button |
| Expected Results: The teacher inserted should be added to the system without any problems |
| Test Case termination: Tester clicking submit button |

**Test case 7 - Adding a teacher - Not all fields entered**

| |
|---|
| Test Case Name: Adding teacher without all fields entered |
| Intent: The tester should be notified that all |
| Precondition: Tester enters in all of the required fields |
| Example: Input1: (empty) |
| Dialog:<br>Step 1: Tester does not enter in all of the input fields to add a teacher<br>Step 2 : Tester then clicks the submit button |
| Expected Results: The tester should be prompted with a message stating that they have to input all fields |
| Test Case termination: Tester clicking submit button |

**Test case 8 - Adding a subject to the system**

| |
|---|
| Test Case Name: Adding a subject to the system |
| Intent: The tester  should be able to see that teacher being added to the system |
| Precondition: Tester enters in all of the required fields |
| Example:  input: history, 6, 5A, 27, higher |
| Dialog:<br>Step 1: Tester enters in all of the input fields to add a subject<br>Step 2 : Tester then clicks the submit button |
| Expected Results: The subject inserted should be added to the system without any problems |
| Test Case termination: Tester clicking submit button |

**Test case 9 - Adding a subject to the system - Not all fields entered**

| |
|---|
| Test Case Name: Adding a subject without the all input fields entered |
| Intent: The tester  should be notified that the input fields that entered are required |
| Precondition: Tester only enters in some input fields |
| Example:  input: (empty), 6, 5A, 27, higher |
| Dialog:<br>Step 1: Tester enters in all of the input fields to add a subject<br>Step 2 : Tester then clicks the submit button |
| Expected Results: The subject inserted should be added to the system without any problems |
| Test Case termination: Tester clicking submit button |

**Test case 10 - Deleting a teacher - Should delete from subjects table**

| |
|---|
| Test Case Name: Deleting a teacher should delete the that teacher and the subjects that the teacher teaches from the subjects table |
| Intent: The tester  should see that the teacher and the subjects that belong to that teacher should be deleted |
| Precondition: Tester clicks the delete link on the teachers page |
| Dialog:<br>Step 1: Tester clicks delete teacher link in the teachers page |
| Expected Results: The teacher should be deleted from the system and also the subjects associated with that teacher should be deleted |
| Test Case termination: Tester clicking submit button |

**Test case 11 - Updating the student group in the student class table - should update student group in Subjects table**

| |
|---|
| Test Case Name: Updating student group in the student class table |
| Intent: The tester should be able to update the student class table and it should update the student group in the timetable and the subjects table |
| Precondition: Tester updates the student group in the student class table |
| Dialog:<br>Step 1: Tester enters in a new student group in the update section of the student class |
| Expected Results: The student group should be updated in the subjects table and the timetable as well as the student class table |
| Test Case termination: Tester clicking submit button |

#### *7.4.1.2 Evaluation of black box testing*

Black box testing was a crucial testing process that was used during the development of this project, as there is a lot of user interaction there would be a lot of UI bugs and limitations. Inexperienced users testing the system meant that the user (who knows nothing about the system compared to the developer) is able to see the users interaction and user control and freedom. Instant feedback about terms that were non technical allowed for the developer to take these terms into account and take the feedback into coding level.

#### *7.4.1.3 Feedback & Changes made from Black Box Testing*

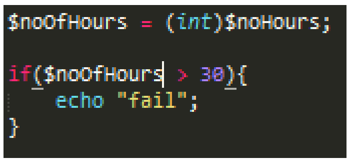
Feedback was essential from the users, even though the users tested on the systems front, many bugs were found. Feedback also about the appearance of the web application, and the users ease of navigation between pages was essential. The observation of the testing that was carried out on the system was crucial to the action that had to be taken into account to a development level, making the system robust and also allowing for a better front end interface.

The result from black box testing meant that there were changes to be made, concern with improving the user's experiences within the application and improving the robustness of the application. Improving the users experiences with using the application included changing colour schemes, different fonts were appropriate and adding more help options, not all user feedback was not implemented as some feedback was not necessary. Improving the systems robustness was essential, the users were able to break the system in certain cases such as the updating the student group in the student class page and not in the subject page, implementing MYSQL triggers made the system more robust and hence make the system less prone to bugs.

