EMPLOYEE MANAGEMENT SYSTEMS

By

GLORIA PADUA

A thesis submitted in partial fulfilment of the
requirements for the degree of

BSc Computer Science (Hons)

University of the Western Cape

2012

University of the Western Cape

Abstract

EMPLOYEE MANAGEMENT SYSTEM

By

Padua Gloria

Supervisor: Ms I Venter

Our project is based on problems faced by a manager of an automobile workshop in Rwanda. The manager faces problems such as:

- Knowing how many employees showed up to work on a specific day.
- Knowing who worked on a specific car.
- How long they took to complete their task.
- Which employee is free to take up a new task?

In order for these problems to be solved, all employees will be registered on database the time they clocked into work, which cars they are working on and how long they took to work on it will be recorded. The manager will be able to allocate jobs to the employees.
The system is web enabled and it will be possible to access from anywhere.

Contents

LIST OF FIGURES

LIST OF TABLES

Number *Page*

ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to my advisor and supervisor Professor. Isabel Venter for the continuous support of my honours study research and project, for her patience, motivation, enthusiasm, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better supervisor and mentor for my Honours study. I would also like to thank my parents, brothers and sisters for having given me their undisputable support throughout, as always, for which my mere expression of thanks likewise does not suffice. Also I thank my fellow postgraduate students University Of the Western cape In particular, I am grateful to Ms Ghislaine Ngangom for enlightening me and promoting a stimulating and welcoming academic experience for me.

GLOSSARY

PHP (Hypertext pre-processor): General purpose scripting language that is used for web development and is embedded into HTML.

UML (Unified modelling language): standardized general-purpose modelling language in the field of object-oriented software engineering

WWW (World Wide Web): a system of interlinked hypertext documents accessed via the Internet.

HTTP (Hypertext transfer protocol): Set of rules for transferring files on World Wide Web.

MySQL (My structured Query language): relational database management system that runs as a server providing multi-user access to number of databases.

THE USER REQUIREMENTS DOCUMENT

INTRODUCTION

Employees are the backbone of any company, management of employee performance plays a major role in deciding the success of the organization. The workshop is situated in Rwanda has a problem in management of employee performance. The current system running in the workshop is paper based. That is the workshop is still using cabinet files to store records of stock and employee information. Useful data is scattered all over the place. In this chapter we shall discuss the solutions to the problems being caused by the current system. We shall try to understand the manager's expectations of the new system we are to develop for him.

DATA COLLECTION

A questionnaire in the form of a survey was used to gather information. Why a survey??The database program proposed for this project is based on an automobile repair shop in Rwanda. So similar businesses' in South Africa and were surveyed to find out how they managed their employees. I asked them to respond to a few questions. Their responses to the questions are summarised in Appendix A. These were used to compare with the responses from the workshop in Rwanda.

PROBLEM DOMAIN

The problem domain is the computerisation of an employee management system that can be accessed online by the administrator and the stakeholders such as the manager.

THE CURRENT SYSTEM

People have different personalities and work ethics .So in order to manage their work efficiently and fairly, there has to be a system in place to allocate tasks to different workers. Currently a manual system is used in the Rwandan business that will provide most of the requirements for this project. The system used in Rwanda is based on "TRUST" the employer trusts the employees. Although noble the manager has little it control over his business. In a manual system data is stored in a cabinet. Files are thus often misplaced or lost. And at times is difficult to find relevant files. Records for stocks are also not always filed correctly and thus information is not centralised and not easily accessible.

EXPECTATIONS FOR THE NEW SYSTEM

The system should be:

- Secure.

- The system should be able to provide a list of the employees, the times they worked, the tasks they have been doing etc.

- The system must be able to list what is currently in stock.

- The system must provide easy access to employees' details (name, id number, employee number, address).

CONCLUSION

This chapter dealt with the user's needs and expectations for the new system. The next chapter deals with requirements analysis that is requirements needed to solve the problem being faced by the workshop in Rwanda. An employee management system will be the best solution for the problem. It will provide easy online access to the employees that are currently at work and what they are working on. Also easy to allocate jobs to the employees that are done with what they were doing. The system will provide quick and reliable access to the running of the business saving the manager a whole lot of time and money.

REQUIREMENTS ANALYSIS DOCUMENT

INTRODUCTION

In the previous chapter the user requirements were collected. In this chapter the requirements will be analysed. All the software development tools that will be used to implement the system will be identified.

DESIGNER'S INTERPRETATION OF THE USER

The manager and stakeholders' would like to have **remote access** to the employee system application. This means that it has to have a web-based interface but should also provide an interface for a mobile device. The data will be stored in a database.

HIGH LEVEL CONSTITUENTS PARTS

The system will consist of two constituents. That is Software management and database management. Database management will be managed by the manager/administrator. Here are features or characteristics of these two constituents:

DATABASE AND SOFTWARE MANAGEMENT:

- Will be accessible on different devices or platforms.
- Will be to add or delete from database.
- Will enable editing of data on the database.
- Will enable retrieval of data from the database.
- Will enable searching through the data and make reports from his findings.

Below is figure 1.1 which depicts how the users will interact with the system:

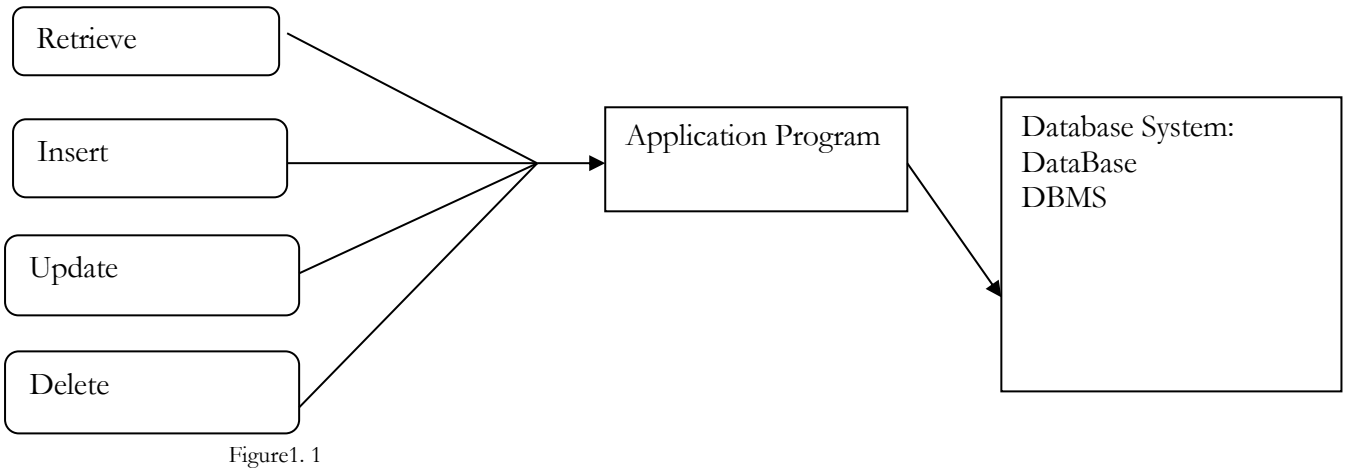

Figure1. 1

EXISTING SOLUTIONS

- There similar systems to employee management system. For example:
- Macros
- Pastel HR management.
- Halogen software (Strategic talent management).

