FACULTY OF SCIENCE, ENGINEERING AND COMPUTING

School of *Computing Science and Mathematics*

BSc DEGREE
IN

Computer Science

PROJECT DISSERTATION

Name: Dipika Lad

ID Number: K1309019

Project Title: Online Rota Application for Everyday Sports

Project Type: Build

Date: 15/04/2016

Supervisor: Graeme Jones

Kingston University London

Word count: 10,056

Did you submit a draft of your dissertation to your supervisor?	Most of the report
Did you receive feedback from your supervisor on any submitted draft?	Yes

Plagiarism Declaration

The following declaration should be signed and dated and inserted directly after the title page of your report:

Declaration

I have read and understood the University regulations on plagiarism and I understand the meaning of the word *plagiarism*. I declare that this report is entirely my own work. Any other sources are duly acknowledged and referenced according to the requirements of the School of Computer Science and Mathematics. All verbatim citations are indicated by double quotation marks ("…"). Neither in part nor in its entirety I have made use of another student's work and pretended that it is my own. I have not asked anybody to contribute to this project in the form of code, text or drawings. I did not allow and will not allow anyone to copy my work with the intention of presenting it as his or her own work.

Date: 15/04/2016

Signature: dipikalad

Table of Contents

Chapter 1 Introduction

This report will explain how and why the online rota website was created, this will be done by using requirement analysis, design methods, implementation tools and techniques as well as test strategies. The report has also included a documentation of how database was used in the website. The website is using HTML, CSS, Bootstrap, Formoid, PHP and MySQL. During the duration of this project certain changes have been made from what was initially stated in the proposal. The reason for this is that new ideas arose while working on the project that changed the outlook on the end result of the project.

The language that interacts with the database is PHP, specifically PDO which is mainly used in Object Oriented PHP. This scripting language is well used in web development as most website need to access a database of some kind. PHP connects and communicate with the MySQL database, which is used with HTML webpages that read, write, update and delete information that is stored in the MySQL database.

The website that has been created currently only has two out of the three sides to it, these sides are store employees and store managers which are the main users in this project. The first main usability side of the website is for the store employees can look at the hours they have worked, the wages they have earned and to see when they are working. The second main usability side of the website is for the store managers is the reports, rota for the store and checking the store employees' availability. The third side of the website is for the system administers who creates the rotas for each store and runs reports on each area. The website is the front end interface that is used by users which is simple and easy to use.

During the course of the project the database changed as certain attributes need to be added to certain tables. This is due to the fact that extra data is needed for certain web pages such as the hours and wages pages. If the database isn't created correctly later on in time when the project is handed over the client and the database crashes or doesn't work properly, this can cause more problems, as the errors will be difficult to find due to the implementation of an inadequate structure.

In this report an overview of the project will be included with detailed explanations into the requirement analysis, design process, implementation and testing. In the report there will be a section that will include research that has been carried out on data sources to gain better understanding of how projects get the data that is needed. During the creation of the website some ideas arose about how the website could be improve to make it more effect but by the time they were discovered it was too late in the project. However they will be discuss in the future development section of the critical review in the report.

The project is using the Kingston University Student Webserver to host the website on. The reason for this is that the project doesn't have a real-time client but it allows me to use and test the website on many end user devices which will represent a real user using the website.

SWOT Analysis

SWOT analysis is a commonly used tool in requirement analysis to determine whether a proposed project will be effective and feasible. It does this by first enumerating the potential strengths of the project compared to the previous system, its weakness, the new opportunities the way the system is implemented provides, and any threats that are introduced by the new system. This evidence is then weighed to determine whether or not the project should proceed and additionally to determine how the potential threats may be mitigated.

A current rota system is normally based on a spreadsheet stored in one or more locations showing employees the times of their shifts. The proposed online system allows user access to a centralised database from different end devices such as tables, smartphones etc.

Strengths	Weaknesses
• Will be able to access system from anywhere • Will be able to access system from any device • Employees will be able to view their shifts. • Managers won't need to print paper copies of the rota or send photographs to employees. • Managers will have less paperwork as the system is web based. • Training managers on how to use the system will be easy as the system is user friendly.	• The system won't be able open correctly on a device • The system may not run correctly when the internet connection is inefficient. • Saver issues can be troublesome when errors are unknown.
Opportunities	Threats
• Staff will be able to work at different stores. • It will allow store managers to change rotas a lot more easier and not worry about notifying employees as the system will notify the employees automatically of the changes	• Some employees may be scheduled in two different stores at the same time • The storage of data must comply with the Data Protection Act. • There is a possibility that data can be corrupted when it is entered into the database or read from the database.

From the above SWOT analysis, the strengths and opportunities of this project clearly outweigh the weakness and threats. Therefore the project is viable and should proceed. Another benefit of this system is that it will save the company time and money in administration work. Nonetheless there are threats with the new system. To avoid having an employee scheduled in two places at once there will need to be rules in the database that only allows an employee to be used once per date. The website will show information that is relevant to each user type which will compile with the Data Protection Act as only data which is needed will be used. There is a slight chance that the website could be hack by an unauthorised individual, however that is unlikely.

Chapter 2 Literature Review

2.1 Introduction

The topic of review is how data sources are protected by law and what types of data are actually protected. Section 2.2 will look into data protection law. Section 2.3 will discuss data generators as a source of data and are they protected. Section 2.4 will discuss data and the different data types.

2.2 Law

The law is there to protect all types of data and most data sources. Different countries have different laws on how data is protected and how the data sources should protect data. In this section the Data Protection Act 1998 will be discuss on what is it and what its key principles are. Also European data laws will be looked at to find out how they protect data.

2.2.1 Data Protection Act 1998

What is the Data Protection Act 1998? The Data Protection 1998 is a piece of legislation that protects personal data from being misused or abused by government, organisations and business [1 and 2]. From source [2] there are 6 major Parts and 16 Schedules that make up the DPA 1998. The Parts of the DPA 1998 outline methods that data is handled by those who process it, basic rights of data subjects, special exemptions and different modes of enforcement [2]. The Schedules explain in greater detail the Parts of the DPA 1998, legal clarifications and elaborate on varied contingencies [2]. The next paragraph will look at what are the key principles of DPA 1998?

From sources [2] key principles are that data must: be handled fairly and lawfully, be acquired only for lawful purposes and not handled in any way unsuited with those purposes, be suitable, relevant and not excessive, be correct and up-to-date, not be held for no longer than necessary, be handled in agreement with the rights and freedoms of data subjects, be protected against unauthorized or unlawful handling and against accidental loss, destruction or damage, not be transferred to a country or territory outside the European Economic Area unless that country or territory protects the rights and freedoms of the data subjects.

2.2.2 European Law

The current EU law that protects data is the EU Data Protection Directive (Directive 95/46/EC). The EU Directive 95/46/EC was "designed to protect the privacy and protection of all personal data" as stated in source [4]. The EU Directive 95/46/EC is founded upon seven key principles. These principles are notice, purpose, consent, security, disclosure, access and accountability [4]. From source [4] the rules of data protection not only apply to the controller which is established or operates within the EU but when the personal data is processed by the controller using equipment located in the EU. This means that personal data processed in the EU by a controller from outside the EU must still comply with the EU Directive 95/46/EC. From looking at source [3] it states that "in January 2015 the European Commission revealed a drafted of its European Data Protection regulation to replace the pervious Data Protection Directive". The reason the European Commission is replacing the Data Protection Directive is so that the regulation is directly applied to all the member states of the EU without implementing a national legislation [3]. This means that the change to current EU legislation will mean greater protection will be applied to personal data.

2.3 Data Generators

From looking at source [6] it states that test data generation is the process of making a set of data that can be used for the testing of new or revised software applications. Another way to describe data generators is that it is a tool which creates random and large quantities of data for testing. Also pure randomness in data is something you don't want either sometimes [7]. The data created by the generators should look real as much as possible [7]. The data used in the generation may be artificial data made for this purpose or actual data from previous operations [6].

Test data generator comes in different types. These types are random test data generator, goal-oriented generator, pathwise test data generator an intelligent test data generator. Random test data generator is the simplest kind of generator; it can be used to test different programs by the randomly generated bit stream which is then represented as the required data type [9]. Goal- oriented generator generates inputs for any specific path instead of generating the input from the code's entry and exit, with this type of generator there is very little change of infeasible paths being generated and can gather an input for any given path [9]. Pathwise test data generator is similar to goal-oriented generator as it also deals with specific paths. The way this generator works it is assigned a specific path to track instead of having a choice surrounded by many pathways, as a result it gains greater knowledge of the path and can predict coverage [9]. Intelligent test data generator is only able to work when the analysis of the code selected for testing in which it is needed to guide the search aimed at the test data [9].

When using data generators is the data needed for a database be designed in a specific manner? From source [8] on page 403 it states that the tools are able to read the database entities directly so that it can determine the specific fields and their type, length and format. It also states that rules, relationships and constrains can be added by user to the data generated. There are ways of using data generating and seeding create test data that can be put into an application the user has chosen [8].

2.4 Data

What is Data? "Data is distinct pieces of information, usually formatted in a special way. All software is divided into two general categories: *data* and *programs*. Programs are collections of instructions for manipulating data. Data can exist in a variety of forms -- as numbers or text on pieces of paper, as bits and bytes stored in electronic memory, or as facts stored in a person's mind. Strictly speaking, data is the plural of *datum*, a single piece of information. In practice, however, people use *data* as both the singular and plural form of the word." as quoted from source [10, Webopedia.com, (2015)].

How is data protected? From doing research not all data is protected this is due to the fact that only sensitive data as such personal data is protected this shown in the results of a Google search in source [11]. Looking at section 2.2 above it is obvious to see that the law protects sensitive personal data. Data comes in different types such personal, real, factual etc. The next paragraph is going to look into some of these data types. All the different types of data will fall into one of three main categories; they are qualitative, discrete quantitative and continuous quantitative data [12].

2.5 Conclusion

The literature review conducted has established that data generators are a good source for fictitious. This will impact the project as it will reduce the amount of time needed to create dummy data from scratch and there are generators for different purposes. The data generated will still have to compile with the Data Protection Act and the EU Data Protection Directive as the data being used in this project is replicating a real life project for a client. An impact of this research is that it will produce data insight on how the data generated at be used in different applications and how wide the uses of data generators can be expanded.

Chapter 3 Requirement Analysis

3.1 Stakeholder Analysis

A stakeholder analysis as stated by Wikipedia.org [13] is a process of identifying the stakeholder which are likely to affect and will be affected by a proposed action. Then the stakeholder will need to be sorted according to their impact on the action and the impact the action will have on them. A stakeholder analysis of a subject involves weighing and balancing all the demands on a firm by each of the stakeholders. A Stakeholder analysis cannot prevent the interest of the stakeholders overruling the interest of the other stakeholder affected but considers all the affected stakeholders [13].

From conducting interviews with store employees and store managers the stakeholders could be identified. The stakeholders of this project are the store managers, store employees, system administrators, human resources department, area managers, regional managers and customers. They have been placed into a power/interest gird to determine how they are affected by the new system.

Power

High	Regional managers System administrators	Store managers Area managers
Low	Human resources department Customers	Store employees
	Low	High

Interest

From the stakeholder analysis the users are determined and the benefits of the new system for certain stakeholders. The users of the system will be the store managers, system administrators and the store employees. The benefit to regional and area managers is the ability to view which employees are working in the stores in their district. The benefits for store employees are the ability to receive notifications of their shifts and also view their work schedule online.

3.2 Use Cases

The Use Case methodology as described by SearchSoftwareQuality.com [14] is used to identify, clarify and organize the system requirements during the system analysis stage. A use case is a set of potential sequences of interactions between a system and its users in a specific environment, which are related to a specific goal, is what makes the use case. A use case consists of a group of elements that could be used together in a technique that has a greater effect than the sum of all the separate elements joined together. All the system activities that have importance to the user should be contain in the use case.

3.2.1 Persona

Persona – Store Employee

Akhil is 23 years old who has recently graduated in Pharmacy from University of Hertfordshire. He is currently working at Everyday Sports Ltd while looking for a job in Pharmacy. When the rota for the store he works at is published he has to go into the store on his day off to find out when he is working for the following week. Having to go to store every time to look at the rota on his day off is highly time consuming and annoying.

Persona – Store Manager

Ang is a 33 years old and is the current store manager of the Wembley store of Everyday Sports Ltd. Currently Ang has to create the stores rota on a system that isn't user friendly and has to manually enter all the shift for each store employee. Creating the rota takes a lot of time to do as she has to constantly look at the staff availability sheet while she is doing the rota so that each store employee is working on the correct days and double check that it is all correct before printing it and pinning it to the notice broad in the staff room.

3.2.2 User Stories

User Story 1 – Store Employee

Logging into the system must be quick and easy. The user wants to access the current rota to see the shifts that have been assigned to them for that week. Also they want to be able to get a SMS reminding them of when their shift is.

User Story 2 – Store Manager

The user want to be able to look at the current rota so they know which employees are working in their store. They also want to be able to edit the rota by adding more shifts in or being able to change or remove an employee's shift.

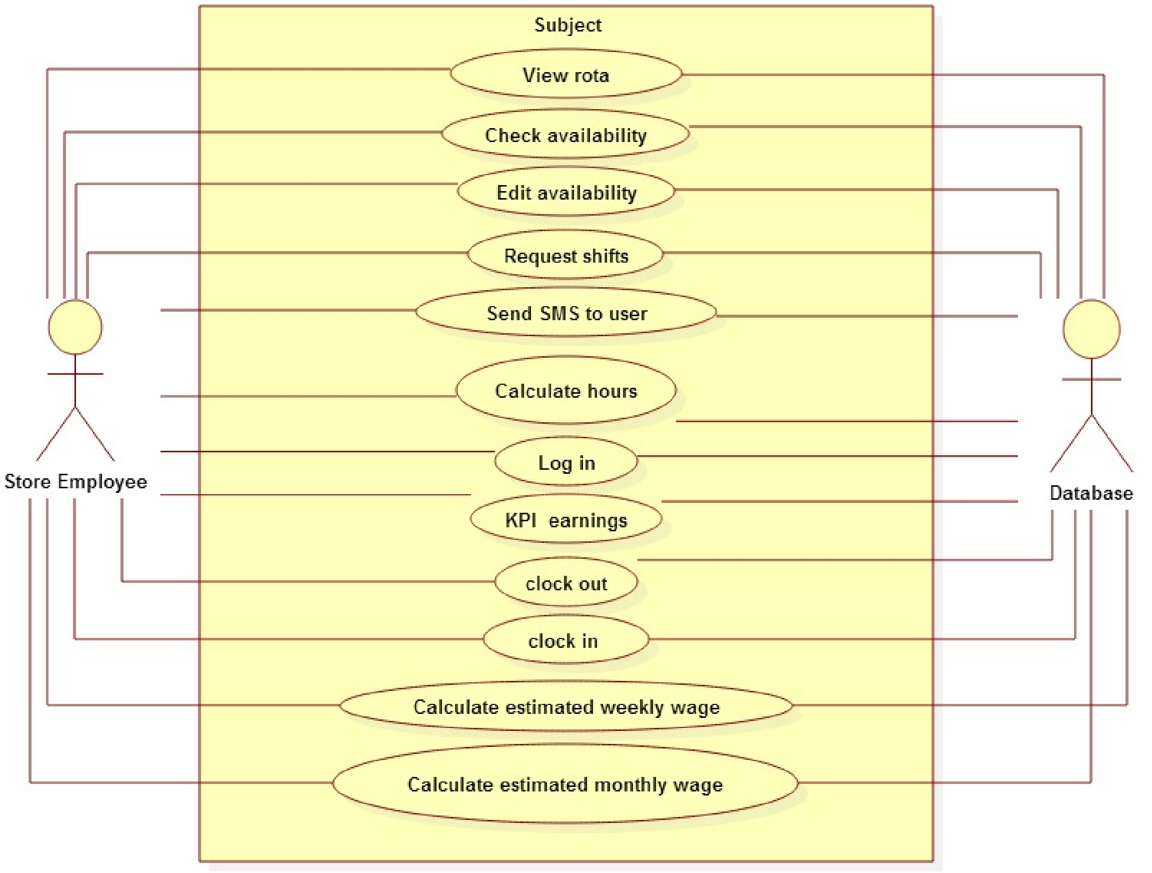
User Story 3 – System Administrator

The user wants to be able to create weekly rotas for the each of the stores for each of the companies the client has. They also want to create and regularly update a list of key dates each stores must have in their stores.

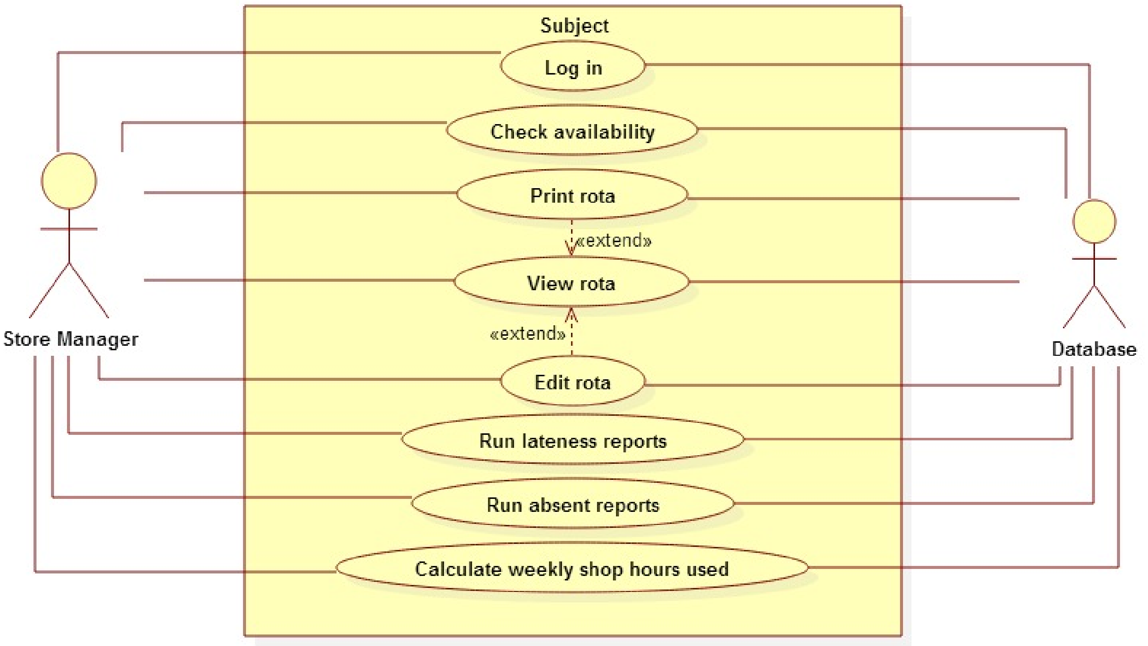
3.2.3 Use Case Diagrams

Use Case Diagrams as stated by WhatIs.com [15] is a graphical depiction of the interactions between the elements of a system. Use case diagrams are used in UML (Unified Modelling Language) is a standard notation that is used for modelling real world objects and systems. Use case diagrams have four key components that create the diagram. These components are the boundary, the actors, the use cases and the relationships. The boundary is the system that is defined in relation to the world that surrounds it. The actors are users that are involved with the system which are defined by their roles. The use cases are specific roles the actors play within and around the system. The relationships are created between and among the actors and use cases [15].

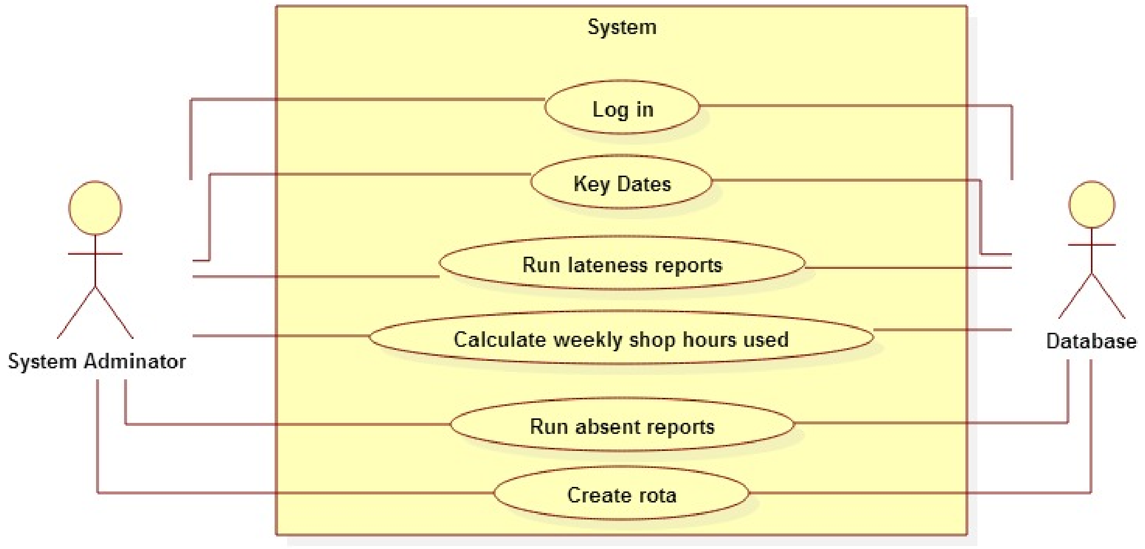
Store Employee Use Case Diagram



Store Manager Use Case Diagram



System Administrators Use Case Diagram



3.2.4 Use Case Texts

3.2.4.1 Use Case Text - UC1: Request Shift

Primary Actor: Store Employee
Stakeholders and Interest:

- Store employee: Wants to have an extra shift.
- Store manager: Reviews the request and makes a decision.

Preconditions:

- ♦ A store employee logs into the rota system.
- ♦ They hover over rota section of the menu.
- ♦ A dropdown list appears and they click on Request Shift.

Success Guarantee (Post conditions):

- ♦ Then store employee is returned to rota page.
- ♦ The request is put on the system.
- ♦ The store manager will review the request.

Main Success Scenario:

1. They fill out the request form.
2. They click submit.
3. They are shown a preview of the request to double check everything is correct.
4. Then the click continue.

Extensions:
The store employee's logon details might not get authenticated correctly due to incorrect input or a system failure.

Open Issues:
System may crash and or fail.

3.2.4.2 Use Case Text - UC2: Edit Rota

Primary Actor: Store Manger
Stakeholders and Interest:

- Store manager: Want to edit the rota.

Preconditions:

- ♦ A store manager logs into the rota system.
- ♦ They hover over rota section of the menu.
- ♦ A dropdown list appears and they click on Request Shift.

Success Guarantee (Post conditions):

- ♦ Then store manager is returned to rota page.
- ♦ The changes is put into the system.

Main Success Scenario:

1. They make changes to the rota.
2. They click submit.
3. They are shown a preview of the rota to double check everything is correct.
4. Then the click continue.

Extensions:
The store logon details might not get authenticated correctly due to incorrect input or a system failure.