### 7.4.2 White Box Testing

The smart school system was being constantly tested using white box testing throughout its development with frequent iterative testing. As code was being developed in sections it was then tested, test cases were created to make sure the system was doing what was expected by examining results of the code developed against the test cases. Any errors that arose from the coding had to be dealt with to advance to the next stage in the development cycle. In *figure*

*7.2* this code should never be executed and if the code is executed the system should crash, as the admin should not be allowed to enter more than 30 hours per week for a subject.



```
$noOfHours = (int)$noHours;

if($noOfHours > 30){
    echo "fail";
}
```

**Figure 7.2 Example of a White Box Test**

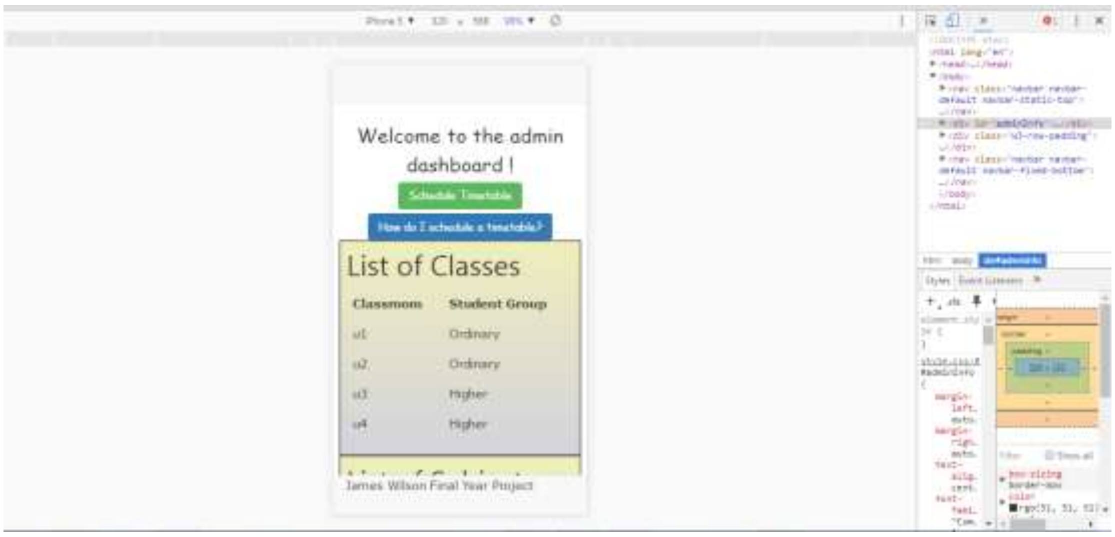
**Evaluation of white box testing:** White box testing was a major part to the ongoing development of the system. When code is written it should be test so that the system is robust and increases the quality of code written. As a result of white box testing it notified the developer that there is an error in the code, that they would have to go back and fix the problem. Again, the changes that were made from white box testing didn't mean that there was a progress in the development phase, it meant that the system would be more robust.

## 7.4.3 System Testing

System testing was a crucial part of testing the web application. In order for the application not to be limited to only one browser for example Google Chrome system testing had to be implemented, which in turn would allow the end user to use the application on their preferred search engine. Enough though the application is primarily going to be used on a PC or laptop it was also essential that the system could be used on any device (rendering for different size screens from PC to a mobile device.

| | |
|---|---|
| Test that the web application runs on Firefox as expected | Passed |
| Test the web application runs on Google Chrome as expected | Passed |
| Test that the web application runs on Safari as expected | Passed |
| Test that the web application runs on Internet Explorer as expected | Passed |
| Test that the web application runs on different size devices | Passed |

During the system testing phase, as the application was hosted on a local server it was not accessible via mobile devices. The Google Chrome device mode served its purpose. It allowed for the use of different devices which comes great use. In the figure below, it can be seen that the iphone 5 (*see figure 7.3*) has been tested. in *figure 7.4* it can be seen that the application has passed the different size devices successfully.