ALTERNATIVE TECHNICAL

With the legacy system in mind, we shall computerize the whole system it will be flexible.

BEST SOLUTION

The best solution will be an online database which will be flexible for the manager to access wherever he is. He can access from his mobile device (Laptop or cell phone). It can achieved by making a web- based software. For better planning we shall break down the work load in form of model and technology to be used.

MODEL

In order to put into consideration all the needs of the user, we will use Unified modelling language (UML) to model the system. UML will act as a blueprint to the whole system we are going to implement. UML will help me to break down the scope of the whole system so that it's flexible to implement the system.

TECHNOLOGY

Since its web-based software, we decided to use:

- Apache (webserver)
- PHP
- MySQL

WHY??

This because they are:

- (Web slave/advantages of apache, 2011) Flexible.
- Its open source so it's cheap.
- It is efficient and easily managed.

We shall be using the following software.

APACHE:

Free available web server. This allows **MySQL** and **PHP** to run on it.

PHP:

Hypertext Pre-processor. It is a widely used general- purpose scripting language that is especially suited for web development and can be embedded into HTML. Above all its free that is open source.

MySQL:

It's a relational database management system (**RDBMS**) that runs as a server providing multi- user access to a number of databases.

PHPMyAdmin:

It is open source software written in PHP with the intention of handling the administration of MySQL over the World Wide Web. It also supports wide range of MySQL operations.

JAVASCRIPT:

It is a scripting language that used to make web pages interactive.

HTTP:

Hyper Transfer Protocol ***(HTTP)*** is a set of rules for transferring files (text, graphic, images, sound, video and other multimedia files) on the World Wide Web (**WWW**).

SYSTEM TESTING:

WHITE BOX TESTING

This is a verification technique for software developers use to examine if their code works as expected. We will use this method to detect errors during software development (williams). White box which is also known as clear box helps in optimizing of my code. Using the aspect of unit testing we shall be go through each and every line of code. This makes it easy for us to remove extra lines of code that can bring about hidden errors.

USUABILITY TESTING

This is a technique used to evaluate the quality of the software. This is where random users are used to test the product using different data and their response to the system is put into consideration.

BLACK BOX TESTING:

It is also known as functional testing. This technique is used to test the complete system. Black box testing ignores all the whole system components and focuses only on the outputs generated for a particular input. Different data will be used for each function to check if the right output is gotten. This is to check the behaviour of the system as per different input data.

CONCLUSION

Employee management system (**EMS**) is going to be implemented using **PHP, MySQL**, **Apache** and **PHPMyAdmin** which are all open source applications. They are all a solution to web based system.

WHY??

After examining all the possible solutions, we decided on a web based system to address this problem.

USER INTERFACE SPECIFICATION

INTRODUCTION

The previous chapters dealt with the user requirements specification. This chapter documents the current user interface and the various elements needed to fulfil the user requirements. The images of the user interface pages are included to demonstrate the application's look and feel.

DESCRIPTION OF COMPLETE USER INTERFACE SPECIFICATION:

The Login page:

The main screen will contain the login page (See Figure 1.2). The login page consists of Workshop link, contact link, Employee id, password and the login button. It consists of two text boxes that are employee id and password. There is also a submit button with the value login.

The contact link leads to a webpage that contains all the contact details of the administrators. The workshop link also leads to a webpage that gives a brief history of the workshop.

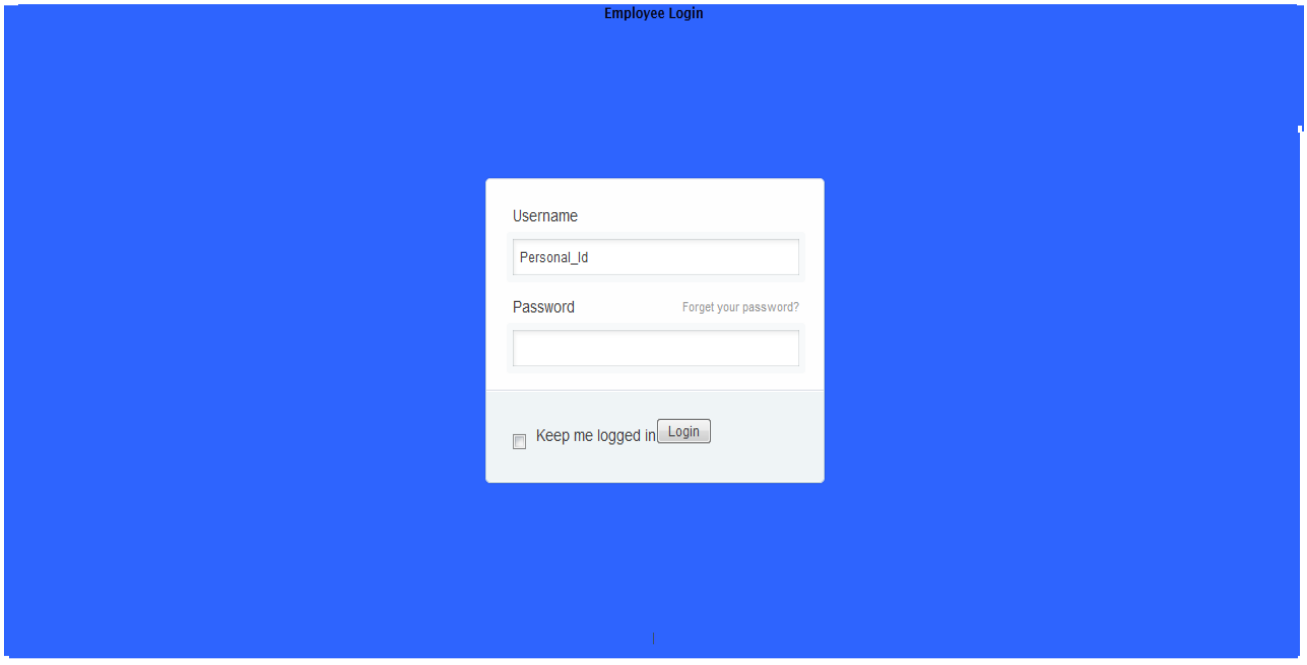

Figure1. 2

The application page:

There will be two kinds of users. They are employees and administrators. Once an employee is logged in, they are only able to view but not update anything in the database. They will be view links for:

- Work hours
- Contact details of fellow workmates and their own
- Holiday days they have left.