Open Issues:
System may crash and or fail.

3.3 List of Requirements

In this project both functional and non-functional requirements will be included for all users as the functionality depending on which user is logged into the web application. Some of the requirements will be shared between the users as they need the same functionality in the web application.

3.3.1 Functional Requirements

Functional requirements are the user focus goals that a system or application will need to fulfil. The list of functional requirements has been derived from the use case diagrams above.

1. Log in
2. View rota
3. Print rota
4. Edit rota
5. Create rota
6. Run absent report
7. Run lateness report
8. Calculate hours
9. Calculate weekly shop hours used
10. Calculate estimated weekly wage
11. Calculate estimated monthly wage
12. Holiday request
13. Key dates
14. Edit availability
15. Check availability
16. Request shifts
17. Clock in
18. Clock out
19. KPI earnings
20. Log out
21. KPI’s
22. 5 Star products
23. Company store league table

3.3.2 Non-Functional Requirements

- User should be able to log into the system within 20 seconds.
- The system must be reliable.
- The system must be available 24/7.

3.4 MoSCoW

MoSCoW as stated on Project Smart [16] that stands for must, should, could and would which is used to prioritise a list of functional requirements. Must have are the requirements that is needed to meet the needs of the system. Should have are requirements that are possible but do not contribute to the success of the project. Could have are requirements that does not affect anything else in the project. Would like are requirements that can be implemented at a different point in time [16].

Must have	Should have
1. Log in 2. View rota 6. Run lateness report 7. Run absent report 10. Calculate estimated weekly wage 12. Holiday request 13. Key dates 14. Edit availability	3 .Print rota 4. Edit rota 8. Calculate hours 15. Check availability 16. Request shifts
Could Have	**Would like**
5. Create rota 9. Calculate weekly shop hours used 11. Calculate estimated monthly wage 19. KPI earnings 20.Log out	17. Clock in 18. Clock out 21. KPI's 23. Company store league table 22. 5 Star products

Chapter 4: Design

The design process of the project involves both the front-end designs of the web application and the back end of the web application. The user interface designs will be discussed first followed by the discussion of the database design.

4.1 User Interface design

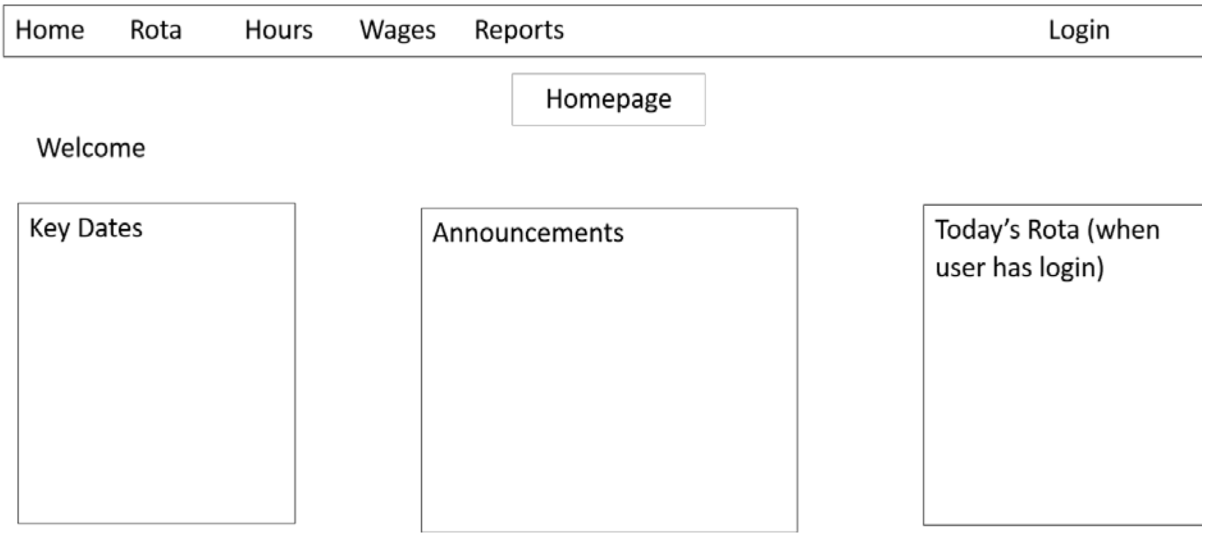
Designing the user interface is important because as it determines whether the users of this project will use the web application once the project is finished. This is due to the fact that user will only use the application if it is easy to use and doesn't require a lot of technical skill to use it. To insure the user ability of the user interface of the application was well designed the users of the project were consulted to find out their ideas on how the web application should look. Based on their ideas the designs that were created represent how each webpage will look at the end of the project.

4.1.1 Wireframes

Wireframes are used in the design process to suggest how the content will be arranged in the finished product. Wireframes do not have colour, style and graphics as it focuses on priority of content, functionality and the behaviour of the product.

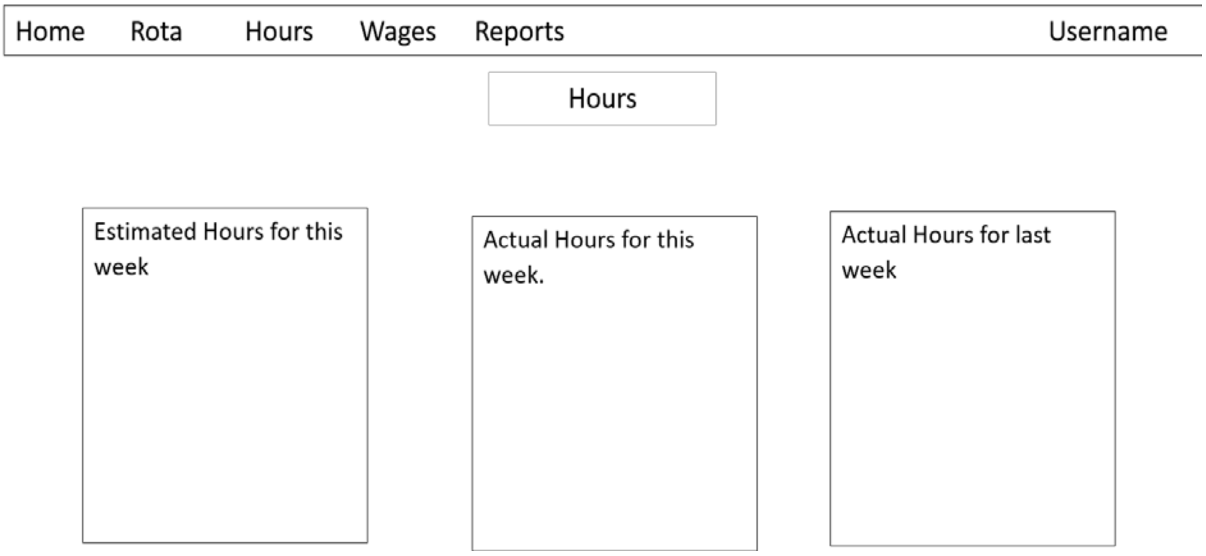
Homepage

The homepage will display a simple user interface where the user can navigate to different pages by using the navigation bar. The homepage contents information about key dates that every employee will need to know such as cut off paid date, pay day, stocktake etc. Also show the rota for that day when a user is logged into the system. The login will change to the user's name when they have logged into the system.



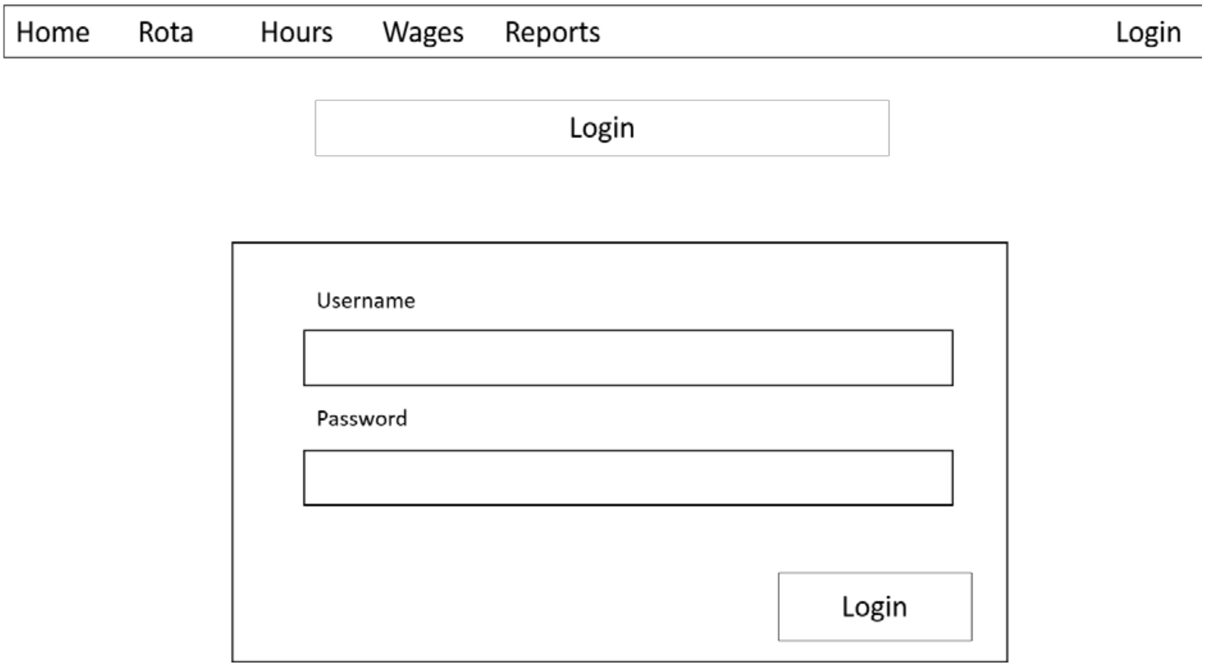
Hours page

The hours page show the user the estimated hours they will work for that week based on the hours the number of shifts they have been put for. The actual hours for that week and the previous week will be based on the shifts that have worked which will be calculated by the number of hours they have worked for each shift.


Home
Rota
Hours
Wages
Reports
Username
Hours
Estimated Hours for this week
Actual Hours for this week.
Actual Hours for last week

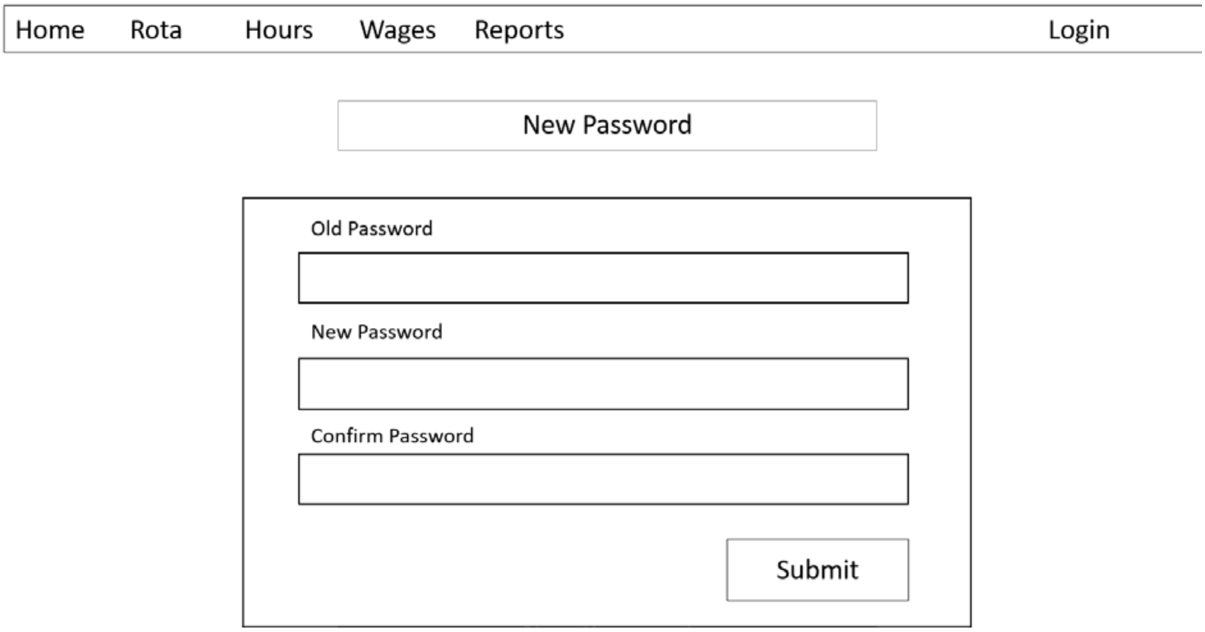
Login page

The login page is a form with fields that need to be populated by the user with their username and password. Once a user has logged in they are able to access the all the information on the website.



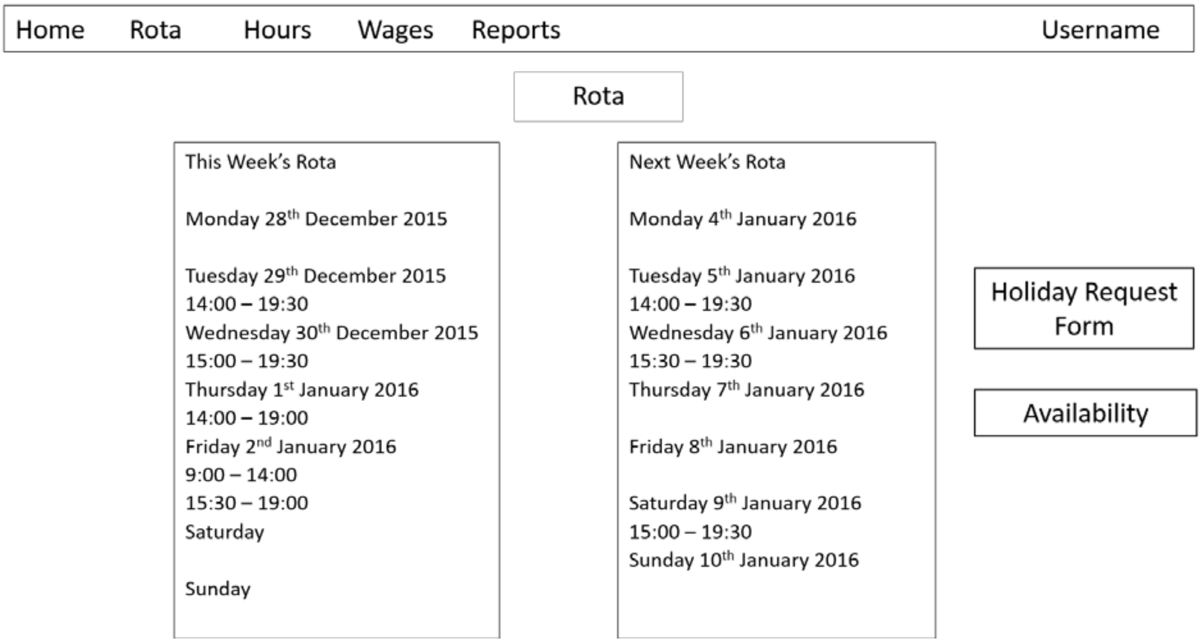
New password page

The new password page is only required when the user logs in for the first time and must change their password. Also they must confirm their new password as a security check to make sure the two passwords match before it is changed.



Rota page for store employee

The rota page for store employee shows what day the employee is working for that week and what days they wil be working the week after. The holiday request form button on the right hand side go to form where the employee can request time off work. The availability button allows the employee to look at their availability and edit their availability.



This Week's Rota

Monday 28th December 2015

Tuesday 29th December 2015
14:00 – 19:30
Wednesday 30th December 2015
15:00 – 19:30
Thursday 1st January 2016
14:00 – 19:00
Friday 2nd January 2016
9:00 – 14:00
15:30 – 19:00
Saturday

Sunday

Next Week's Rota

Monday 4th January 2016

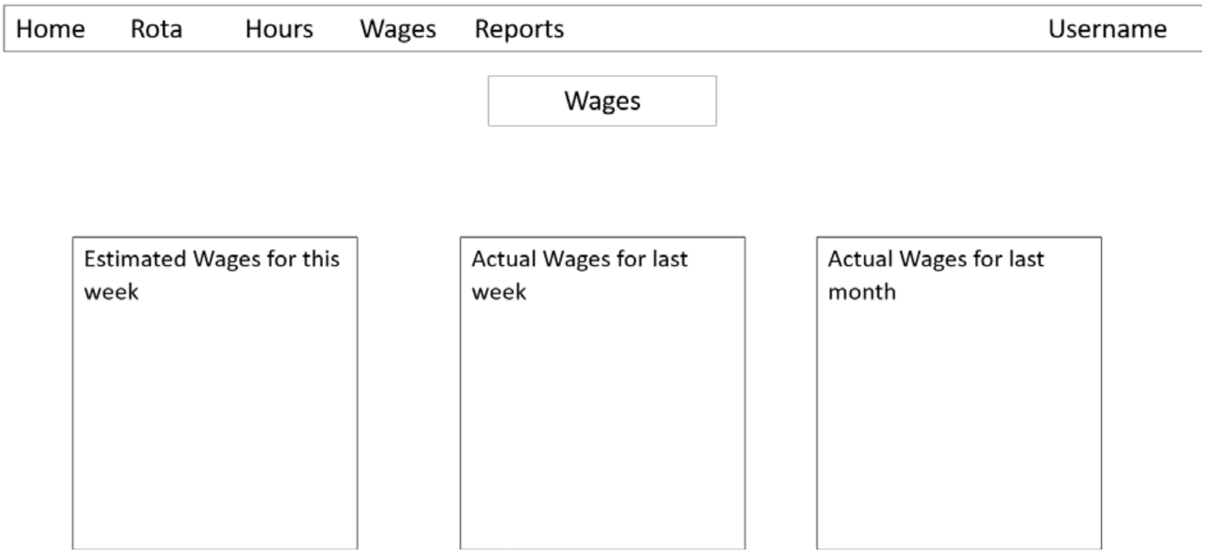
Tuesday 5th January 2016
14:00 – 19:30
Wednesday 6th January 2016
15:30 – 19:30
Thursday 7th January 2016

Friday 8th January 2016

Saturday 9th January 2016
15:00 – 19:30
Sunday 10th January 2016

Wages page

The wages page shows the user the estimated wages that they will earn by working the shifts they have been scheduled for that week. The actual wages for that week and the previous week will be based on the hours that they have worked that week which will be multiplied by their salary.



Home Rota Hours Wages Reports Username

Wages

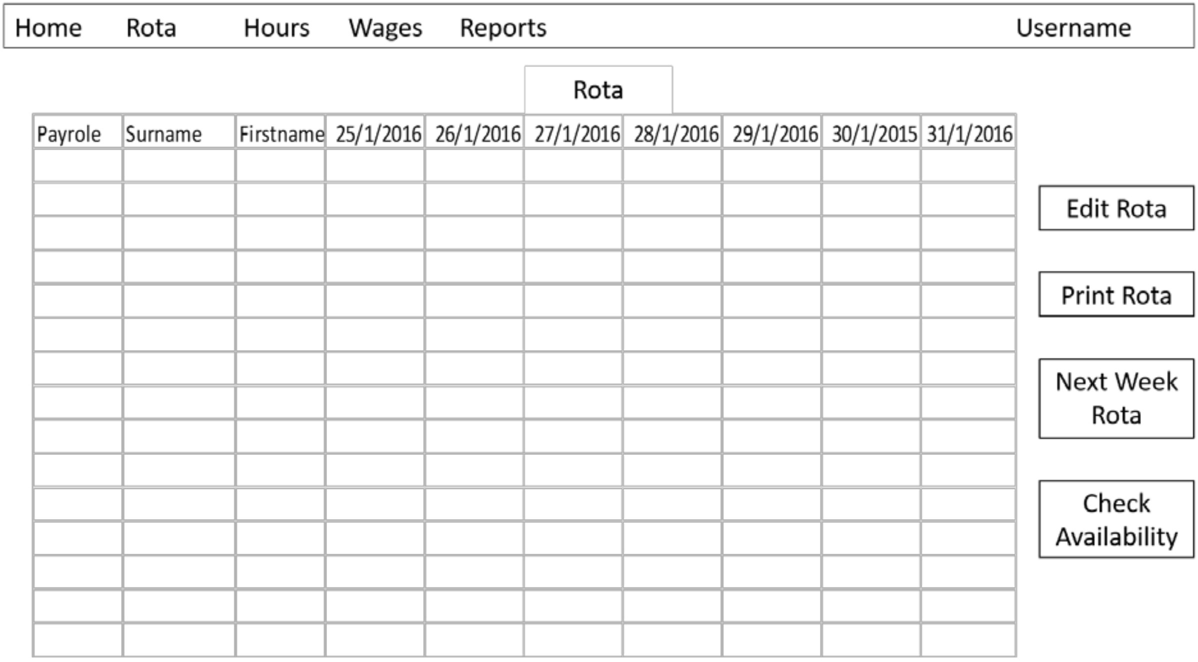
Estimated Wages for this week

Actual Wages for last week

Actual Wages for last month

Rota page for Store Manager

The rota page for the store manager shows the rota for all the employees and managers working in the store for that week. The next week button brings up the store rota for the following week.



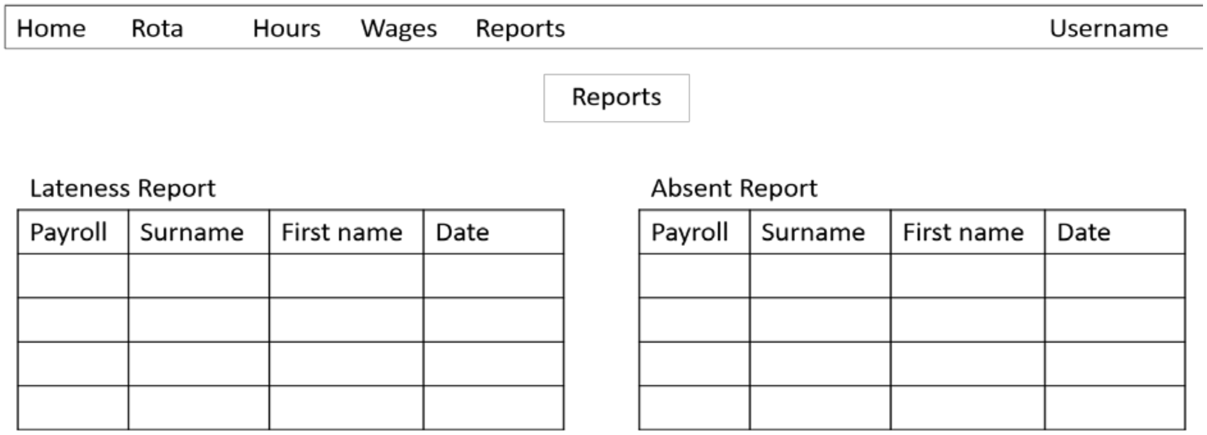
Home Rota Hours Wages Reports Username

Rota

Payrole	Surname	Firstname	25/1/2016	26/1/2016	27/1/2016	28/1/2016	29/1/2016	30/1/2015	31/1/2016

Reports Page

The reports page shows the lateness and absent reports that a manager would need to see on a weekly basis to monitor staff punctuality and reliability.



