**CONFERENCE FACILITY MANAGEMENT SYSTEM**

KIBURU BENSON MUIR

A software project submitted in partial fulfillment for the requirement of the award of Bachelor of **Business Information Technology**. Degree of **Jomo Kenyatta University of Agriculture and Technology**.

August, 2015

# DECLARATION

This software project is my original work, except where otherwise stated and has not been presented for a degree in any other University or any other award.

……………………………………………………. ………………………

(Kiburu Benson Muiru) Date

(HD 232-C007-0137/2013)

# CERTIFICATION

The undersigned certify that he has read and hereby recommend for acceptance of Jomo Kenyatta University a software project entitled: “Sustainability Assessment Model to Manage Conference facilities in Developing Countries”

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. Simon M Karume Date

*School of Human Resource Development*Jomo Kenyatta University

# COPYRIGHT

This software project is copyright material protected under the Berne Convection, the copyright Act 1999 and other international and national enactments in that behalf, on intellectual property. It may not be reproduced by any means in full or in part except for short extracts in fair dealing so for research or private study, critical scholarly review or discourse with acknowledgement, with written permission of the Dean School of Science and Applied Technology on behalf of both the author and Jomo Kenyatta University.

# DEDICATION

God saw me through the entire course. I therefore dedicate this work of my hands to my sweet wife and son for the encouragement, support and inspiration they gave me during the study. The boundaries love for you cannot be estimated. May God bless you as you grow to witness what he has kept in store for you.

I also dedicate to all my classmates who gave me support and inspiration to work with them even in the future.

# ACKNOWLEDGEMENT

First and foremost, I would like to acknowledge the blessing of my heavenly father who provided guidance and avenues of getting funds, time and knowledge to enable me accomplish this project. I am greatly indebted to all people who lent a hand in the preparation of this project.

In particular I want to thank Dr. Simon M. Karume for having availed himself consultation and thorough criticism while undertaking the research. I wish also to pay tribute to my beloved family, my sweet daughter immaculate for their special prayers they always offered while pursuing the course.

I can’t forget my course mates, Friends and colleagues who encouraged me all through the course.

# ABSTRACT

A proper facility management system is important as it helps ensure good stewardship and administration of the use of venues and equipments within a premise. Many conference centres seeks to build facilities with integrity by allowing usage of their premise for healthy activities. Currently, most conference centres uses a manual recording system for venue and equipments booking. This system, unfortunately poses a lot of problems for the management of conference facilities such as improper data management, duplication of processes and no data privacy. Conference Facility Management System (CFMS) is a system that is developed based on database technology. The aim of developing this system is to facilitate the management in dealing with matters relating to the activities of the conference. This system provides a user-friendly interface that helps the user to better understand guidelines to use. This system has been developed using Adobe Dreamweaver CS6, PHP as the programming language, Apache as the server and MySQL as the database. The methodology that is used to build up this system is the Extreme Programming. Extreme programming is used as guides consistently complete each development phase of the CFMS.

**TABLE OF CONTENTS**

[DECLARATION i](#_Toc427821841)

[CERTIFICATION i](#_Toc427821842)

[COPYRIGHT ii](#_Toc427821843)

[DEDICATION iii](#_Toc427821844)

[ACKNOWLEDGEMENT iv](#_Toc427821845)

[ABSTRACT v](#_Toc427821846)

[CHAPTER ONE 1](#_Toc427821847)

[INTRODUCTION 1](#_Toc427821848)

[1.1 Background information 1](#_Toc427821849)

[1.2 Problem definition 1](#_Toc427821850)

[1.3 Description of the Current System 2](#_Toc427821851)

[1.3.1 How the current system works 2](#_Toc427821852)

[1.3.2 Weaknesses of the current system 3](#_Toc427821853)

[1.4 Proposed solution 4](#_Toc427821854)

[1.4.1 Justification 5](#_Toc427821855)

[1.4.2 Objectives: 6](#_Toc427821856)

[1.4.2.1 General objectives 6](#_Toc427821857)

[1.4.2.2 Specific objectives 6](#_Toc427821858)

[1.5 Project schedule 7](#_Toc427821859)

[1.6 Project budget 8](#_Toc427821860)

[1.7 Feasibility Study 9](#_Toc427821861)

[1.7.1 Economic feasibility. 9](#_Toc427821862)

[1.7.2 Technical feasibility study 9](#_Toc427821863)

[CHAPTER TWO 10](#_Toc427821864)

[LITERATURE REVIEW 10](#_Toc427821865)

[2.1 Introduction 10](#_Toc427821866)

[2.2 Fact and Finding 10](#_Toc427821867)

[2.2.1 Case Study 1: Web-Based System for Communication and Scheduling 10](#_Toc427821868)

[2.2.2 Case Study 2: Web Booking and Hotel Management System 12](#_Toc427821869)

[2.2.3 Case Study 3: A Meeting Scheduling System, GRACE 13](#_Toc427821870)

[CHAPTER THREE 14](#_Toc427821871)

[METHODOLOGY 14](#_Toc427821872)

[3.1 Introduction 14](#_Toc427821873)

[3.2 Software process model adopted 14](#_Toc427821874)

[3.2.1 Extreme Programming (XP) 14](#_Toc427821875)

[3.2.2 Extreme programming Features 15](#_Toc427821876)

[3.3.3 Illustration of Extreme Programming System Development Process 15](#_Toc427821877)

[3.2.4 System Development Lifecycle 15](#_Toc427821878)

[3.2.5 Strengths 18](#_Toc427821879)

[3.2.6 Weaknesses 18](#_Toc427821880)

[3 3 Requirement Gathering Tools 18](#_Toc427821881)

[3.3.1. Observation method. 18](#_Toc427821882)

[3.3.2. Interview method. 19](#_Toc427821883)

[3.4 System requirement 19](#_Toc427821884)

[3.4.1 Hardware Requirements 19](#_Toc427821885)

[3.4.2 Software requirements 20](#_Toc427821886)

[CHAPTER FOUR 21](#_Toc427821887)

[SYSTEM ANALYSIS AND DESIGN 21](#_Toc427821888)

[4.1 Introduction 21](#_Toc427821889)

[4.2 Architectural design: 21](#_Toc427821890)

[4.3 System Analysis 22](#_Toc427821891)

[4.3.1 Context diagram 22](#_Toc427821892)

[4.3.2 Domain analysis 22](#_Toc427821893)

[4.3.3 Use case model 24](#_Toc427821894)

[4.4 System Design 25](#_Toc427821895)

[4.4.1 Entity relationship diagram 25](#_Toc427821896)

[4.4.2 Sequence diagrams 26](#_Toc427821897)

[4.5 Database design 27](#_Toc427821898)

[4.5.1 Logical design 27](#_Toc427821899)

[CHAPTER FIVE 28](#_Toc427821900)

[SYSTEM IMPLEMENTATION AND TESTING 28](#_Toc427821901)

[5.1 Introduction 28](#_Toc427821902)

[5.2 Summary of the modules 29](#_Toc427821903)

[5.2.1 Member Module 29](#_Toc427821904)

[5.2.2 Receptionist Module 30](#_Toc427821905)

[5.2.3 Manager Module 31](#_Toc427821906)

[5.3 Summary of how the system works 32](#_Toc427821907)

[5.3.1 Login User 32](#_Toc427821908)

[5.3.2 Administrator Control Panel 33](#_Toc427821909)

[5.3.3 User Control panel 33](#_Toc427821910)

[5.4.1 Screen shot of selected tests 35](#_Toc427821911)

[5.4.1.1 Login test 35](#_Toc427821912)

[5.4.1.2 Making reservation test 35](#_Toc427821913)

[5.4.1.3 Resources management test 36](#_Toc427821914)

[5.5 conclusion 36](#_Toc427821915)

[5.6 recommendations 37](#_Toc427821916)

[REFERENCES 38](#_Toc427821917)

[APPENDICES 39](#_Toc427821918)

[Appendix 1: system code 39](#_Toc427821919)

[5.3.1 Login User 39](#_Toc427821920)

[5.3.2 Administrator Control Panel 40](#_Toc427821921)

[5.3.3 User Control panel 41](#_Toc427821922)

[5.3.4 Selecting Reservation 43](#_Toc427821923)

[5.3.5 Making Reservation 45](#_Toc427821924)

[Appendix 2: Test data 53](#_Toc427821925)

[Test data for Login 53](#_Toc427821926)

[Test data for Reservation 53](#_Toc427821927)

[Test data for Resources 54](#_Toc427821928)

# CHAPTER ONE

## INTRODUCTION

### 1.1 Background information

Kibz hotel and conference center is a hotel situated along Nakuru-Olkalau highway. Our unique meeting venues and business centers deliver focus, flexibility and productivity. Rent by the week, full day, and half day or even by the hour. Whether you seek configurable conference venue spaces of every imaginable size or state-of-the-art conferencing technologies. Our member customers will efficiently and effectively connect to hotel on the ground or on the other side of the globe.

