STUDENT RESIDENCE MANAGEMENT SYSTEM

by

Iraba Marie Louise

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

Bachelor of Science (Honours) in Computer Science

University of the Western Cape

2009

Date: September 9, 2009
The University of the Western Cape (UWC) houses approximately 3500 students. A total of twelve University residences, with eight located on the Campus and four off campus. Occasionally, the University would house students off campus if there is a demand for accommodation. The Student Residence Management System is developed to facilitate application for accommodation online and to help the staff to manage the different residence activities such as controlling booking, payments and room allocation. The Student Residence Management System will be able to notify and confirm all room allocations. Room allocation confirmations would be sent by email to students who were given accommodation. The development of this system also focuses on security of the information and privileges and access rights are attributed to students and administrators.
# Table of Contents

**Table of Contents**

List of Tables \(\text{iv}\)

List of figures \(\text{v}\)

Acknowledgments \(\text{vi}\)

Glossary \(\text{vii}\)

## Chapter 1

Sketching the background \(8\)

1.1 Introduction \(8\)

1.2 User characteristics \(8\)

1.3 Current system \(9\)

1.4 Motivation \(10\)

1.5 Conclusion \(10\)

## Chapter 2

User Requirement DOCUMENT \(11\)

2.1 Introduction \(11\)

2.2 Data collection technique \(11\)

2.3 Problem domain \(11\)

2.4 Users’ problems on the current system \(12\)

Students’ problems \(12\)

Residence Administrators Problems \(12\)

2.5 User expectations from a new computerized system \(13\)

2.6 Limitations to be encountered \(14\)

2.7 Conclusion \(14\)

## Chapter 3

Requirement Analysis document \(15\)

3.1 Introduction \(15\)

3.2 Designer’s interpretation of the user’s requirements \(15\)

3.3 High-level Constituents Parts \(15\)

Database management \(15\)

Software management \(16\)

3.4 Existing solutions \(16\)

3.5 Alternative Technical Solutions \(16\)

3.6 Best Solution \(16\)

3.7 The Model to be used \(16\)

3.8 Technologies to be used \(17\)

PHP \(17\)

MySQL \(17\)
display Login Page 45
Student application 45
Payment 46
Allocate 47
6.6 Conclusion 47

Chapter 7 48
CODE DOCUMENTATION 48
7.1 Introduction 48
7.2 System Operations 48
7.3 Login Code Documentation 50
7.4 Booking codes documentation 51
7.5 Payment codes documentation 56
7.6 Allocate codes documentation 60
7.7 Conclusion 65

bibliography 66

Appendices 67
Student’s Questionnaire 67
Staff Questionnaire 68
Project plan (Term 1) 69
Project Plan (Term 2) 70
Project plan (Term 3) 71
Project plan (Term 4) 72

Index 73
<table>
<thead>
<tr>
<th>Number</th>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1:</td>
<td>Student Object</td>
<td>36</td>
</tr>
<tr>
<td>Table 2:</td>
<td>Administrator Object</td>
<td>36</td>
</tr>
<tr>
<td>Table 3:</td>
<td>ResAdmin Object</td>
<td>37</td>
</tr>
<tr>
<td>Table 4:</td>
<td>Residence Object</td>
<td>37</td>
</tr>
<tr>
<td>Table 5:</td>
<td>Room Object</td>
<td>38</td>
</tr>
<tr>
<td>Table 6:</td>
<td>Payment object</td>
<td>38</td>
</tr>
<tr>
<td>Table 7:</td>
<td>Booking Object</td>
<td>39</td>
</tr>
<tr>
<td>Table 8:</td>
<td>Latest Balance Object</td>
<td>39</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Login Page</td>
<td>21</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Booking Page (Selection of Residence)</td>
<td>22</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Booking Page (Selection of Room)</td>
<td>22</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Booking Page (Feedback)</td>
<td>23</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Confirmation E-mail</td>
<td>24</td>
</tr>
<tr>
<td>Figure 6</td>
<td>UWC Administration Page</td>
<td>25</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Residence Coordinator Page (Choosing Residence)</td>
<td>26</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Residence Coordinator Page (Allocate)</td>
<td>27</td>
</tr>
<tr>
<td>Figure 9</td>
<td>Residence Coordinator Page (SMS Booking Confirmation)</td>
<td>27</td>
</tr>
<tr>
<td>Figure 10</td>
<td>User Case Diagram</td>
<td>29</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Student Activity Diagram</td>
<td>30</td>
</tr>
<tr>
<td>Figure 12</td>
<td>Admin Activity Diagram</td>
<td>32</td>
</tr>
<tr>
<td>Figure 13</td>
<td>ResAdmin Activity Diagram</td>
<td>34</td>
</tr>
<tr>
<td>Figure 14</td>
<td>Class Diagram</td>
<td>40</td>
</tr>
<tr>
<td>Figure 15</td>
<td>State Diagram</td>
<td>43</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Sequence Diagram</td>
<td>44</td>
</tr>
<tr>
<td>Figure 17</td>
<td>System Operations</td>
<td>49</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENTS

The author wishes to [Click and type acknowledgments]
### GLOSSARY

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML</td>
<td>Hyper Text Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol: it is an application-level protocol for distributed, collaborative, hypermedia information systems. Its use for retrieving interlinked resources led to the establishment of the World Wide Web.</td>
</tr>
<tr>
<td>MYSQL</td>
<td>My Structured Query Language: the most popular Open Source SQL database management system, is developed, distributed, and supported by Sun Microsystems, Inc.</td>
</tr>
<tr>
<td>OOA</td>
<td>Object-Oriented Analysis</td>
</tr>
<tr>
<td>OOD</td>
<td>Object-Oriented Design</td>
</tr>
<tr>
<td>PHP</td>
<td>Hypertext Preprocessor: a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.</td>
</tr>
<tr>
<td>SMS</td>
<td>Short Message Service</td>
</tr>
<tr>
<td>SRMS</td>
<td>The Student Residence Management System</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>UWC</td>
<td>University of the Western Cape</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web.</td>
</tr>
</tbody>
</table>
Chapter 1

SKETCHING THE BACKGROUND

1.1 Introduction
The University of the Western Cape (UWC) has decided that by 2010, the entire student registration process will be online. This vision implies that the student accommodation system should also be computerized and possibly become online based.

The current accommodation application process involves students to fill a paper-based application form which takes a long time to process. The Student Residence Management System (SRMS) is introduced as a computerized system to ease the pressure on students and administrators. The new residence application system will help to eliminate the paper-based applications thereby speeding up processing as details will be captured in the database. This will help to eliminate the delays caused by the sequential checking of each residence for the availability of rooms. The system will check automatically for the available rooms to help accelerate the room allocation process thus making it painless for the managers to allocate rooms to students.

At the present time, students must check for the successful or unsuccessful list of accommodation application at the residence administrator. Since the new system to be adopted is computerized, it will automatically send notifications to successful and unsuccessful applicants through email.

1.2 User characteristics
Students’ residence management system currently involves three types of users; these are Students, Residence Administrator, and UWC Administrator.
- Student Users: this type of user includes all the UWC students who apply for accommodation. These users will also adopt the online system when applying for their accommodations.

- Residence Administrator: these are members of staff who are responsible of residence management. Their task will be allocating rooms to students and notifying students of their application results by email.

- UWC Administrator: this will be a member of administration of the university who will be in charge of application fees for residence.

1.3 Current system
The current system of residence application is not yet computerized and this has made the process of application to be very slow and tiresome. During application, forms are provided to students who fill them by hand and deposit them to the Residence Office. The staff in charge classifies all the application forms and processes each case at time.