Lateness Report

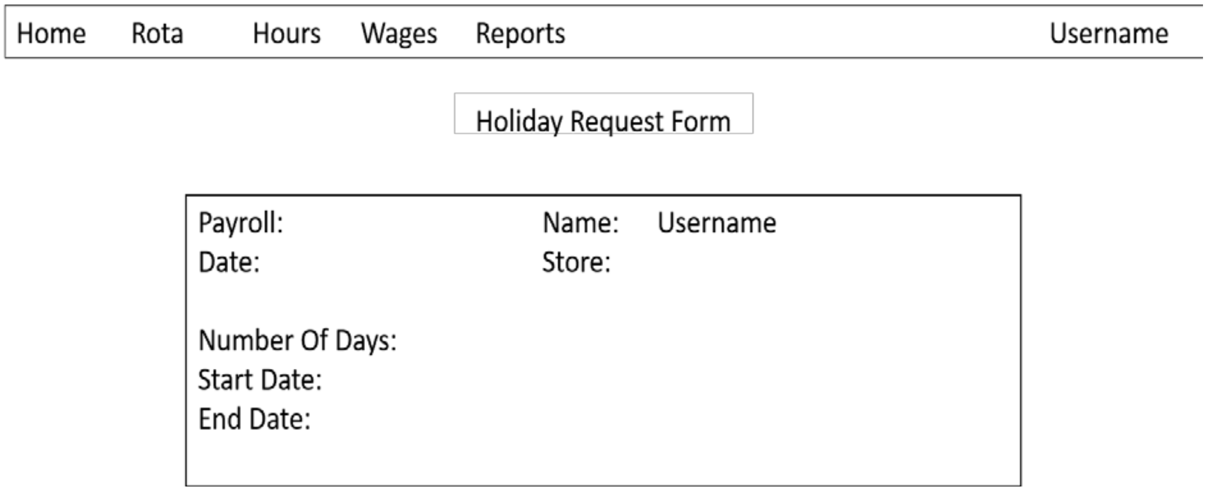
Payroll	Surname	First name	Date

Absent Report

Payroll	Surname	First name	Date

Holiday Request Form

The holiday request form page will allow store employees to submit requests for time off work.

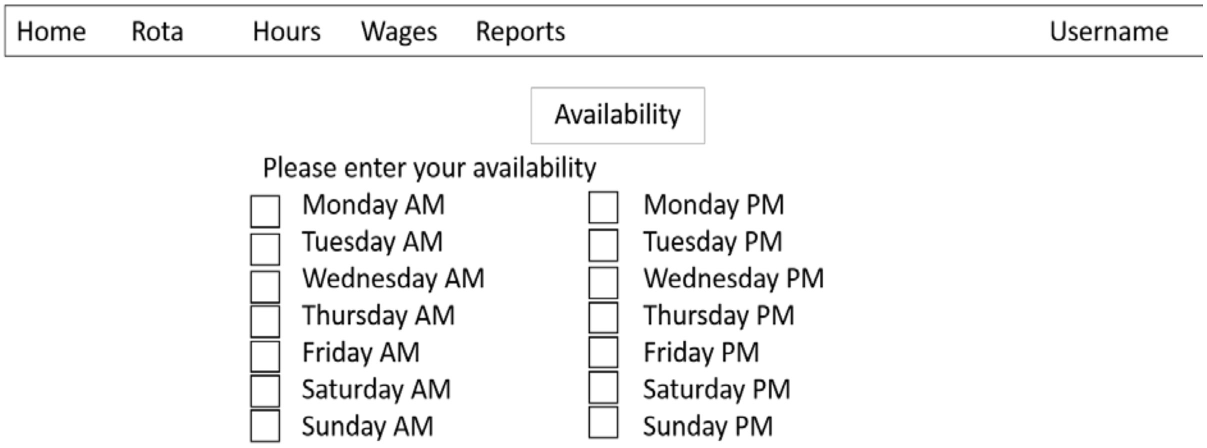


Payroll: Name: Username
Date: Store:

Number Of Days:
Start Date:
End Date:

Availability Form Page

The availability form page show the employee’s current availability and they can edit it by deselecting and selecting the day and time they want to work.



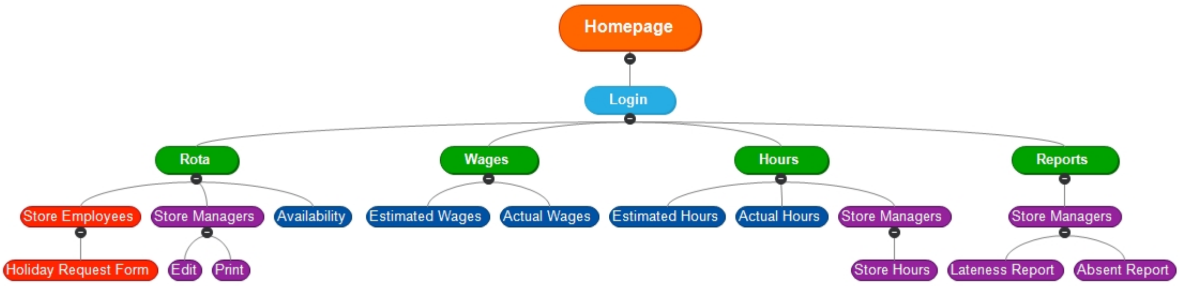
Please enter your availability

□ Monday AM
□ Tuesday AM
□ Wednesday AM
□ Thursday AM
□ Friday AM
□ Saturday AM
□ Sunday AM

□ Monday PM
□ Tuesday PM
□ Wednesday PM
□ Thursday PM
□ Friday PM
□ Saturday PM
□ Sunday PM

4.1.2 Navigation

The navigation of web pages is important as it shows how the pages interact with each other. Also it shows which users have access to certain pages and information. In the diagram there are different colours to represent the users. The red are the webpages used by the store employees. The purple are the webpages that are used by the store managers and the blue are the webpages used by both users. A user can move between the branches, as this will be implemented into the navigation bar in chapter 5.



4.2 Database design

Database design is about all the different components that come together to create the overall design of the database that will be implemented later on the project. The techniques that will be used to for the design of the database are class diagram, entity relationship diagram and data dictionary. The reason for using these techniques is due to the fact that they start as a simple design in terms of a conceptual diagram i.e. class diagram to the data dictionary which hold all the detailed information about each table in the database.

4.2.1 Class Diagram

The class diagram shown overleaf is a simple model of the database with all its established attributes and relationships between the classes. The way the relationships are shown in the class diagram is a one-to-many relationship. This means that one class can be associated in many occurrences with another class for example one store employee can be have more than one shift in a day or week.

4.2.2 Entity Relationship Diagram

The entity relationship diagram shown on page 20 is a more detailed then the class diagram as it show the data types and length of each attribute in the database. It also shows which attributes are primary and foreign keys as shown below.

4.2.3 Data Dictionary

The data dictionary contains all the information need to create the table in MySQL using the Create Table statement. The table statements state each attribute name, data type, size, constraints such as check, unique and not null and whether an attribute is a primary key or foreign key or both in some instances.

Area					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
A_id	Integer	5	PK	NOT NULL	
R_id	Integer	5	FK	NOT NULL	
A_name	Varchar	50		NOT NULL	Unique
A_manager	Varchar	100		NOT NULL	

The area table holds information about all the areas the client has stores and which region each area belongs to.

Availability					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
Av_id	Integer	3	PK	NOT NULL	
Description	Varchar	50		NOT NULL	

The availability table holds information about all the days and time of the availability types.

Clock					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
Cl_id	Integer	5	PK	NOT NULL	
Sh_id	Integer	5	FK	NOT NULL	
Start_time	Time			NOT NULL	
End_time	Time			NOT NULL	

The clock table holds information about the time an employee clocks in and out for each shift.

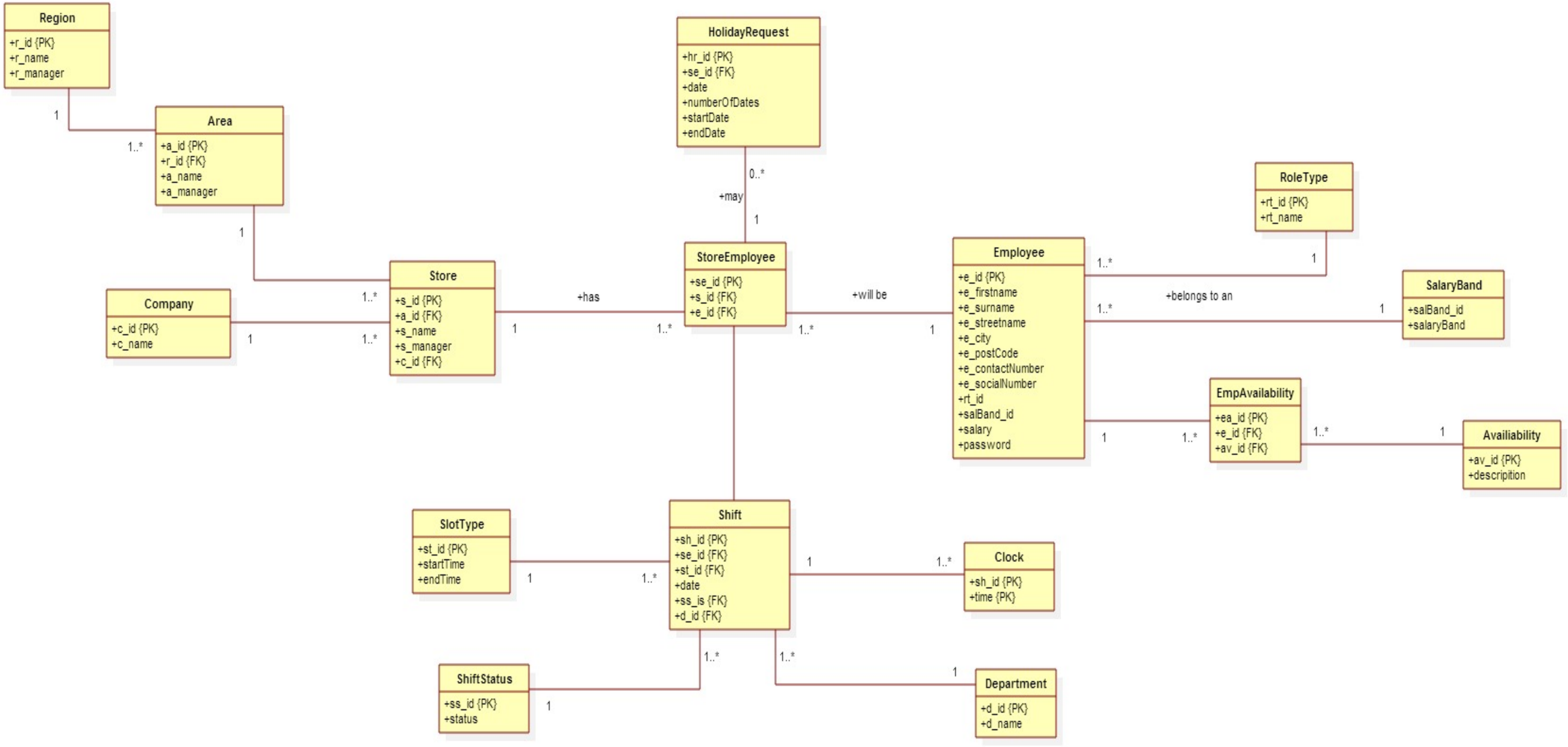


Figure 4.2.1 Class Diagram

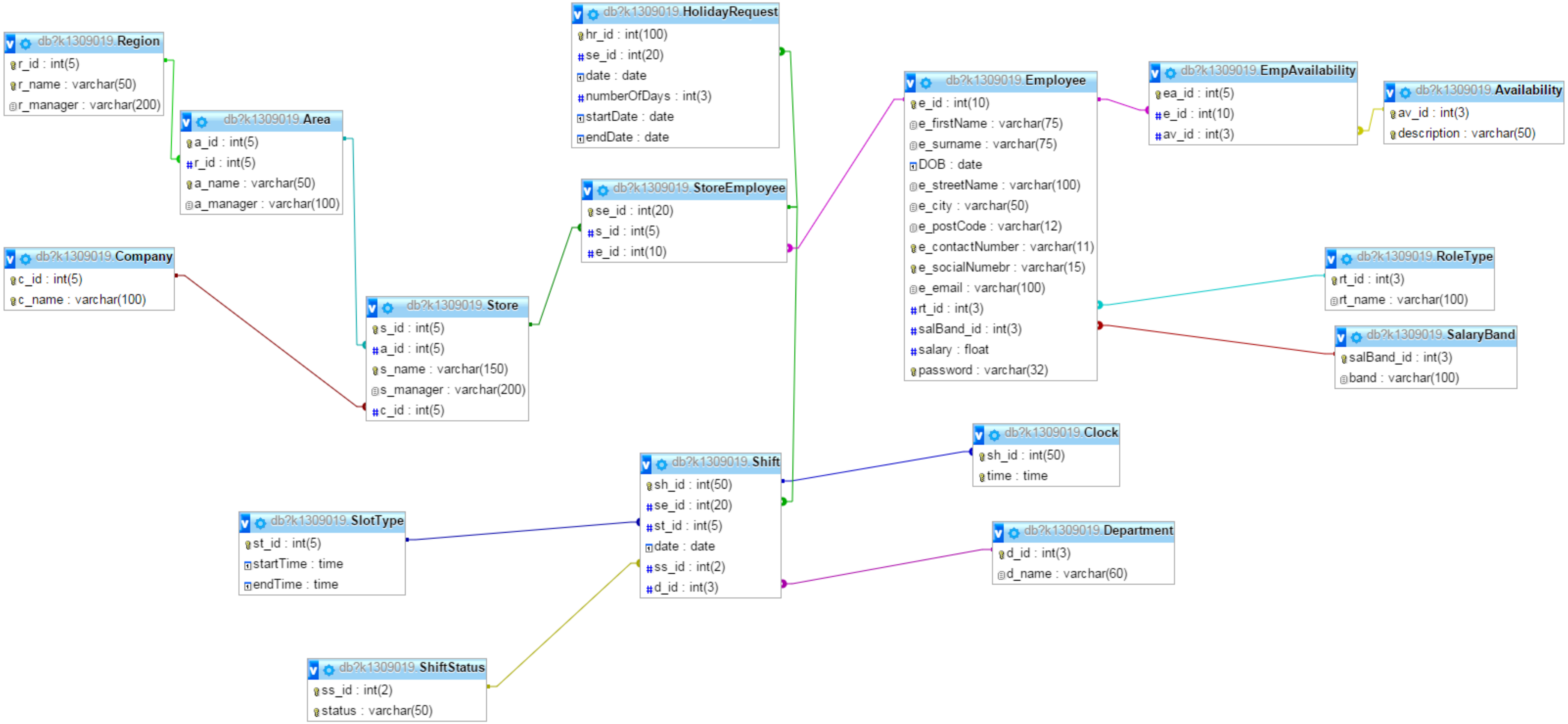


Figure 4.2.3 ERD

Company					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
C_id	Integer	5	PK	NOT NULL	
C_name	Varchar	100		NOT NULL	Unique

The company table holds the name of each retail company the client has in its corporation.

Department					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
D_id	Integer	3	PK	NOT NULL	
D_name	Varchar	60		NOT NULL	

The department table holds the different departments that the client has in its retail companies.

EmpAvailability					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
Ea_id	Integer	5	PK	NOT NULL	
E_id	Integer	10	FK	NOT NULL	
Av_id	Integer	3	FK	NOT NULL	

The emp availability table holds information about each employee's availability.

Employee					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
E_id	Integer	10	PK	NOT NULL	
E_firstName	Varchar	75		NOT NULL	
E_surname	Varchar	75		NOT NULL	
DOB	Date			NOT NULL	
E_streeName	Varchar	100		NOT NULL	
E_city	Varchar	50		NOT NULL	
E_postCode	Varchar	12		NOT NULL	
E_contactNumber	Varchar	11		NOT NULL	Unique
E_socialNumebr	Varchar	15		NOT NULL	Unique
E_email	Varchar	50		NOT NULL	
Rt_id	Integer	3	FK	NOT NULL	
salBand_id	Integer	3	FK	NOT NULL	
Salary	Float			NOT NULL	
Password	Varchar	32		NOT NULL	

The employee table holds the employees personal information as well as their job description, salary and password for the Rota system.

HolidayRequest					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
Hr_id	Integer	100	PK	NOT NULL	
Se_id	Integer	20	FK	NOT NULL	
Date	Date			NOT NULL	
numberOfDays	Integer	3		NOT NULL	
startDate	Date			NOT NULL	
endDate	Date			NOT NULL	

The holiday request table holds all the information about requested holidays by store employees.

Region					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
R_id	Integer	5	PK	NOT NULL	
R_name	Varchar	50		NOT NULL	Unique
R_manger	Varchar	200		NOT NULL	

The region table holds information about all the region's the client uses.

RoleType					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
Rt_id	Integer	3	PK	NOT NULL	
Rt_name	Varchar	100		NOT NULL	

The role type table holds information about all the types of job roles an employee can be associated to.

SalaryBand					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
salBand_id	Integer	3	PK	NOT NULL	
salaryBand	Varchar	100		NOT NULL	

The salary band table hold simple information about which salary band an employee will be associated with.

Shift					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
Sh_id	Integer	50	PK	NOT NULL	
Se_id	Integer	20	FK	NOT NULL	
St_id	Integer	5	FK	NOT NULL	
Date	Date			NOT NULL	
Ss_id	Integer	2	FK	NOT NULL	
D_id	Integer	3	FK	NOT NULL	

The shift table holds information about each employee's shift.

Shift Status					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
Ss_id	Integer	2	PK	NOT NULL	
Status	Varchar	50		NOT NULL	

The shift status table holds information about all the possible statues for a shift.

Slot Type					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
St_id	Integer	5	PK	NOT NULL	
startTime	Time			NOT NULL	
endTime	Time			NOT NULL	

The slot type table holds information about all the different shift timings.

Store					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
S_id	Integer	5	PK	NOT NULL	
A_id	Integer	5	FK	NOT NULL	
S_name	Varchar	150		NOT NULL	Unique
S_manager	Varchar	200		NOT NULL	
C_id	Integer	5	FK	NOT NULL	

The store table holds information about each store the client has.

StoreEmployee					
Attribute Name	**Data Type**	**Size**	**PK/FK**	**Not Null**	**Constraint**
Se_id	Integer	20	PK	NOT NULL	
S_id	Integer	5	FK	NOT NULL	
E_id	Integer	10	FK	NOT NULL	

The store employee table holds information about which store an employee is currently working at.

Chapter 5: Implementation & Testing

The purpose of this chapter is to explain the decisions that were made when implementing the web application for client. The reason for this is show how much thought went into the creation of the web application and why it each aspect is important.

The structure of this chapter will begin with the discussion of the tools and technologies that were used in building of this the project and an explanation as to why they were chosen and how they were used. The discussion of architecture of the system will be discussed next; this will explain the importance of the architecture and how it was adapted for the project. The implantation of the user interface will follow the architecture. The implementation of the code and implementation of the database will then be discussed after the user interface. In these sections there will be a discussion to explain their purpose in project and how it was implemented. Also during this chapter testing was conducted on some aspects of the implementation process. The structure of these test begin with an explanation of the test, followed by the expected result and the actual result of the test conduct.

5.1 Tools and technologies

5.1.1 Text editor

At the beginning of this project Notepad++ was the text editor that was being used but as the project started to move along the text editor wasn't correcting syntax errors. As result Notepad++ was changed to NetBeans as it corrected the errors and shows possible errors that could affect the code. Also it suggested possible syntax that could be needed.

5.1.2 Host

For this project the Kingston University's kunet host was chosen to host the web application. The reason for using kunet is due to the fact it is free and provided by the university. Also if there is problem with kunet the developer is able to speak to someone in person and sort out the problem quickly and efficiently. To upload the web application files to the host FileZilla and Cyber Duck were used to do this action.

5.1.3 Database

For the project a MySQL database was used with phpMyAdmin. The reason for using phpMyAdmin was due to the fact it is needed for a web application. An alternative database software is Oracle SQL developer but using this software doesn't compile with the needs of this project.

5.1.4 Frameworks

In this project two frameworks were used in the user interface these were Bootstrap and Formoid. Bootstrap was used to make the user interface look professional but simple to use. Another reason for using Bootstrap was it compatible with a range of end user devices e.g. tables, smart phones etc. therefore the Bootstrap adjust the web application to the size of the scene of the end user device. Formoid was used to make the forms for the web application as it saves time on creating the forms from scratch.

5.1.5 PHP

In this project PHP will be the server-side language that will be used. The reason for using PHP is that it works well with MySQL databases that use the phpMyAdmin control panel.

PHP is also communicates between the HTML web pages and the webservers a web application is hosted on.

5.1.6 MySQL

MySQL is a type of database will is commonly used to store information for web application. MySQL is commonly used with PHP as it allows content to be inserted, selected, updated and deleted dynamically from the database via web pages.

5.1.7 HTML5

HTML5 is commonly used in the creation and implementation of web pages for web applications. HTML5 has built in features with saves time when creating web pages as the old version of HTML would have to use a scripting language like JavaScript to achieve the same result.

5.1.8 CSS

CSS is used alongside HTML in order to format and add style to the web pages. By using CSS I have been able to make the web pages look unified and constant with each other.