* Maximize convenience and fun with appealing event and conference venues.
* Increase productivity with fully-networked computer labs.
* Select from comprehensive and affordable meeting packages including the ‘day only’ meeting plan.
* Get the most from your time and space with professional on-site personnel at unique meeting venues.

Keeping track of the usage of equipments and venues within a premise is important for management of the hotel. A proper tracking system will help ensure good stewardship and administration of the use of equipments and venues.

### 1.2 Problem definition

Currently, the receptionist of Kibz hotel uses two separate manual recording systems to manage venue and equipments booking. This pose a problem for the receptionist as she is unable to keep track of the equipments used in a venue for a particular event.

The use of manual recording system also creates additional work load for the receptionist. The receptionist needs to obtain booking information (such as date, time, venue and equipments), check for availability and inform booking status to Kibz hotel members via phone or face to face. In addition to that, the receptionist has to constantly check the booking records to ensure borrowed equipments are returned by Kibz hotel members after its use.

In addition to that, the current practice of striking out the booking requests that are not approved by the manager in the venue and equipments record book may cause confusion to the receptionist if it is not done properly.

Besides that, data is not managed properly under the manual recording system. Every time a member books venue and equipments, particulars such as the name and phone number is provided. This creates duplication. In addition to that, the data required for bookings (particulars of members, venue data and equipments data) are treated as three separate entities under the manual system. Therefore, if the phone number is not provided when booking, the receptionist has to look it up in the Kibz hotel Directory Book.

There is also no data privacy in the current manual recording system because the names and phone numbers of Kibz hotel members are listed side by side with the equipments borrowed in a recording book.

Lastly, the manual recording system currently used by Kibz hotel makes it difficult for reports on the use of venues and equipments at Kibz hotel center to be generated.

### 1.3 Description of the Current System

Our current system is a manual based system mostly operated at the reception desk. This is where customer are received, registered, do their booking and also where payment are done. Room services such as checking for room availability and booking are also done at the reception. This is a tiresome job to the worker since all that work is done only by two people. It involves a lot of paper work which is more prone to errors.

All the paper work are organized and stored in large file where its hard to retrieve any information needed urgently since it will take more time to go through all the files.

#### 1.3.1 How the current system works

As stated the current system works manually. When a customer checks in he/she is given a registration form to fill and another form for booking a room. Before booking the customer has to know which room is available thus the worker at the reception checks in the book which room is available. This becomes time consuming and prone to errors such as double booking. Sometime the worker at the reception has to go through all the rooms to see which is not occupied.

Payment of the service is also done manually where cash is given at the reception and the customer is issued a receipt. Since its hard to track which facilities the customer has used then its difficult to calculate the actual amount the customer has spent thus leading to a loss to the hotel

#### 1.3.2 Weaknesses of the current system

* After conducting a feasibility study of the current manual systemand had it analyzed I noticed the following weaknesses:-
* Manual record keeping is not consistent as some errors can creep in while writing manually.
* It is difficult to keep record of room availability and large number of customers.
* More labor and time is required in this current system.
* Managing room availability in order is bit difficult, so it’s not possible for executive to search the particular room.
* Collecting payment is difficult.
* No central database can be created as information is not available in database.
* More funds and paper is wasted to keep the manual record of available rooms and customers.

### 1.4 Proposed solution

|  |  |  |
| --- | --- | --- |
| S. No. | SOLUTIONS | DESCRIPTION |
| 1 | Computerizing the system | The basic problem of existing system was that the system was not computerized. In the newly made system we have removed this problem and we are providing a fully automated system in which the users will be having lots of advantages.   * Better management of data * Highly accessible for customers. * No funds or money is wasted in paper works neither we have to use so much man power though it is a web based application all the data goes directly to database and we can retrieve data according to our choice. * Security * All details will be available at one click * To avoid chaos in registration of a new customer can directly get their login details on to their e-mail id. |
| 2 | 24/7 accessible | The computerized system will accessible from anywhere and at any time since it’s a web based system. So the customers can reach the system very easily. |
| 3 | Records of available rooms and customer report | The customer’s and room’s details will be managed through the computerized based system thus making it highly reliable for the hotel. |
| 4 | Collection of payment | Payment will be collected online while booking rooms, but if some due is there then it can be collected either by cash or card at the time of check-out of customer and also invoice is provided. |
| 5 | Proper controlling system | Executive can view each and every report of the customer just by going through the system which will save his time as the database is centralized now. |
| 6 | Centralized database | The computer database maintains customers and rooms records which will be highly beneficial for the executive. If executive updates any particular field it would notify at other places as all data is fetched through a centralized server only. This will add to the reliability and access of the data. |

#### 1.4.1 Justification

Conference facility management system (CFMS) will bring about greater ease to the receptionist in terms of workload. Instead of the receptionist having to take down booking details for venue and equipments, the responsibility is now transferred to the customers who will fill up an on-line booking form in CFMS. The receptionist tasks are only to approve registration for new customer, record date and time of collected and returned equipments.

In addition to that, a better tracking system as compared to the manual recording system will help receptionist to better administer the uses of venues and equipments at the center.

There will be a better communication and transfer of data between the manager and the receptionist. Under the manual system, confusion may arise during the process of striking out booking requests that are not approved by the manager. With CFMS, the canceling of any booking request is done properly by the system.

Notification via email either to inform members of booking status or to remind members to return borrowed equipments will clearly enhance the communication between customers and receptionist.

#### 1.4.2 Objectives:

##### 1.4.2.1General objectives

* Eliminate Scheduling conflicts
* Intuitive and Easy to Use
* Eliminate Hassles and Save Time
* Improve Communications
* Improve Customer Service

##### 1.4.2.2 Specific objectives

* **Eliminate Scheduling Conflicts**
* CFRS does not allow double-bookings and provides automatic conflict resolution.
* Quickly displays facility use and availability with colored scheme and graphic views in daily and weekly formats.
* Finds available time slots with a powerful search tool and filter.
* **Intuitive and Easy to Use**
* The intuitive and E-Commerce style web interface eliminates any need to train users.
* Users simply click and type on the graphical calendar to reserve facility.
* Features simple navigation tools on the graphical calendar to select facility, dates and time-slots.
* On-line calendars display instant space availability.
* **Eliminate Hassles and Save Time**
* Users can create any kind of recurring schedule in seconds.
* Eliminates phone calls and paper methods by using Forum Remainder Service (FRS) for online requests and/or self-service reservations.
* Reduces the time required to schedule reservations and manage the facilities, enabling facility managers to focus on customer service.
* Its powerful tools efficiently manage all kinds of reservation requests.
* **Improve Communications**
* Facility managers and relevant parties will receive automated change notifications, a confirmation number and a reservation status via e-mail.
* Provides real-time access to the latest information.
* A detailed reservation record log tracks every change made to the reservations.
* Supports multiple facility managers.
* Provides various reminder mechanisms for the facility managers.
* **Improve Customer Service**
* Enables immediate response to user inquiries on availability on the calendar viewers.
* Users can print or email detailed confirmations to customers.
* Provides online, interactive access via the web 24/7.