The current system being manual, the management does not have updated records of vacant rooms or occupied rooms. To know whether there are still some vacant rooms, the person in charge has to go to the residence building and check physically. Also the lack of updated information about the room availability and their types causes some students to be allocated to the type of rooms they did not want.

The manual system of residence management has another constraint of not accessing information from the students’ database. The residence management staff does not get timely information about students who have been suspended from the university, who have failed, students who have not completed registration and payment, unless they go and request this information from the university administration.
1.4 Motivation

The idea of computerizing this system was introduced with a view to address the problems associated with the current residence management system.

This project introduces a new online-based system of application for student residence which will facilitate students to book for the room they want without spending much time on the queue. Also this system will help the management staff to process the application and notify the students timeously. This must be achieved by enabling the residence management staff to access, timely, all student information from the UWC students database; and this system also will provide a facility of record keeping about room allocations and room availability.

Australian Company, Auskor Australia, developed a student accommodation management model consist of the following important components:

Operations management:

- Application and Room Booking Systems,
- Room Inventory Management
- Tenancy Management
- Rules of Conduct
- Financial Reporting and Account Management Systems
- Marketing and Students recruitment. (Auskor Australia, 2008)

After observing this company model, the student residence management system will develop in such away all these components will include.

1.5 Conclusion

After having observed the current problems in the system of student residence management and after seeing the vision of the UWC in the year 2010, this project was chosen as to provide durable solutions.

In this chapter described the characteristics of the current system. The next chapter looks at the user characteristics.
USER REQUIREMENT DOCUMENT

2.1 Introduction
In the previous chapter, the current system and the users of the system were discussed. This chapter tackles the users’ views about the current system and what they expect from the new system. Also, the data collection tools used are described.

2.2 Data collection technique
The data collection instruments used to gather accurate information about the existing system and the requirements for the new system were interviews with stakeholders, questionnaires and observations.

Interviews and questionnaires were administered to students and residence management staff to collect user requirements. Observation of the current existing system was done at the Residence Administration Office in order to find out how the existing system functions, the problems encountered and how they can be solved by the new computerized system.

2.3 Problem domain
The problem domain of this project is the computerization of a university student’s residence management system with a facility of online access for the users.
2.4 **Users’ problems on the current system**

The stakeholders are students and residence administrators and they encounter different problems with the current system. The information collected from both users is as follows:

**STUDENTS’ PROBLEMS**

- Currently students register for accommodation at residence office only.
  In case students are in distant areas, where they can not reach the university, registration for accommodation is quite difficult.

- Paper based application requires students to print, complete and submit the application form at the residence administration. It is difficult because it takes more than three days to register since there is always a long queue.

- Students apply for accommodation and when they are accepted to study at the University, they are told to re-apply even though they applied before and this confuses students and take much time.

- When the students apply for rooms, there is confusion because some students are given the type of the rooms which they did not apply for. This is because administration does not always have updated records of vacant rooms and their types.

- There is no electronic complaint or suggestion box (anonymous).
  The students can not complain, or make suggestions or comments about the problem they face with the system and also they cannot send an email to the administrator.

**RESIDENCE ADMINISTRATORS PROBLEMS**

- Allocation of same room to two different people.
Users make a mistake of giving a single room to different students at the same time. When student makes booking, the system doesn’t show if the room is vacant or not.

- Tedious capturing data and processing of the paper-based student applications. When applying into the residence database, some time the users are not able to read the handwriting of the students and that cause them to enter the wrong information into the system.

- Sometime the users make mistakes when tracing students who paid half or the full amount.

- Users do not have access to database to get the information about students eg. those who fail, those who leave the university for different reasons etc. When the residence administrator wants the information about students they go to the faculty office and administration.

### 2.5 User expectations from a new computerized system

According to the investigations conducted, the following are students’ expectations from the new system:

- The interface must be simple, easy to understand and use.

- The system must be secured

- Remote access to the system of residence application.

- The system must be able to inform the students whether the application has been successful or not. Through email students should be able to receive their room numbers.

According to the survey, the administrators want the system to have the following features:

- The system must be secured

- The system must identify the occupied rooms as well as unoccupied ones.
- The system must be integrated/linked to the university students’ database for easy retrieval of student’s information (name, address, id number etc), academic, financial records and application/admission status.

2.6 Limitations to be encountered
During system implementation, the designer will not have direct access to the UWC students’ database for easy retrieval of student’s information (name, address, id number, academic and financial records and application/admission status) because of the university policy. However, a similar database will be created as to help the system developer to test for the new system.

2.7 Conclusion
The student residence management system will be the best solution to residence management problems as it will provide easy online application, so that students can apply from home or anywhere else via internet. The online system will also provide quick and reliable registration process hence reducing the load of work done by the staff and saving time for students who apply. This chapter assessed the user’s views about the current system and their expectations from the new system. The next chapter discusses the system software requirements.
3.1 Introduction
In the previous chapter the user requirements were identified. In this section we will focus on the system and software requirements needed for implementation of the new system. Also the software development tools which will help to address the problems will be identified.

3.2 Designer’s interpretation of the user’s requirements
- Students want remote access to the system of residence application. This means that the system has to be web-based to enable them to apply from home.
- The system will need a database because it involves a lot of information which has to be stored and retrieved later, for example information about types of room and their availability.

3.3 High-level Constituents Parts

DATABASE MANAGEMENT
The database will be managed by the administrator and it will have the following characteristics:
- The Database shall be accessible by the software.
- The Database shall allow users to store and search for data
- The Database shall allow users to modify stored data.
- The Database shall allow users to delete stored data.
SOFTWARE MANAGEMENT
- The software shall be accessible from all platforms.
- The software shall be able to interact with the database to retrieve and modify data.
- The software shall be able to add data to the database.
- The software shall be able to edit data in the database.
- The software shall be able to delete data from the database.
- The software shall be able to retrieve data from the database.
- The software shall be able to create reports based on information in the database.

3.4 Existing solutions
Currently the University has a website where students find and print application form. Some members of administration use the database of UWC where information about students is found. This information is taken into consideration while processing the applications made by students.

3.5 Alternative Technical Solutions
The whole system of application for residence should be computerized and preferably must be a web-based system.

3.6 Best Solution
The Online-based Student Residence Management System will be easy flexibility to users, consistent and also it will fulfill the user requirements.

3.7 The Model to be used
In order to analyze the user requirements, UML will be used. UML will help to model the exact activities of the new system, to help understand very well how the system will work and to present a list of tasks to the user. Furthermore, UML will help to break down the project to make it much easier to develop.
3.8 Technologies to be used

It is recommended to use a Web-based technology for the system (using PHP & MySQL with Apache as the web server). The advantage is that the system developed using this technology is:

- Easy to use (with user-friendly interfaces)
- Free (doesn’t require any license)
- Cheaper
- Easy to manage and maintain.

**PHP**

The PHP Hypertext Preprocessor allows web developers to create dynamic content that interacts with databases. (Mehdi Achour, Friedhelm Betz and Antony Dovgal, 2009)

It’s a server-side scripting language.

**MySQL**

MYSQL is an open source relational database management system. It is based on the structure query language (Sun Microsystems, Inc., 2009), it is consistent fast performance, high reliability and ease of use.

**Apache**

The apache is a freely available Web server that is distributed under an "open source" license.

**PHPMyAdmin**

As the user graphic, it is the interface free and convivially realized in program language (PHP) and easy to manage the MySQL database on the server (phpmyadmin, 2009)
JavaScript
Is a scripting language used to enable programmatic access to objects within other applications. It is primarily used in the form of client-side JavaScript for the development of dynamic website.