5.2 Architecture

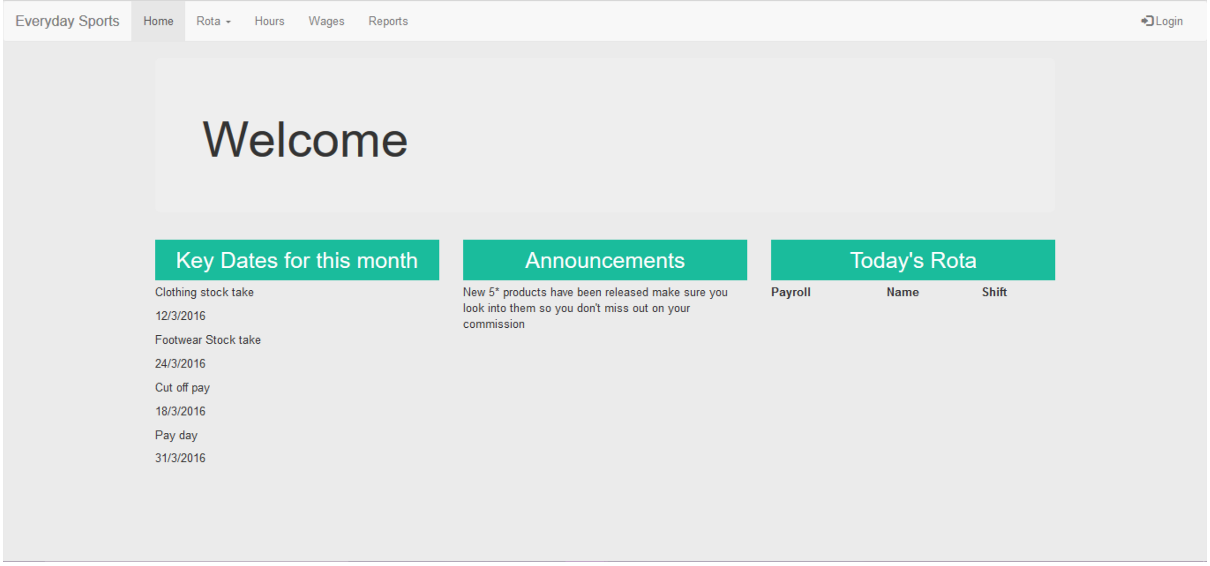
For the system architecture the project will use the basic concepts of MVC but with some adaptations to it. As stated on SitePoint.com MVC was developed in the late 1970's as a software architecture that was built on the basis of separating data from the methods and presentation of data [18]. The basic concept of MVC is that the user uses the controller, the controller manipulates the model, the model updated the view and the view shows the contents to the user [18]. For the project the MVC has been changed slightly, the utils stores all the functions that interact with the database. The model holds all the classes that correspond to each of the tables in the database. The controller manipulates the model and uses the utils, the view shows the content that the controller has retrieved from the model and utils.

5.3 User Interface Implementation & Testing

The aim of the user interface is show the user information in a well presented fashion. The user interface includes features that the users of the client want in the website which enhances the overall effectiveness of the site. The implementation of the user interface caters to two of the three users that were mentioned in the requirement analysis. The sides of the website that have been implemented are for the store employees and the store managers. A lot of the webpages are used for both users but there are certain pages i.e. reports, store rota etc. which require managerial access.

5.3.1 Homepage

The homepage welcomes the user to the website. The homepage is updated weekly so that the uses can see up to date information on the key dates and announcements. Today's rota will only be seen once the user has logged into the system.



Welcome

Key Dates for this month

Clothing stock take
12/3/2016
Footwear Stock take
24/3/2016
Cut off pay
18/3/2016
Pay day
31/3/2016

Announcements

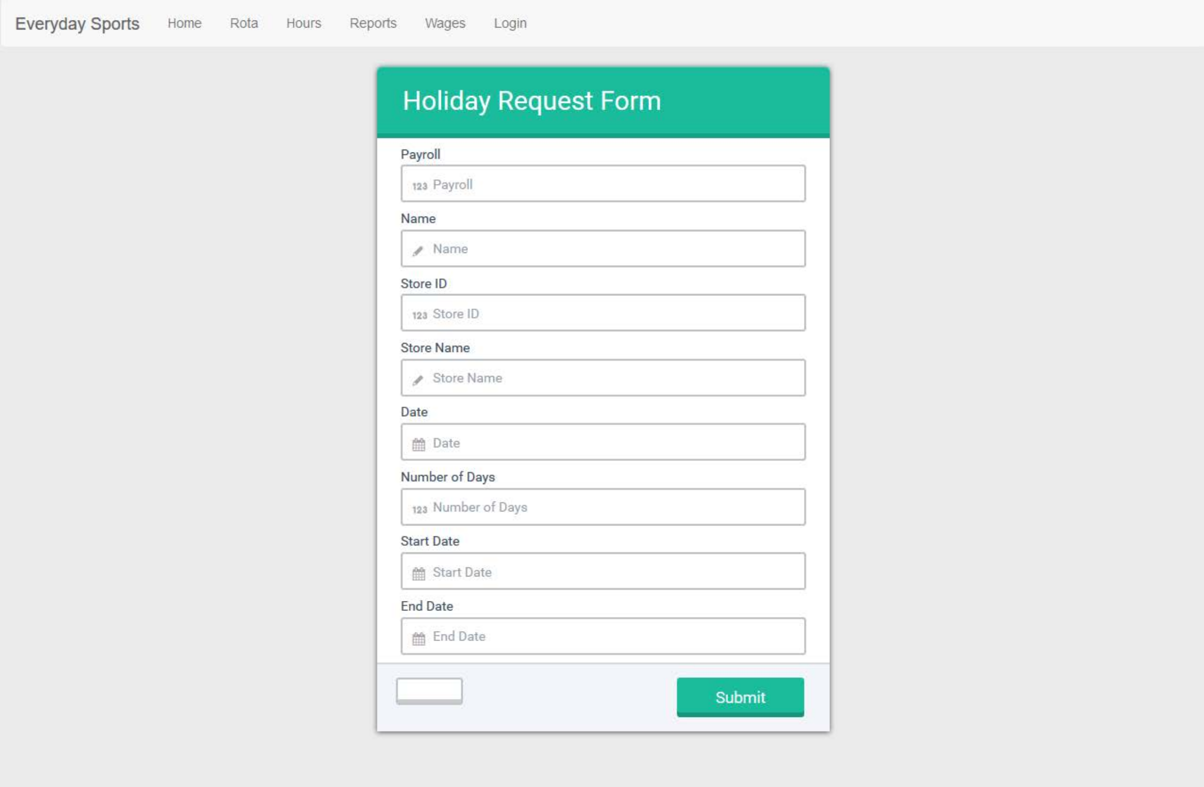
New 5* products have been released make sure you look into them so you don't miss out on your commission

Today's Rota

Payroll	Name	Shift

5.3.2 Holiday Request Form

The holiday request form is to be used by store employees and must be filled out when they want to request time off from work. The form needs submitted at least two weeks before the start date on the form as it will increase the chance of the form will be accepted.



Holiday Request Form

Payroll
123 Payroll

Name
Name

Store ID
123 Store ID

Store Name
Store Name

Date
Date

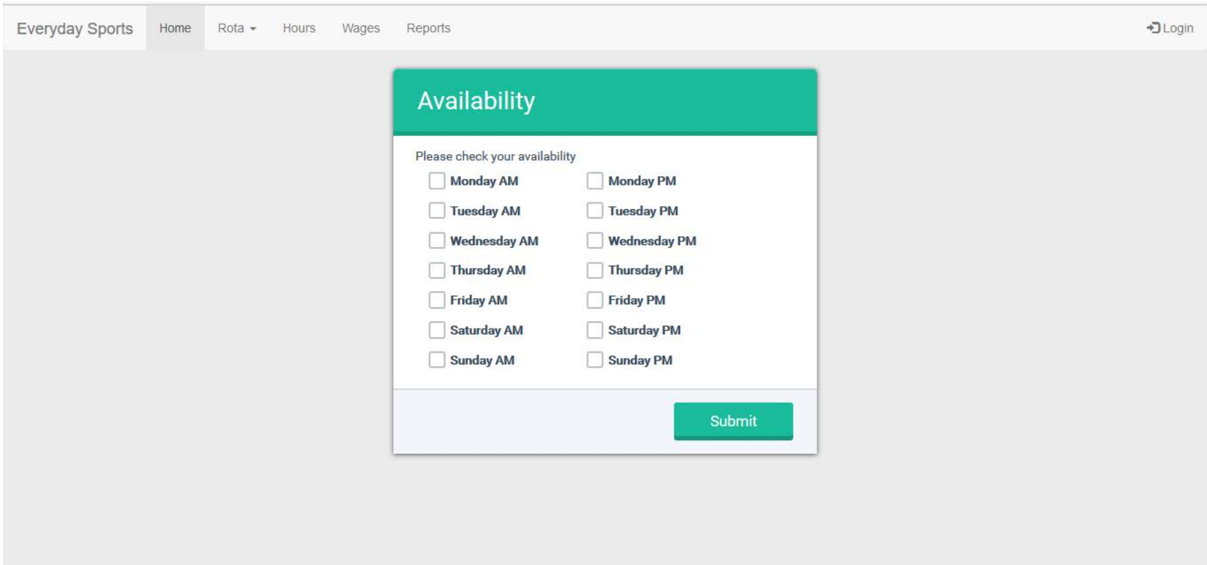
Number of Days
123 Number of Days

Start Date
Start Date

End Date
End Date

5.3.3 Availability Form

The availability form allows the store employees to change and update their availability. This will give store managers and system administer accurate and update information of each store employee availability as it will help when in the creation of the weekly rota.



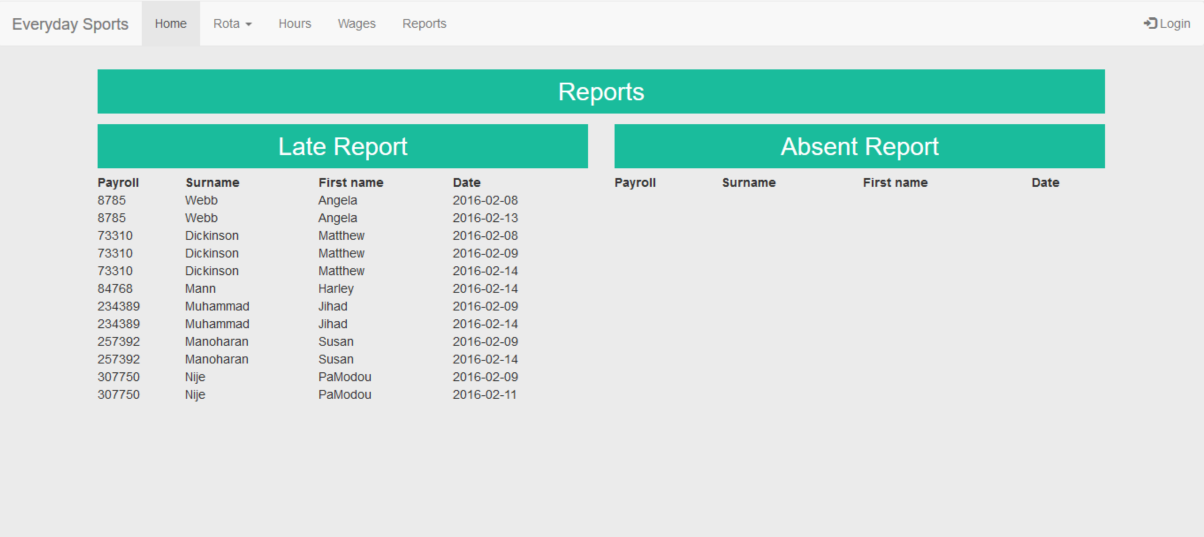
Availability

Please check your availability

☐ Monday AM
☐ Tuesday AM
☐ Wednesday AM
☐ Thursday AM
☐ Friday AM
☐ Saturday AM
☐ Sunday AM
☐ Monday PM
☐ Tuesday PM
☐ Wednesday PM
☐ Thursday PM
☐ Friday PM
☐ Saturday PM
☐ Sunday PM

5.3.4 Reports

The reports page shows the store manager which store employees have been late and absent between two dates. The reason for these reports are that the store managers and the system administrators know which employees are constantly late and absent to work. Also the store managers can use this information when they are conducting an investigation into a store employee. The system administrators will use the information to question the store managers about why there is high number of late and absent employees at a particular store.



Reports

Late Report

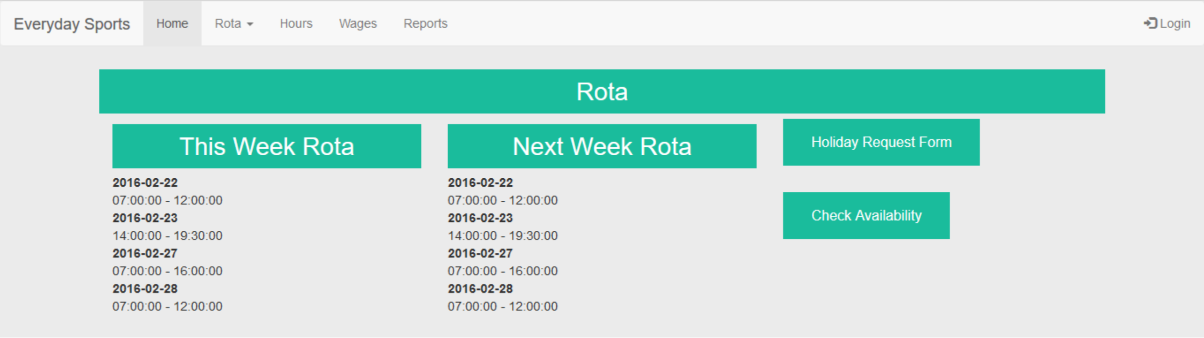
Payroll	Surname	First name	Date
8785	Webb	Angela	2016-02-08
8785	Webb	Angela	2016-02-13
73310	Dickinson	Matthew	2016-02-08
73310	Dickinson	Matthew	2016-02-09
73310	Dickinson	Matthew	2016-02-14
84768	Mann	Harley	2016-02-14
234389	Muhammad	Jihad	2016-02-09
234389	Muhammad	Jihad	2016-02-14
257392	Manoharan	Susan	2016-02-09
257392	Manoharan	Susan	2016-02-14
307750	Nije	PaModou	2016-02-09
307750	Nije	PaModou	2016-02-11

Absent Report

Payroll	Surname	First name	Date

5.3.5 Rota

The rota page is a report that shows the store employee that is logged in the days that they working for the current week and the following weeks rota. This page also has buttons that allows the store employee change their availability and submit a holiday request form.

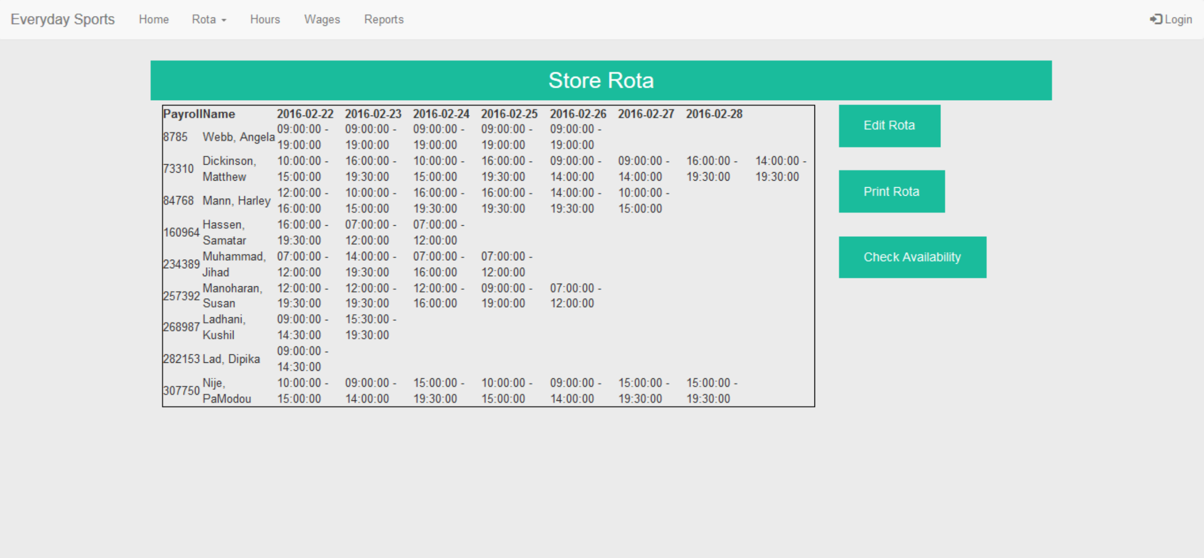


Rota

This Week Rota	Next Week Rota
2016-02-22 07:00:00 - 12:00:00 **2016-02-23** 14:00:00 - 19:30:00 **2016-02-27** 07:00:00 - 16:00:00 **2016-02-28** 07:00:00 - 12:00:00	**2016-02-22** 07:00:00 - 12:00:00 **2016-02-23** 14:00:00 - 19:30:00 **2016-02-27** 07:00:00 - 16:00:00 **2016-02-28** 07:00:00 - 12:00:00

5.3.6 Store Rota

The store rota is a report that shows all the employee that are working at the store for that week. This page can only be accessed by store managers and system administers.



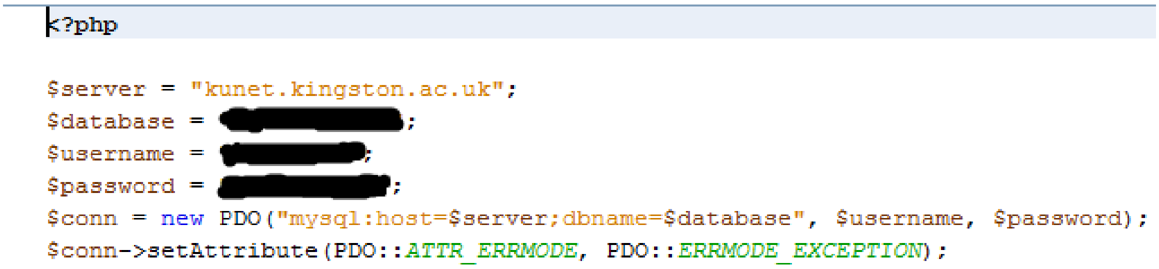
Store Rota

PayrollName		**2016-02-22**	**2016-02-23**	**2016-02-24**	**2016-02-25**	**2016-02-26**	**2016-02-27**	**2016-02-28**	
8785	Webb, Angela	09:00:00 - 19:00:00	09:00:00 - 19:00:00	09:00:00 - 19:00:00	09:00:00 - 19:00:00	09:00:00 - 19:00:00			
73310	Dickinson, Matthew	10:00:00 - 15:00:00	16:00:00 - 19:30:00	10:00:00 - 15:00:00	16:00:00 - 19:30:00	09:00:00 - 14:00:00	09:00:00 - 14:00:00	16:00:00 - 19:30:00	14:00:00 - 19:30:00
84768	Mann, Harley	12:00:00 - 16:00:00	10:00:00 - 15:00:00	16:00:00 - 19:30:00	16:00:00 - 19:30:00	14:00:00 - 19:30:00	10:00:00 - 15:00:00		
160964	Hassen, Samatar	16:00:00 - 19:30:00	07:00:00 - 12:00:00	07:00:00 - 12:00:00					
234389	Muhammad, Jihad	07:00:00 - 12:00:00	14:00:00 - 19:30:00	07:00:00 - 16:00:00	07:00:00 - 12:00:00				
257392	Manoharan, Susan	12:00:00 - 19:30:00	12:00:00 - 19:30:00	12:00:00 - 16:00:00	09:00:00 - 19:00:00	07:00:00 - 12:00:00			
268987	Ladhani, Kushil	09:00:00 - 14:30:00	15:30:00 - 19:30:00						
282153	Lad, Dipika	09:00:00 - 14:30:00							
307750	Nije, PaModou	10:00:00 - 15:00:00	09:00:00 - 14:00:00	15:00:00 - 19:30:00	10:00:00 - 15:00:00	09:00:00 - 14:00:00	15:00:00 - 19:30:00	15:00:00 - 19:30:00	

5.4 Code Implementation and Testing

5.4.1 Connecting to the Database

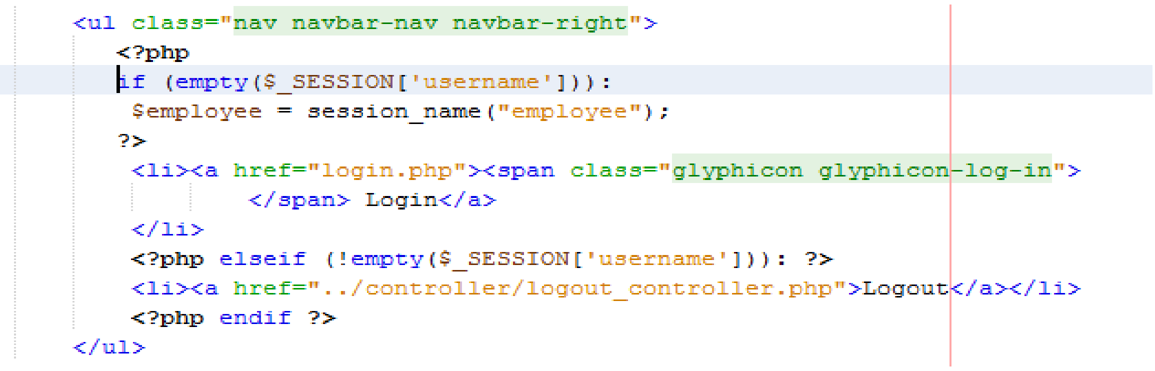
Connecting to the database in very important as the website requires information to be selected, inserted, updated, deleted and checked from the database. The website connects to the database using PHP PDO as shown in the figure below.



```
<?php

$server = "kunet.kingston.ac.uk";
$database = [redacted];
$username = [redacted];
$password = [redacted];
$conn = new PDO("mysql:host=$server;dbname=$database", $username, $password);
$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

5.4.2 Navigation

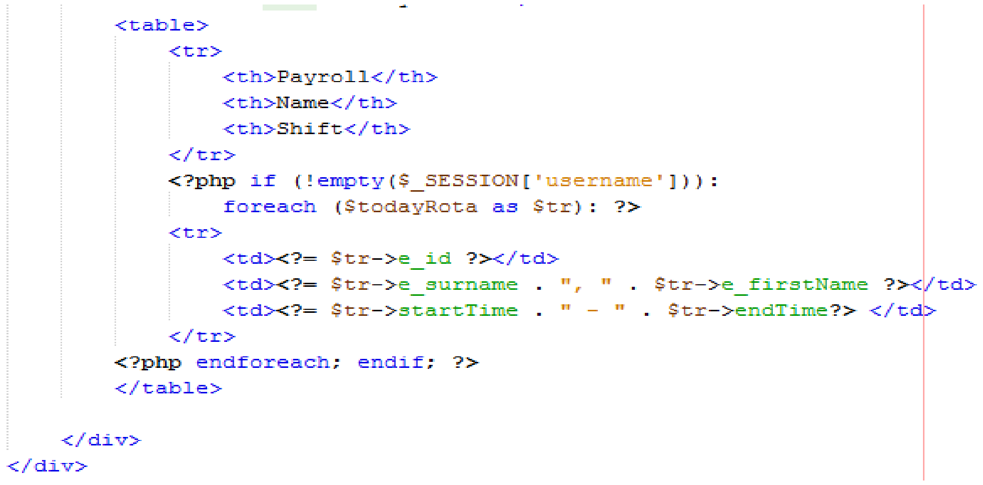
The navigation code of the web application is important as it allows the user to move between the different webpages of the application. The navigation bar also had the user name to show that they are logged in the web application. Using PHP the navigation will show log in if a user hasn’t logged in yet and log out once they have logged in.