### 1.5 Project schedule

|  |  |  |  |
| --- | --- | --- | --- |
| **ACTIVITY** | **STARTING DATE** | **ENDING DATE** | **DURATION (Days)** |
| Problem identification and Approval | 28/05/2015 | 04/06/2015 | 5 |
| Feasibility Study | 28/05/2015 | 04/06/2015 | 5 |
| Requirement Analysis and Literature Review | 04/06/2015 | 11/06/2015 | 7 |
| System Analysis/Design/Methodology | 11/06/2015 | 25/06/2015 | 14 |
| System coding and Implementation | 25/06/2015 | 30/07/2015 | 35 |
| System Testing | 30/07/2015 | 13/08/2015 | 14 |

### 1.6 Project budget

|  |  |
| --- | --- |
| **Budget Item** | **Amount in Ksh.** |
| **1. EQUIPMENT** | |
| System Development | 100,000 |
| Hardware Installation | 91,000 |
| System Software | 50,000 |
| Licensing | 25,000 |
| Training | 50,000 |
| Services | 80,000 |
| **Sub-Total** | **396,000** |
| **2. COMMUNICATION** | |
| Airtime @400per week\*16wks | 6,400 |
| Transport per month @ 3000 \*4 months | 12,000 |
| Sub-Total | **18,400** |
| **Grand total** | **414,400** |

Is a part of a system budgeting and accounting practices which helps the system analyst to determine the approximate cost and savings for an organization.

### 1.7 Feasibility Study

A feasibility study was carried out to determine the benefits of the current manual system and the proposed computerized system. The system is indeed viable:

#### 1.7.1 Economic feasibility.

The estimated costs of the system will indeed outweigh the estimated costs of development of the system. The estimated profits and benefits are as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Previous monthly  Expenditure | Proposed monthly  Expenditure | Amount cut  down monthly |
| Paper work | 400,000 | 50,000 | 350,000 |
| Salaries | 5,000,000 | 2,000,000 | 3,000,000 |
| Guest complaints | 350,000 | 50,000 | 300,000 |
| Grand total | 5,750,000 | 3,000,000 | 3,650,000 |

The following shows the monthly net profits of the current manual and proposed computerized system:

|  |  |  |
| --- | --- | --- |
|  | Manual system | Computerized system |
| Gross profit | 50,000,000 | 95,000,000 |
| Expenditure | 35,750,000 | 30,000,000 |
| Net profit | 14,250,000 | 65,000,000 |

#### 1.7.2 Technical feasibility study

The system will be easy to maintain for the technical staff. The system structure is easy to modify by the experts in order to meet the hotel needs and maintain its competence in the business world in the future.

# CHAPTER TWO

## LITERATURE REVIEW

### 2.1 Introduction

Literature reviews are part of the planning phase for Conference Facility Management System (CFMS) project. The objective of literature reviews is to collect relevant and required information from reliable sources such as journals and conference paper to come up with the most suitable solution for the implementation of CFMS. In this chapter, several case studies that involve calendaring, scheduling, booking and facilities management that is relevant to CFMS are studied. In addition to that, approach to web applications that could be used to enhance user friendliness is also looked into.

### 2.2 Fact and Finding

Several case studies that involve calendaring, scheduling, booking and facilities management that is relevant to CFMS are looked into. These case studies include;

1. Web-Based System for Communication and Scheduling (WBSCS)
2. Web Booking and Hotel Management System (HotMot) and
3. Meeting Scheduling System (GRACE).

The advantage and disadvantage of using the AJAX approach of web applications will also be explored.

#### 2.2.1 Case Study 1: Web-Based System for Communication and Scheduling

This case study is based on the research paper written by El Aarag and Hartford (2003). This research paper describes the requirements and implementations of an integrated on-line email and calendar system using wired computers and wireless device such as a cellular telephone with Internet capabilities. Since CFMS is a web-based application, studies on system implementation using wired computers is given the emphasis.

WBSCS has two main functionalities:

1. Allow users to check a remote mail server and send email.
2. Allow users to create and manage a limited number of calendars.

Next, the research paper gives good reasoning to every tool used in the development of WBSCS. Below are the list of tools and its reasoning:

1. **PHP as the server side scripting language.**

WBSCS must be easily portable to all major operating system. PHP is the best choice as it has libraries for Windows, Mac OSX, Linux and various versions of UNIX. Even if it does not have a library, it can be still be used as a Common Gateway Interface (CGI) application.

1. **MySQL as the database.**

The database used in WBSCS has to be portable as well. Furthermore, since MySQL is often used alongside with PHP, it is the best choice for the system.

1. **Microsoft Internet Information Services (11s) as the web server.**

The IIS is chosen because Windows computer is used.

1. **JavaScript as the client side scripting language.**

User interface has to be taken into consideration when developing a web based system because different user will use different browser to access the system. To ensure standardization, JavaScript is chosen because it is the only common language that is supported by all web browsers.

1. **Cascading Style Sheet (CSS).**

A web-based system that is similar in look will help improve user interaction. To achieve that, CSS is used to handle the design and formatting aspect of the Hypertext Markup Language (HTML) code.

1. **CompuServe Graphics Interchange Format (GIF) and Joint Photographic Experts Group (JPEG) as the image format for icons and thumbnail.**

While the latter keep thinking the web page had not been downloaded completely. Therefore, in view of the problems, the standard image format *i.e.* GIF and JPEG is used.

In regards to the system layout, the design mirrors that of Microsoft Outlook.

The purpose of designing a layout based on an interface that is familiar with users is to enhance user usability.

#### 2.2.2 Case Study 2: Web Booking and Hotel Management System

Web Booking and Hotel Management System (HotMot) is an on-line hotel reservation system developed for Hotel Lannentie in Finland (Anckar and Walden, 2001).

HotMot as depicted in Figure 2.6 is an integration between the reception module (indicated by a lighter shade of grey) and web module (indicated by a darker shade of grey).

The key features in reception module are listed below

* Speedy booking using an activated availability table.
* Multiple bookings at one time.
* Easy modification or cancellation of bookings.
* Real time availability of rooms can be viewed by downloading new reservation made by customers at the web module.
* Provide greater flexibility by reserving a room in a chosen accommodation category instead of a specific room.
* Automation of data collection followed by report generation.

The key features in web module are listed below:

* Customers can quickly check availability of rooms in each accommodation category before placing a reservation.
* Automation of discounts calculation especially for regular customers.

In terms of database, bookings made by user through the web module will be saved in a secondary reservation database on the Web server therefore eliminating the need for the hotel to be connected to the Internet.

The development of HotMot also takes into consideration the use of dial-up Internet connections. Therefore, when the booking application of the reception module is opened, the program will automatically connect with the Internet, download new reservation from the web module and update new bookings made at reception module to the reservation database then disconnects.

However, the above feature may not present a true real time reservation data as new bookings will constantly be made at the web module and reception module. Furthermore, the web module does not provide users the ability to cancel booking.

#### 2.2.3 Case Study 3: A Meeting Scheduling System, GRACE

GRACE is a prototype developed by Pino and Mora (1998) that helps schedule meeting using the latitude modal. The latitude modal does not just look at the availability of a time slot for meeting but takes into consideration user preference on time, subject or place. Using this modal, the computer based system helps people to schedule meetings -themselves instead of the system scheduling the meeting for them. This approach is characterized as "computer supported cooperative work" (Schmidt and Bannon, 1992). In fact, HotMot in the previous case study uses this approach as well because reservations are only made for rooms of an accommodation category. The receptionist or the hotel staff will later reserve a particular room based on guest's preference.

GRACE utilizing the "computer supported cooperative work" approach as well runs on a network of Sun Sparc workstations with SunOs/Xwindows.

# CHAPTER THREE

## METHODOLOGY

### 3.1 Introduction

A system development methodology refers to the framework that is used to structure, plan, and control the process of developing an information system. A wide variety of such frameworks have evolved over the years, each with its own recognized strengths and weaknesses. One system development methodology is not necessarily suitable for use by all projects. Each of the available methodologies is best suited to specific kinds of projects, based on various technical, organizational, project and team considerations.

The systems development methodology is used to describe the process for building systems, intended to develop systems in a very deliberate, structured and methodical way.