HyperText Transfer Protocol (HTTP)
Is an application-level protocol for distributed, collaborative, hypermedia information system? Its use for retrieving inter-linked resources led to the establishment of the World Wide Web. (Hypertext Transfer Protocol, 2009)

3.9 System testing
Paper Prototyping
Drawing of the system will be put on paper and shown to the users so that they can test the system. This will be useful because;

Paper Prototyping saves time and money because it allows developers to test the system before code is written.

Allows easy and cheap amendment to the design.

Lowers support burden and increase an overall quality of the software.

Usability Testing
Usability testing is a technique used to evaluate a product by testing it on users. During usability testing random users will be brought in to test the system.

Complete System Testing
Black box testing method will be used to test the complete system. Appropriate data will be selected as per functionality and test it against the functional specifications in order to check for normal and abnormal behavior of the system.
3.10 Conclusion

After examination of possible solutions, we chose the web-based system as it will be able to address problems encountered by users within the current residence administration system. The proposed system is to find optimal matching for user requirements, and management of residences. System implementation will be carried out using PHP, MySQL, PHPMyAdmin and Apache, which are the open source applications. This chapter looked at the requirement analysis document and the next chapter looks at the user interface.
Chapter 4

USER INTERFACE SPECIFICATION

4.1 Introduction
In the previous chapter we discussed about the user requirements specification, the proposed solutions and alternatives to those requirements. This chapter describes the user interface, how it works and how the user interacts with the program. Also the images of the user interface pages are included as to show how application looks like.

4.2 Description of the complete user interface
The application user interface has of the following pages:

LOGIN PAGE
The Login Page (see Figure 1) consists of two text boxes, namely Username and Password; and a Login command button allowing the users to log into the system. The login page helps the users to login as a Student who makes the application, as an Administrator who receives application fees for accommodation or as a ResManager whose duty is to allocate room to students.

The login page comprises necessary contact address and other links containing information related to residences, facilities and services available:
The **Contact** link: displays the contacts of the Residence Office and where users can write their comments.

The **About Residence** link: displays information about the residences (pictures, types of room, prices etc).

The **Facilities** link: shows the other facilities available at the residence (restaurant, cyber café, swimming pool, cinema etc).

The **Help** link: provides the information that helps the user on how to use the program.
Figure 2: Booking Page (Selection of Residence)

As shown in Figure 2, the Booking page contains a combo box used to select a residence and a Display command button which is used to displays the rooms available in the selected residence. The Booking page below (see Figure 3) shows the rooms available and room information in Liberty Residence. The student uses the option buttons to book for a particular room and clicks the Submit command button to submit his/her application.

Figure 3: Booking Page (Selection of Room)
Figure 4: Booking Page (Feedback)

After booking the system provides the details to the applicant. These include the Reference Number, Date of Booking, Student Number, Payment Balance etc (See Figure 4). It also contains the Delete Command Button that can be used in case the applicant wishes to cancel the booking.
After booking the system sent a confirmation e-mail to the applicant. The Confirmation E-mail (see Figure 5) provides all booking details such as student number and reference number which the student presents when he/she is making payment. Apart from the confirmation e-mail the system also sends the confirmations by SMS to students.
The UWC Admin Page (see figure 6) contains a text box in which the Administrator enters student reference number and the amount paid. Then he clicks the Submit command button update the amount paid. Below the link of balance statement helps to displays a mini statement of payments made by the student, the dates of payment and the outstanding balance.
Residence Coordinator Page:

Figure 7: Residence Coordinator Page (Choosing Residence).

The Residence Coordinator Page (see Figure 7) consists of two combo boxes used in selecting academic year of student and the residence then by clicking, the Display command button below, the page displays the list of all applicants who have paid in that selected residence (see figure 8). The Residence Coordinator page also has a link “SMS” allows the Coordinator to view the list of students who booked and enables to send booking confirmation SMS to student (see figure 9).
**Figure 8:** Residence Coordinator Page (Allocate).

**Figure 9:** Residence Coordinator Page (SMS Booking Confirmation)
4.3 How the User Interface behaves

The interface allows the users to input login username and password which must be authenticated before the access is granted. In case of wrong input of username or password, the application produces an error message prompting the user to input correct username or / and password. Also if the user input the correct username and password, the program informs the user that he has successfully logged in. The following User Case Diagram (see Figure 10) displays the functionality of the system.
As shown in Figure 10, the student can view personal details, chooses the residence and room, and finally sends the application form. He/she can view and print reference number, retrieve, edit or cancel his application form. The administrator processes payments for each student who applies while the Residence Manager allocates room to student and both can retrieve, edit or cancel information.
4.4 How the user interacts with the system

HOW THE STUDENT INTERACTS WITH THE SYSTEM

Interactions between the student and the system are summarized in the Student Activity Diagram (see Figure 11).

Figure 11  Student Activity Diagram

The online application is done after the student has read and agrees to the accommodation application information provided on the website. Also the list of activities below gives an overview of how the student interacts with the system in the process of the application.
1. The student will login into the Online Accommodation System by entering his/her student number and password on the login screen.
2. The system authenticate the student
3. The system will show the student’s details and application form
4. The student checks his details
5. The student choose room and send the application form
6. The system checks the room availability and correspondence between the room chosen and the student’s year of study.
7. The system provides the Student Application Reference Number.
8. Student prints the reference number
9. Student edit / cancel the application
10. The system update information
11. The system shows the updated information
12. The student logout
HOW THE ADMINISTRATOR INTERACTS WITH THE SYSTEM

Interaction between the administrator and the system is summarized in the following Administrator Activity Diagram, (see Figure 12).

1. The administrator login
2. The system authenticates the user
3. The system displays the payment page
4. The administrator enters the reference number of the student
5. The system retrieves the student information from the database
6. The system displays the student information
7. The administrator enters the amount for payment
8. The administrator delete / edit / search information about payment
9. The system updates the information
10. The system shows the updated information
11. Logout.

**HOW THE RESMANAGER INTERACTS WITH THE SYSTEM**

Interaction between the ResManager and the system is summarized in the following ResManager Activity Diagram (see Figure 13).

1. The ResManager login
2. The system authenticate the user
3. The system displays the main page
4. The ResManager selects the student’s year of study
5. The system displays the list of applicants in that year
6. The system shows the list of unoccupied rooms for that year
7. The ResManager allocated rooms to students
8. The ResManager search / edit / delete information
9. The system updates information
10. The system displays the information
11. The ResManager sends the confirmation to the student.
12. Logout
This chapter described the user interface. The log in page, main page and other pages’ functionalities were illustrated. The activity diagrams showed how the different types of users (Student, Administrator and ResManager) interact with the system. Furthermore, the use cases showed what the users will do with the system. The next chapter will discuss the objects needed to implement the system.
5.1 Introduction

The user interface was described in the previous chapter showing what the interface can do and how the user will interact with the system. In this chapter, each object is described and documented and a data dictionary provides the details of each object. Furthermore, the objects’ attributes and methods are identified.

5.2 Data Dictionary

The dictionary contains information describing the content of the Online Registration System.