```
<ul class="nav navbar-nav navbar-right">
   <?php
   if (empty($_SESSION['username'])):
    $employee = session_name("employee");
   ?>
    <li><a href="login.php"><span class="glyphicon glyphicon-log-in">
            </span> Login</a>
    </li>
    <?php elseif (!empty($_SESSION['username'])): ?>
    <li><a href="../controller/logout_controller.php">Logout</a></li>
    <?php endif ?>
</ul>
```

5.4.3 Homepage

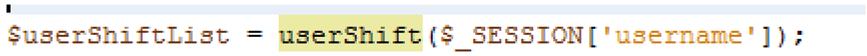
In the homepage there is a column that will show the rota for that day. It can only been seen when a user as logged into the web application. The reason for this is due to the fact this information is only for the store employees and store managers. Also it is a security measure that has been put in place.



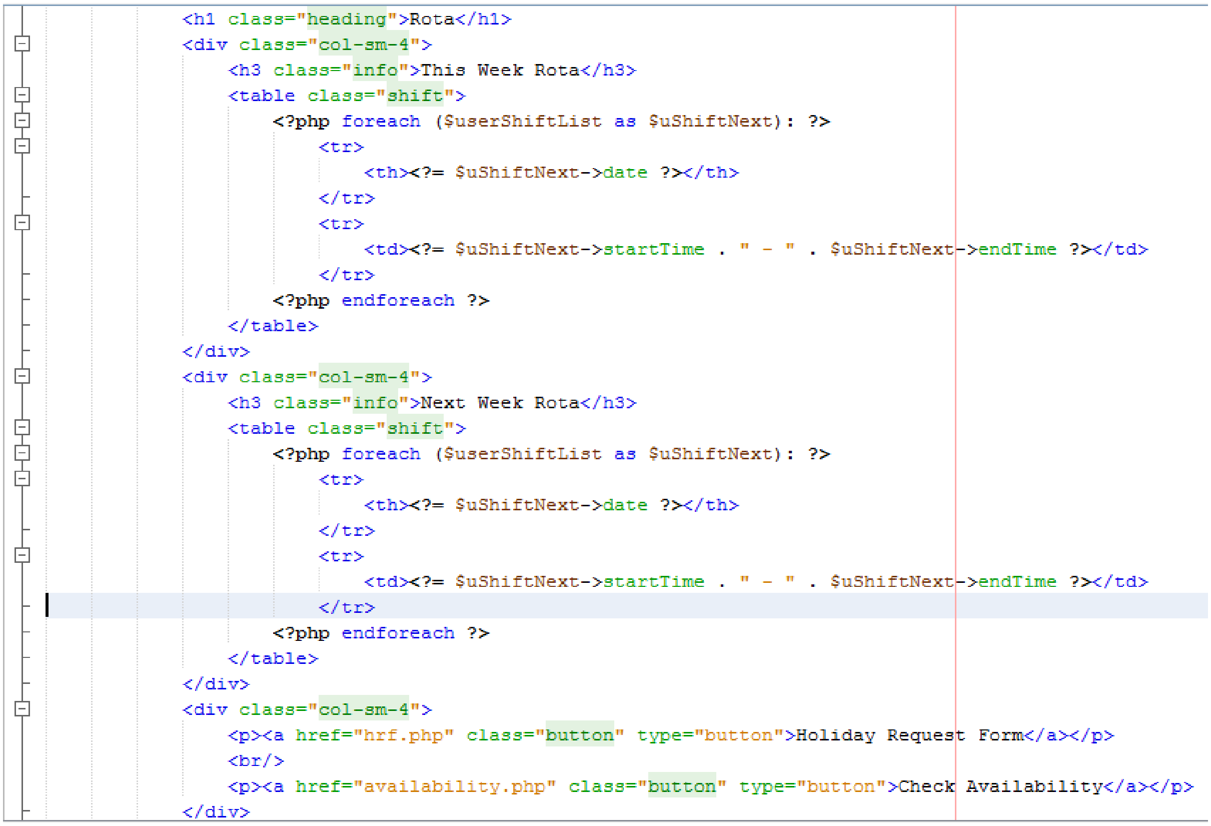
```
        <table>
            <tr>
                <th>Payroll</th>
                <th>Name</th>
                <th>Shift</th>
            </tr>
            <?php if (!empty($_SESSION['username'])):
                foreach ($todayRota as $tr): ?>
            <tr>
                <td><?= $tr->e_id ?></td>
                <td><?= $tr->e_surname . ", " . $tr->e_firstName ?></td>
                <td><?= $tr->startTime . " - " . $tr->endTime?> </td>
            </tr>
        <?php endforeach; endif; ?>
        </table>

    </div>
</div>
```

5.4.4 Rota

The rota page uses the user shift function from the connection file to read from the database all the records for a specific user between two dates and displays it on the webpage using a for each on each table.



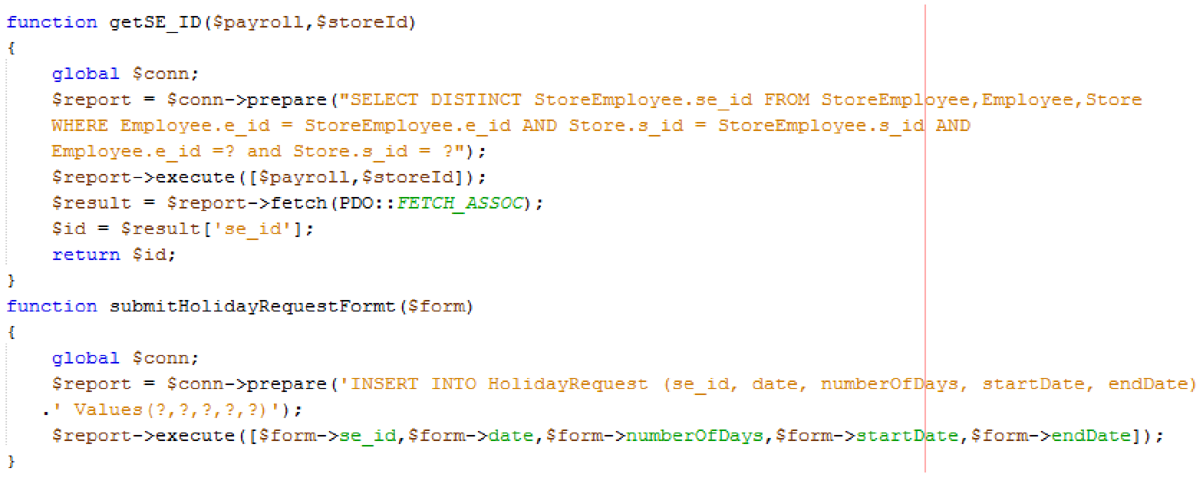
```
$userShiftList = userShift($_SESSION['username']);
```



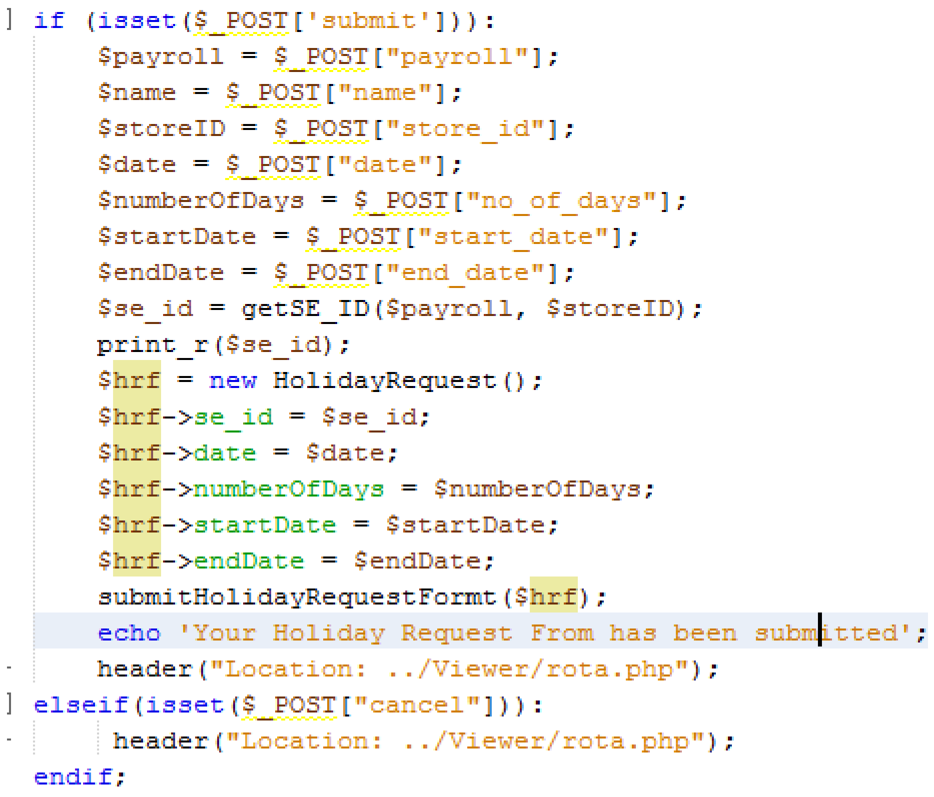
```
<h1 class="heading">Rota</h1>
<div class="col-sm-4">
    <h3 class="info">This Week Rota</h3>
    <table class="shift">
        <?php foreach ($userShiftList as $uShiftNext): ?>
            <tr>
                <th><?= $uShiftNext->date ?></th>
            </tr>
            <tr>
                <td><?= $uShiftNext->startTime . " - " . $uShiftNext->endTime ?></td>
            </tr>
        <?php endforeach ?>
    </table>
</div>
<div class="col-sm-4">
    <h3 class="info">Next Week Rota</h3>
    <table class="shift">
        <?php foreach ($userShiftList as $uShiftNext): ?>
            <tr>
                <th><?= $uShiftNext->date ?></th>
            </tr>
            <tr>
                <td><?= $uShiftNext->startTime . " - " . $uShiftNext->endTime ?></td>
            </tr>
        <?php endforeach ?>
    </table>
</div>
<div class="col-sm-4">
    <p><a href="hrf.php" class="button" type="button">Holiday Request Form</a></p>
    <br/>
    <p><a href="availability.php" class="button" type="button">Check Availability</a></p>
</div>
```

5.4.5 Holiday request form

The holiday request form code has two parts. The first part use the getSE_ID function to find an employee's store employee ID from the database using the employee's payroll and the store ID. This then returned as string. The second part is to insert all the information in the database using the se_id.



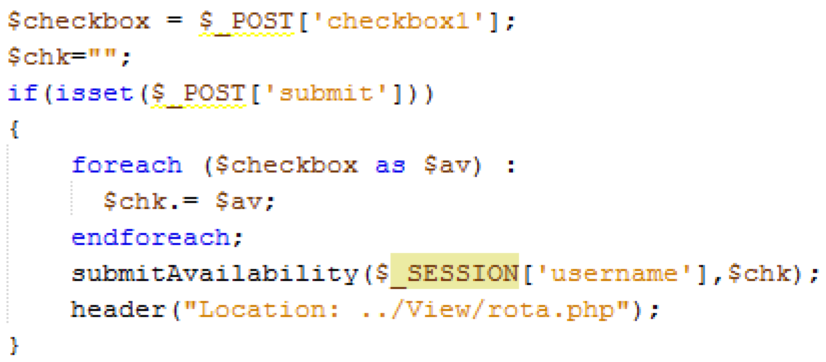
```
function getSE_ID($payroll,$storeId)
{
    global $conn;
    $report = $conn->prepare("SELECT DISTINCT StoreEmployee.se_id FROM StoreEmployee,Employee,Store
    WHERE Employee.e_id = StoreEmployee.e_id AND Store.s_id = StoreEmployee.s_id AND
    Employee.e_id =? and Store.s_id = ?");
    $report->execute([$payroll,$storeId]);
    $result = $report->fetch(PDO::FETCH_ASSOC);
    $id = $result['se_id'];
    return $id;
}
function submitHolidayRequestFormt($form)
{
    global $conn;
    $report = $conn->prepare('INSERT INTO HolidayRequest (se_id, date, numberOfDays, startDate, endDate)
   .' Values(?,?,?,?,?)');
    $report->execute([$form->se_id,$form->date,$form->numberOfDays,$form->startDate,$form->endDate]);
}
```



```
if (isset($_POST['submit'])):
      $payroll = $_POST["payroll"];
      $name = $_POST["name"];
      $storeID = $_POST["store_id"];
      $date = $_POST["date"];
      $numberOfDays = $_POST["no_of_days"];
      $startDate = $_POST["start_date"];
      $endDate = $_POST["end_date"];
      $se_id = getSE_ID($payroll, $storeID);
      print_r($se_id);
      $hrf = new HolidayRequest();
      $hrf->se_id = $se_id;
      $hrf->date = $date;
      $hrf->numberOfDays = $numberOfDays;
      $hrf->startDate = $startDate;
      $hrf->endDate = $endDate;
      submitHolidayRequestFormt($hrf);
      echo 'Your Holiday Request From has been submitted';
      header("Location: ../Viewer/rota.php");
elseif(isset($_POST["cancel"])):
       header("Location: ../Viewer/rota.php");
endif;
```

5.4.6 Availability

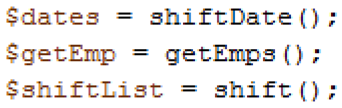
This code in the availability controller allows the use’s checkbox inputs to be inserted into the database. This will then give the store managers the most update information on their store employees’ availability.



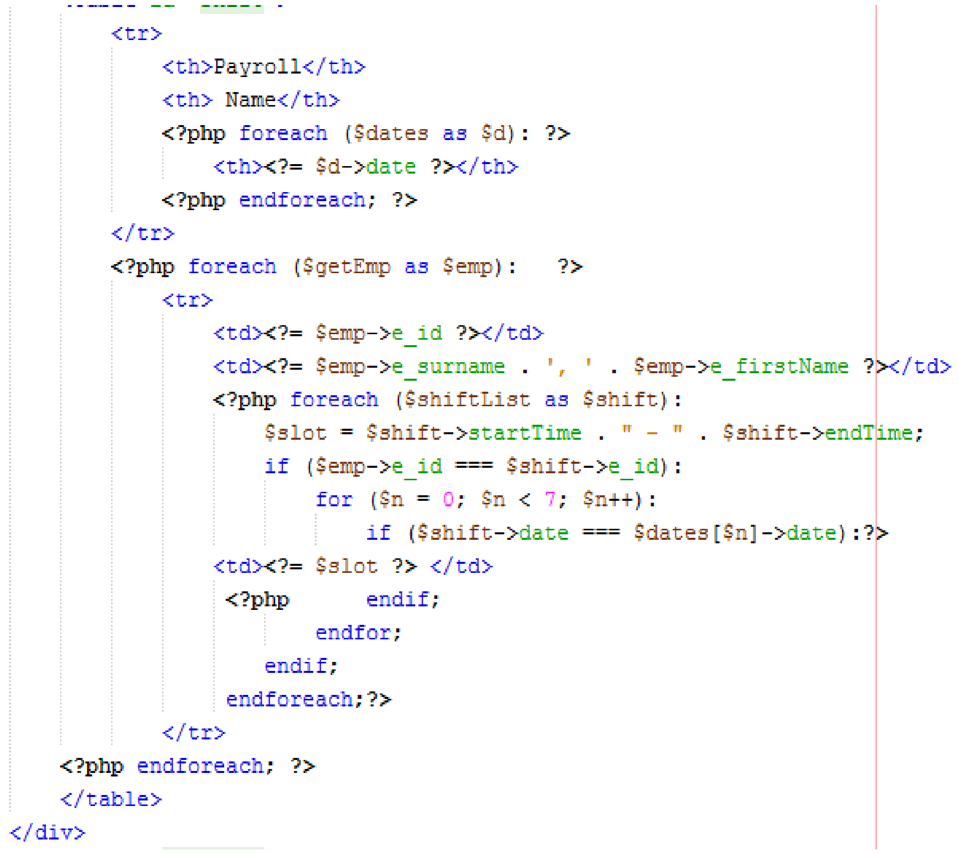
```
$checkbox = $_POST['checkbox1'];
$chk="";
if(isset($_POST['submit']))
{
    foreach ($checkbox as $av) :
      $chk.= $av;
    endforeach;
    submitAvailability($_SESSION['username'],$chk);
    header("Location: ../View/rota.php");
}
```

5.4.7 Store rota

This code will allow the store managers to see the store weekly rota. To crate this view three different functions were need. The first function was to get the dates of a week from the database for the table headings. The second function retrieves all the employees that are working between the start and end of the week dates. The third function retrieves each employee’s shift for that week and is put into the table.



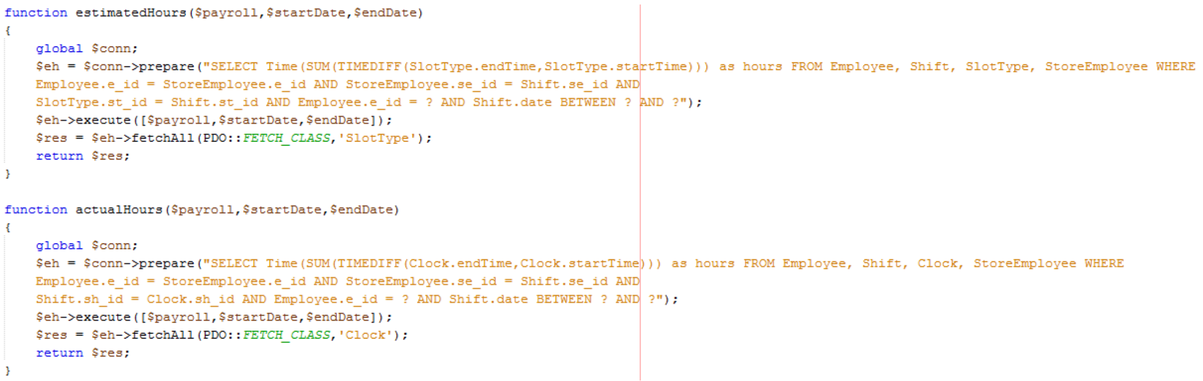
```
$dates = shiftDate();
$getEmp = getEmps();
$shiftList = shift();
```



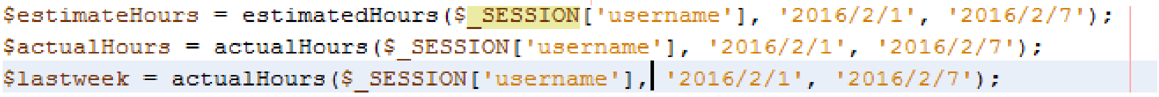
```
        <tr>
            <th>Payroll</th>
            <th> Name</th>
            <?php foreach ($dates as $d): ?>
                <th><?= $d->date ?></th>
            <?php endforeach; ?>
        </tr>
        <?php foreach ($getEmp as $emp):   ?>
            <tr>
                <td><?= $emp->e_id ?></td>
                <td><?= $emp->e_surname . ', ' . $emp->e_firstName ?></td>
                <?php foreach ($shiftList as $shift):
                    $slot = $shift->startTime . " - " . $shift->endTime;
                    if ($emp->e_id === $shift->e_id):
                        for ($n = 0; $n < 7; $n++):
                            if ($shift->date === $dates[$n]->date):?>
                <td><?= $slot ?> </td>
                 <?php      endif;
                        endfor;
                    endif;
                 endforeach;?>
            </tr>
    <?php endforeach; ?>
    </table>
</div>
```

5.4.8 Hours

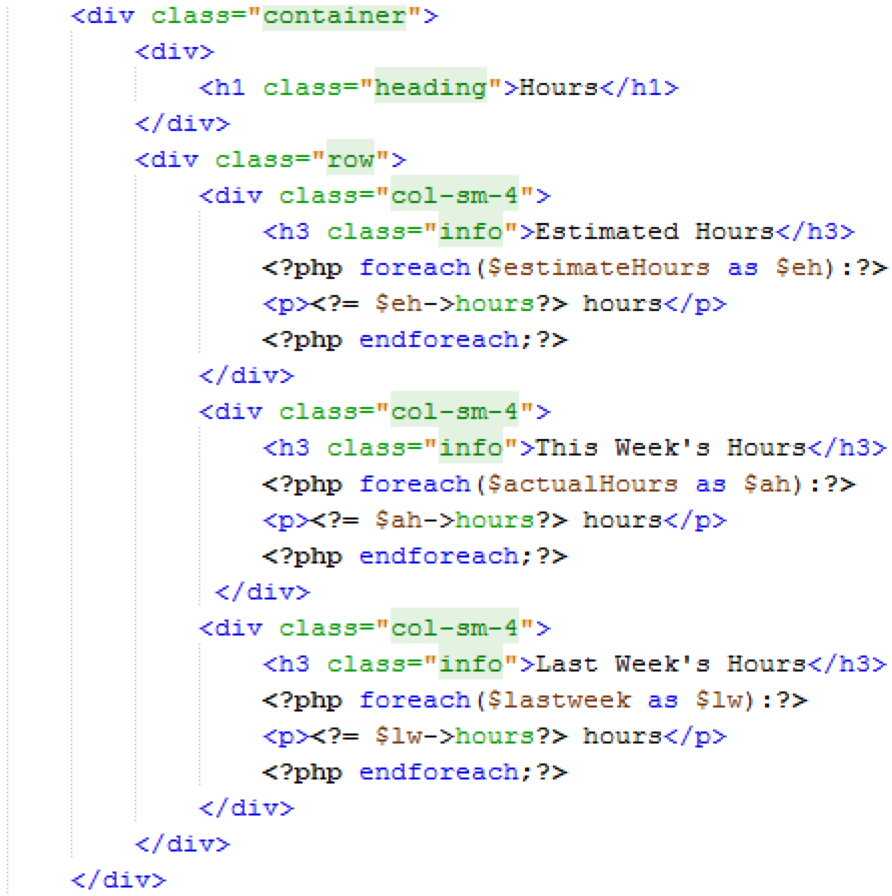
In the hours controller page two functions were called from the connection PHP file. These functions were used to display information about the estimated and actual hours for a user.