Extreme programming is the methodology of my choice in developing Conference Facility Management system

### 3.2 Software process model adopted

#### 3.2.1 Extreme Programming (XP)

Extreme programming is a software development methodology which is intended to improve software quality and responsiveness to changing customer requirements. As a type of agile software development, it advocates frequent "releases" in short development cycles. This is intended to improve productivity and introduce checkpoints where new customer requirements can be adopted. The main goal of XP is to lower the cost of change in software requirements.

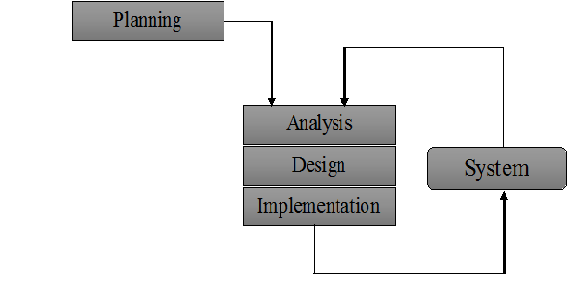
Extreme programming is carried out in the following manner; the phases are carried out in extremely small steps. First, one writes automated tests, to provide concrete goals for development. Next is coding (by a pair of programmers). Design and architecture emerge out of refactoring, and come after coding. Design is done by the same people who do the coding. The incomplete but functional system is deployed or demonstrated for the users. At this point, the practitioners start again on writing tests for the next most important part of the system.

#### 3.2.2 Extreme programming Features

Extreme programming has the following features/ core practices

* Fine scale feedback which involves**,** Test driven development, Planning game, Whole team and Pair programming
* Continuous process rather than batch. This also involves, Continuous Integration, Design Improvement ,and Small Releases
* Shared understanding including Simple design, System metaphor, Collective code ownership and Coding standards or coding conventions
* And Programmer welfare that involves Sustainable pace that is forty hour week.

#### 3.3.3 Illustration of Extreme Programming System Development Process

****

#### 3.2.4 System Development Lifecycle

In developing the Conference Facility Management System, the following steps were taken;

**i. Planning**

A project plan was developed as well as other planning documents. It provided the basis for acquiring the resources needed to achieve a solution. This phase ensured that the problem solved was the one that needed to be solved and that the initial description was complete and consistent.

Under the planning phase of the project, a project timeline, work plan and Budget were developed. (Please refer to appendices). Under this phase;

* The project team was formed and a project leader appointed
* The system flowcharts were prepared
* The characteristics of the proposed system were defined and identified

**ii. Analysis**

At this point, the system in place was analyzed to determine where the problem was in an attempt to fix the system. This step involved breaking down the system in different pieces to analyze the situation, analyzing project goals, breaking down what needed to be created and attempting to engage users so that definite requirements could be defined.

Under analysis, Requirement gathering is the most crucial aspect as many times communication gaps arise in this phase and this leads to validation errors and bugs in the software program. Therefore, the following techniques were used to gather information

Under analysis, the following data collection techniques were used.

1. **Semi-structured interviews**

Semi-structured interviews are conducted with a fairly open framework which allow for focused, conversational, two-way communication. They can be used both to give and receive information.

This tool was used as a data collection methodology of choice because it is; less intrusive to those being interviewed as the semi-structured interview encourages two-way communication.

1. **Direct (Reactive) Observation**

Direct Observation is a method in which a researcher observes and records behavior / events / activities / tasks / duties while something is happening. This was used in correspondence to interviewing in order to gain a more holistic view of the Conference Facility Management System. Observations give additional, more accurate information on behavior of people than interviews or questionnaires. They can also check on the information collected through interviews especially on sensitive topics.

**c) Using available information**

This is a data collection method that involves the process of examining and evaluating already existent literature material to obtain facts and data regarding a specific subject. Locating these sources and retrieving the information can help in data collection.

In the development of the records management system, this research methodology was mainly used in the analysis and design phases of the system development process. This is because it permitted the researcher(s) to analyze changes in trends.

**iii. Design**

In systems design the design functions and operations is described in detail, including screen layouts, business rules, process diagrams and other documentation. The output of this stage described the new system as a collection of modules or subsystems. The design stage took as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements was produced as a result of interviews, workshops, and/or prototype efforts.

Design elements described the desired system features in detail, and generally included functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary.

**iv. Implementation phase**

Here all the iterations were brought together and integrated to make one working system. Modular and subsystem programming code was accomplished during this stage. Unit testing and module testing was done in this stage

#### 3.2.5 Strengths

* Customer focus increase the chance that the software produced will actually meet the needs of the users
* The focus on small, incremental release decreases the risk on your project:
* By showing that your approach works and
* By putting functionality in the hands of your users, enabling them to provide timely feedback regarding your work.
* Continuous testing and integration helps to increase the quality of your work
* XP is attractive to programmers who normally are unwilling to adopt a software process, enabling your organization to manage its software efforts better.

#### 3.2.6 Weaknesses

* XP is geared toward a single project, developed and maintained by a single team.
* XP is particularly vulnerable to "bad apple" developers who:
* Don't work well with others
* Who think they know it all, and/or
* Who are not willing to share their "superior” code
* XP will not work in an environment where a customer or manager insists on a complete specification or design before they begin programming.
* XP will not work in an environment where programmers are separated geographically.
* XP has not been proven to work with systems that have scalability issues (new applications must integrate into existing systems).

### 3 3 Requirement Gathering Tools

#### 3.3.1. Observation method.

The team closely observed the hotel’s activities as the guests are served at the hotel and how the management carried out its managerial activities. The team found out that the hotel indeed was in a dire need for automation of its activities to realize its maximum potential. The guest admission and booking process was not well managed by the current manual system, hence, the team came up with this system to clearly outline the management boundaries of the two departments. The guest payment process also needed an urgent checkup as it led to congestion on the accountant’s desk leading to time wastage. The team also observed that the guest records were not well protected from unauthorized access as ill-motivated guests could easily gain access to the other guest files by posing as the real guest. From our observation we noted that the manual system denied the hotel of international guests who would have rather been potential customers through online booking.

#### 3.3.2. Interview method.

This was the most exhaustible method of data collection. The team used their data superior collection skills to extract data out of the hotel management team, the hotel employees and the hotel’s guests at the moment of data collection. The interviewers created a conducive environment in which the interviewees could feel free to let out the best they could that helped the team come up with the most effective system to fulfill their needs. Among the data collected the following was highlighted to be of a great consideration: The guests’ feelings about thecurrent management of the hotel, the employees’ comfortability with the proposal of introducing a new computerized system. Many computer illiterate employees had fears of replaced displaced by the computer literate employees but they were assured of their survival should they be ready to prove their competence in their activities. The management’s dire need formanagement of the hotel’s proceedings was pu8t into consideration by the system developers.

### 3.4 System requirement

These are both hardware and software require for the smooth running of the proposed system. These are:

#### 3.4.1 Hardware Requirements

* 500GB hard disks
* RAM 4GB
* Corei3 Intel processor speed of 2.0GHz
* Input devices: Keyboards, Mouse
* Output devices: Monitors, speakers, Printers

#### 3.4.2 Software requirements

1. **MYSQL DBMS-**it allows combination, extraction, manipulation and organization of data in the voters’ database. It is platform independent and therefore can be implemented and used across several such as Windows, Linux server and is compatible with various hardware mainframes. It is fast in performance, stable and provides business value at a low cost.
2. **HTML -Hypertext Markup Language-**This is currently the core of the web world, it is a language used to makeup web page. It is the glue that holds everything together. Although HTLM was used for the implementation of the CFMS, it is highly compatible with extensible HTML (XHTML) which is designed to be a replacement of HTML made to handle data and is also portable between different browsers and platforms with little or no alterations in code. Macromedia Dreamweaver is a prefer tool for designing HTML pages and that is the tool used in coming up with this CFMS system.
3. **PHP coding-**This is for advanced user who find PHP codes easy to work with.
4. **Testing** is done via XAMPP SERVER.
5. **Web browsers**: Mozilla Firefox, Google chrome, Opera and Internet Explorer
6. **Reporting Tool** i.e. through Data Report.