Figure 1.2 shows the user application page

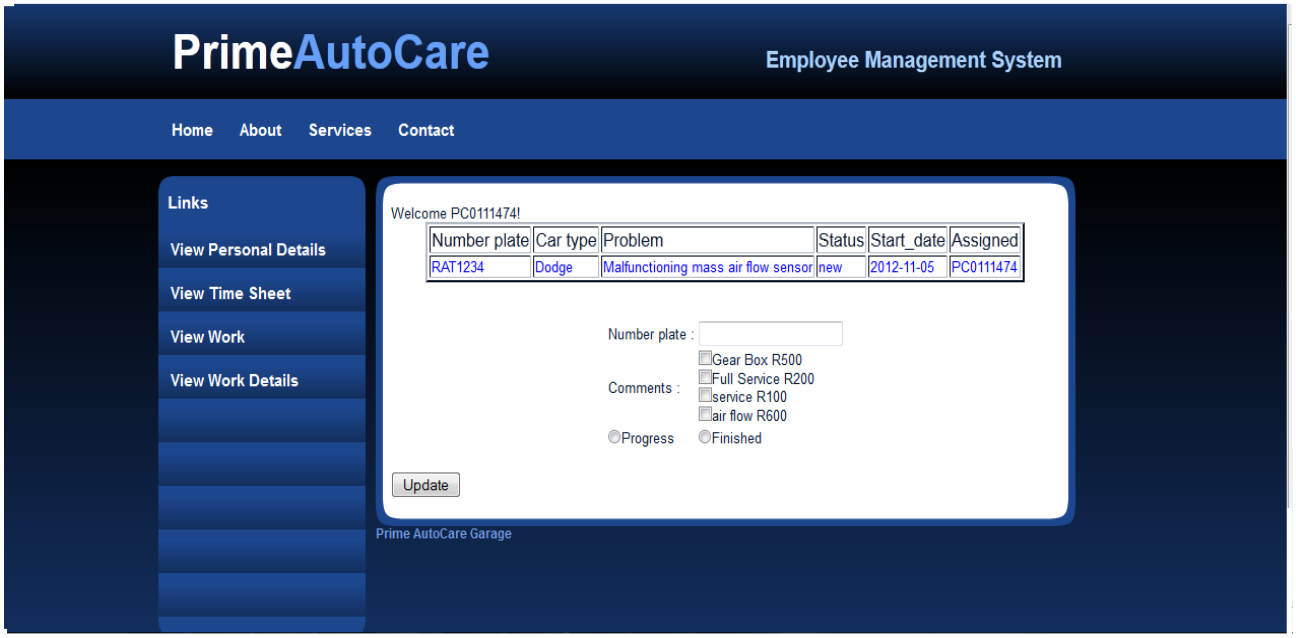

Figure1. 3

The feedback page:

The page below is shown as feedback after clicking on the work hour's icon. This shows all the work hours of the employees. This however doesn't allow the employees' to change anything.

The administrator's application page on the other hand has very many different options mainly:

- **Salary link:** This deals with all the aspects concerned with salary of each individual employee. That is deducting taxes and insurances on their monthly allowances.

- **Employee link:** This deals with adding, deleting and updating all the employees' information.

The figure 1.4 shows how the administrator application page looks like:

Figure1. 4

HOW IT USER INTERFACE BEHAVES:

When users input their username and password these must be authenticated before access is granted to the various applications. An error message is issued when the user enters the wrong username or password. The error message prompts the user that they have entered the wrong username or password. In the case of correct password and username then the user is gains access to the application that they are privileged to use.

As shown in figure 1.3 the employee will only be able to view and edit his personal details and that of his /her emergency contact. On the other hand the administrator is privileged to use all the functionality of the application.

Figure 1.5 shows the use case diagram of the system's functionalities:

USE CASE DIAGRAM

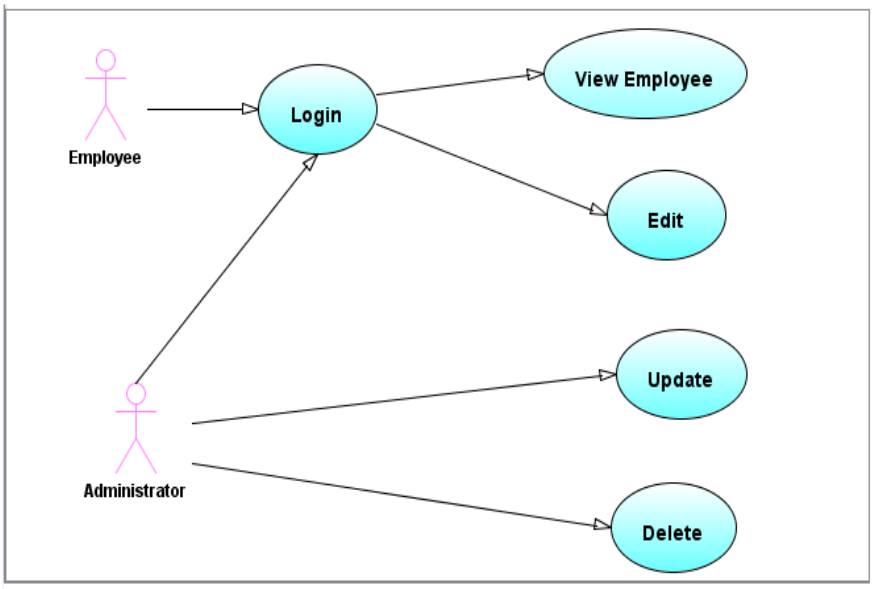

Figure1. 5

HOW USER INTERACTS WITH THE INTERFACE:

The activity diagrams (see figure 1.6) shows a summary of all the user activities. This all starts with logging into the system. The user enters his or her employee id and password. The input is authenticated by the system and when it's been successfully authenticated then the privileges are checked. The privileges are checked to ensure that the right main page is shown for each user. This is because the administrator and the employees have different privileges to the application system. For example an employee interacts with a system such as when he/ she logs in successful is selects a link to view the worksheet. The system displays the particular employee's work sheet. There is also an option for downloading the work sheet. The employee selects this option; the system returns a download of the spread sheet of the work sheet:

ACTIVITY DIGRAM : (EMPLOYEES)

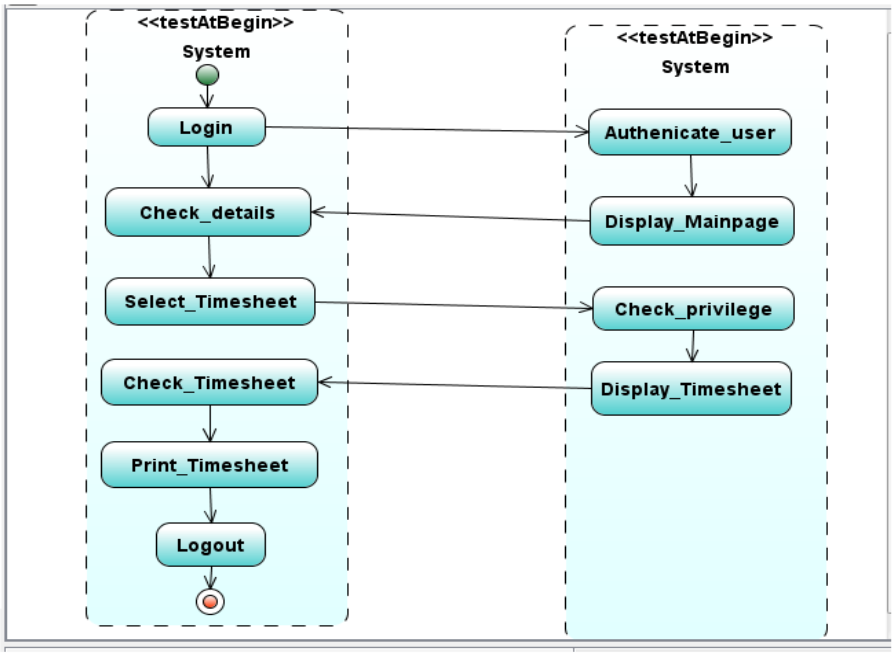

Figure1. 6

In the case of the administrator, he/she interacts with the system such as when the administrator is successfully logged in. The activity diagram (Figure 1.7) shows how an administrator can add data into the database:

ACTIVITY DIAGRAM (ADMINISTRATOR)

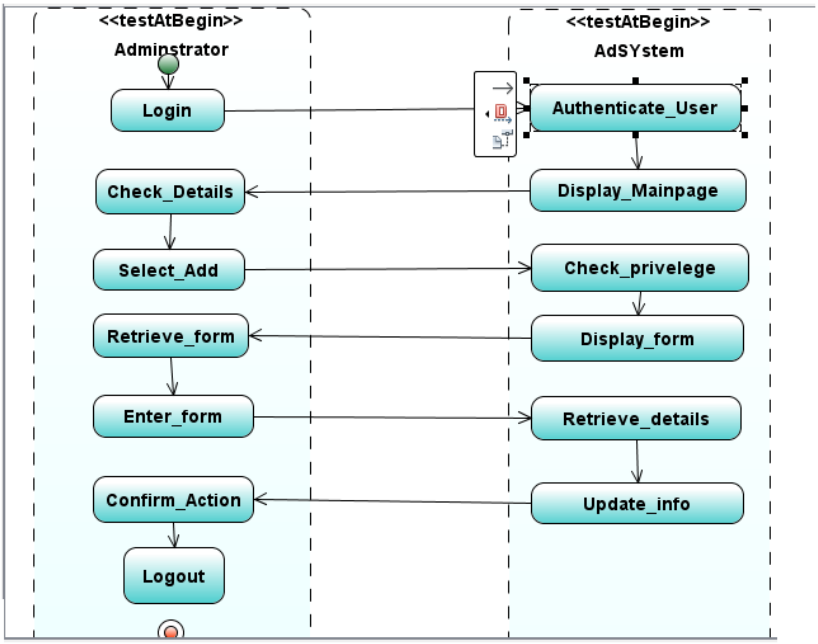

Figure1. 7

CONCLUSION:

In this chapter we described the user interface specification: the application pages, the main page, the login page and all the other functionality pages. The activity diagrams shown in this chapter describe how users interact with the system. In the next chapter the objects that would be needed to implement the user interface will be discussed.

HIGH LEVEL DESIGN

INTRODUCTION

This chapter concentrates on the object oriented analysis (OOA) or high level design of the problem. In this chapter every object is described and documented and the data dictionary provides the detail of all the objects. The class diagrams will show how the subsystems interact with each other. Furthermore, the objects, attributes and methods will be identified.

Data dictionary:

A data dictionary is a collection of descriptions of the data objects or items in a data model for the benefit of programmers and others who need to refer to them. The dictionary gives a brief description of the objects that will be needed to form the online employee management system.

Detailed breakdown of the technical solution

Employee details Object (Table 1.1):

This contains employees' personal details and qualification. This object contains record for only one employee.

NAME	MEANING	VALUE
Personal_id	Passport or id number and also acts as the employee_id used for login	20 characters
First_name	Their given name	20 characters
Last_name	Family name	20 characters
Date of birth	Their birthday/ age	10 numbers
Cell phone	Cell phone number for the employee	10 numbers
Address	Home address of employee	50characters
Qualification	Level of qualification the employee has reached	100 characters

Email	Email address of the employee	20 characters
Password	Password given to the employee in order to login to the system.	20 characters

Table1. 1

Administrator's details (Table 1.2):

This contains administrator's personal information:

NAME	MEANING	VALUE
Adminstrator_id	Passport or id number and also acts as the Adminstrator_id used for login	20 characters
First_name	Their given name	20 characters
Last_name	Family name	20 characters
Password	Password given to the adminstrator in order to login to the system	20 characters

Table1. 2

Salary Object (Table 1.3):

This contains the employee's salary details which have been calculate in respect to how many cars they have worked on that month:

NAME	MEANING	VALUE
Personal_id	Passport or id number and also acts as the employee_id used for login	20 characters
Salary	Amount of salary the employee is entitled to this month	10 Numbers

Currency	The currency the salary is calculated in	20 characters
Tax	The taxes deducted from the salary.	10 Numbers
Insurance	Employee insurance and pension fund	10 Numbers

Table1. 3

Time sheet Object (Table 1.4):

This contains the time they took a break and holiday break for the employees.

NAME	MEANING	VALUE
Break id	The break taken	20 characters
Holiday id	The holiday taken	20 characters
Time	Timestamp	Time in YY MM DD:H.M.S

Table1. 4

Work Sheet objects (Table 1.5):

This contains the worksheet and the time they started to work, the car they working on and the status. This also contains the car information like the number plate, make of the car and the problem car has:

NAME	MEANING	VALUE
Number_plate	Registered number plates of car that has come to the workshop	20 characters
Make of the car	The type/make of the car.	20 characters
Problem	The reason why the car came to the workshop	100 characters
Entry_date	Date and time it entered the	Time in YY MM DD:H.M.S

	workshop	
End_date	Date and time the services on the car were finished	Time in YY MM DD:H.M.S
Status	The work progress of the car.	20 characters
Job_assigned	The employee who has been assigned a specific car	20 characters
Price	The total price of the services performed on a specific car	10 Number

Table1. 5

Detailed interaction between subsystems:

The ERD (See figure 1.8) gives a brief description of how the tables would be linked together in the system in order to make data retrieval and update easy for the user