**STUDENT OBJECT**

The Student object contains student personal information (see Table 1). This object contains one record for every student.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudentNumber</td>
<td>Student’s number given by the university</td>
<td>7 numbers</td>
<td>1234959</td>
</tr>
<tr>
<td>Title</td>
<td>Student’s title</td>
<td>20 Characters</td>
<td>Miss</td>
</tr>
<tr>
<td>Surname</td>
<td>Last name of the student</td>
<td>20 Characters</td>
<td>Sara</td>
</tr>
<tr>
<td>Name</td>
<td>First names of student</td>
<td>20 Characters</td>
<td>Mary Jane</td>
</tr>
<tr>
<td>Gender</td>
<td>Student’s gender coded M (Male), F (Female)</td>
<td>20 character</td>
<td>Female</td>
</tr>
<tr>
<td>DateBirth</td>
<td>Student’s birth date</td>
<td>Date</td>
<td>03-03-1990</td>
</tr>
<tr>
<td>Email</td>
<td>Student’s Email</td>
<td>20 Characters</td>
<td><a href="mailto:iraba@yahoo.fr">iraba@yahoo.fr</a></td>
</tr>
<tr>
<td>HomeAddress</td>
<td>Student’s address during holiday</td>
<td>20 Characters</td>
<td>19 Oakwood, Woodstock, CPT</td>
</tr>
<tr>
<td>Email</td>
<td>Student’s email address</td>
<td>20 Characters</td>
<td><a href="mailto:studuwc@hotmail.com">studuwc@hotmail.com</a></td>
</tr>
<tr>
<td>MobileNo</td>
<td>Student’s Cell phone number</td>
<td>Numbers</td>
<td>0724858774</td>
</tr>
<tr>
<td>Faculty</td>
<td>Student’s program of study</td>
<td>20 Characters</td>
<td>Engineering</td>
</tr>
<tr>
<td>YearEntry</td>
<td>The year the student entered the University</td>
<td>4 numbers</td>
<td>2002</td>
</tr>
</tbody>
</table>

Table 1: Student Object

**ADMINISTRATOR OBJECT**

The administrator object (see Table 2) contains administrator personal information.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdminID</td>
<td>Administrator’s Identification number</td>
<td>7 numbers</td>
<td>0054959</td>
</tr>
<tr>
<td>DateBirth</td>
<td>Date of Birthday</td>
<td>Date</td>
<td>1970/12/12</td>
</tr>
<tr>
<td>Name</td>
<td>Administrator’s name</td>
<td>20 Characters</td>
<td>Musa</td>
</tr>
<tr>
<td>MobilePhone</td>
<td>Number of his/her phone</td>
<td>10 Number</td>
<td>078888800</td>
</tr>
</tbody>
</table>

Table 2: Administrator Object
**RESADMIN OBJECT**

The ResAdmin Object (see Table 3) contains ResAdmin’s personal information.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResAdmin ID</td>
<td>ResAdmin’s Identification number</td>
<td>7 numbers</td>
<td>0034959</td>
</tr>
<tr>
<td>DateBirth</td>
<td>Date of Birthday</td>
<td>Date</td>
<td>1945/1/14</td>
</tr>
<tr>
<td>Name</td>
<td>ResAdmin’s name</td>
<td>20Characters</td>
<td>Mussa</td>
</tr>
<tr>
<td>Mobile Phone</td>
<td>Number of his/her phone</td>
<td>10 Number</td>
<td>078844400</td>
</tr>
</tbody>
</table>

Table 3: ResAdmin Object

**RESIDENCE OBJECT**

The Residence Object (see Table 4) contains information about the residence of the UWC.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residence Code</td>
<td>Code given to each residence</td>
<td>7 numbers</td>
<td>001</td>
</tr>
<tr>
<td>Residence Name</td>
<td>Actual Name of the residence</td>
<td>20Characters</td>
<td>Cassinga</td>
</tr>
<tr>
<td>NumberRoom</td>
<td>Number which are in residence</td>
<td>20Number</td>
<td>3000rooms</td>
</tr>
</tbody>
</table>

Table 4: Residence Object

**ROOM OBJECT**

The Room Object (see Table 5) contains information about a particular room of a given residence.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Room Code</td>
<td>Code given to each room</td>
<td>7 character</td>
<td>001/0001</td>
</tr>
<tr>
<td>ResidenceID</td>
<td>Actual code of the residence</td>
<td>Characters</td>
<td>001</td>
</tr>
<tr>
<td>Room Type</td>
<td>The type of rooms in that residence</td>
<td>10 characters</td>
<td>Single</td>
</tr>
<tr>
<td>Price</td>
<td>The Charges per room</td>
<td>10 numbers</td>
<td>R 1864</td>
</tr>
<tr>
<td>Status</td>
<td>The status of room</td>
<td>10 characters</td>
<td>Vacant</td>
</tr>
</tbody>
</table>

Table 5: Room Object

**PAYMENT OBJECT**

The Payment Object (see Table 6) contains information about the payment for accommodation.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>PaymentID</td>
<td>Code of payment</td>
<td>7 character</td>
<td>001/0001</td>
</tr>
<tr>
<td>StudentNumber</td>
<td>Student Number of student who paid</td>
<td>10 Number</td>
<td>Vincent Biruta</td>
</tr>
<tr>
<td>Amount</td>
<td>Price of application fees</td>
<td>10 Number</td>
<td>R 500</td>
</tr>
<tr>
<td>Date</td>
<td>Date of payment</td>
<td>Date</td>
<td>02/03/2009</td>
</tr>
<tr>
<td>Amountpaid</td>
<td>Cash student paid</td>
<td>10 Number</td>
<td>R 300</td>
</tr>
<tr>
<td>Balance</td>
<td>Rest payment</td>
<td>10 Number</td>
<td>R 200</td>
</tr>
<tr>
<td>ReferenceNumber</td>
<td>Number system display after application</td>
<td>5 Number</td>
<td>10005</td>
</tr>
</tbody>
</table>

Table 6: Payment object
**BOOKING OBJECT**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>BookingID</td>
<td>Code given to each application made</td>
<td>7 character</td>
<td>45738</td>
</tr>
<tr>
<td>Student Number</td>
<td>Name of student who paid</td>
<td>10 characters</td>
<td>Vincent Biruta</td>
</tr>
<tr>
<td>Room Code</td>
<td>Room code applied for</td>
<td>10 Number</td>
<td>001/0001</td>
</tr>
<tr>
<td>Date</td>
<td>Date of application</td>
<td>Date</td>
<td>2009/11/11</td>
</tr>
</tbody>
</table>

Table 7: Booking Object

**LATEST BALANCE OBJECT**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ref_ID</td>
<td>Helps to retrieve payment information</td>
<td>7 character</td>
<td>14</td>
</tr>
<tr>
<td>Latest Balance</td>
<td>Last payment</td>
<td>Double</td>
<td>R 700.00</td>
</tr>
</tbody>
</table>

Table 8: Latest Balance Object

### 5.3 Class Diagram

The Class Diagram (see Figure 14) shows the relationship between classes. The student applies for a room and s/he pays for the room. The administrator controls and records the payments. Also, the residence has rooms and rooms are managed by the ResAdmin. The ResAdmin manages the applications and allocates rooms to successful applicants.
Figure 14: Class Diagram
5.4 Conclusion

In this chapter, each class and its attributes were explained in detail using data dictionary. The data dictionary described the attributes and the data type and an example was given to each attribute. Furthermore, the class diagram demonstrated the relationship between classes in the system. The next chapter looks at the object oriented design.
Chapter 6

OBJECT ORIENTED DESIGN (OOD) OR LOWER LEVEL DESIGN

6.1 Introduction
In the previous chapter, each object was described and documented in data dictionary. However, in this chapter, the Object Oriented Design (OOD) will take the classes in the OOA a level deeper into the realm of pseudo-code. The OOD defines the data types for the attributes, the algorithms and implementation details of classes. Also the state and sequence diagrams of the system are explained.

6.2 Inner Details of Class Attributes and Methods
The class diagram described in Chapter 5 shows all classes that will be used within the system. Also class attributes (data types) were defined in the data dictionary and methods (functions) are described within the class diagram.
6.3 State Diagram

The State Diagram (see Figure 15) depicts the dynamic behavior of login and application functions of the system. The student enters the username and password, and then the system validates the entries. If valid, the student is accepted into the system otherwise the student is rejected. For the student to be allocated a room, the system checks if the room is available, if not, the application is rejected. If the room is available the system will check whether the student has paid the room fee and has no outstanding balance.