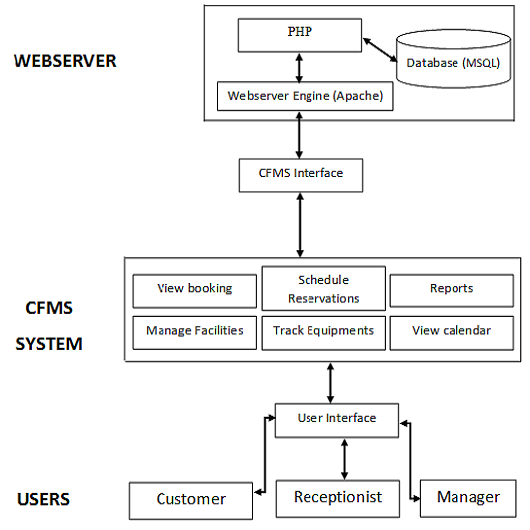
# CHAPTER FOUR

## SYSTEM ANALYSIS AND DESIGN

### 4.1 Introduction

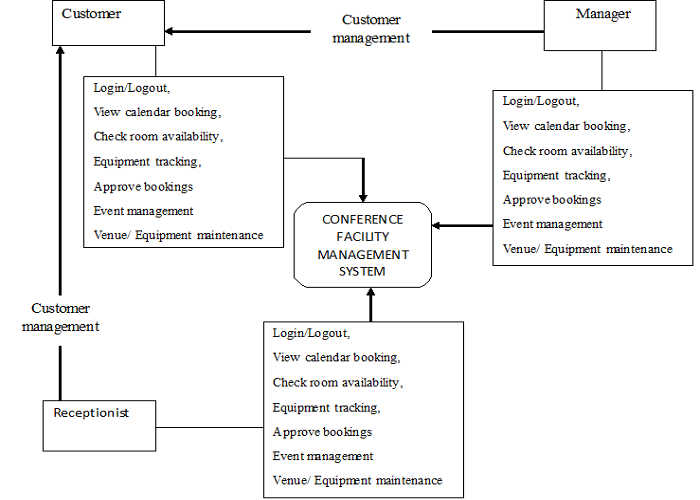
### 4.2 Architectural design:

The system is designed in the following manner. The Conference Facility Management system (CFMS) has a backend engine that consists of a MYSQL database, PHP as the programming language and Apache as the webserver and the user interface modules. The system architecture is illustrated below

****

### 4.3 System Analysis

#### 4.3.1 Context diagram



#### 4.3.2 Domain analysis

**Introduction-** This is the process that identifies the relevant objects of an application domain. It describes background information that has been gathered about events in organizations and how they are handled.

**Glossary.**

**Event-** Events includeLogging in & Outof the system. Several categories of events have been identified:

***-Open event***: An event that starts at a precise instant but with no predetermined duration. Meetings held at the consulate and requests sent by applicants often fall into this category.

***-Fixed event***: An event that starts at a precise instant and with a predetermined duration. Staff training sessions are examples of this kind of event.

***-Day events***: An event associated with a particular day without precise start and end times. Weekly database queries on end of the week are such events by users and database administrator.

***-Recurrent event***: An event that occurs repeatedly on some regular schedule (for example daily, weekly or monthly). The event normally has a starting date and an ending date. Deadlines for booking interviews at the consulate.

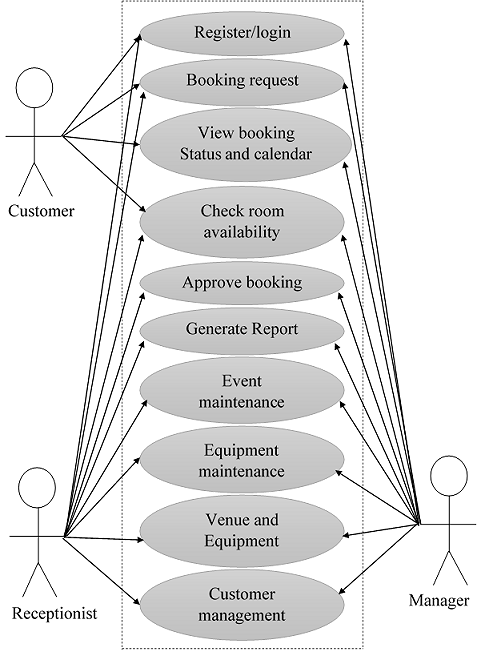
**General knowledge about the domain-** This domain is all about viewing the guest house and then the user will be able to register into the system; this will help determine the level of performance the system must achieve.Out-dated events are of little interest. Each event has a title, a location and the name of a contact person associated with it.

**Customers and users-** A system administrator normally manages the computer environment. Technicians typically install software that must be available to all users.

**The environment-** The receptionist will have a computer on the desk; it is most common for this to be MS-Windows based, but a significant minority of potential clients use other platforms.A wide variety of software is installed on these computers, with each actor having a unique configuration. Some software may be installed on every computer using a site-wide license.

**Competing software-** Several software tools exist that can manage events. However, since these are usually included as part of a larger system they are quite complex to use. Hence they are not generally adopted by the entire staff. These applications allow you to check for the availability of rooms and to book them. They might also maintain the personal schedules of staff members such that it is possible to find the best schedule for a given event.

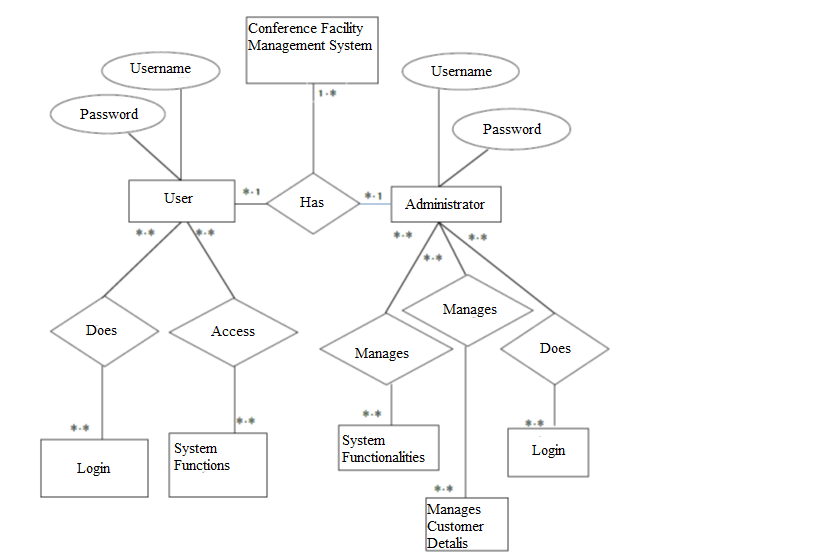
#### 4.3.3 Use case model



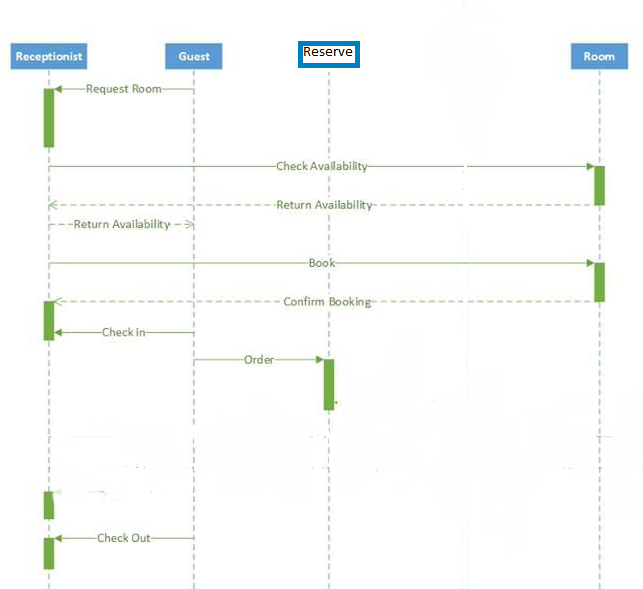
### 4.4 System Design

System design is how the systems looks like. It show how the system work and various sections and functions of the system.