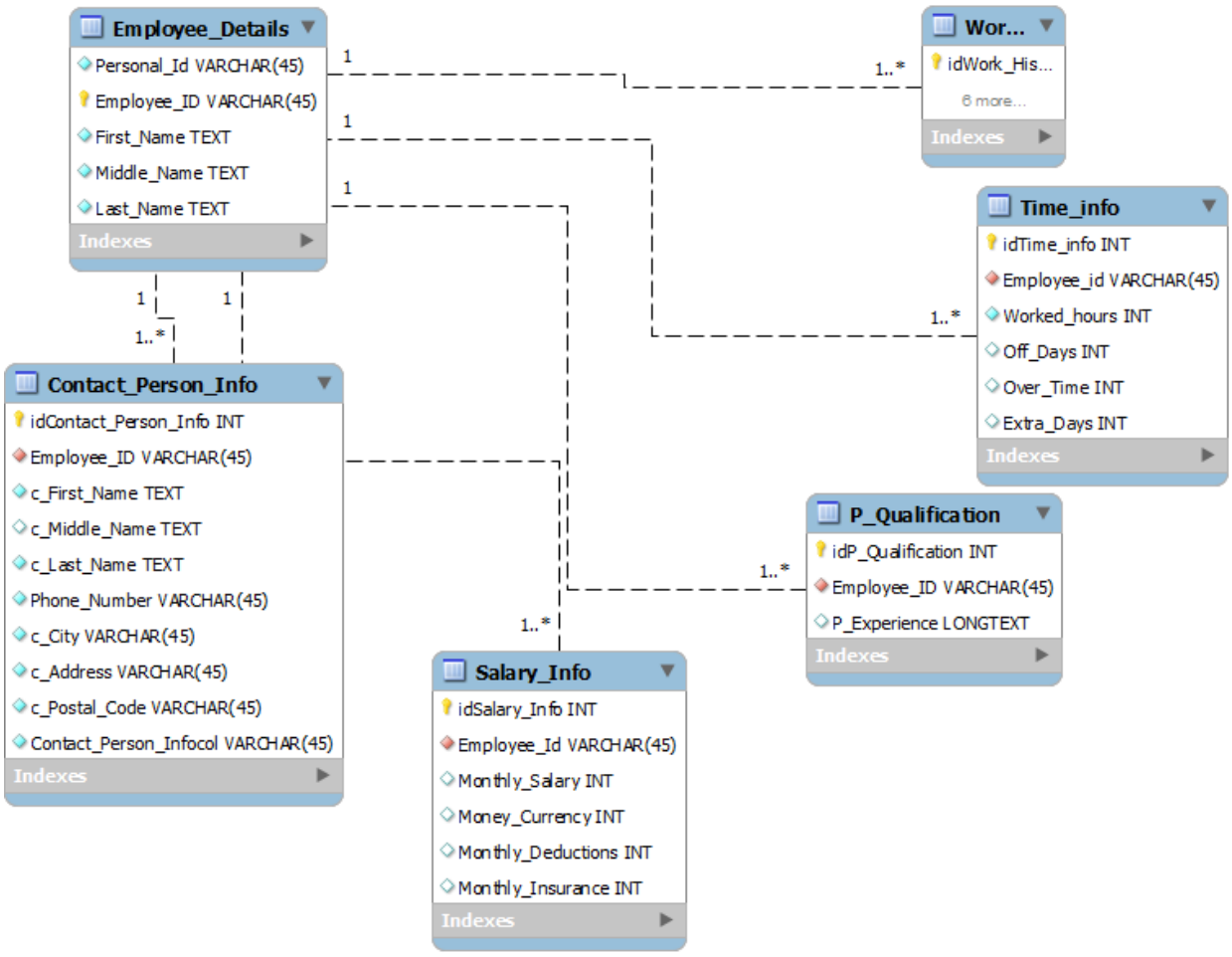

Figure1. 8

Conclusion:

The data dictionary described the attributes and the data type. In this chapter, each class and its attributes are explained in detail using a data dictionary. The ERD explains the relationship between the different classes and attributes that are needed to implement the user interface. In the next chapter the object oriented design, which is also known as low level design will be discussed.

OBJECT ORIENTED DESIGN (OOD) OR LOW LEVEL DESIGN (LLD)

INTRODUCTION

In the previous chapter, each object was described and documented in terms of a data dictionary. In this chapter, the Object Oriented Design (OOD) will be described.The data types for the attributes, the algorithms and implementation particulars of each classetc. will be explained. The OOD will present all the classes mentioned in the OOA in terms of pseudo-code. The state diagram and sequence diagram will also be explained in this chapter.

The inner details of class attributes (data types) and methods (functions):

The ERD (See figure 1.8) defined in chapter 4 shows all classes/tables that will be used inside the system. Also data types were well-defined in the data dictionary and functions are pronounced within the ERD.

State Diagram:

The state diagram (See figure 1.9) portrays the dynamic performance of login and application functions of the system. Both the employee and the administrator log in using the same application page. So both the users enter the username and password, and then the system authenticates the entries. If valid the user is recognized by the system or else the user is rejected. The user then selects an option and if he is privileged

to use the option then the system confirms it. The state diagram below explains itall:

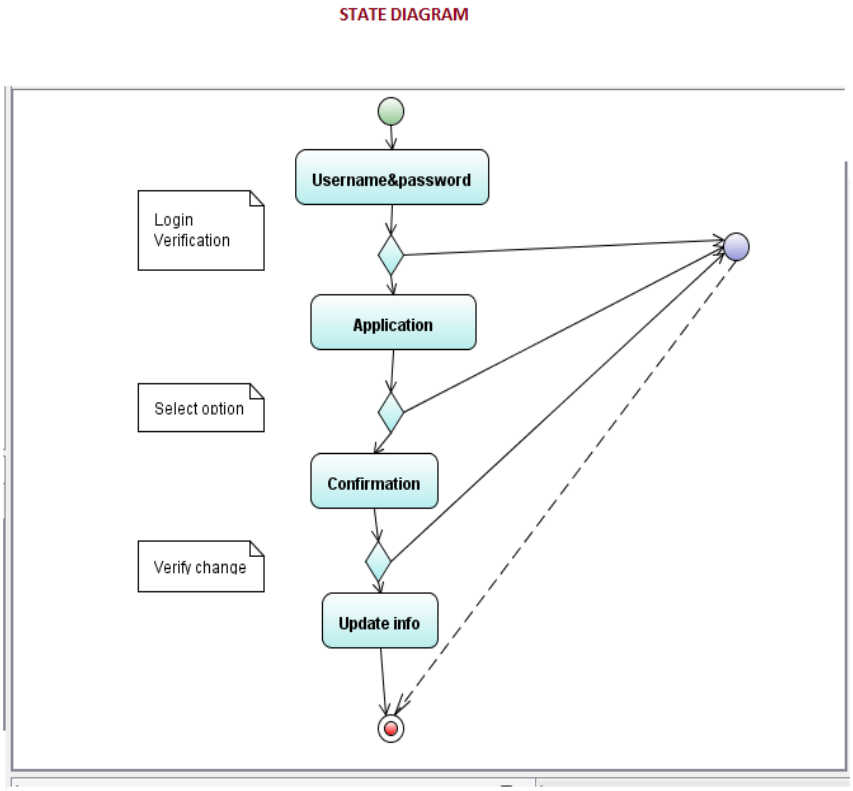

Figure1. 9

The event diagram or sequence diagram:

(See figure 1.10) Demonstrates the sequence of activities where the system is in operation.