![State Diagram](image)

Figure 15: State Diagram
6.4 Sequence Diagram

The following Sequence Diagram (see Figure 16) illustrates the sequence of activities as the system is into operation.

The student applies on application page, after that the system checks for room availability and makes confirmation to the application page. The application page displays the reference number to student. Then student make payment to the UWC Administrator and the Administrator input payment information and update the database. After that the ResAdmin selects room and students in the system, and the system displays the information about rooms and students’ applications. Thereafter the ResAdmin allocates rooms to students, update the database and send notifications to successful applicants.

Figure 16: Sequence Diagram
6.5 Pseudo code

DISPLAY LOGIN PAGE

input "ID" and "Password" text boxes
if "LOGIN" button clicked then
{
  Search user in database for User's member if is not found then
  {
    displays error in page of login "User not found" start Procedure LOGIN again
  }
  Else
  {
    if username or password correct
    {
      if user = Student then
      {
        display Application Page
      }
      if user=Administrator then
      {
        System displays payment page
      }
    }
    else
    {
      display allocation Page
    }
  }
}

STUDENT APPLICATION

If student is not accepted in university
{
  display “You cannot login. You are not acceptable to continue your application.”
  then "Go Back" link clicked }
else
{
  display application Page which has all student details and where s/he can choose room
  if "submit" button clicked {
    if there is not the place{
      display error page with "no place of room in this residence try again"
    }
  else
  { Display the reference number}
Print the reference number and send to email

/*After event of “save” button the system display and send directly the reference number in email of student.*/
“save” button clicked
if there is room
    { select email from student and then put reference number }
After display the reference number
“print” button clicked{
Reference number is printed}

Edit and canceling the application

/*When the student want to modify or to canceling the application*/
"Search" button clicked{
search user database for student (by reference number ){
If it’s wrong reference number then {
display “incorrect reference number ”
then "Go Back" link clicked { go back to Student details page }
}
Else if "Edit" link clicked { Display Student details page } then
update information about room only
else {"Delete" link clicked { canceling the application }
}

PAYMENT

if user=Administrator then
{
System displays payment page
“input” reference number
Display the student details
“input” the amount
}
if "SUBMIT" button clicked then { update information in DB }
ALLOCATE

if user=ResAdmin then
{
  //System displays allocate page
  “select” student year of study from student Table
  Echo ‘the students details’
  And
  select” res from room Table
  Echo ‘the list of residence which are unoccupied’
  //system displays the list of students how paid
  if "SUBMIT" button clicked then { send email to student and update information in DB }
else
“RESET” button clicked then{modify the information }

6.6 Conclusion

This Chapter shows the pseudo-codes of the proposed programs, written in a form which is understandable and which will be easy to convert into a programming language. It also covered the state diagram and sequence diagram. The next Chapter presents the detailed documentation of the code (Auskor Australia, 2008).
Chapter 7

CODE DOCUMENTATION

7.1 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.

7.2 System Operations
The following sketch (see Figure 17) 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 (Students, Admin and Coordinator), booking code documentation for student user, payment code documentation for the admin user and allocate code documentation for coordinator.
Figure 17: System Operations
7.3 Login Code Documentation

```php
/**
 * @author IRABA Maire Louise
 * @email 2960564@uwc.ac.za
 *
 * This page (login.php) helps the three users to login we can login as student, admin and coordinator.
 * This code contains following pages:
 * connect.php: which helps to connect to the database
 * index.php: which has heading info such as links, logo of UWC etc.
 * mail.php: helps to send Email
 * admin.php: represents the Admin Page. In this code it is used to display the Admin Interface if the user login as Admin.
 * coordinator.php: represents the Coordinator Page. In this code it is used to display the Coordinator Page if the user login as Coordinator.
 * booking.php: represents the Booking Page. In this code it is used to display the Booking Page if the user login as Student.
 * In this page every query has its comment. The code also includes some other important comments.
 */

// Get info from database in the users table.[Operation 1]
$query = "SELECT * FROM users WHERE id ="."$_POST["user"]."" AND pass_dob="."$_POST["pass"]."""
include("connect.php")
$result = mysql_query($query) or die("Unable to verify the user because :
.mysql_error()");
$row = mysql_fetch_array($result);
if ($row) {
    session_start();
    $_SESSION["id"] = $row["id"];
    // Check user type[Operation 2]
    $user = $row["type"];
    if($user == "admin") {
    
```
header("Location: admin.php");
exit;
} else if($user == "coord") {
    header("Location: coordinator.php");
    exit;
}
else if($user == "stud") {
    header("Location: booking.php");
    exit;
}
}
mysql_free_result($result);
mysql_close($conn);
?>

7.4 Booking codes documentation

<?php
/**
 ********************************************
*********
 *****************************
**********
payment.php
 ********************************************
*********
 *****************************
**********
*Author: IRABA Maire Louise
*Email: 2960564@uw.ac.za
 ********************************************
*********
 *****************************
**********
this page(payment.php) comes when the user login as Admin. It has the button Submit which helps the Admin to update the payment table and a Search Button which helps him to find out the student details. This page also has a link "Balance Statement" which shows the payment statement.
 ********************************************
*********
 *****************************
**********
this page containts:
connect.php : it helps to connect to the database
Mail.php: it helps to send an email.
 ********************************************
*********
 *****************************
**********
In this page every query has its comment. The code also includes some other important comments.
 ***/?>
<html>
<body>
<? include("connect.php");
require_once "Mail.php";

// Display User Name[Operation 1,2]

$q="SELECT * from users where id ="."$_SESSION["id"]."";";
$result = mysql_query($q);
$arr = mysql_fetch_array($result);

// Generate ref_no
function generateRandStr($length)
{
    $randstr = "";
    for($i=0; $i<$length; $i++){
        $randnum = mt_random(0, 61);
        if($randnum < 10){
            $randstr .= chr($randnum+48);
        }else if($randnum < 36){
            $randstr .= chr($randnum+55);
        }else{
            $randstr .= chr($randnum+61);
        }
    }
    return $randstr;
}

// Make a booking
if(isset($_POST['book'])) {
    // Check if student has already booked[Operation 4]
    $q = "SELECT * from booking WHERE user_id = "."$_SESSION["id"]."";";
    $result = mysql_query($q);
    $check = mysql_fetch_array($result);
    if($check){
        echo "";
    }
    else {
        // Generate the ref number[Operation 5]
        $ref_no = generateRandStr(4);
        $student =$_SESSION["id"];;
        $q = "INSERT INTO booking VALUES('$ref_no',"".$_SESSION["id"].""'.$room',now())";
        $book = mysql_query($q);
    }
}
</body>
// Get the room being booked
$q = "SELECT booking.room_id from booking where user_id = " . "$_SESSION["id"]" . ";"
$result = mysql_query($q);

// Update room status
$update = "UPDATE room SET status = 'pending' WHERE id = " . ",room" . ";"
mysql_query($update);

// Send Email and SMS[Operation 6]
$sql="select * from users where id = " . ",student" . ";"
$Email = $arr['email'];
$Telephone = $arr['cell'];
$telephone= $Telephone;
$from = "<residence@uwc.ac.za>";
$to = "<" . ",student." . @uwc.ac.za", ";
$to .= "<" . ",$Email.", ";
$to .= "<" . ",telephone." . @mtn.co.za", ";
$to .= "<" . ",telephone." . @vodacom.co.za", ";
$subject = "Residence Booking Confirmation";