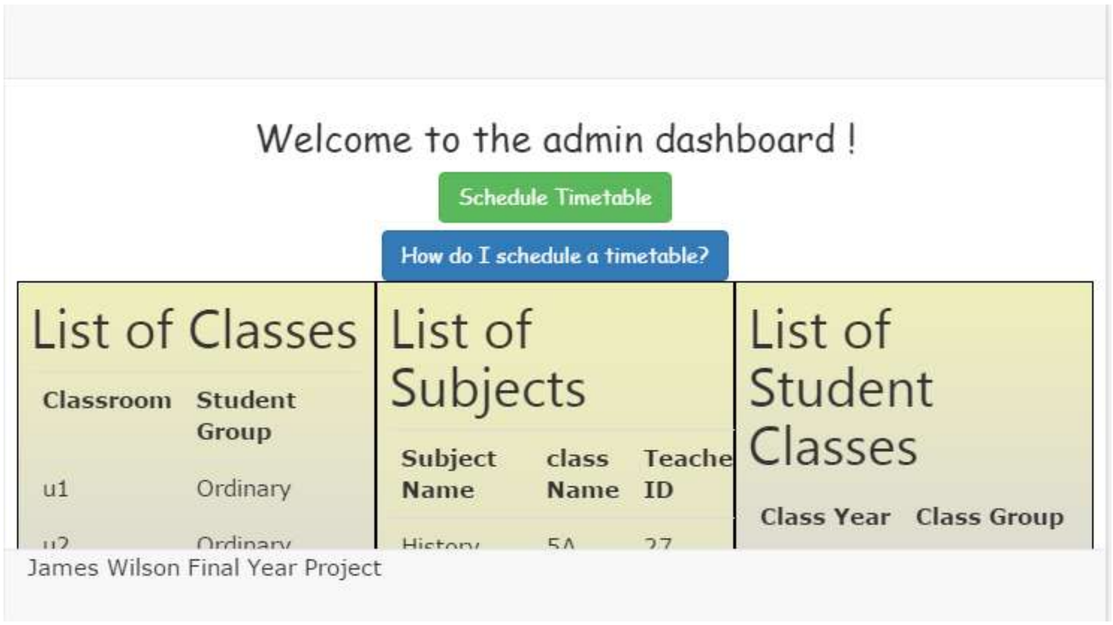
Welcome to the admin dashboard !

List of Classes

| Classroom | Student Group |
|---|---|
| u1 | Ordinary |
| u2 | Ordinary |
| u3 | Higher |
| u4 | Higher |

James Wilson Final Year Project

**Figure 7.3 System testing - Multiple Devices**



Welcome to the admin dashboard !

List of Classes

| Classroom | Student Group |
|---|---|
| u1 | Ordinary |

List of Subjects

| Subject Name | class Name | Teache ID |
|---|---|---|

List of Student Classes

| Class Year | Class Group |
|---|---|

James Wilson Final Year Project

**Figure 7.4 System Testing - iphone 6s Plus (sideward's)**

## 7.4 Conclusion

In conclusion in evaluating the different testing methods that are available the testing methods that will be predominately used will be White-Box testing and Black-Box testing. These testing method will be used to ensure that the system runs fully with no unexpected errors preventing the user to be not able to use the system efficiently. Black-Box testing is a very important method of testing in this application, it allows the system to meet the user's needs and when Nielsen's Heuristics are applied to it the system can score highly. When using the Black-Box testing method the developer is able to see first time users to complete some of the functional tasks in the system thus giving the developer a opportunity into observing how the user interact with the system. White-Box testing will also be used to check of software errors within the code, to make sure that the code does exactly what it is intended to do. White-Box testing will be used throughout the development of the system, consistently testing to make sure that the code is running sufficiently that there is for example no infinite loops or unreachable code this is known as control flow problems.

# 8 Project Plan

## 8.1 Introduction

Unfortunately not all of the project requirements that were established before the development phase in Chapter 1 (1.3) were implemented, this was due to the restrictions in time, workload from other college assignments, despite having managed the time researching and developing the project properly and efficiently. Many functional requirements have changed from the original proposal, these functional requirements are nothing major and could be completed in two to three weeks. This led to the change in the

## 8.2 What has changed & Why

Addressing the functional requirements have changed due to insufficient time in the development phase. There was a lot of changes that were made because of some of the system functional requirements that were not implemented in the system, again addressing that these requirements were not the main complexity of the system. A list of the following functional requirements were not implemented in the final system version:

**Publish Homework:** The publish homework function that the teacher should be able to do was not implemented. For instance the teacher is enrolled in a specific subject,  they should be able to publish what homework needs to be done for the students of that class.