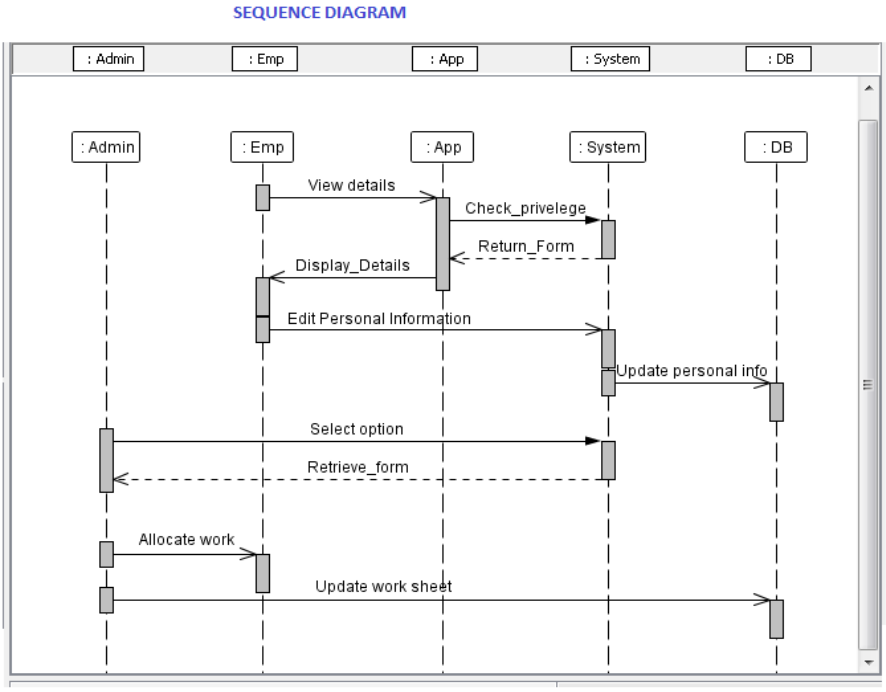

Figure1. 10

Pseudo code:

The login page

GETEmployeeid GETPassword IF (Username == EnteredEmployee&&Password == EnteredPassword)

THEN LoginSuccessful

ELSE LoginFailed.

ENDIF

CONCLUSION:

This chapter undertook the pseudo-code showed an outline of programs, written in a form which is clear and which will be easy to convert into a programming language. It also covered the state diagram and sequence diagram.

CODE DOCUMENTATION

INTRODUCTION

In the previous chapter the data types for the attributes, the algorithms and implementation details of classes were defined. Also, the state and sequence diagrams of the system have been explained. In this chapter a sketch of the entire system that helps to view that all activities performed in the system. The code is fully documented. For each PHP file, we defined MySQL query which helps to retrieve the information from the database according our user requirements. The code also has comments to explain the algorithm.

System Operations

The following sketch (see Figure 1.8) represents all the operations between the users and the system. For every user, the operations are numbered according to the order in which they occur. The number of the operation will be used in the code documentation (the number of the action will be reflected as a comment in the code). The code documentation is provided according to the type of the user. For example the Login code will be given for all users (Employee and Admin)

Login page:

```
<?php

/** NAME: PADUA GLORIA STUDENT NUMBER:2916574 This code recieves a form from the login page and checks whether the user exists and then is logged in otherwise permission isn't

Granted*/

//if(isset($_POST['submit'])){

   $dbHost = "localhost";       //Location Of Database usually its localhost

   $dbUser = "root";            //Database User Name

   $dbPass = "";           //Database Password

   $dbDatabase = "new";    //Database Name


   $db = mysql_connect($dbHost,$dbUser,$dbPass)or die("Error connecting to database.");

   //Connect to the databasse

   mysql_select_db($dbDatabase, $db)or die("Couldn't select the database.");

   $usr = $_GET['username'];

   $pas = $_GET['password'];
```

```
    $sql = mysql_query("SELECT Personal_id,password FROM employee_details");
$user="";
$password="";
while($row = mysql_fetch_array($sql)){
if((strcmp($row['Personal_id'],$usr)==0)&&(strcmp($row['password'],$pas)==0)){
$user=$usr;
$password=$pas;
}
}
if (strcmp($user,"")!=0 && strcmp($password,"")!=0){
setcookie("myuser", $user);
 header("Location: user.php");
}else{
     header("Location: login.php");
    }
?>
```

Work sheet:

This is the code that makes the employee able to pick the work for the day and they able to choose the services they performed and calculate the total price for the services done.

```
/** NAME: PADUA GLORIA
STUDENT NUMBER: 2916574
This code enables the employee able to pick the work for the day
and they able to choose the services they performed and calculate the total price for the services done*/
if (isset($_POST['submit'])){
// Connect to server and select database.
$con = mysql_connect("localhost","root","");
```

```
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }// Connecting to the server


mysql_select_db("new", $con);//choosing a database and connecting to it.
//$usr='PC0111474';
//$usr="new"; //Status of the car
$id=$_POST['categories'];//Number_plate
$price1=0;
$price2=0;
$price3=0;
$price4=0;
if (isset($_POST['gear'])){
$price1=$_POST['gear'];
}
if (isset($_POST['Fservice'])){
$price1=$_POST['Fservice'];
}
if (isset($_POST['service'])){
$price1=$_POST['service'];
}
```

```
if (isset($_POST['airflow'])){

$price1=$_POST['airflow'];

}

$status=$_POST['submit'];//Status

$today = date("Y-m-d H:i:s");//date of today

$total=$price1+$price2+$price3+$price4;

$sql="UPDATE car_info SET Status='$status',End_date='$today',Price='$total'

WHERE Number_plate='$id' ";



if (!mysql_query($sql,$con))

  {

  die('Error: ' . mysql_error());

  }

echo "1 record added";



mysql_close($con);

}



//Pick a car and work on it

if (isset($_POST['choose'])){

// Connect to server and select database.
```

```
$con = mysql_connect("localhost","root","");

if (!$con)

  {

  die('Could not connect: ' . mysql_error());

  }// Connecting to the server



mysql_select_db("new", $con);//choosing a database and connecting to it.

//$usr='PC0111474';

//$usr="new"; //Status of the car

$id=$_POST['box'];



$sql="UPDATE car_info SET Status='In progress'

WHERE Number_plate='$id' AND Status='new' ";



if (!mysql_query($sql,$con))

  {

  die('Error: ' . mysql_error());

  }

echo "1 record added";



mysql_close($con);

}
```

```
?>
```

Break and Holiday time (Employee):

```
<?php

/** NAME: PADUA GLORIA

STUDENT NUMBER:2916574

if (isset($_POST['submit'])){

// Connect to server and select database.

$con = mysql_connect("localhost","root","");

if (!$con)

  {

  die('Could not connect: ' . mysql_error());

  }// Connecting to the server



mysql_select_db("new", $con);//choosing a database and connecting to it.

//$usr='PC0111474';

//$usr="new"; //Status of the car

$id= $_COOKIE["myuser"];



$sql="INSERT INTO Persons (Personal_id, Holiday_type)

VALUES

('$id','$_POST[Hols]')";
```

```
if (!mysql_query($sql,$con))

  {

  die('Error: ' . mysql_error());

  }

echo "1 record added";



mysql_close($con);

}




 ?>
```