// Select room details information
$sql="select * from room where id = " . ",room" . ";"
$residencename =$arr['res'];
$roomname =$arr['id'];
$typename =$arr['type'];
$roomcost =$arr['cost'];
$body .= "Ref. Number :" . 
$body .= "Room Number :" . 
$body .= "Room Type :" . 
$body .= "Application Fee :" . 
$body .= "Residence :" . 
$body .= "you must pay after two days, if not we will replace to other
one \n";
$host = "itsnw.uwc.ac.za";
$username = "2960564";
$password = "19821129";
$mail = $smtp->send($to, $headers, $body);
if (PEAR::isError($mail)) {
    echo("<p>" . $mail->getMessage() . "</p>");
} else {
    echo("<p>Message successfully sent!</p>".Completed);        
}  

// Delete the booking
else if(isset($_POST['delete'])) {
    // Get the room to be updated
    $q = "SELECT room_id from booking where user_id = 
'.$_SESSION['id']."';
    $q = "DELETE FROM booking WHERE user_id = 
'.$_SESSION['id']."';
    $res = mysql_query($q) or die();
    // Update the status in room table
    $update = "UPDATE room SET status = 'available' WHERE id = 
'.$_SESSION['id']."';
    mysql_query($update) or die();
}  
?>
</head>
<body>
// View balance
<?
<table>
<tr>
<td>
    <div>
        <a href="javascript:toggleLayer('balance')">Balance statement</a>
    </div></td>
<tr>
<td>
    <div id = "balance">
        <div class="studNo">Stud No: <? echo $_SESSION['id']; ?></div>
    </div></td>
<tr>
<td>
    <div id = "balance">
        <div class="studNo">Stud No: <? echo $_SESSION['id']; ?></div>
    </div></td>
</tr>
</table>
// Get the ref num for this student
$q = "SELECT ref_id FROM booking WHERE user_id = 
'.$_SESSION['id']."';
$result = mysql_query($q);
$ref_num = mysql_fetch_array($result);
$q = "SELECT * FROM payment WHERE ref_id = ".$ref_num['ref_id']."";
$result = mysql_query($q);
if(mysql_num_rows($result) < 1) {
    echo "<div class ="perror"">Found no payments on the database.";
} else {
    echo '<br>
    echo '<table style ="color: #000; font-size: 12px;" align = "left" width= "100%" cellspacing = "2" cellpadding = "2" border = "0" bgcolor = "white"">
    <tr>
        <td align = "left"><b>Payments</b></td>
        <td align = "left"><b>Balance</b></td>
        <td align = "left"><b>Date of Payment</b></td>
    </tr>
    $bg = 'FFFFFF';
    while ($row = mysql_fetch_array($result)) {
        $bg = ($bg == '#FFFFFF' ? '#FFFFFF' : '#FFFFFF');
        echo '<tr bgcolor ="'. $bg .'">
            <td align = "left">'. $row['instalment'] .'</td>
            <td align = "left">'. $row['balance'] .'</td>
            <td align = "left">'. $row['pay_date'] .'</td>
        </tr>;
    }
    echo '</table>;
} else{
    // Student has not booked[Operation 3]
    //Check student level
    $q="SELECT level FROM student WHERE id = "'.$_SESSION['id'].'"."";
    $result=mysql_query($q);
    $studentyear = $row['level'];
    switch ($studentyear) {
    case 'second':
        ?></td><strong>Choose Residence</strong></td><br />
 Payment codes documentation

7.5 Payment codes documentation

菜戶：IRABA Maire Louise
电邮：2960564@wac.ac.za

this page(payment.php) comes when the user login as Admin. It has the button Submit which
helps the Admin to update the payment table and a Search Button which helps him
to find out
the student details. This page also has a link "Balance Statement" which shows the
payment statement.
***************************************************************************
this page contains:
connect.php : it helps to connect to the database
***************************************************************************
In this page every query has its comment. The code also includes some other
important comments.
/**/ include("connect.php");
// Display User Name[Operation 1,2]
$q="SELECT * from users where id ="."._SESSION["id"]."";
$result=mysql_query($q);
$arr = mysql_fetch_array($result);
// Enter payment
if(isset($_POST["proc_pay"])) {

// Get the ref num and payment entered
$ref_id = $_POST["ref_no"];
$pay = $_POST["pay"]; 
if($ref_id && $pay) {

// Get the latest balance[Operation 3]
$q = "SELECT balance FROM payment WHERE ref_id = "".".$ref_id."" ORDER BY balance DESC";
$result = mysql_query($q);
while($row = mysql_fetch_array($result)) {
    $latest_bal = $row[0];
}

// Compute balance
$balance = (int)($latest_bal) - (int)($pay);

// Insert the very latest balance in payment table
$q = "INSERT INTO payment values("".".$ref_id."","".$pay."","".$balance."",now()"");
$upd_bal = mysql_query($q);
}
?>
// View student information[Operation 3]
<?
if(isset($ref_num) & & !$ref_num) {
    echo "</div>";<div class ="perror">Please enter reference number.</div>";
}
else if($ref_num) {
$q = "SELECT ref_id, booking.user_id, users.name, room.stud_year, room.res, booking.room_id FROM booking, room, users WHERE booking.ref_id = ".$ref_num." AND room.id = booking.room_id AND users.id = booking.user_id ";
$result = mysql_query($q);

if(mysql_num_rows($result) < 1) {
    echo "<div class ="perror">Found no details on the database.</div>";
}
else {
    echo '<br>
    echo '<table style ="color: #000; font-size: 12px;" align = "left" width= "100%" cellspacing = "2" cellpadding = "2" border = "0" bgcolor = "white"">
    <tr>
    <td align = "left"><b>Ref No</b></td>
    <td align = "left"><b>Stud No</b></td>
    <td align = "left"><b>Name</b></td>
    <td align = "left"><b>Level</b></td>
    <td align = "left"><b>Residence</b></td>
    <td align = "left"><b>Room</b></td>
    </tr>
    $bg = 'FFFFFF';
    while ($row = mysql_fetch_array($result)) {
        $bg = ($bg == '#FFFFFF' ? '#FFFFFF' : '#FFFFFF');
        echo '<tr bgcolor = "' . $bg . '">
        <td align = "left">'. $row[0] . '</td>
        <td align = "left">'. $row[1] . '</td>
        <td align = "left">'. $row[2] . '</td>
        <td align = "left">'. $row[3] . '</td>
        <td align = "left">'. $row[4] . '</td>
        <td align = "left">'. $row[5] . '</td>
    </tr>";
    }
    echo '</table>';
}
//View the balance [Operation 3]
<!-- Payment -->
<html>
<body>
<?
$q = "SELECT * FROM payment WHERE ref_id = ".$ref_id."";
$result = mysql_query($q);
if(mysql_num_rows($result) < 1) {
    echo "<div class ="perror">Found no payments on the database.";
} else {