**Publish Grades:** Another function that was not implemented was the teacher awarding grades for the students. For instance the teacher can give the student a grade based on an assignment or a recent test they have sat.

**Teacher/ Student Log in:** The system was entirely designed for the admin to create schedule a timetable so he/ she was able to schedule classes for each room in the school.

**Editing the Timetable:** Once the timetable has been scheduled, the admin should be able to edit this timetable, to their liking. Unfortunately this functionality was not added to the system, it would have made for a complete timetable management system.

**Overview of requirements that were not met:** Taking into a account the system was a complex system and being a little too ambitious with the requirements specification for the project. Considering my current skills with PHP the functionality that was not completed could be completed in a week or less as it is not complex it is just extending the functionality of the web application. None the less, editing the schedule was a functional requirement that would have been a bonus if it was met.

## 8.3 What approach would be taken if starting the project again

If I was to start the project over, I would of started the development process during the interim phase and having a set project idea in the summer, even setting up the environment. A set idea would of been crucial as the original project was changed in week 6 (late October) meaning that there was a lot of background research to catch up on, on top of researching the similar technologies and the design phase.

Starting the development earlier as stated,  would decrease the workload of development in the second semester. With the extra time at hand it would of meant that all functional and non functional requirements would of been met with a better user interface. Again, this would of taken some of the pressure off the development in the early stage while attending college, completing other assignment, with work and also looking after my son.

Stating that, the interim stage was successful, the research conducted was invaluable, especially when starting the development phase. The methodology chosen (agile) also proved to be crucial, as it is constantly adapting to the ever changing requirements.

# 9 Project Evaluation & Conclusion

## 9.1 Introduction

This chapter evaluates the work that has been completed, the goals and objectives that have been established before the project development and what goals and objectives that have been met. This chapter also deals with the future work of the project and it also reviews the personal reflection about the project and what would be done differently.

## 9.2 Key Learning

### 9.2.1 Project Management

It was very important that the project timeline that I planned out was kept to plan completing different stages. This was key when completing the interim phase and also the development phase reaching the desired standard.

### 9.2.2 Programming Skills

I have extended my knowledge greatly about Java during the development of the system. I would now be very comfortable now coding in Java. I have also learned a lot more about PHP, making my web application more robust was also crucial when testing with PHP.

### 9.2.3 Programming Language Interoperability

Using one than one programming language was very new to me, it was the first project that I have developed that two programming languages worked together, even though working interoperating two different programming languages can cause problems.

## 9.3 Future Work

The projects source files and also the code have the possibility in the future for further development expansion. The scheduling can also be tweaked if necessary to make it even more accurate, and also can be further developed to be scheduled for classrooms or for student classes.

The extra features such as publishing homework and grades can also be implemented, to further develop the web application so it can be used as a full online system, and hence making secondary schools grades and personal information online.

## 9.4 Overview

### 8.4.1 Introduction

The idea behind the project and an overview of the functionality of the web application is provided in **chapter 1**. It lists the functional/ non functional requirements, the goals and objectives of the project, the project challenges, followed the timeline describing all the key delivery dates.

### 9.4.2 Background Research

In **chapter 2**, the necessary background research was conducted going into detail about timetable management, the NP hard problem that exists, also into a lot of detail about the scheduling algorithm and how the genetic algorithm is the optimum solution for scheduling a timetable.

### 9.4.3 Research of Similar Systems

In **chapter 3**, research of similar systems was carried out, it gave an insight into how the system should be designed and implemented. Nielsen's Heuristics was key in the case of grading each of the systems, giving each a score in certain areas was crucial, it was able to outline all of the good and bad aspects of each of the systems.