View and Edit Personal details (employee):

```
<?php

/** NAME: PADUA GLORIA

STUDENT NUMBER: 2916574

This code enables the employee to view and edit their personal details*/

if (isset($_GET['Edit'])){

$usr= $_COOKIE["myuser"];

$con=mysql_connect("localhost","root","");

if(! $con)

{
```

```
                                                    die('could not connect: '. mysql_error());
}
mysql_select_db("new", $con);
$name="";
$lname="";
$cell="";
$add="";
$dob="";
$quali="";
$gender="";
$email="";

if (isset($_GET['username'])){
$name= $_GET['username'];
}
if (isset($_GET['last'])){
$lname=$_GET['last'];
}
if (isset($_GET['cell'])){
$cell=$_GET['cell'];
}
if (isset($_GET['add'])){
```

```
$add=$_GET['add'];

}

if (isset($_GET['dob'])){

$dob=$_GET['dob'];

}

if (isset($_GET['quali'])){

$quali=$_GET['quali'];

}

if (isset($_GET['gender'])){

$gender=$_GET['gender'];

}

if (isset($_GET['email'])){

$email=$_GET['email'];

}

//Insert Into database

if($name!=""){

$sql="UPDATE employee_details SET First_Name='$name' WHERE Personal_id='$usr' ";

if (!mysql_query($sql,$con))

  {

  die('Error: ' . mysql_error());

  }

}
```

```
if($lname!=""){
$sql="UPDATE employee_details SET Last_name='$lname' WHERE Personal_id='$usr' ";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
}
if($cell!=""){
$sql="UPDATE employee_details SET Cell_phone='$cell' WHERE Personal_id='$usr' ";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
}
if($add!=""){
$sql="UPDATE employee_details SET Address='$add' WHERE Personal_id='$usr' ";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
}
if($dob!=""){
```

```
$sql="UPDATE employee_details SET DOB='$dob' WHERE Personal_id='$usr' ";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
}
if($quali!=""){
$sql="UPDATE employee_details SET Qualification='$quali' WHERE Personal_id='$usr' ";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
}
if($gender!=""){
$sql="UPDATE employee_details SET Gender='$gender' WHERE Personal_id='$usr' ";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
}
if($email!=""){
$sql="UPDATE employee_details SET email='$email' WHERE Personal_id='$usr' ";
```

```
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
}

mysql_close($con);

}
?>
```

View salary:

```
/** NAME: PADUA GLORIA
STUDENT NUMBER:2916574
This code enables the employee to view The status of their salary For the month*/
if (isset($_COOKIE["myuser"])){
$con=mysql_connect("localhost","root","");
if(! $con)
{
                                                    die('could not connect: '. mysql_error());
}
```

```
mysql_select_db("new", $con);



$usr= $_COOKIE["myuser"];

$query = "SELECT First_Name,Last_name FROM employee_details WHERE Personal_id='$usr'";

$result = mysql_query($query);

 while($row = mysql_fetch_array($result))

{



  $name=$row['First_Name'];

  $last=$row['Last_name'];

}

echo "<h1><font size=6>Welcome </font>"."<font color='blue' size=4>".$name ."
".$last."!</font></h1> ";

//echo " " . $name ." ".$last. "!";

}



else{

  echo "Welcome guest!<br />";}// retrieve the cookie



echo "<br><br>";
```

```
//$table = 'car_info';



$query = "SELECT * FROM salary_info WHERE Salary_id='$usr'";


$result = mysql_query($query);


$response = "<div id='wrapperdetail'><div id='form-divdetail'>";

$response .= "<form class='form' id='form1' action='viewSalary.php' method='POST'>";

$response .= "<div><table border=2 bgcolor='#FFFFFF' color='blue'><tr>";

//$response .= "<td><font size=5><input type='submit' name='delete' value='Delete'></input></font></td>";

$response .= "<td><font size=2>Employee_id</font></td>";

$response .="<td><font size=2>Salary</font></td>";

$response .="<td><font size=2>Currency</font></td>";

$response .="<td><font size=2>Tax</font></td>";

$response .="<td><font size=2>Insurance</font></td></tr>";

//$response .="<td><font size=2>password</font></td></tr>";

while($row = mysql_fetch_array($result))

{


 $response .="<tr>";
```

```
 //$response .="<td><font color='blue' size=4><input type='checkbox' name='box' value=". $row['Personal_id']."></input></font></td>";

 $response .="<td><font color='blue' size=2.3>". $row['Salary_id']. "</font></td>";

 $response .="<td><font color='blue' size=2.3>". $row['Salary']. "</font></td>";

 $response .="<td><font color='blue' size=2.3>". $row['Currency']. "</font></td>";

 $response .="<td><font color='blue' size=2.3>". $row['Tax']. "</font></td>";

$response .="<td><font color='blue' size=2.3>". $row['insurance']. "</font></td>";



$response .="</tr>";







}

$response .= "</table></div></form></div></div>";

echo $response;

?>




</div>

      <div id="content_bottom"></div>



        <div id="footer"><h3><a href="http://www.bryantsmith.com">Prime AutoCare Garage</a></h3></div>
```

```
    </div>
  </div>
</body>
</html>
```

Admin Login:

```
<?php
```

/** NAME: PADUA GLORIA

STUDENT NUMBER:2916574

This code recieves a form from the login page and checks whether the user exists and then is logged in otherwise permission isn't

Granted*/

```
//if(isset($_POST['submit'])){
   $dbHost = "localhost";        //Location Of Database usually its localhost
   $dbUser = "root";             //Database User Name
   $dbPass = "";             //Database Password
   $dbDatabase = "new";    //Database Name

   $db = mysql_connect($dbHost,$dbUser,$dbPass)or die("Error connecting to database.");
   //Connect to the databasse
   mysql_select_db($dbDatabase, $db)or die("Couldn't select the database.");
   $usr = $_GET['username'];
   $pas = $_GET['password'];
   $sql = mysql_query("SELECT Admin_personal,password FROM admin");
```

```
$user="";

$password="";

while($row = mysql_fetch_array($sql)){

        if((strcmp($row['Admin_personal'],$usr)==0)&&(strcmp($row['password'],$pas)==0)){

$user=$usr;

$password=$pas;

}

}

if (strcmp($user,"")!=0 && strcmp($password,"")!=0){

setcookie("myuser", $user);

 header("Location: Admin.php");

}else{

    header("Location: AdminLogin.php");

   }

?>
```

Administrator job allocation code:

```
/** NAME: PADUA GLORIA

STUDENT NUMBER: 2916574

This code enables the administrator to register a new car and allocate tasks to employees

*/

if (isset($_POST['Submit'])){

// Connect to server and select database.
```

```
$con = mysql_connect("localhost","root","");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }

mysql_select_db("new", $con);
$usr="new"; //Status of the car
$today = date("Y-m-d H:i:s");//date of today
$sql="INSERT INTO car_info (Number_plate,Job_assigned,Car_type,Problem,Status,Start_date)
VALUES
('$_POST[value]','$_POST[categories]','$_POST[type]','$_POST[comment]','$usr','$today')";

if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
echo "1 record added";

mysql_close($con);
}
```

```
?>
```