#### 4.4.1 Entity relationship diagram



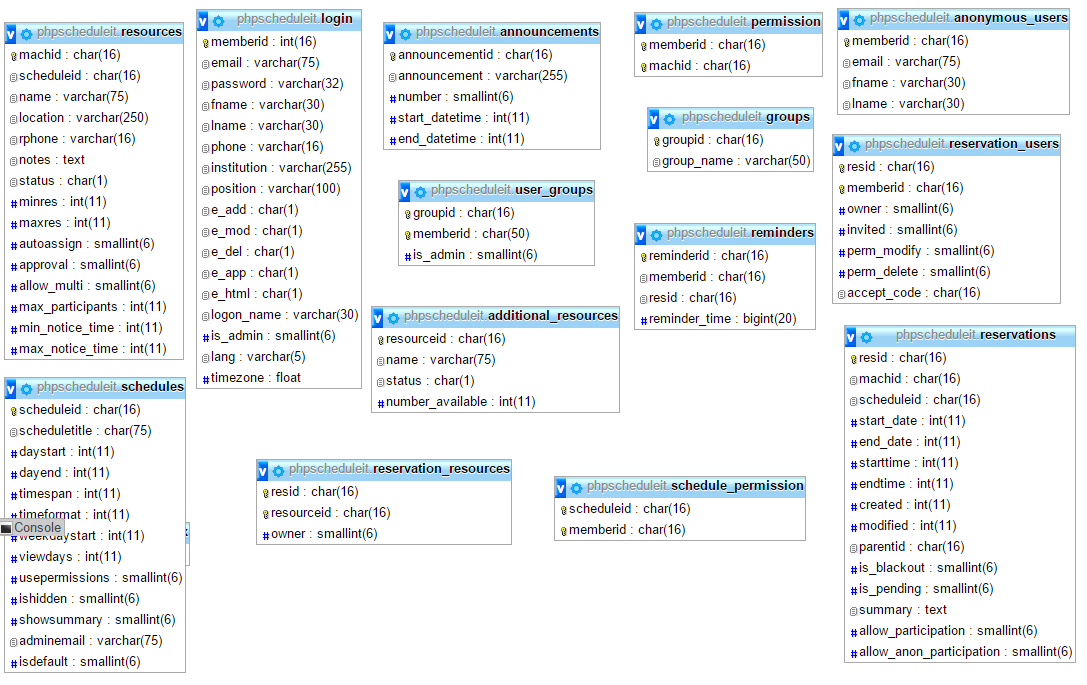
#### 4.4.2 Sequence diagrams



### 4.5 Database design

This how system database looks. It shows tables contained in the database as shown below.

#### 4.5.1 Logical design



# CHAPTER FIVE

## SYSTEM IMPLEMENTATION AND TESTING

### 5.1 Introduction

In this chapter we talk of how the system is going to be made into use after being developed. First the system has to go through several tests before it is fully functional so as to make sure the system will run smoothly and it will meet the requirements specified by the user. Below are the tools used to develop the system.

**PHP (Hypertext Pre-processor)**

PHP is probably the most popular scripting language on the web. It is used to enhance web pages. With PHP, you can do things like create username and password login pages, check details from a form, create forums, picture galleries, surveys, and a whole lot more.

**HTML**

HTML, or Hypertext Markup Language, is used to create web pages. Site authors use HTML to format text as titles and headings, to arrange graphics on a webpage, to link to different pages within a website, and to link to different websites. HTML is a set of codes that a website author inserts into a plain text file to format the content. The author inserts HTML tags, or commands, before and after words or phrases to indicate their format and location on the page. HTML tags are also used to add tables, lists, images, music, and other elements to a webpage.

**Database used.**

MySQL runs on virtually all platforms, including Linux, UNIX, and Windows. Although it can be used in a wide range of applications, MySQL is most often associated with web-based applications and online publishing and is an important component of an open source enterprise stack called LAMP. MySQL is the relational database management system and PHP as the object-oriented scripting language.

**Web server**

**Apache** is generally recognized as the world's most popular Web server (HTTP server). Originally designed for UNIX environments, the Apache Web server has been ported to Windows and other network operating systems. The Apache Web server provides a full range of Web server features, including CGI, SSL, and virtual domains. Apache also supports plug-in modules for extensibility. Apache is free software, distributed by the *Apache Software Foundation* that promotes various free and open source advanced Web technologies.

### 5.2 Summary of the modules

Systems modules are interfaces or the users of the system. Modules shows how the user is going to use the system and how is going to access the system.

#### 5.2.1 Member Module

The main functions of the member module will encompass:

**a) Login**

CFMS Members need to be authenticated by providing the right email address and password before they are allowed to access the module. Once login is successful, the member module is personalized as according to their particulars.

**b) Booking Request Maintenance**

CFMS members can place booking for venue and equipments by filling in an online reservation page. CFMS members can also edit or delete their booking requests provided that it has not been approved by the manager yet. Once the booking requests have been approved, only the delete operation can be done.

**c) View Booking Status**

CFMS members who have place bookings on a venue can check their booking status *(i.* ***e.*** pending, approved or not approved) through the system. Once the manager has cleared pending bookings for the day, the booking status (either approved or not approved) will be sent to members via email.

**d) View Booking Calendar**

CFMS members can view booking calendar which shows the day slots of all venues at CFMS center. The calendar will be color coded for easier identification; pink will indicate that the day contains only approved booking, blue will indicate that the day contains only the member's pending bookings, purple will indicate both pending and approved bookings while white will indicate that there are no bookings for whichever venue.

#### 5.2.2 Receptionist Module

The main functions of the receptionist module will encompass:

**a) Login**

The receptionist likewise has to be authenticated by providing the right email address and password before he / she is allowed to access the module.

**b) View Booking Calendar**

Receptionist can view booking calendar which shows the day slots of all venues at conference center. The calendar will be color coded for easier identification; pink will indicate that the day contains only approved booking, yellow will indicate that the day contains bookings with overdue items and white will indicate that there are no bookings for whichever venue.

**c) Generate Report**

Reports that are generated can either be in a textual form or in a graphical form (such as bar chart or pie chart). Two (2) examples of reports are:

* Report on venue and equipments booking according to name of members.
* Report on the frequency of venues booked in a specified length of time.

The former example will be in a textual form while the latter will be presented using a bar chart.

**d) Equipments Tracking**

CFMS will check the database for any overdue equipment. If there are, a reminder to return the equipment will be sent to the borrower via email by the receptionist. Once equipments have been returned, the return date and time will be recorded at the online booking form by the receptionist.

**e) Member Management**

The receptionist can approve or disapprove the registration of new CFMS members. The receptionist can also delete members which have already been approved earlier.

#### 5.2.3 Manager Module

The main functions of the manager module will encompass:

**a) Login**

The manager likewise has to be authenticated by providing the right email address and password before he / she is allowed access to the module.

**b) Approve Bookings**

Pending bookings are listed for the manager's approval. Once bookings have been finalized, members who have placed booking for the venue will be notified via email.

**c) View Booking Calendar**

Manager can view booking calendar which shows the day slots of all venues at conference center. The calendar will be color coded for easier identification; pink will indicate that the day contains only approved booking, blue will indicate that the day contains only pending bookings, purple will indicate both pending and approved bookings while white will indicate that there are no bookings for whichever venue.

**d) Generate Report**

Reports that are generated can either be in a textual form or in a graphical form (such as bar chart or pie chart). Two (2) examples of reports are:

* Report on venue and equipments booking according to name of members.
* Report on the frequency of venues booked in a specified length of time.

The former example will be in a textual form while the latter will be presented using a bar chart.

**e) Event Management**

The manager can add, edit or delete a CFMS event.