    //Display payment and Balance [Operation 4]
    echo '<br>);
    echo '<table style ="color: #000; font-size: 12px;" align = "left" width= "100%" cellspacing = "2" cellpadding = "2" border = "0" bgcolor = "white"" '>
    <tr>
    <td align = "left"><b>Payments</b></td>
    <td align = "left"><b>Balance</b></td>
    <td align = "left"><b>Date of Payment</b></td>
    </tr>
    $bg = 'FFFFFF';
    while ($row = mysql_fetch_array($result)) {
        $bg = ($bg == '#FFFFFF' ? '#FFFFFFF' : '#FFFFFFF');
        echo '<tr bgcolor = "'.$bg.'">
        <td align = "left">'. $row['instalment'] .'</td>
        <td align = "left">'. $row['balance'] .'</td>
        <td align = "left">'. $row['pay_date'] .'</td>
        </tr>);
    }
    echo '</table>";
    }
?>
</div>
</body>
</html>
7.6 Allocate codes documentation

```php
/**  
* Coordinator.php  
* Author: IRABA Maire Louise  
* Email: 2960564@uw.ac.za  
* This page (coordinator.php) comes after the user has logged in as a Coordinator. It contains two combo boxes used to select academic year and residence. The Display Button helps display the list of all students who already booked for residence. The Coordinator sends automatically a confirmation e-mail to the student after clicking the Submit Button. This page has also a link "SMS" which helps to send booking confirmation to students. 
*/

include("connect.php");  
$q="SELECT * from users where id ="."."._SESSION["id"].".";  
$result=mysql_query($q);  
$arr = mysql_fetch_array($result);  
if($a_status == "pending") {  
    // Send eMail and SMS [Operation 6]  
    // Get user id
```
$q = "select room.id, booking.room_id, booking.user_id from room, booking where room.id = "'.$room.'" AND booking.room_id = room.id";
$result = mysql_query($q);
$arr = mysql_fetch_array($result);
$student = $arr['user_id'];

$sql="select * from users where id = "'.$student.'";";
$sql =mysql_query($sql);
$arr = mysql_fetch_array($sql,MYSQL_ASSOC);
$Email = $arr['email'];
$Telephone = $arr['cell'];
$telephone= $Telephone;
from = "<residence@uwc.ac.za>";
to = "<".$student."@uwc.ac.za>,";
to .= "<".$Email.">,";
to .= "<".$telephone."@mtn.co.za>,";
to .= "<".$telephone."@vodacom.co.za>,";
to .= "<".$telephone."@cellc.co.za>,";
subject = "Allocation Confirmed!";
$sql="select * from room where id ="'.$room.'";";
$sql =mysql_query($sql);
$arr = mysql_fetch_array($sql,MYSQL_ASSOC);

$residencename =$arr['res'];
$studentidname =$_SESSION['id'];
$roomname =$arr['id'];
$typename =$arr['type'];
$roomcost =$arr['cost'];
$body = "Dear Student".$studentidname."n;
$body .= "Your details for the booking "n;
$body .= "Ref. Number :".$ref_no."n";
$body .= "Room Number :".$roomname."n";
$body .= "Room Type :".$typename."n";
$body .= "Application Fee :R".$roomcost."n";
$body .= "Residence :".$residencename."n";

// Update booking table, delete this booking
$del = "DELETE FROM booking WHERE booking.room_id = "'.$room.'";";
mysql_query($del);
else if($o_status == "occupied") {
    // Set status occupied [Operation 5]
    $update = "UPDATE room SET status = ",\$o_status," WHERE id = "",$room,"";
    mysql_query($update);
    
</div>

//Select Res and Year [Operation 3]
<form>
<? echo "action="$_SERVER["PHP_SELF"]" method="post" name="reservedForm" ?>
<div class="reservedForm">
<table style="width: 100%; border: 0; text-align: left;">
<tr>
    <td colspan="2" style="font-size: 16px;">Reserved Rooms</td>
</tr>
<tr>
    <td>Student Level:</td>
    <td>
        <select class="formInputField" size = "1" name = "searchType" id="searchType" onchange="changeSearch(this.value, this.form);">
            <option value="apt_status">Select level:</option>
            <option value="first_year">First Year</option>
            <option value="second_year">Second Year</option>
            <option value="third_year" >Third Year</option>
            <option value="post" >Post Graduate</option>
        </select>
    </td>
</tr>
<tr>
    <td>Residence:</td>
    <td>
        <select class="formInputField" size = "1" name = "searchBy" id="searchBy" >
            <option value="opt_status">Select residence:</option>
        </select>
    </td>
</tr>
</table>
</div>
</form>
<input type="hidden" name="id" value ="<? $_POST['id'] ?>" />
<input class="formInputButton" type="submit" name="reserve" id="reserve" value="Display" />
</table>
</div>
</form>
<form>
<? echo "action="$_SERVER[PHP_SELF]"" method="post" ?>
<?
if((isset($apt_status) && !($apt_status == "apt_status")) && (isset($opt_status) && !($opt_status == "opt_status"))) {

// Get info [Operation 4]
$q="SELECT payment.ref_id, payment.balance, booking.ref_id, booking.user_id, booking.room_id, type, stud_year, res, status FROM room, booking, payment WHERE payment.balance = '0' AND booking.ref_id = payment.ref_id AND room.id = booking.room_id AND room.status ="pending";
$result = mysql_query($q);
if(mysql_num_rows($result)) {

<table style=" background-color: #FFFFFF; width: 716px; border: 0; margin: 15px auto 0 14px; float: left; text-align: left; font-size: 12px; "  >
<tr style="background:inherit; ">
<td><strong>Stud No</strong></td>
<td><strong>Stud Level</strong></td>
<td><strong>Res</strong></td>
<td><strong>Room</strong></td>
<td><strong>Type</strong></td>
<td><strong>Cancel</strong></td>
<td><strong>Allocate</strong></td>
</tr>
<?php
while($arr = mysql_fetch_array($result)) {