Add a new employee code:

```
/** NAME: PADUA GLORIA

STUDENT NUMBER:2916574

This code enables the administrator to register a new employee

*/

//Pick a car and work on it

if (isset($_POST['add'])){

// Connect to server and select database.

$con = mysql_connect("localhost","root","");

if (!$con)

  {

  die('Could not connect: ' . mysql_error());

  }// Connecting to the server



mysql_select_db("new", $con);//choosing a database and connecting to it.

//$usr='PC0111474';

//$usr="new"; //Status of the car

//$id=$_POST['box'];

$new_date = $_POST['year'] . "-" . $_POST['month'] . "-" . $_POST['day'];

$today = date("Y-m-d H:i:s");//date of today
```

```
$sql="INSERT INTO employee_details (Personal_id,First_Name,Last_name,Cell_phone,Address,Start_date,DOB,Qualification,Gender,email,password)

VALUES

('$_POST[id]','$_POST[fname]','$_POST[lname]','$_POST[cell]','$_POST[add]','$today','$new_date','$_POST[comment]','$_POST[gender]','$_POST[email]','$_POST[pass]')";




if (!mysql_query($sql,$con))

  {

  die('Error: ' . mysql_error());

  }

echo "1 record added";



mysql_close($con);

}

?>
```

Calculate the salary:

```
/** NAME: PADUA GLORIA

STUDENT NUMBER:2916574

This code enables the administrator to calculate the salary of each and every employee in their employement

*/
```

```
if (isset($_POST['submit'])){

$con = mysql_connect("localhost","root","");

if (!$con)

  {

  die('Could not connect: ' . mysql_error());

  }



mysql_select_db("new", $con);

$id=$_POST['user'];

$total=0;

$insurance=50;

$tax=50;

$salary=0;

$result = mysql_query("SELECT Job_assigned,SUM(Price) FROM car_info

WHERE Job_assigned='$id'");

while($row=mysql_fetch_array($result))

{

$total= $row['SUM(Price)'];

$salary=$salary=($total*0.1)-($insurance+$tax);

}

$sql="INSERT INTO salary_info (Salary_id, Salary, Currency, Tax, insurance)

VALUES
```

('$id','$salary','Rand','$tax','$insurance')";

if (!mysql_query($sql,$con))

{

die('Error: ' . mysql_error());

}

echo "1 record added";

mysql_close($con);

}

?>

Delete an employee code:

This code enables the administrator to delete an employee. Although I shall show the main functionality which is deletion of an employee that is for the case when an employee is terminated or leaves the workshop.

/** NAME: PADUA GLORIA

STUDENT NUMBER:2916574

This code enables the administrator to view and delete an employee*/

if (isset($_POST['delete'])){

// Connect to server and select database.

$con = mysql_connect("localhost","root","");

```
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }// Connecting to the server

mysql_select_db("new", $con);//choosing a database and connecting to it.
//$usr='PC0111474';
//$usr="new"; //Status of the car
$id=$_POST['box'];

$sql="DELETE FROM employee_details WHERE Personal_id='$id'";

if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
echo "1 record added";

mysql_close($con);
}
?>
```

CONCLUSION

This chapter shows the code that was programmed to create the main functionalities of the system. Only a small portion is shown. The portion show is for the functionality. Code used to create the user interface which is html code is not shown in this chapter.

APPENDIX A

QUESTIONAIRE

1. Do you have a timesheet for your workers? How does it work?

……
……
……
……

2. Do you have stock?

……
……
……
……

3. How do you monitor/ manage your stock?

……
……
……
……

4. What kind of system do you use to allocate workload or jobs to the employees?

……
……
……
……..

5. Who is in charge of this whole system?

……
……
……
……
……

6. Do you have a timesheet for your workers? How does it work?

……
……
……
……

7. Do you have stock?

……
……
……
……

8. How do you monitor/ manage your stock?

……

……

……

……

9. What kind of system do you use to allocate workload or jobs to the employees?

……

……

……

……

10. Who is in charge of this whole system?

……

……

……

……

APPENDIX B

Tasks	~~16th July~~	23th July	30th July	6th Aug	13th Aug	20th Aug	27th Aug
~~Thesis Document~~	~~Finalise the editing of the documentation~~	~~Finalise the editing of the documentation - & editing~~ ~~Update any changes to the design e.g. objects~~	~~Make changes to object's pseudo code as you develop the software, document all changes etc. in the code & **start** on the User's guide (User's Guide a deliverable for the next term only!)~~				
~~Re-visit the GUI and make changes or redesign~~	~~Check the GUI and see if you are happy that it deals with all the options~~ ~~.~~	~~Re-design parts of the GUI or the whole GUI or just change it for the time being to red – so that you can change it later~~					~~Replace screenshots with screenshots of the current program (it will have changed)~~

~~Create & populate database~~		~~Create & populate (add a few data references to) the MySQL database or put together files to be used in programme~~					
~~Programming task~~		~~Plan the approach by breaking task into objects or modules to program~~	~~Program 1st task/module /object~~	~~Program 1st & 2nd task/module /object~~	~~Programming~~	~~Program 4th task/module /object~~	~~Finalise programming & testing~~
~~Testing and refining with a basic data set~~	~~Read about MySQL database and decide on its structure or if you use files how it will be used.~~	~~Read about software & tools you wish to implement~~	~~Read about software & tools you wish to implement~~		~~Decide on a subset of testing data~~	~~Testing and refining~~	
~~Presentation~~							
~~Website~~	~~**Update NB**~~	~~**Update NB**~~					~~**Update NB**~~

Holiday Complete Still needs to be done

-

REFERENCES

-
- All about PHPMyAdmin http://www.phpmyadmin.net/home_page/index.php
- All about Apache http://www.webopedia.com/TERM/W/Web_server.html
- All about PHP http://php.net/manual/en/intro-whatis.php

Bibliography

MySQL and PHP Reference. (1997). Oracle and/or its affiliates.

All about PHPADMIN. (n.d.). Retrieved from http://www.phpmyadmin.net/home_page/index.php

D.D., D. (n.d.). *Installing, configuring and developing with XAMPP*. . 2007.

Lerdorf, R. (December 6, 2002.). *PHP advanced tutorial*. Veracruz, Mexico.

Viteb. (n.d.). *employee-management*. Retrieved March 22, 2012, from http://www.viteb.com/employee-management

Webslave/advantages of apache. (n.d.). Retrieved march 15, 2012, from Webslave: http://www.webslave.ws/apache/the-advantages-of-apache/

williams, L. (n.d.). *White box testing*. Retrieved March 04, 2012, from agile.csc.ncsu.edu:: http://agile.csc.ncsu.edu/SEMaterials/WhiteBox.pdf

INDEX