### 9.4.4 Technology Review

In **chapter 4**, research was conducted on the technologies that would be suitable for this project. The technology review proved to be crucial, choosing the appropriate technologies that will be used in the project development, it allowed for an understanding of why the technologies should be used and proved to be invaluable before starting the development phase. All of the components worked well together, hence proving that the research conducted was very valuable and successful.

### 9.4.5 System Design

In **chapter 5**, consisted of the overall design of the system. Here, the system architecture (3 tier system) was explained in detail, also the use case diagram also describes the actors that are in the system and the ERD diagram explains the database and the relationship between the tables in the database. In this chapter, it also talks about what development methodology was chosen for the project, which was the agile methodology. Every time code was developed it was then soon tested, with ever changing aims and objectives along with the frequent short bursts of working software as seen throughout the implementation section.

### 9.4.6 Implementation

In **chapter 6,** the implementation of the application is discussed in detail, describing all of the main components developed.

### 9.4.7 Testing & Evaluation

**Chapter 7**, the testing and evaluation chapter gives detailed information about how the project was tested both during the implementation phase. As there were issues that needed to be overcome these issues and errors were detected and logged to the console log in eclipse. Testing a project of this size was very important and proved to be a major factor from the results of black box, white box and also system testing.

### 9.4.8 Project Plan

**Chapter 8**, comprised of the functionality of the system that has not been implemented in the final version of the system. It list some of the functional requirements of the system were not met, and this chapter also overviews what would be differently if I was to implement the project again.

## 9.5 Reflections

The smart school system was definitely one of the biggest challenges and project in my life. There were a lot of challenges that I had to overcome during the development of the project. It was very tough and seemed a daunting task taking on such a big project and being a father. As mentioned earlier not all of the specifications, but development after the project deadline is a certainty.

Reflecting back over my time in DIT I have learned a lot of valuable lessons. I definitely have serious gratitude to all the lectures over the last four years, the skills that I have gathered are invaluable. The skills I gained from DIT range from my programming skills to my presentation and communication skills. Having a child through my  final year was a difficult process, putting it mildly, but the achievement at the end of the project was very rewarding.

My time in DIT has been a memorable one, meeting a lot of amazing people. It is said that it is coming to an end of my college time, it has been a huge life changer for me and it has given me so much opportunities for the years to come. I will not miss the countless long nights of coding, completing assignments and studying for the exams.