**f) Venue Maintenance**

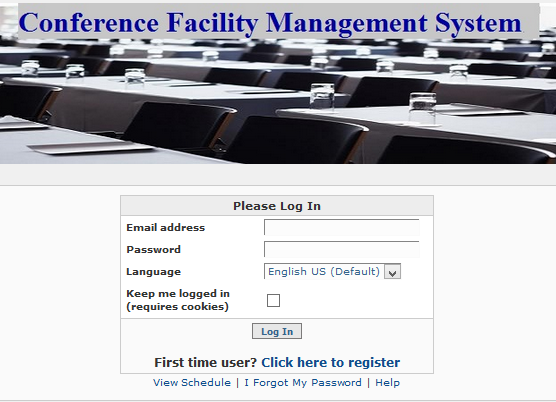
The manager can add details for a new venue. He */* She can also edit or delete the venue details.

**g) Equipments Maintenance**

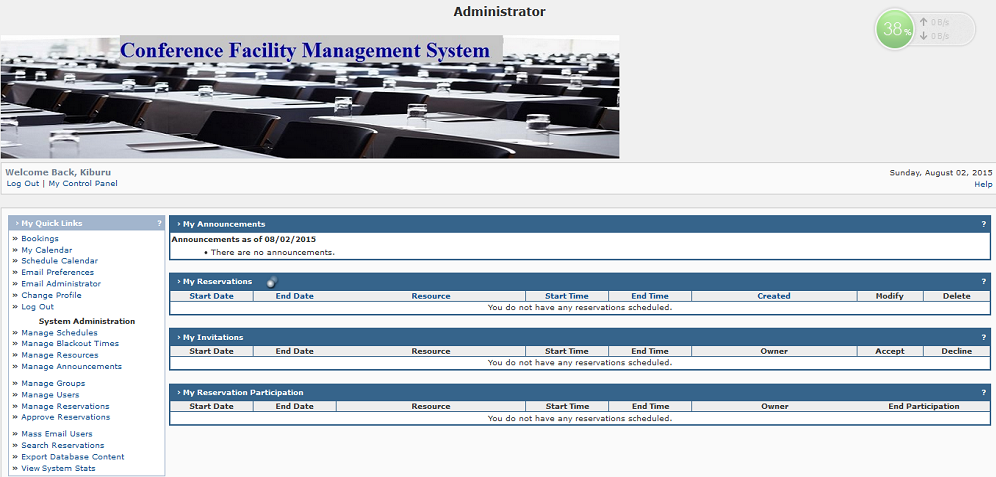
The manager can add details for new equipment. He / She can also edit or delete the details for the equipment.

### 5.3 Summary of how the system works

#### 5.3.1 Login User



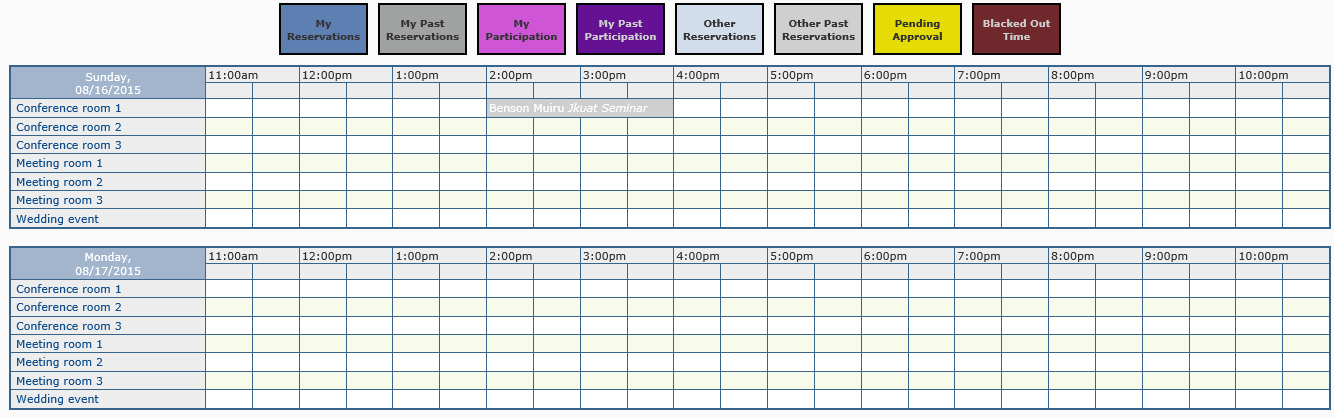
#### 5.3.2 Administrator Control Panel

****

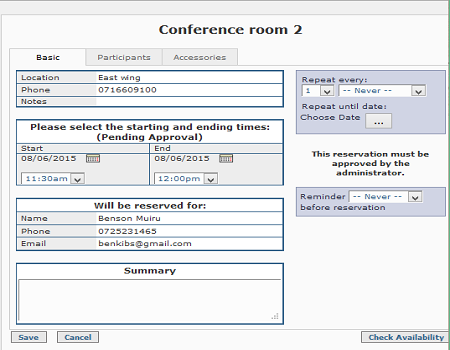
#### 5.3.3 User Control panel



**5.3.4 Selecting Reservation**

****

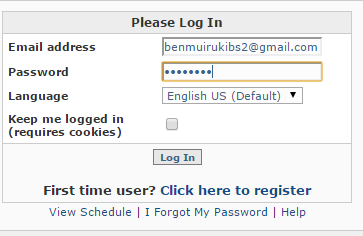
**5.3.5 Making Reservation**

****

#### 5.4.1 Screen shot of selected tests

The following are screen shots of selected test to show how the systems work

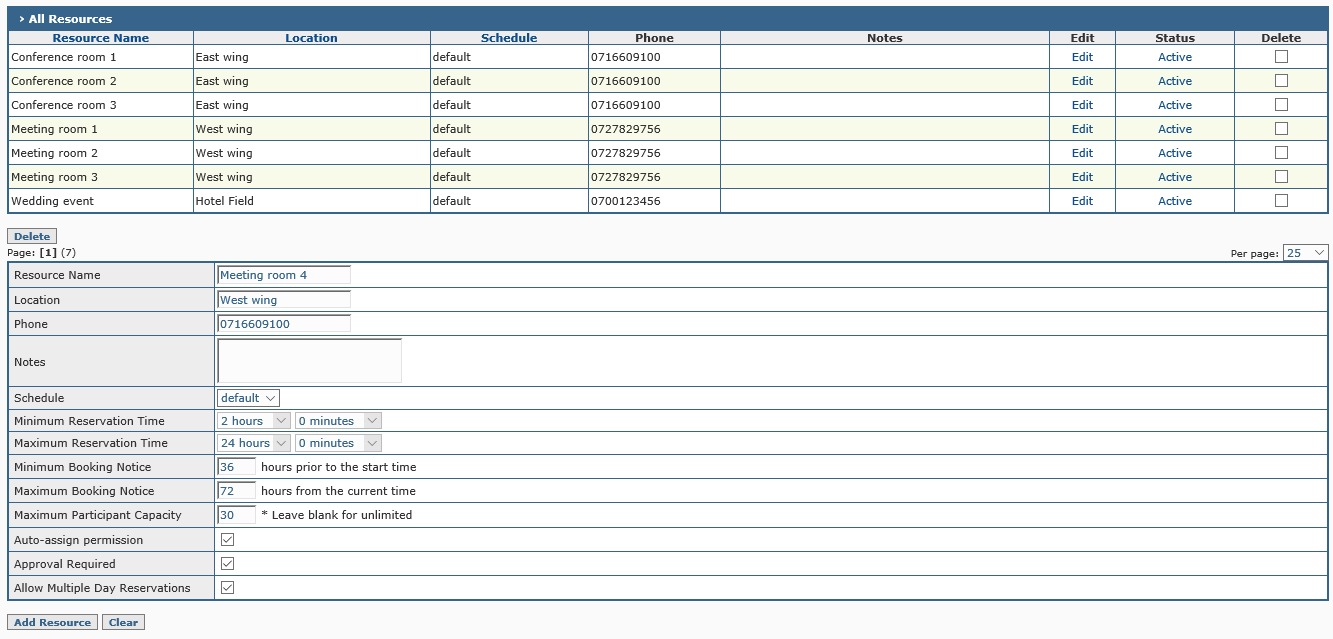
##### 5.4.1.1 Login test



##### 5.4.1.2 Making reservation test



##### 5.4.1.3 Resources management test



Include a copy of test data as appendix 3

### 5.5 conclusion

In conclusion, this Conference Facility Management System provides a lot benefits as it allow Receptionist and the Manager to organize and manage conference facilities i.e. events and equipments associated with the conference centers. The system hopefully would provide the necessary functions to the Event Manager in order to manage the event effectively, efficiently and thus knowing that the system will function smoothly. Besides that, it would be nice and interesting if users found the procedures used in booking or reservation and review to be user-friendly and flexible. In addition, this system will also help the Event Manager of the conference to users and alert them of the important issues such approval of their reservation, the availability of resources and cancelling of their reservations. This will provide a better understanding between the customers and facility management regarding the conference in order to effectively manage the conference. The Conference Facility Management System is expected to be used in conference centers Nakuru and outside Malaysia as this would portray not only IIUM but also Malaysia as a technology and educational hub.