```
function estimatedHours($payroll,$startDate,$endDate)
{
    global $conn;
    $eh = $conn->prepare("SELECT Time(SUM(TIMEDIFF(SlotType.endTime,SlotType.startTime))) as hours FROM Employee, Shift, SlotType, StoreEmployee WHERE
    Employee.e_id = StoreEmployee.e_id AND StoreEmployee.se_id = Shift.se_id AND
    SlotType.st_id = Shift.st_id AND Employee.e_id = ? AND Shift.date BETWEEN ? AND ?");
    $eh->execute([$payroll,$startDate,$endDate]);
    $res = $eh->fetchAll(PDO::FETCH_CLASS,'SlotType');
    return $res;
}

function actualHours($payroll,$startDate,$endDate)
{
    global $conn;
    $eh = $conn->prepare("SELECT Time(SUM(TIMEDIFF(Clock.endTime,Clock.startTime))) as hours FROM Employee, Shift, Clock, StoreEmployee WHERE
    Employee.e_id = StoreEmployee.e_id AND StoreEmployee.se_id = Shift.se_id AND
    Shift.sh_id = Clock.sh_id AND Employee.e_id = ? AND Shift.date BETWEEN ? AND ?");
    $eh->execute([$payroll,$startDate,$endDate]);
    $res = $eh->fetchAll(PDO::FETCH_CLASS,'Clock');
    return $res;
}
```



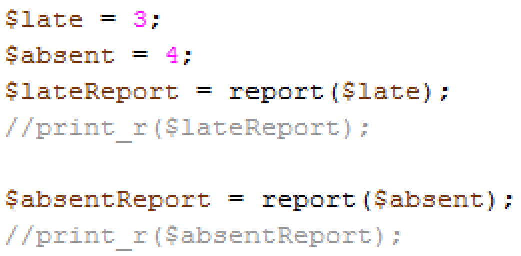
```
$estimateHours = estimatedHours($_SESSION['username'], '2016/2/1', '2016/2/7');
$actualHours = actualHours($_SESSION['username'], '2016/2/1', '2016/2/7');
$lastweek = actualHours($_SESSION['username'], '2016/2/1', '2016/2/7');
```



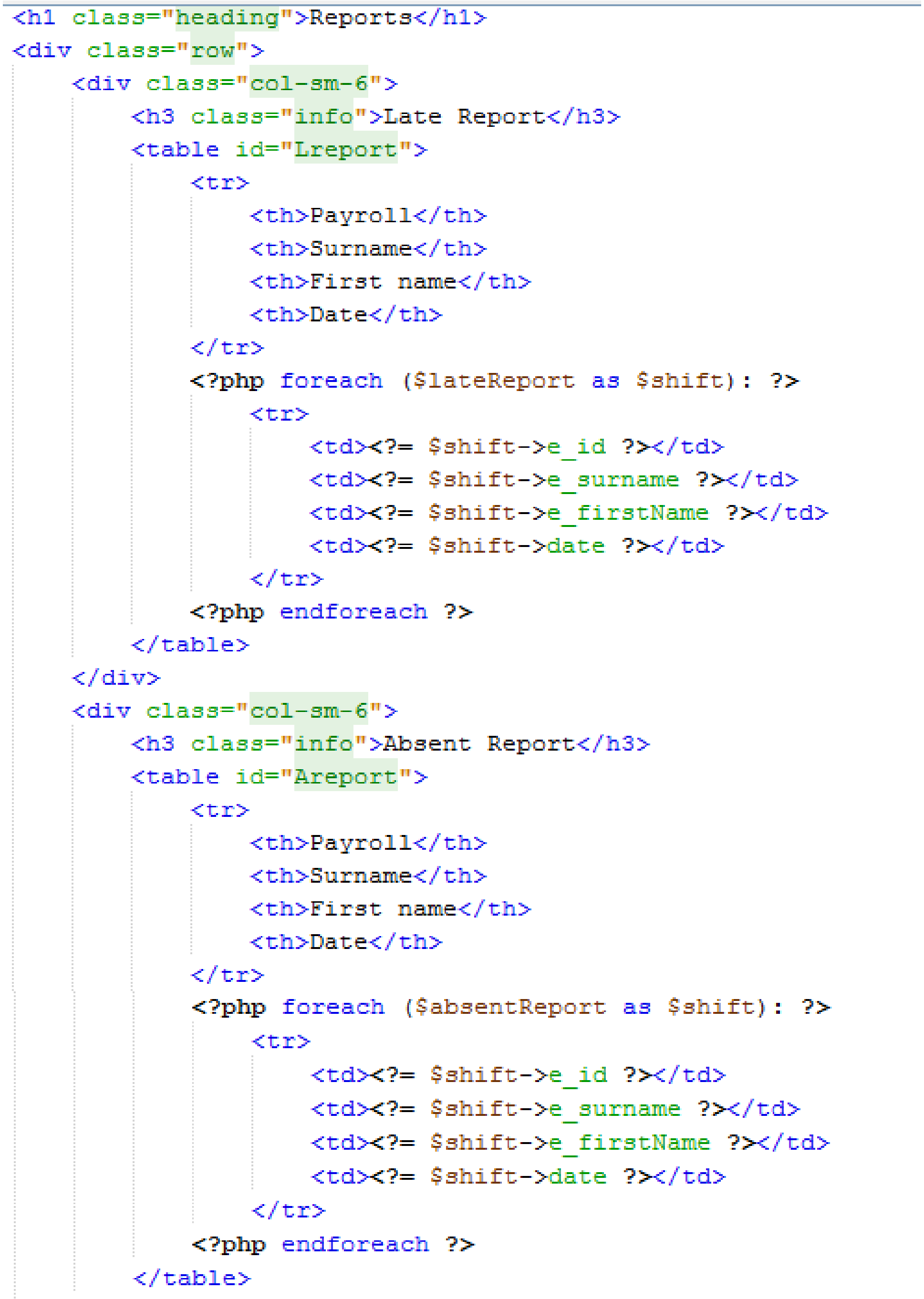
```
<div class="container">
    <div>
        <h1 class="heading">Hours</h1>
    </div>
    <div class="row">
        <div class="col-sm-4">
            <h3 class="info">Estimated Hours</h3>
            <?php foreach($estimateHours as $eh):?>
            <p><?= $eh->hours?> hours</p>
            <?php endforeach;?>
        </div>
        <div class="col-sm-4">
            <h3 class="info">This Week's Hours</h3>
            <?php foreach($actualHours as $ah):?>
            <p><?= $ah->hours?> hours</p>
            <?php endforeach;?>
         </div>
        <div class="col-sm-4">
            <h3 class="info">Last Week's Hours</h3>
            <?php foreach($lastweek as $lw):?>
            <p><?= $lw->hours?> hours</p>
            <?php endforeach;?>
        </div>
    </div>
</div>
```

5.4.9 Reports

Using the reports function form the connection file enable the reports page to display the lateness and absent reports by entering the id for each of the statuses into the function and recalled them using for each.



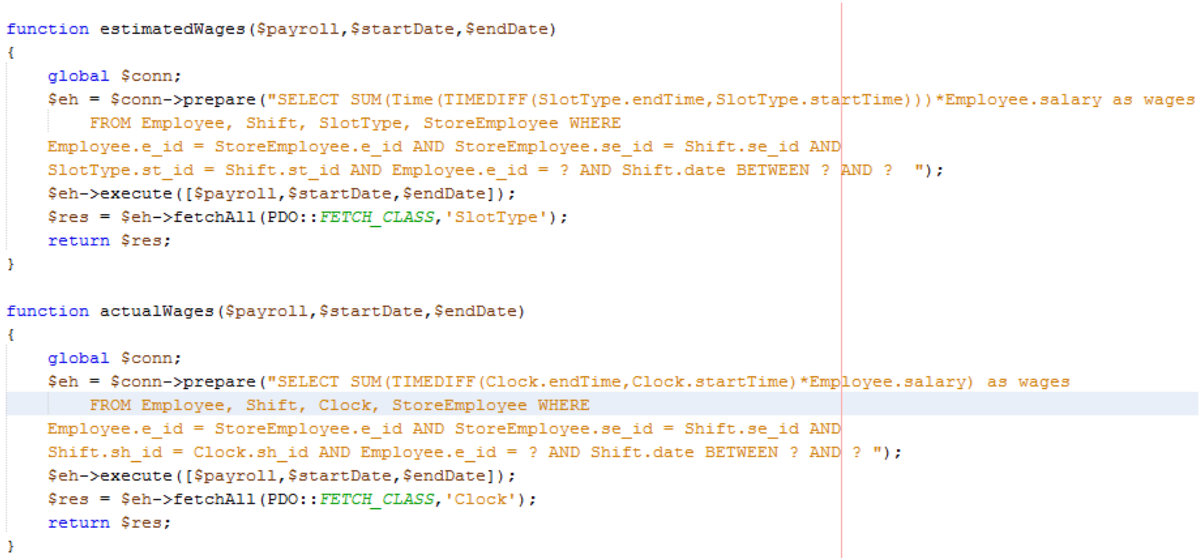
```
$late = 3;
$absent = 4;
$lateReport = report($late);
//print_r($lateReport);

$absentReport = report($absent);
//print_r($absentReport);
```



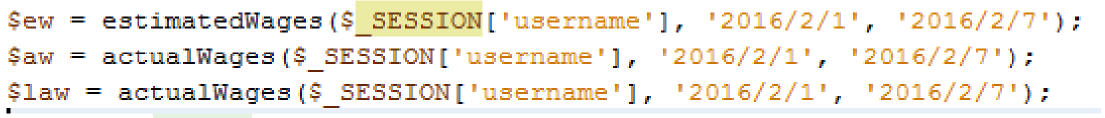
```
<h1 class="heading">Reports</h1>
<div class="row">
    <div class="col-sm-6">
        <h3 class="info">Late Report</h3>
        <table id="Lreport">
            <tr>
                <th>Payroll</th>
                <th>Surname</th>
                <th>First name</th>
                <th>Date</th>
            </tr>
            <?php foreach ($lateReport as $shift): ?>
                <tr>
                    <td><?= $shift->e_id ?></td>
                    <td><?= $shift->e_surname ?></td>
                    <td><?= $shift->e_firstName ?></td>
                    <td><?= $shift->date ?></td>
                </tr>
            <?php endforeach ?>
        </table>
    </div>
    <div class="col-sm-6">
        <h3 class="info">Absent Report</h3>
        <table id="Areport">
            <tr>
                <th>Payroll</th>
                <th>Surname</th>
                <th>First name</th>
                <th>Date</th>
            </tr>
            <?php foreach ($absentReport as $shift): ?>
                <tr>
                    <td><?= $shift->e_id ?></td>
                    <td><?= $shift->e_surname ?></td>
                    <td><?= $shift->e_firstName ?></td>
                    <td><?= $shift->date ?></td>
                </tr>
            <?php endforeach ?>
        </table>
```

5.4.10 Wages

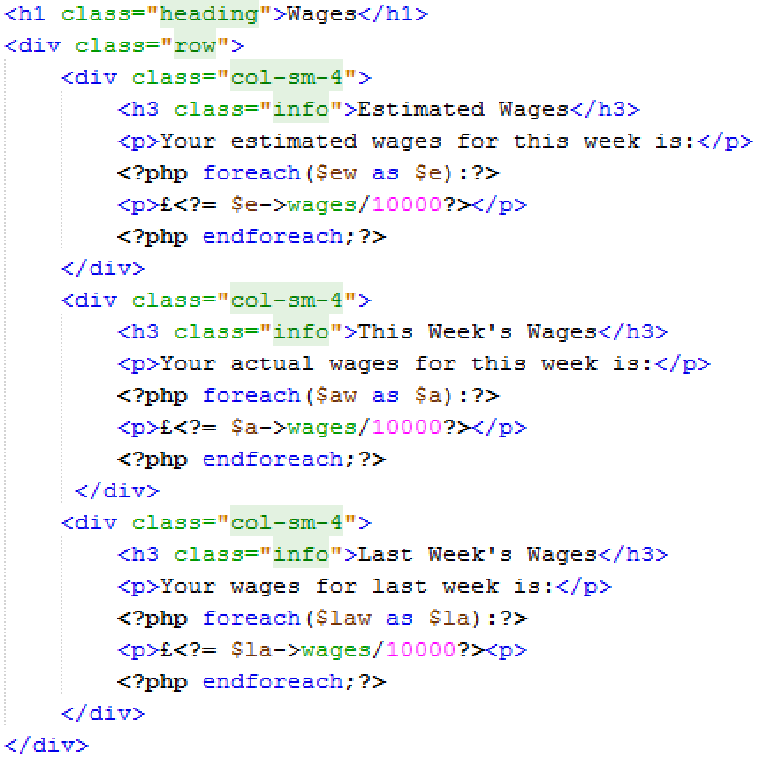
These two function from the connection PHP file were call to the wages controller page were they were used in for each functions to get the information as displayed on the pages. But the wages had to be divided by 100000 first before it is displayed on the webpage.



```
function estimatedWages($payroll,$startDate,$endDate)
{
    global $conn;
    $eh = $conn->prepare("SELECT SUM(Time(TIMEDIFF(SlotType.endTime,SlotType.startTime)))*Employee.salary as wages
        FROM Employee, Shift, SlotType, StoreEmployee WHERE
    Employee.e_id = StoreEmployee.e_id AND StoreEmployee.se_id = Shift.se_id AND
    SlotType.st_id = Shift.st_id AND Employee.e_id = ? AND Shift.date BETWEEN ? AND ?  ");
    $eh->execute([$payroll,$startDate,$endDate]);
    $res = $eh->fetchAll(PDO::FETCH_CLASS,'SlotType');
    return $res;
}

function actualWages($payroll,$startDate,$endDate)
{
    global $conn;
    $eh = $conn->prepare("SELECT SUM(TIMEDIFF(Clock.endTime,Clock.startTime)*Employee.salary) as wages
        FROM Employee, Shift, Clock, StoreEmployee WHERE
    Employee.e_id = StoreEmployee.e_id AND StoreEmployee.se_id = Shift.se_id AND
    Shift.sh_id = Clock.sh_id AND Employee.e_id = ? AND Shift.date BETWEEN ? AND ? ");
    $eh->execute([$payroll,$startDate,$endDate]);
    $res = $eh->fetchAll(PDO::FETCH_CLASS,'Clock');
    return $res;
}
```



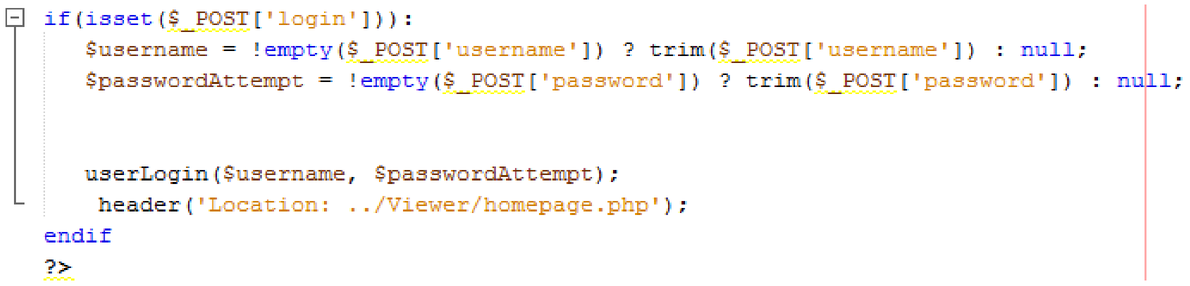
```
$ew = estimatedWages($_SESSION['username'], '2016/2/1', '2016/2/7');
$aw = actualWages($_SESSION['username'], '2016/2/1', '2016/2/7');
$law = actualWages($_SESSION['username'], '2016/2/1', '2016/2/7');
```



```
<h1 class="heading">Wages</h1>
<div class="row">
    <div class="col-sm-4">
        <h3 class="info">Estimated Wages</h3>
        <p>Your estimated wages for this week is:</p>
        <?php foreach($ew as $e):?>
        <p>£<?= $e->wages/10000?></p>
        <?php endforeach;?>
    </div>
    <div class="col-sm-4">
        <h3 class="info">This Week's Wages</h3>
        <p>Your actual wages for this week is:</p>
        <?php foreach($aw as $a):?>
        <p>£<?= $a->wages/10000?></p>
        <?php endforeach;?>
     </div>
    <div class="col-sm-4">
        <h3 class="info">Last Week's Wages</h3>
        <p>Your wages for last week is:</p>
        <?php foreach($law as $la):?>
        <p>£<?= $la->wages/10000?><p>
        <?php endforeach;?>
    </div>
</div>
```

5.4.11 Login

In the login page when a user clicked on the login button the login controller file checks that each filed is not empty. Then process to enter each variable into the user login function, which then checks the database for the user. Once that is done if the user exits the system will login the user otherwise it will return incorrect password.



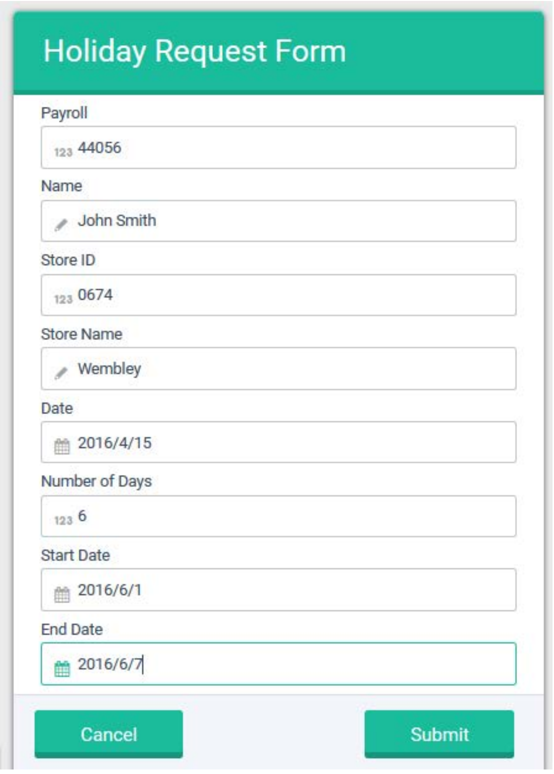
```
if(isset($_POST['login'])):
    $username = !empty($_POST['username']) ? trim($_POST['username']) : null;
    $passwordAttempt = !empty($_POST['password']) ? trim($_POST['password']) : null;


    userLogin($username, $passwordAttempt);
     header('Location: ../Viewer/homepage.php');
endif
?>
```

5.4.12 Testing

Holiday Request Form Testing

Test 1

To test the holiday request an incorrect payroll will entered to check the integrity of the data is correct. The expected outcome of this test is a fatal error will occur and the data that was inputted into will not be inserted in to the database.



Holiday Request Form

Payroll: 44056

Name: John Smith

Store ID: 0674

Store Name: Wembley

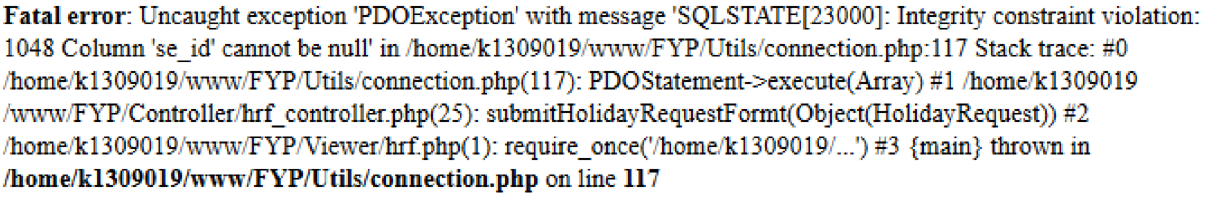
Date: 2016/4/15

Number of Days: 6

Start Date: 2016/6/1

End Date: 2016/6/7

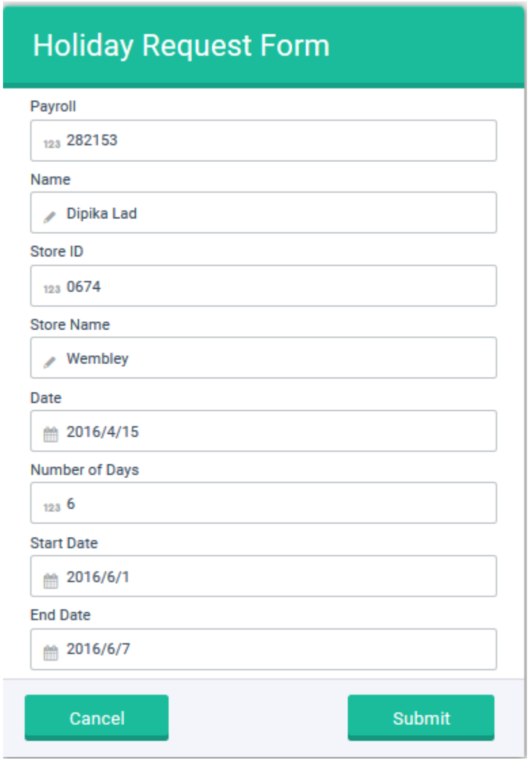
The screen clipping below shows the result of the test that was conducted on the holiday request form with incorrect data. This test proves that any incorrect data inputted into the payroll or store ID field will proceed to show an error and therefore will not be inserted into the database.



Fatal error: Uncaught exception 'PDOException' with message 'SQLSTATE[23000]: Integrity constraint violation: 1048 Column 'se_id' cannot be null' in /home/k1309019/www/FYP/Utils/connection.php:117 Stack trace: #0 /home/k1309019/www/FYP/Utils/connection.php(117): PDOStatement->execute(Array) #1 /home/k1309019 /www/FYP/Controller/hrf_controller.php(25): submitHolidayRequestFormt(Object(HolidayRequest)) #2 /home/k1309019/www/FYP/Viewer/hrf.php(1): require_once('/home/k1309019/...') #3 {main} thrown in **/home/k1309019/www/FYP/Utils/connection.php** on line **117**

Test 2

To test the holiday request form correct data will be inputted to show that the data will be inserted in the database. The expected result is the user is redirected to rota page when the form has been submitted.



Holiday Request Form

Payroll: 282153

Name: Dipika Lad

Store ID: 0674

Store Name: Wembley

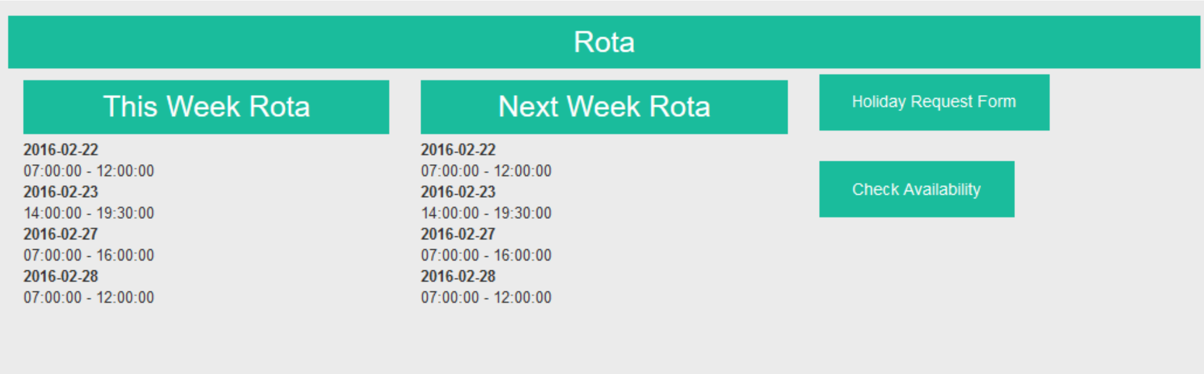
Date: 2016/4/15

Number of Days: 6

Start Date: 2016/6/1

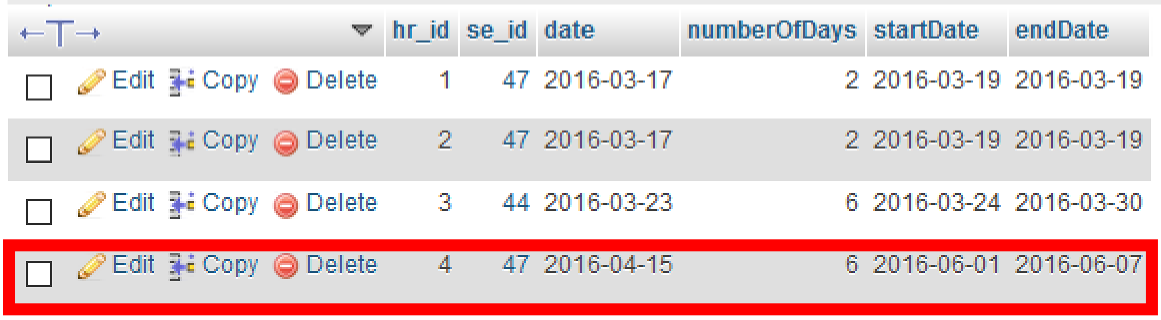
End Date: 2016/6/7

The test was successful as it expected result came true when the holiday request form was submitted. The screen clipping of the database shows the data from the test has successfully been committed to the database and is highlighted by a red outline.



Rota

This Week Rota	Next Week Rota
2016-02-22 07:00:00 - 12:00:00 **2016-02-23** 14:00:00 - 19:30:00 **2016-02-27** 07:00:00 - 16:00:00 **2016-02-28** 07:00:00 - 12:00:00	**2016-02-22** 07:00:00 - 12:00:00 **2016-02-23** 14:00:00 - 19:30:00 **2016-02-27** 07:00:00 - 16:00:00 **2016-02-28** 07:00:00 - 12:00:00



←T→	hr_id	se_id	date	numberOfDays	startDate	endDate
□ Edit Copy Delete	1	47	2016-03-17	2	2016-03-19	2016-03-19
□ Edit Copy Delete	2	47	2016-03-17	2	2016-03-19	2016-03-19
□ Edit Copy Delete	3	44	2016-03-23	6	2016-03-24	2016-03-30
□ Edit Copy Delete	4	47	2016-04-15	6	2016-06-01	2016-06-07

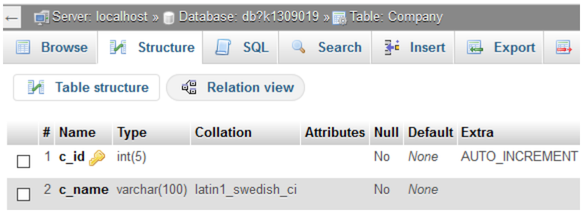
5.5 Database Implementation & Testing

5.5.1 Table statements

Statement 1

In this first statement a simple table was created in the database for all the different retail companies the client has. This table has only two attributes which are c_id and c_name as shown in the code below. Also the data in this table is very simple.