$user_id = $arr['user_id'];
$residence = $arr['stud_year'];
$res = $arr['res'];
$room = $arr['room_id'];
$type = $arr['type'];
}
<table>
<thead>
<tr>
<th>User ID</th>
<th>Residence</th>
<th>Room</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```php
else if((isset($apt_status) && ($apt_status == "apt_status")) && (isset($opt_status) && ($opt_status == "opt_status"))) {
    echo "<div class ="error">Please select student level and residence.</div>";
}
?>
</form>
</body>
</html>
```
7.7 Conclusion

This chapter provided the code documentation. The PHP file consisted of its name, Mysql queries and comments. The next chapter describes how the system will be tested to prove that it works successfully.
Student’s Questionnaire

My name is Marie Louise, I am a student at UWC doing Computer Sciences. I am currently carrying out an academic project “Student Residence Management System” which intends to solve problems related to student’s residence booking. I would wish to collect information from you through this questionnaire. Please fill this questionnaire in the space provided.

1. **Respondent details**
   - Undergraduate □  Postgraduate □
   - Course : _______________
   - Year of study : _______________
   - Sex : _______________
   - Residence (out side or in side?): _______________

2. How do you rate the current booking system for students’ residence?
   - Very efficient □
   - Efficient □
   - Undecided □
   - Inefficient □
   - Very inefficient □

3. What difficulties do you observe on the current residence booking system?
   ______________________________________________________
   ______________________________________________________
   ______________________________________________________

4. In the new computerized system of booking, what facilities do you wish to be included?
   ______________________________________________________
   ______________________________________________________
   ______________________________________________________
   ______________________________________________________
Staff Questionnaire

My name is Marie Louise, a student at UWC majoring in Computer Sciences. I am currently carrying out an academic project “Student Residence Management System” which intends to solve problems related to student’s residence management. I would wish to collect information from you through this questionnaire. Please fill this questionnaire in the space provided.

1. Respondent details

Department: _________________________
Position : ______________________
Duties : ______________________

2. How do you rate the current students’ residence management system?

Very efficient  □
Efficient  □
Undecided  □
Inefficient  □
Very inefficient  □

3. What difficulties do you observe in this system?

______________________________________________________________________________
______________________________________________________________________________
______________________________________________________________________________
______________________________________________________________________________

4. In the new computerized student residence management system, what facilities do you wish to be included?

______________________________________________________________________________
______________________________________________________________________________
______________________________________________________________________________
______________________________________________________________________________

Thank you for your support
## Project plan (Term 1)

<table>
<thead>
<tr>
<th>Meeting dates &amp; times /Tasks</th>
<th>29th Feb</th>
<th>5th March</th>
<th>12th March</th>
<th>19th March</th>
<th>25(^{th}) March</th>
<th>28(^{th}) March</th>
<th>1(^{st}) April</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comments</td>
<td>.Meeting with supervisor Prof Nyongesa</td>
<td>.Meeting with supervisor Prof Nyongesa</td>
<td>.Meeting with supervisor Prof Nyongesa</td>
<td>.Meeting with supervisor Prof Nyongesa</td>
<td>.Meeting with supervisor Prof Nyongesa</td>
<td>.Meeting with supervisor Prof Nyongesa</td>
<td></td>
</tr>
<tr>
<td>Thesis Document</td>
<td></td>
<td></td>
<td>Writing the Documentation</td>
<td>Writing the Documentation</td>
<td>Sent it to Prof. Nyongesa</td>
<td></td>
<td></td>
</tr>
<tr>
<td>URD</td>
<td>Observation of one residence (Tygerbeg)</td>
<td>Reading the Documentation on Interviews</td>
<td>Interview some Students and give them the questions papers</td>
<td>Interview some Staff of admin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAD</td>
<td></td>
<td>Reading the documentation about RAD</td>
<td>Trying to analyze the System</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Presentation /deliverable</td>
<td>Do the slides</td>
<td>Presentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Website</td>
<td>.</td>
<td>.</td>
<td>Designing the Website</td>
<td>Hosting online</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Project Plan (Term 2)

<table>
<thead>
<tr>
<th>Meeting and date</th>
<th>11&lt;sup&gt;th&lt;/sup&gt; April</th>
<th>20&lt;sup&gt;th&lt;/sup&gt; April</th>
<th>24&lt;sup&gt;th&lt;/sup&gt; April</th>
<th>4&lt;sup&gt;th&lt;/sup&gt; May</th>
<th>12&lt;sup&gt;th&lt;/sup&gt; May</th>
<th>19&lt;sup&gt;th&lt;/sup&gt; May</th>
<th>26&lt;sup&gt;th&lt;/sup&gt; May</th>
<th>1&lt;sup&gt;st&lt;/sup&gt; June</th>
<th>3&lt;sup&gt;rd&lt;/sup&gt; June</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Meeting with supervisor</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Meeting with supervisor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Thesis document</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Writing Document</td>
<td>Hand in final document</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>UIS</strong></td>
<td>To Read the contents of UIS</td>
<td>To analyze how they will be</td>
<td></td>
<td>To build the interfaces</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>OOA</strong></td>
<td>To Read the contents of OOA</td>
<td></td>
<td>Drawing the diagrams</td>
<td>Use the net beans</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>OOD</strong></td>
<td>To Read the contents of OOD</td>
<td></td>
<td>Consult the other documents</td>
<td>Use the net beans</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Prototype</strong></td>
<td></td>
<td>Consult the websites about the codes</td>
<td>Trying to code</td>
<td>Test it</td>
<td>Show supervisor</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Deliverable &amp; Presentation</strong></td>
<td></td>
<td></td>
<td></td>
<td>To do slides &amp; send to supervisor</td>
<td>Modify depending on feedback</td>
<td>Show supervisor how I'll present</td>
<td>Presentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tasks</td>
<td>13\textsuperscript{th} July</td>
<td>20\textsuperscript{th} July</td>
<td>27\textsuperscript{th} July</td>
<td>3\textsuperscript{rd} Aug</td>
<td>10\textsuperscript{th} Aug</td>
<td>17\textsuperscript{th} Aug</td>
<td>24\textsuperscript{th} Aug</td>
<td>28\textsuperscript{th} August – 6\textsuperscript{th} Sept</td>
<td>7\textsuperscript{th} Sept</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Thesis Document</td>
<td>Finalise the editing of the documentatio n - &amp; editing</td>
<td>Update any changes to the design – e.g. objects</td>
<td>Make changes to object’s pseudo code as you develop the software, document all changes etc. in the code &amp; start on the User’s guide (User’s Guide a deliverable for the next term only!)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Finalise Documentation and hand in on the 29\textsuperscript{th}</td>
<td></td>
</tr>
<tr>
<td>Re-visit the GUI and make changes or redesign</td>
<td>Check the GUI and see if you are happy that it deals with all the options</td>
<td>Re-design parts of the GUI or the whole GUI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Replace screenshots with screenshots of the current program (it will have changed)</td>
<td>Finalise GUI</td>
<td></td>
</tr>
<tr>
<td>Create &amp; populate database</td>
<td>Create &amp; populate (add a few data references to) the MySQL database or put together files to be used in programme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programming task</td>
<td>Plan the approach by breaking task into objects or modules to program</td>
<td>Program 1\textsuperscript{st} task/modul e/object</td>
<td>Program 2\textsuperscript{nd} task/modul e/object</td>
<td>Program 3\textsuperscript{rd} task/modul e/object</td>
<td>Program 4\textsuperscript{th} task/module/object</td>
<td>Finalise programming &amp; testing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing and refining with a basic data set</td>
<td>Read about MySQL database and decide on its structure or if you use files how it will be used, Read about software &amp; tools you wish to implement Fuzzy logic and how to address misspelt words</td>
<td>Read about software &amp; tools you wish to implement How to send SMS with the programme</td>
<td>Read about software &amp; tools you wish to implement How to send SMS with the programme</td>
<td>*Decide on a subset of testing data *How to send SMS with the programme</td>
<td></td>
<td>Testing and refining *How to send SMS with the programme</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Presentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Prepare for presentation</td>
<td></td>
</tr>
<tr>
<td>Website</td>
<td>Update NB</td>
<td>Update NB</td>
<td></td>
<td>Update NB</td>
<td>Update NB</td>
<td></td>
<td></td>
<td>Presentation 9\textsuperscript{th} September</td>
<td></td>
</tr>
</tbody>
</table>
## Project plan (Term 4)

<table>
<thead>
<tr>
<th>Meeting dates &amp; times /Tasks</th>
<th>20\textsuperscript{th} Sept</th>
<th>30\textsuperscript{th} Sept</th>
<th>7\textsuperscript{th} Sept</th>
<th>13\textsuperscript{rd} Sept</th>
<th>20\textsuperscript{th} Sept</th>
<th>27\textsuperscript{th} Sept</th>
<th>4\textsuperscript{th} Nov</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thesis Document</td>
<td>Try to read the all comments and to modify</td>
<td>Start by identifying all the tasks that the program must be able to do - write the User's guide</td>
<td>Finalize user's guide &amp; Thesis documentation.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programming task</td>
<td>Modify depending on feedback</td>
<td>Add reports in the system</td>
<td>Automating SMS</td>
<td>Finalize Installation disc.</td>
<td></td>
<td></td>
<td>Presentation</td>
</tr>
<tr>
<td>Testing</td>
<td>Doing a research about testing</td>
<td></td>
<td>System Testing</td>
<td>System Testing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Presentation</td>
<td></td>
<td></td>
<td>Designing a PowerPoint Presentation</td>
<td>Showing the Presentation to the Supervisor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Website</td>
<td>Update NB</td>
<td>Update NB</td>
<td></td>
<td></td>
<td>Hosting Website</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
INDEX

A
 algorithms · 41

C
 characteristics · i, 9, 11, 16

D
 Design · 41

H
 hypermedia · 19
 Hypertext · 18

J
 JavaScript · 19

P
 plan · 51
 protocol · 19
 pseudo · 46

Q
 QUESTIONNAIRE · 50

T
 Technologies · 18