# Bibliography (research sources)

[1] 2015. [Online]. Available: http://www.cooper.oxon.sch.uk/docs/usingtechnologytoimproveteachingandearninginsecondaryschools.pdf. [Accessed: 3- Oct- 2015]
[2] TeachHUB, 'Top 12 Ways Technology Changed Learning', 2015. [Online]. Available: http://www.teachhub.com/how-technology-changed-learning. [Accessed: 11- Oct- 2015]
[3] Online.purdue.edu, 'How Has Technology Changed Education?', 2015. [Online]. Available: http://online.purdue.edu/ldt/learning-design-technology/resources/how-has-technology-changed-education. [Accessed: 4- Oct- 2015]
[4] Melanie Mitchell (1996),An Introduction To Genetic Algorithm. Cambridge, Massachusetts: The MIT Press. p5-6
[5] Nngroup.com, '10 Heuristics for User Interface Design: Article by Jakob Nielsen', 2015. [Online]. Available: http://www.nngroup.com/articles/ten-usability-heuristics/. [Accessed: 11- Oct- 2015]
[6] Davia, Christopher. "Method and apparatus for the separation of web layout, logic, and data when used in server-side scripting languages." U.S. Patent Application 09/838,653, filed April 19, 2001
[7] Php.net, 'PHP: What can PHP do? - Manual', 2015. [Online]. Available: http://php.net/manual/en/intro-whatcando.php. [Accessed: 19- Oct- 2015]
[8] A. Lukaszewski, 'Here's What Python Is', *About.com Tech*, 2015. [Online]. Available: http://python.about.com/od/gettingstarted/ss/whatispython.htm#step2. [Accessed: 22- Oct- 2015]
[9] Afterhoursprogramming.com, 'Python Introduction Tutorial', 2015. [Online]. Available: http://www.afterhoursprogramming.com/tutorial/Python/Introduction/. [Accessed: 23- Oct- 2015]
[10] Railsapps.github.io, 'What is Ruby on Rails? · RailsApps', 2015. [Online]. Available: http://railsapps.github.io/what-is-ruby-rails.html. [Accessed: 25- Oct- 2015]
[11] SearchSOA, 'What is Ruby on Rails (RoR or Rails)? - Definition from WhatIs.com', 2015. [Online]. Available: http://searchsoa.techtarget.com/definition/Ruby-on-Rails. [Accessed: 26- Oct- 2015]
[12] SearchSOA, 'What is JavaScript? - Definition from WhatIs.com', 2015. [Online]. Available: http://searchsoa.techtarget.com/definition/JavaScript. [Accessed: 26- Oct- 2015]
[13] Devguru.com, 'VBScript >> Introduction', 2015. [Online]. Available: http://www.devguru.com/technologies/vbscript/home. [Accessed: 29- Oct- 2015]
[14] SearchEnterpriseDesktop, 'What is VBScript? - Definition from WhatIs.com', 2015. [Online]. Available: http://searchenterprisedesktop.techtarget.com/definition/VBScript. [Accessed: 29- Oct- 2015]
[15] Cs.ait.ac.th, '1 General Information About MySQL', 2015. [Online]. Available: http://www.cs.ait.ac.th/laboratory/database/manual/manual_Introduction.shtml. [Accessed: 29- Oct- 2015]
[16] Slideshare.net, 'Firebase - A real-time server', 2015. [Online]. Available: http://www.slideshare.net/AneeqAnwar/firebase-new. [Accessed: 04- Nov- 2015]
[17] Firebase.com, 'Features - Firebase', 2015. [Online]. Available: https://www.firebase.com/features.html. [Accessed: 05- Nov- 2015]
[18] A. Glover, 'Flexing NoSQL: MongoDB in review', InfoWorld, 2015. [Online]. Available: http://www.infoworld.com/article/2618931/database/flexing-nosql--mongodb-in-review.html. [Accessed: 15- Nov- 2015]
[19] David Budgen (1994). Software Design. Great Britain: Addison-Wesley Publishing Company Inc. p44-47.
[20] Base36.com,. 'Agile & Waterfall Methodologies – A Side-By-Side Comparison | Base36'. N.p., 2013. Web. 10 Nov. 2015.]
[21] Agilenutshell.com, 'What is Agile?', 2015. [Online]. Available: http://www.agilenutshell.com. [Accessed: 12- Nov- 2015]

[22] P. Sparrow, 'Spiral Model : Advantages and Disadvantages ~ I Answer 4 U', *Ianswer4u.com*, 2015. [Online]. Available: http://www.ianswer4u.com/2011/12/spiral-model-advantages-and.html#axzz3snWvyKNU. [Accessed: 12- Nov- 2015]
[23] Stevens Pooley (1999). Using UML. Dorset, Great Britain: Pearson Education Limited . p96-103.
[24] SearchSoftwareQuality, 'What is white box (white box testing)? - Definition from WhatIs.com', 2015. [Online]. Available: http://searchsoftwarequality.techtarget.com/definition/white-box. [Accessed: 13- Nov- 2015]
[25] Softwaretestingfundamentals.com, 'Black Box Testing | Software Testing Fundamentals', 2010. [Online]. Available: http://softwaretestingfundamentals.com/black-box-testing/. [Accessed: 22- Nov- 2015]
[26] Webopedia.com, 'What is Java? A Webopedia Definition', 2015. [Online]. Available: http://www.webopedia.com/TERM/J/Java.html. [Accessed: 15- Nov- 2015]
[27] Techopedia.com, 'What is C++ Programming Language? - Definition from Techopedia', 2015. [Online]. Available: https://www.techopedia.com/definition/26184/c-programming-language. [Accessed: 15- Nov- 2015]
[28] SearchSQLServer, 'What is C++? - Definition from WhatIs.com', 2015. [Online]. Available: http://searchsqlserver.techtarget.com/definition/C. [Accessed: 15- Nov- 2015]
[29] Docs.oracle.com. (2016). DateFormatSymbols (Java Platform SE 7 ). [online] Available at: https://docs.oracle.com/javase/7/docs/api/java/text/DateFormatSymbols.html [Accessed 31 Mar. 2016].