```
CREATE TABLE Company (
c_id int(5) NOT NULL,
c_name varchar(100) NOT NULL
CONSTRAINT c_id_fk PRIMARY KEY (c_id),
CONSTRAINT c_name_un UNIQUE (c_name));
```

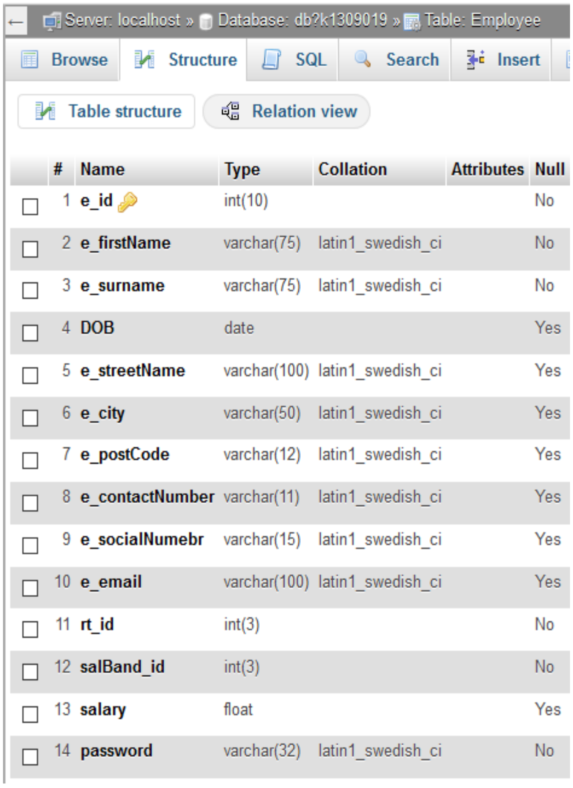


	#	Name	Type	Collation	Attributes	Null	Default	Extra
□	1	**c_id**	int(5)			No	*None*	AUTO_INCREMENT
□	2	**c_name**	varchar(100)	latin1_swedish_ci		No	*None*	

Statement 2

In this second statement a table was created in the database for the clients employees. The table has more than one foreign key and has multiple unique constraints on some of the attributes. The foreign keys in this table are used to access information from the role type table and the salary band table.

```
CREATE TABLE Employee (
e_id int(10) NOT NULL,
e_firstName varchar(75) NOT NULL,
e_surname varchar(75) NOT NULL,
DOB date,
e_streetName varchar(100),
e_city varchar(50),
e_postCode varchar(12) ,
e_contactNumber varchar(11) ,
e_socialNumebr varchar(15) ,
e_email varchar(100) ,
rt_id int(3) NOT NULL,
salBand_id int(3) NOT NULL,
salary float ,
password varchar(32) NOT NULL,
CONSTRAINT e_id_pk PRIMARY KEY (e_id),
CONSTRAINT pass_un UNIQUE (password),
CONSTRAINT sn_un UNIQUE (e_socialNumebr),
CONSTRAINT cn_un UNIQUE
(e_contactNumber),
CONSTRAINT rt_id_fk FOREIGN KEY (rt_id)
REFERENCES RoleType (rt_id),
CONSTRAINT salBand_id_fk FOREIGN KEY (salBand_id) REFERENCES SalaryBand
(salBand_id));
```


← Server: localhost » Database: db?k1309019 » Table: Employee
Browse
Structure
SQL
Search
Insert
Table structure
Relation view

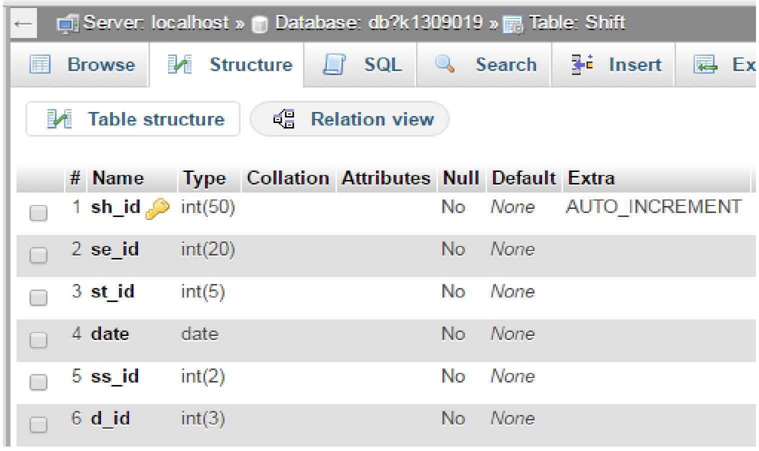
	#	Name	Type	Collation	Attributes	Null
□	1	**e_id**	int(10)			No
□	2	**e_firstName**	varchar(75)	latin1_swedish_ci		No
□	3	**e_surname**	varchar(75)	latin1_swedish_ci		No
□	4	**DOB**	date			Yes
□	5	**e_streetName**	varchar(100)	latin1_swedish_ci		Yes
□	6	**e_city**	varchar(50)	latin1_swedish_ci		Yes
□	7	**e_postCode**	varchar(12)	latin1_swedish_ci		Yes
□	8	**e_contactNumber**	varchar(11)	latin1_swedish_ci		Yes
□	9	**e_socialNumebr**	varchar(15)	latin1_swedish_ci		Yes
□	10	**e_email**	varchar(100)	latin1_swedish_ci		Yes
□	11	**rt_id**	int(3)			No
□	12	**salBand_id**	int(3)			No
□	13	**salary**	float			Yes
□	14	**password**	varchar(32)	latin1_swedish_ci		No

The foreign keys in this table are rt_id and salBand_id. Rt_id is a foreign key because the employee table needs to give each employee record a role type also it reduces the likely hood of human error occurring as only the role type id is inputted into the record and not the name of the role type. salBand_id is a foreign key because the employee table needs to needs to give each employee record a salary band id. The reason being each band has a different minimum salary value which depends on the age of an employee or if the employee is on a contract.

Statement 3

In this third statement the table created is to store all the information about each store employee’s shifts. Each shift has a unique id and imports data from four different tables by using foreign keys.

```
CREATE TABLE Shift (
sh_id int(50) NOT NULL  AUTO_INCREMENT,
  se_id int(20) NOT NULL,
  st_id int(5) NOT NULL,
  date date NOT NULL,
  ss_id int(2) NOT NULL,
  d_id int(3) NOT NULL,

CONSTRAINT sh_id_pk PRIMARY KEY (sh_id),
CONSTRAINT st_if_fk FOREIGN KEY (st_id) REFERENCES SlotType (st_id),
CONSTRAINT ss_id_fk FOREIGN KEY (ss_id) REFERENCES ShiftStatus (ss_id),
CONSTRAINT d_id_fk FOREIGN KEY (d_id) REFERENCES Department (d_id),
CONSTRAINT se_id_fk FOREIGN KEY (se_id) REFERENCES StoreEmployee (se_id));
```



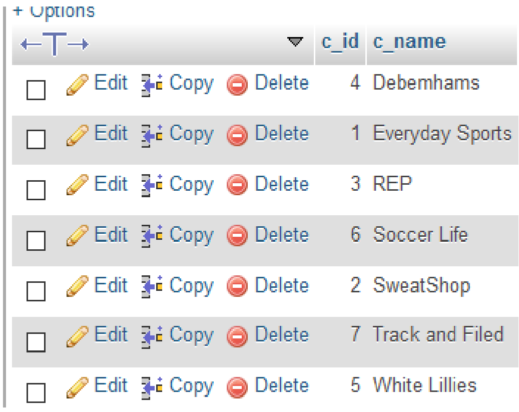
#	Name	Type	Collation	Attributes	Null	Default	Extra
1	sh_id	int(50)			No	None	AUTO_INCREMENT
2	se_id	int(20)			No	None	
3	st_id	int(5)			No	None	
4	date	date			No	None	
5	ss_id	int(2)			No	None	
6	d_id	int(3)			No	None	

The foreign keys in the shift table are se_id, st_id, ss_id and d_id. The reason that these attributes are foreign key is due to the fact that this table relies on infomatin from store employee, slot type, shift status and departement. Also it reduces the amount of humman errors that can occur during data entry as the imformation is already set in the database and just needs to be called using the foreign keys.

5.5.2 Populated tables

Table 1 – Company Table

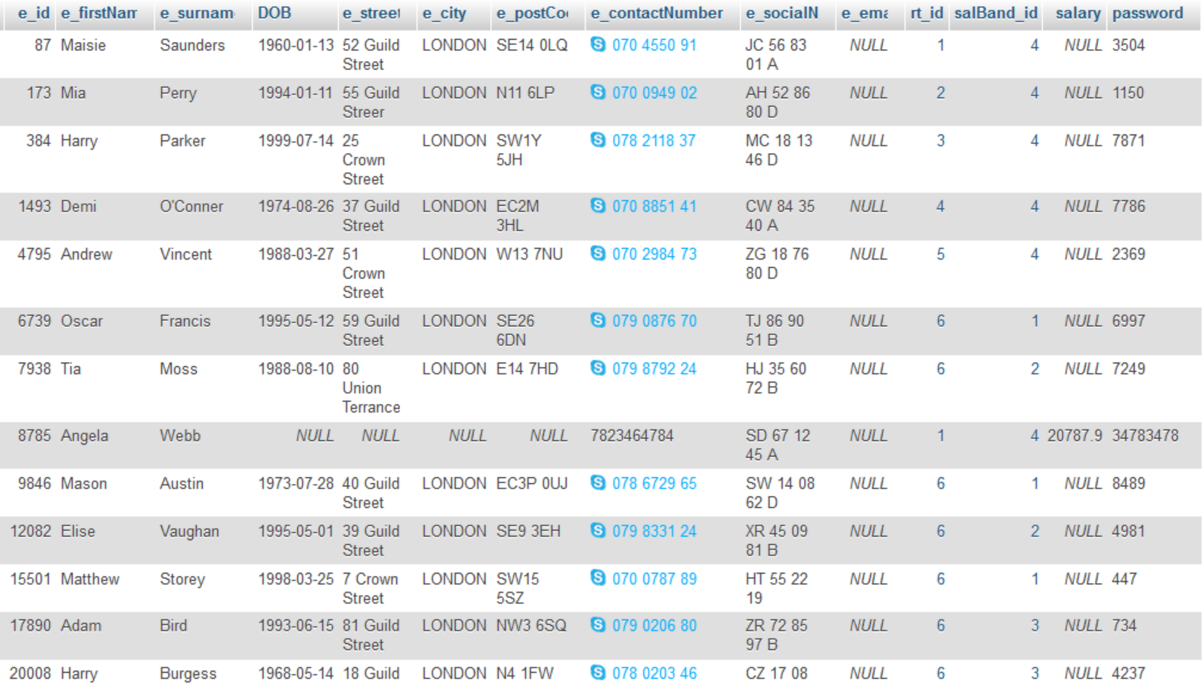
This first table shows all the data that is stored in the company table about all the retail companies the client has.



		c_id	c_name
□	Edit Copy Delete	4	Debemhams
□	Edit Copy Delete	1	Everyday Sports
□	Edit Copy Delete	3	REP
□	Edit Copy Delete	6	Soccer Life
□	Edit Copy Delete	2	SweatShop
□	Edit Copy Delete	7	Track and Filed
□	Edit Copy Delete	5	White Lillies

Table 2 – Employee Table

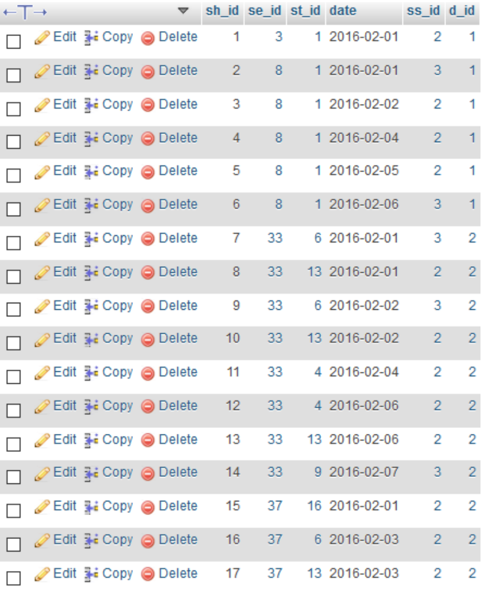
This table shows some of the employees the company has in two of their stores at this moment in time. The table shows each employee's information such as address, DOB, email, contact number etc.



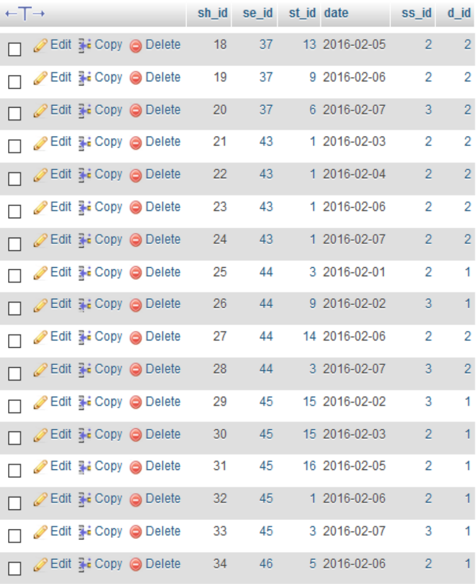
e_id	e_firstNam	e_surnam	DOB	e_street	e_city	e_postCo	e_contactNumber	e_socialN	e_ema	rt_id	salBand_id	salary	password
87	Maisie	Saunders	1960-01-13	52 Guild Street	LONDON	SE14 0LQ	070 4550 91	JC 56 83 01 A	*NULL*	1	4	*NULL*	3504
173	Mia	Perry	1994-01-11	55 Guild Streer	LONDON	N11 6LP	070 0949 02	AH 52 86 80 D	*NULL*	2	4	*NULL*	1150
384	Harry	Parker	1999-07-14	25 Crown Street	LONDON	SW1Y 5JH	078 2118 37	MC 18 13 46 D	*NULL*	3	4	*NULL*	7871
1493	Demi	O'Conner	1974-08-26	37 Guild Street	LONDON	EC2M 3HL	070 8851 41	CW 84 35 40 A	*NULL*	4	4	*NULL*	7786
4795	Andrew	Vincent	1988-03-27	51 Crown Street	LONDON	W13 7NU	070 2984 73	ZG 18 76 80 D	*NULL*	5	4	*NULL*	2369
6739	Oscar	Francis	1995-05-12	59 Guild Street	LONDON	SE26 6DN	079 0876 70	TJ 86 90 51 B	*NULL*	6	1	*NULL*	6997
7938	Tia	Moss	1988-08-10	80 Union Terrance	LONDON	E14 7HD	079 8792 24	HJ 35 60 72 B	*NULL*	6	2	*NULL*	7249
8785	Angela	Webb	*NULL*	*NULL*	*NULL*	*NULL*	7823464784	SD 67 12 45 A	*NULL*	1	4	20787.9	34783478
9846	Mason	Austin	1973-07-28	40 Guild Street	LONDON	EC3P 0UJ	078 6729 65	SW 14 08 62 D	*NULL*	6	1	*NULL*	8489
12082	Elise	Vaughan	1995-05-01	39 Guild Street	LONDON	SE9 3EH	079 8331 24	XR 45 09 81 B	*NULL*	6	2	*NULL*	4981
15501	Matthew	Storey	1998-03-25	7 Crown Street	LONDON	SW15 5SZ	070 0787 89	HT 55 22 19	*NULL*	6	1	*NULL*	447
17890	Adam	Bird	1993-06-15	81 Guild Street	LONDON	NW3 6SQ	079 0206 80	ZR 72 85 97 B	*NULL*	6	3	*NULL*	734
20008	Harry	Burgess	1968-05-14	18 Guild	LONDON	N4 1FW	078 0203 46	CZ 17 08	*NULL*	6	3	*NULL*	4237

Table 3 – Shift Table

This table shows all the information about shifts each employee has. Each shift has a status stating whether is it an upcoming shift, completed shift, cancelled shift, absent shift or late. Also each shift has an ID so that a record can be kept of past shifts for report purposes.



←T→	sh_id	se_id	st_id	date	ss_id	d_id
□ Edit Copy Delete	1	3	1	2016-02-01	2	1
□ Edit Copy Delete	2	8	1	2016-02-01	3	1
□ Edit Copy Delete	3	8	1	2016-02-02	2	1
□ Edit Copy Delete	4	8	1	2016-02-04	2	1
□ Edit Copy Delete	5	8	1	2016-02-05	2	1
□ Edit Copy Delete	6	8	1	2016-02-06	3	1
□ Edit Copy Delete	7	33	6	2016-02-01	3	2
□ Edit Copy Delete	8	33	13	2016-02-01	2	2
□ Edit Copy Delete	9	33	6	2016-02-02	3	2
□ Edit Copy Delete	10	33	13	2016-02-02	2	2
□ Edit Copy Delete	11	33	4	2016-02-04	2	2
□ Edit Copy Delete	12	33	4	2016-02-06	2	2
□ Edit Copy Delete	13	33	13	2016-02-06	2	2
□ Edit Copy Delete	14	33	9	2016-02-07	3	2
□ Edit Copy Delete	15	37	16	2016-02-01	2	2
□ Edit Copy Delete	16	37	6	2016-02-03	2	2
□ Edit Copy Delete	17	37	13	2016-02-03	2	2

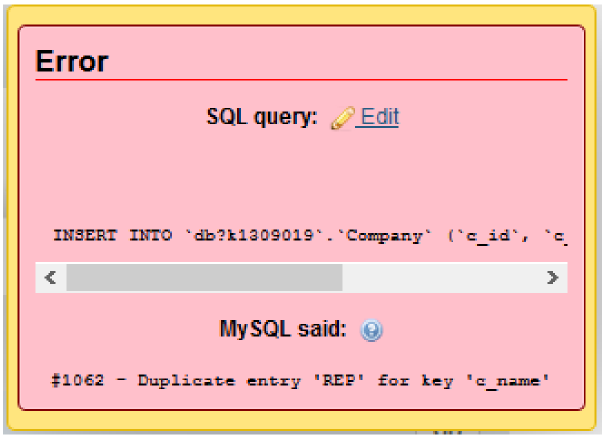


←T→	sh_id	se_id	st_id	date	ss_id	d_id
□ Edit Copy Delete	18	37	13	2016-02-05	2	2
□ Edit Copy Delete	19	37	9	2016-02-06	2	2
□ Edit Copy Delete	20	37	6	2016-02-07	3	2
□ Edit Copy Delete	21	43	1	2016-02-03	2	2
□ Edit Copy Delete	22	43	1	2016-02-04	2	2
□ Edit Copy Delete	23	43	1	2016-02-06	2	2
□ Edit Copy Delete	24	43	1	2016-02-07	2	2
□ Edit Copy Delete	25	44	3	2016-02-01	2	1
□ Edit Copy Delete	26	44	9	2016-02-02	3	1
□ Edit Copy Delete	27	44	14	2016-02-06	2	2
□ Edit Copy Delete	28	44	3	2016-02-07	3	2
□ Edit Copy Delete	29	45	15	2016-02-02	3	1
□ Edit Copy Delete	30	45	15	2016-02-03	2	1
□ Edit Copy Delete	31	45	16	2016-02-05	2	1
□ Edit Copy Delete	32	45	1	2016-02-06	2	1
□ Edit Copy Delete	33	45	3	2016-02-07	3	1
□ Edit Copy Delete	34	46	5	2016-02-06	2	1

5.5.3 Testing

Testing table 1

In the company table there is a unique constraint on the c_name attribute. To test this constraint a company name that already exist in the table will be re-entered into the database. The expected outcome of this test is an error message stating that there is a duplicate entry in c_name.



Error

SQL query: Edit

```
INSERT INTO `db?k1309019`.`Company` (`c_id`, `c
```

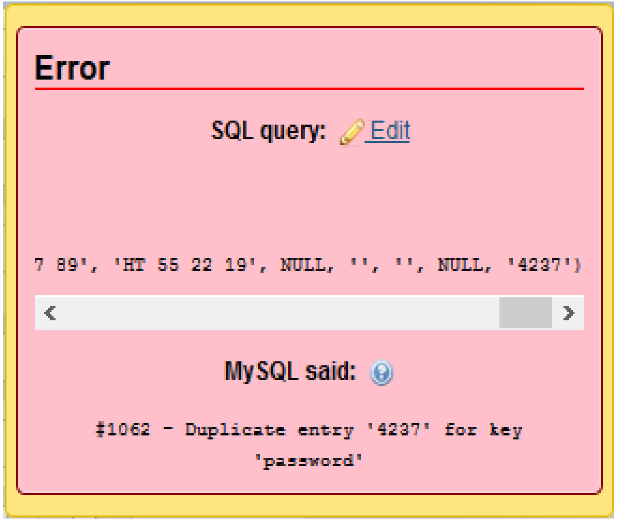
MySQL said:

```
#1062 - Duplicate entry 'REP' for key 'c_name'
```

This error message appears when the unique constraint has been violated on c_name in the company table and this is the actual outcome of the test.

Testing table 2

To test the unique constraints and foreign keys the data for the unique will be data that already in the table and for the foreign keys no data will be inputted. The expected outcomes of these teach is error messages for the violation of the unique constraints and foreign keys.



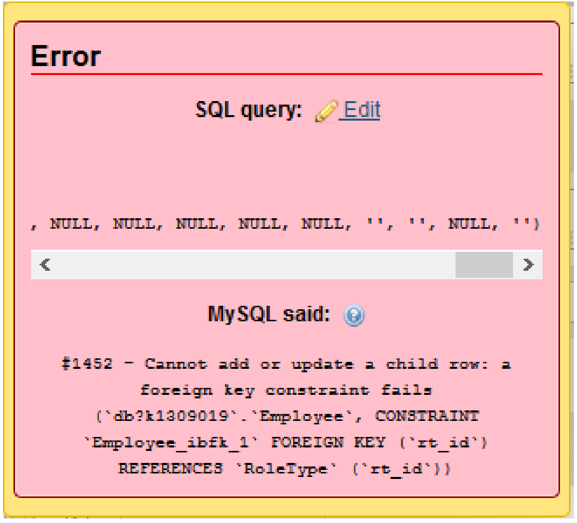
Error

SQL query: Edit

```
7 89', 'HT 55 22 19', NULL, '', '', NULL, '4237')
```

MySQL said:

```
#1062 - Duplicate entry '4237' for key
'password'
```



Error

SQL query: Edit

```
, NULL, NULL, NULL, NULL, NULL, '', '', NULL, '')
```

MySQL said:

```
#1452 - Cannot add or update a child row: a
foreign key constraint fails
(`db?k1309019`.`Employee`, CONSTRAINT
`Employee_ibfk_1` FOREIGN KEY (`rt_id`)
REFERENCES `RoleType` (`rt_id`))
```

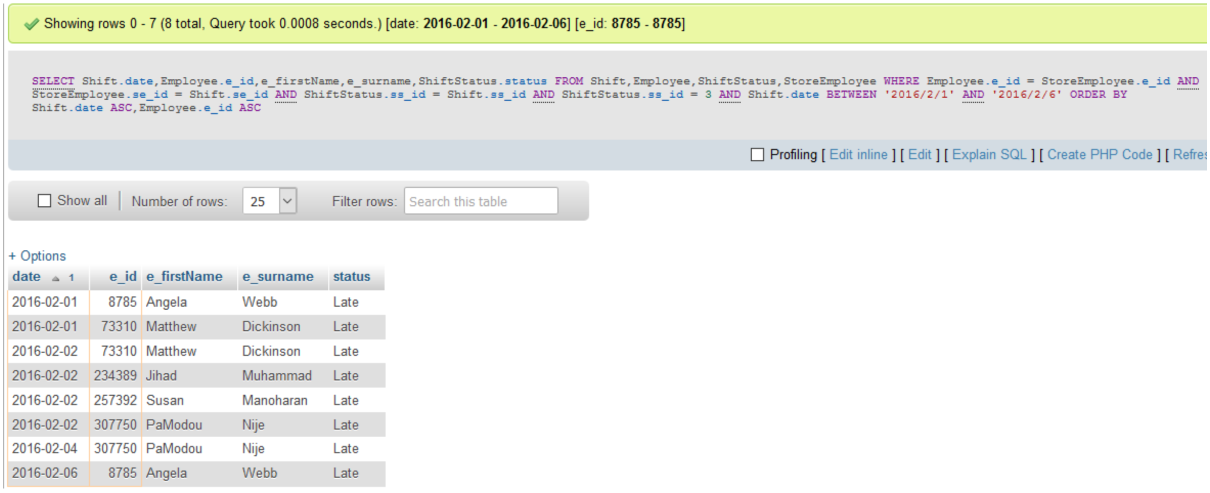
These error messages are the actual outcome of the tests that were performed to test the integrity of the unique constraints and foreign keys.

5.5.4 Queries & Testing

This query will be used in a function to request all the employees that were late between two dates.

```
SELECT Employee.e_id,e_surname,e_firstname,Shift.date
FROM Employee,Shift,ShiftStatus,StoreEmployee
WHERE Employee.e_id = StoreEmployee.e_id AND
StoreEmployee.se_id = Shift.se_id AND
ShiftStatus.ss_id = Shift.ss_id AND
```

ShiftStatus.ssId = "3" AND
Shift.date BETWEEN "2016/2/8" AND "2016/2/14"
ORDER BY Employee.e_id



✓ Showing rows 0 - 7 (8 total, Query took 0.0008 seconds.) [date: **2016-02-01** - **2016-02-06**] [e_id: **8785** - **8785**]

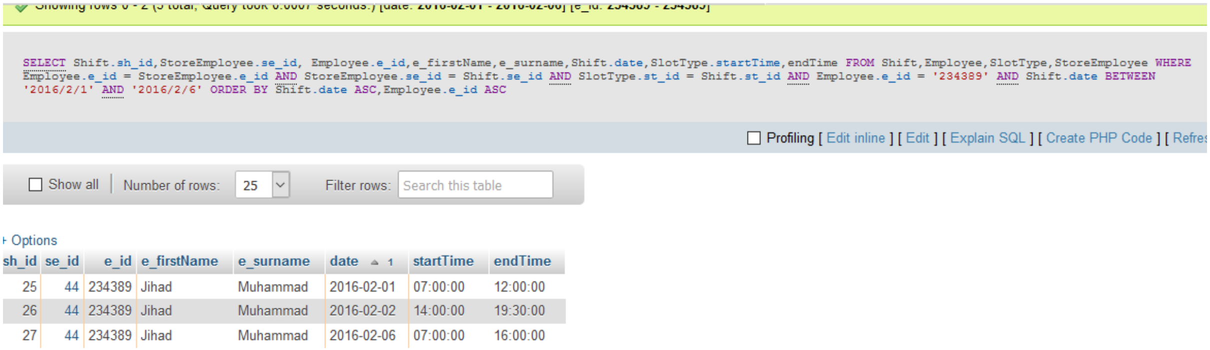
```
SELECT Shift.date,Employee.e_id,e_firstName,e_surname,ShiftStatus.status FROM Shift,Employee,ShiftStatus,StoreEmployee WHERE Employee.e_id = StoreEmployee.e_id AND
StoreEmployee.se_id = Shift.se_id AND ShiftStatus.ss_id = Shift.ss_id AND ShiftStatus.ss_id = 3 AND Shift.date BETWEEN '2016/2/1' AND '2016/2/6' ORDER BY
Shift.date ASC,Employee.e_id ASC
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP Code] [Refres
□ Show all | Number of rows: 25 Filter rows: Search this table
+ Options

date ▲ 1	e_id	e_firstName	e_surname	status
2016-02-01	8785	Angela	Webb	Late
2016-02-01	73310	Matthew	Dickinson	Late
2016-02-02	73310	Matthew	Dickinson	Late
2016-02-02	234389	Jihad	Muhammad	Late
2016-02-02	257392	Susan	Manoharan	Late
2016-02-02	307750	PaModou	Nije	Late
2016-02-04	307750	PaModou	Nije	Late
2016-02-06	8785	Angela	Webb	Late

This query will be used in a function to request the shifts of an employee's between two dates.

SELECTS Shift.date, Employee.e_id, e_firstName, e_surname, SlotType.startTime, endTime
FROM Employee, Shift, SlotType, StoreEmployee
WHERE Employee.e_id = StoreEmployee.e_id AND StoreEmployee.se_id = Shift.se_id
AND SlotType.st_id = Shift.st_id AND Shift.date BETWEEN '2016/2/22' AND '2016/2/28'
ORDER BY Shift.date ASC, Employee.e_id ASC



✓ Showing rows 0 - 2 (3 total, Query took 0.0007 seconds.) [date: 2016-02-01 - 2016-02-06] [e_id: 234389 - 234389]

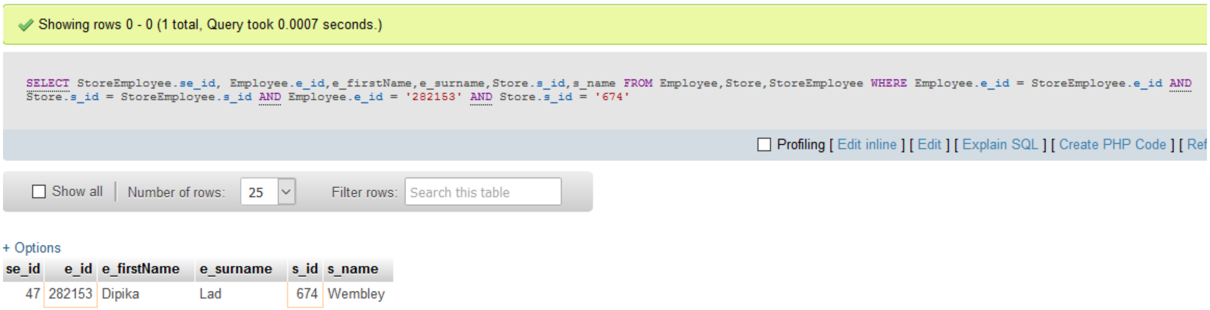
```
SELECT Shift.sh_id,StoreEmployee.se_id, Employee.e_id,e_firstName,e_surname,Shift.date,SlotType.startTime,endTime FROM Shift,Employee,SlotType,StoreEmployee WHERE
Employee.e_id = StoreEmployee.e_id AND StoreEmployee.se_id = Shift.se_id AND SlotType.st_id = Shift.st_id AND Employee.e_id = '234389' AND Shift.date BETWEEN
'2016/2/1' AND '2016/2/6' ORDER BY Shift.date ASC,Employee.e_id ASC
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP Code] [Refres
□ Show all | Number of rows: 25 Filter rows: Search this table
+ Options

sh_id	se_id	e_id	e_firstName	e_surname	date ▲ 1	startTime	endTime
25	44	234389	Jihad	Muhammad	2016-02-01	07:00:00	12:00:00
26	44	234389	Jihad	Muhammad	2016-02-02	14:00:00	19:30:00
27	44	234389	Jihad	Muhammad	2016-02-06	07:00:00	16:00:00

This query will be used in a function to request an employee's store employee ID from the database to be used as an insert value in the holiday request insert query.

SELECT DISTINCT StoreEmployee.se_id
FROM StoreEmployee, Employee, Store
WHERE Employee.e_id = StoreEmployee.e_id AND Store.s_id = StoreEmployee.s_id AND
Employee.e_id = "282153" and Store.s_id = "0674"



✓ Showing rows 0 - 0 (1 total, Query took 0.0007 seconds.)

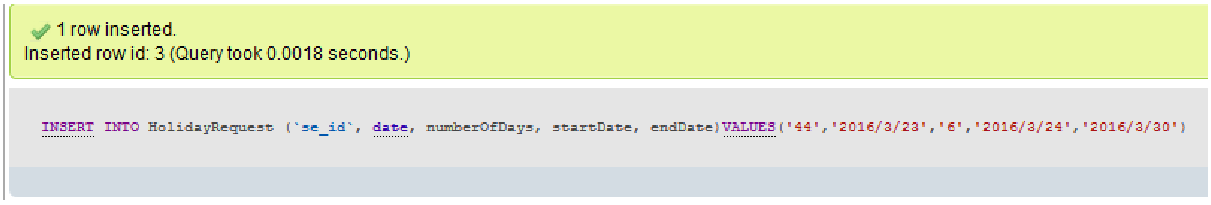
```
SELECT StoreEmployee.se_id, Employee.e_id,e_firstName,e_surname,Store.s_id,s_name FROM Employee,Store,StoreEmployee WHERE Employee.e_id = StoreEmployee.e_id AND
Store.s_id = StoreEmployee.s_id AND Employee.e_id = '282153' AND Store.s_id = '674'
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP Code] [Ref
□ Show all
Number of rows: 25
Filter rows: Search this table
+ Options

se_id	e_id	e_firstName	e_surname	s_id	s_name
47	282153	Dipika	Lad	674	Wembley

This query will be used in a function to insert information from the holiday request form in the user interface to the database table holiday request.

INSERT INTO HolidayRequest (se_id, date, numberOfDays, startDate, endDate)
Values(‘44’, ‘2016/3/23’, ‘6’, ‘2016/3/24’, ‘2016/3/30’)



✓ 1 row inserted.
Inserted row id: 3 (Query took 0.0018 seconds.)

```
INSERT INTO HolidayRequest (`se_id`, date, numberOfDays, startDate, endDate)VALUES('44','2016/3/23','6','2016/3/24','2016/3/30')
```

Chapter 6: Testing and Evaluation

6.1 Testing Strategies

6.1.1 Black box testing

Black box testing is also known as functional testing. Black box testing as stated by Webopedia.com [18] is used to test the internal workings of an application that are not known to the tester. An advantage of black box testing is that the test is unbiased as neither designer nor tester are dependent on the other. A disadvantage is that the test can be redundant if a test case has been run already by the designer [18].

6.1.2 White box testing

White box testing is also known as glass box, structural, clear box or open box testing. White box test as stated by Webopedia.com [19] is used to test with extensive knowledge of the internal workings of an application to select the test data. White box testing uses the extensive knowledge of programming code to inspect the outputs unlike black box testing [19].

6.2 Testing

Function	Test procedure	Expected Result	Result
Login (Valid)	• Click on Login page • Enter valid username and password • Click on login button	Redirects to homepage and username appears on the navigation bar	FAIL
Login (Invalid)	• Click on Login page • Enter invalid username and password • Click on login button	Error message appears stating incorrect details used and stays on the login page	PASS
Homepage (Not Logged in)	• Enter online rota URL into the browser	Homepage to appear welcoming user to the site and show key dates, announcements and the table headings of today's rota	PASS
Hours	• Click on the hours page	Hours page appears showing the users hours	PASS
Wages	• Click on the wages page	Wages page appears showing the users wages	PASS
Rota	• Click on the rota dropdown • Click on the rota page	Rota page appears showing the users rota for this week and fallowing week and the buttons for holiday request form and availability	PASS
Store Rota	• Click on the rota dropdown • Click on the store rota page	Store rota page appears showing the store's rota for this week and the buttons for editing and printing the rota and availability	PASS
Holiday Request Form (page)	• Click on the rota dropdown • Click on the holiday request form page	Holiday request form page appears showing the form.	PASS
Holiday Request Form (Button)	• Click on the rota dropdown	Holiday request form page appears showing the form.	PASS

	• Click on the rota page • Click on the holiday request form button		
Availability Form (page)	• Click on the rota dropdown • Click on the availability page	Availability page appears showing the form.	PASS
Availability Form (Button)	• Click on the rota dropdown • Click on the rota page • Click on the availability button	Availability page appears showing the form.	PASS
Submitting Availability Form	• Click on the rota dropdown • Click on the availability page • Check the boxes • Click submit	The page is then redirected to the rota page.	FAIL

In chapter 5 some other tests were conducted on the holiday request form using valid and invalid data and passed. Also test were conducted on the quires that were used in the project to make sure they work correctly and produce the expected results. In chapter 5 a couple of test were conduct on two tables to investigate the data integrity of these tables, which produced the expected results.

Chapter 7: Critical Review

This point of the project report shall conduct a review of the critical issues that were handled during the design and implementation of the online rota application. Additionally this review will discuss the functionalities that have yet to be implemented into the application. Also there will be a discussion of what lessons that have been learnt during the course of this project. In addition this review will discuss any potential development that could be carried out on the functionality and use of the application.

7.1 Design and Implementation Issues

The biggest issue that I faced during the implementation phase of this project was the use of PHP script language as I have not had much experience with it. Due to this, it created a number of challenges that meant I had to spend a lot of time trying to resolve them. Unfortunately as the end of the project drew nearer I wasn't able to resolve some of the challenges that occurred. An aspect of PHP that I was able to learn very quickly was the creation of functions that used PDO to access the database.

An issue that occurred was the inserting of an employee's availability into the database as I used checkboxes that I wanted to input into the database with an employee's payroll and the current date. I researched different ways that I could potentially use to code the inputting of data into the insert function. I tried some of the examples that I had found but none of them worked in the way I wanted it to and as result it produced fatal errors in the PHP code.

Another issue that occurred during the implementation phase was the creation of the store rota table. Initial I had not thought of how difficult it would be to code the table using PHP. I knew that data that was being retrieved from the database was correct but the issue was using the PHP language to display the information in the table. I started by using only one function to retrieve the data from the database then try to display it but unfortunately that did not work out to be effective and created a lot of errors that could not be resolved with my knowledge of PHP. I then decided to look online for any examples that could be used to try and create the table. I came across a few examples that looked promising to help me create the table. I tried to replicate the code but changed it the way I thought it would work, however that to prove unsuccessful. I then decided to split the function I was using into three separate functions. The first function retrieved the dates of a single week from the shift table in the database and used them as headings in the HTML table. The second function retrieves the employees that are working between the two dates from the database and was displayed in the HTML table. The final function retrieves the shifts for each employee. By using this final function with the combination of if statements and for loops I was able to get the table to display the information in a similar to how I wanted it to be shown but was not able to then refine it further due to the time constraints.

7.2 Missing Functionality

One of the missing functionality that cause some issues was the login. I wrote code that was similar to code I had used in a different project and tested it but it wasn't logging the user into the application. There was any error that occurred so I had no idea as to why the problem was happening and in the end I could not fix it.

Another missing functionality was getting the retrieval of employee's availability from the database as I could not figure out how to check the box of the availability that was currently sorted in the database. Due to this, the edit availability functionality could not be completed.

7.3 Future Development

Future development for this project will begin with resolving the missing functionalities that were discussed in section 7.2. The next stage of the future development process would have to be fixing the issues that were discussed in section 7.1 as they have some impact on the application that need to resolve promptly. An area where development should go into next is the implementation of employee’s KPI earnings as it can be incorporated into the wages page.

7.4 Lesson Learned

From doing this project I have learnt that in theory certain ideas may seem simple but when it comes to implementing they create challenges that are not easy to solve with very basic knowledge. I have also learnt that I should have spent more time trying to get as much of the application implemented as possible and considered help earlier in project instead of nearer to the end of the project. Therefore I felt that better time management would have been needed due to the fact that I had other modules that also needed my attention during the semesters.

From doing this project I have learnt that any programming problem can be solved when thought through logically. Also that most problems are fixed through trial and error which I should use more to try and achieve the outcomes I desire.

During the course of this project I have gained a greater understanding of PHP scripting language. Having now gained this better understanding of PHP I hope to implement this knowledge in future projects I will have to undertake.

If I was to carry out this project again I would ensure that a greater amount of time be allocated to the implementation stage of the project so that greater functionality can be gained and thereby being able to achieve most of the requirements set out by the users.

References

1. GOV.UK, (2015). *Data Protection*. [Online] Available at: https://www.gov.uk/data-protection/the-data-protection-act [Accessed on 30/10/2015].
2. SearchStorage.co.uk, (2015). *What is U.K. Data Protection Act 1998 (DPA 1998)? - Definition from WhatIs.com*. [Online] Available at: http://searchstorage.techtarget.co.uk/definition/Data-Protection-Act-1998 [Accessed on 30/10/2015].
3. Computerweekly.com, (2015). *Essential guide: What the EU Data Protection Regulation changes mean to you*. [Online] Available at: http://www.computerweekly.com/guides/Essential-guide-What-the-EU-Data-Protection-Regulation-changes-mean-to-you [Accessed on 06/11/2015].
4. SerchSecurity.co.uk (2015). *What is EU Data Protection Directive (Directive 95/46/EC)? - Definition from WhatIs.com*. [Online] Available at: http://searchsecurity.techtarget.co.uk/definition/EU-Data-Protection-Directive [Accessed on: 06/11/2015].
5. Ec.europa.eu, (2015). *Protection of personal data - European Commission*. [Online] Available at: http://ec.europa.eu/justice/data-protection/ [Accessed on 07/11/2015].
6. Wikipedia, the free encyclopaedia (2015). *Test data generation*. [Online] Available at: https://en.wikipedia.org/wiki/Test_data_generation [Accessed on: 07/11/2015]
7. C2.com, (2015). *Test Data Generator*. [online] Available at: http://c2.com/cgi/wiki?TestDataGenerator [Accessed on 10 Nov. 2015].
8. Lewis, W., Dobbs, D. and Veerapillai, G. (2009). *Software testing and continuous quality improvement*. Boca Raton, Fla.: CRC Press, pages 400 to 405.
9. Techopedia.com, (2015). *What is a Test Data Generator? - Definition from Techopedia*. [online] Available at: https://www.techopedia.com/definition/30405/test-data-generator [Accessed on 10 Nov. 2015].
10. Webopedia.com, (2015). *What is Data? A Webopedia Definition*. [online] Available at: http://www.webopedia.com/TERM/D/data.html [Accessed on 11 Nov. 2015].
11. Google.co.uk, (2015). *How data is protected - Google Search*. [online] Available at: https://www.google.co.uk/search?q=how+data+is+protected&ie=utf-8&oe=utf-8&gws_rd=cr&ei=Pc5DVvSSCcyRaIvTp8gK [Accessed on 11 Nov. 2015].
12. Cimt.plymouth.ac.uk, (2015). *Unit 11 Section 1 : Types of Data*. [online] Available at: http://www.cimt.plymouth.ac.uk/projects/mepres/book7/bk7i11/bk7_11i1.htm [Accessed on 11 Nov. 2015].
13. Wikipedia, the free encyclopaedia. (2015). *Stakeholder analysis*. [online] Available at: https://en.wikipedia.org/wiki/Stakeholder_analysis. [Accessed 20 Nov. 2015].
14. SearchSoftwareQuality, (2015). *What is use case? - Definition from WhatIs.com*. [online] Available at: http://searchsoftwarequality.techtarget.com/definition/use-case [Accessed 27 Nov. 2015].
15. WhatIs.com. (2016). *What is use case diagram (UML use case diagram)? - Definition from WhatIs.com*. [online] Available at: http://whatis.techtarget.com/definition/use-case-diagram [Accessed 11 Apr. 2016].
16. Project Smart. (2016). *MoSCoW Method*. [online] Available at: https://www.projectsmart.co.uk/moscow-method.php [Accessed 11 Apr. 2016].
17.
18. Webopedia.com, (2016). *What is Black Box Testing? Webopedia*. [online] Available at: http://www.webopedia.com/TERM/B/Black_Box_Testing.html [Accessed 4 Mar. 2016].

19. Webopedia.com, (2016). *What is White Box Testing? Webopedia*. [online] Available at: http://www.webopedia.com/TERM/W/White_Box_Testing.html [Accessed 4 Mar. 2016].