### 5.6 recommendations

For the system to function to its best capability, the analyst recommends the following:

* Networking of the hotel’s computers to enable data communication at the hotel.
* The hotel server be connected to the internet. In addition the hotel should have a web site onto which the booking page will be loaded to enable online booking over the internet.
* The system should be troubleshoot and updated to maintain the high competence standards of the system. This is because, being a system, the Conference Facility Management System is bound to undergo system entropy.
* High security measures should be upheld in order to avoid theft crimes at the hotel.
* The latest versions of antivirus be installed and updated frequently in order to detect the many virus programs that are emerging daily in the IT society.
* Every employee of the hotel is identified uniquely by a user name and password to his/her office computer.

# REFERENCES

Hamad A. Aleshidi, and Akram M. Zeki VIRTUAL LEARNING ENVIRONMENT. Chapter in the book of Advancement in ELearning.

ISBN: 978-967-418-105-5. IIUM Press, Malaysia

Maimona J. Badi, Akram M. Zeki and Murni Mohmud. RELEVANCE OF MULTIMEDIA IN EDUCATION. Chapter in the book of Advancement in E-Learning. ISBN: 978-967-418-105-5.

IIUM Press, Malaysia.

Akram M. Zeki, Nor Hatini Saleh and Shayma Senan. ONLINE TEACHING NOTES. Chapter in the book of Advancement in ELearning.

ISBN: 978-967-418-105-5. IIUM Press, Malaysia

Akram M. Zeki, Iffah Nadhirah binti Rosdi, Azliza binti Aszenan and Aisha Hassan A. Hashim. DEVELOPMENT OF USER FRIENDLY E-LECTURE NOTES. Chapter in the book of Advancement in ELearning.

ISBN: 978-967-418-105-5. IIUM Press, Malaysia

EasyChair Conference System. http://www.easychair.org/ [Accessed on 2011]

OpenConf Conference 2012. http://www.openconf.com/ [Accessed on 211].

System Development Life cycle (Planning, Analysis, Design, Implementation, Testing and Maintenance) by Prashant Vadher, 2010. Retrieved from <http://prashantvadher.blogspot.com/2010/08/system> developmentlife- cycle-sdlc.html [Accessed on 2011]

Connolly, T. and Begg, C.(2005).Database systems;a practical approach to design, implementation and management. 5th Ed. England:Addison Wesley, Pearson Education.

Finiasz. M. and Baigneres, T. (2009). EPFL: iChair. School of computer and communication sciences.

# APPENDICES

## Appendix 1: system code

### 5.3.1 Login User

<?php

include\_once('lib/Template.class.php');

// Auth included in Template.php

$auth = new Auth();

$t = new Template();

$msg = '';

$resume = (isset($\_POST['resume'])) ? $\_POST['resume'] : '';

// Logging user out

if (isset($\_GET['logout'])) {

$auth->doLogout();

}

else if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

$msg = $auth->doLogin($\_POST['email'], $\_POST['password'], (isset($\_POST['setCookie']) ? 'y' : null), false, $resume, $\_POST['language']);

}

else if (isset($\_COOKIE['ID'])) {

$msg = $auth->doLogin('', '', 'y', $\_COOKIE['ID'], $resume); // Check if user has cookies set up. If so, log them in automatically

}

$t->printHTMLHeader();

// Print out logoImage if it exists

echo (!empty($conf['ui']['logoImage']))

? '<div align="center"><img src="' . $conf['ui']['logoImage'] . '" alt="logo" vspace="5"/></div>'

: '';

$t->startMain();

if (isset($\_GET['auth'])) {

$auth->printLoginForm(translate('You are not logged in!'), $\_GET['resume']);

}

else {

$auth->printLoginForm($msg);

}

$t->endMain();

// Print HTML footer

$t->printHTMLFooter();

?>

### 5.3.2 Administrator Control Panel

<?php

include\_once('lib/Template.class.php');

include\_once('lib/Admin.class.php');

$admin = new Admin(trim($\_GET['tool']));

$admin->user = new User(Auth::getCurrentID());

if (!$admin->user->get\_isadmin()) { $admin->user->is\_admin = Auth::isAdmin(); }

$t = new Template(translate('System Administration'));

$t->printHTMLHeader();

// Make sure this is the admin

if (!$admin->isUserAllowed()) {

CmnFns::do\_error\_box(translate('This is only accessable to the administrator') . '<br />'

. '<a href="ctrlpnl.php">' . translate('Back to My Control Panel') . '</a>');

}

$t->printWelcome();

$t->startMain();

if (!$admin->is\_error()) {

$admin->execute();

}

else {

CmnFns::do\_error\_box($admin->get\_error\_msg());

}

$t->endMain();

$t->printHTMLFooter();

?>

### 5.3.3 User Control panel

<?php

include\_once('lib/Template.class.php');

include\_once('lib/Utility.class.php');

include\_once('templates/cpanel.template.php');

if (!Auth::is\_logged\_in()) {

Auth::print\_login\_msg(); // Check if user is logged in

}

$t = new Template(translate('My Control Panel'));

$db = new DBEngine();

$t->printHTMLHeader();

$t->printWelcome();

$t->startMain();

$user = new User(Auth::getCurrentID());

$is\_group\_admin = $user->is\_group\_admin();

// Break table into 2 columns, put quick links on left side and all other tables on the right

startQuickLinksCol();

showQuickLinks(Auth::isAdmin(), $is\_group\_admin); // Print out My Quick Links

startDataDisplayCol();

$order = array('number');

$announcements = $db->get\_announcements(mktime());

showAnnouncementTable( $announcements, $db->get\_err() );

printCpanelBr();

// Valid order values in reservation retreival

$order = array('start\_date', 'name', 'starttime', 'endtime', 'created', 'modified');

$res = $db->get\_user\_reservations(Auth::getCurrentID(), CmnFns::get\_value\_order($order), CmnFns::get\_vert\_order());

showReservationTable($res, $db->get\_err()); // Print out My Reservations

printCpanelBr();

showInvitesTable($db->get\_user\_invitations(Auth::getCurrentID(), true), $db->get\_err());

printCpanelBr();

showParticipatingTable($db->get\_user\_invitations(Auth::getCurrentID(), false), $db->get\_err());

printCpanelBr();

if ($conf['app']['use\_perms']) {

showTrainingTable($db->get\_user\_permissions(Auth::getCurrentID()), $db->get\_err());

}

endDataDisplayCol();

$t->endMain();

$t->printHTMLFooter();

?>

### 5.3.4 Selecting Reservation

<?php

list($s\_sec, $s\_msec) = explode(' ', microtime()); // Start execution timer

/\*\*

\* Include Template class

\*/

include\_once('lib/Template.class.php');

/\*\*

\* Include scheduler-specific output functions

\*/

include\_once('lib/ResCalendar.class.php');

// Check that the user is logged in

if (!Auth::is\_logged\_in()) {

Auth::print\_login\_msg();

}

$t = new Template(translate('Resource Calendar'));

// Print HTML headers

$t->printHTMLHeader();

// Print welcome box

$t->printWelcome();

// Begin main table

$t->startMain();

$type = isset($\_GET['view']) ? $\_GET['view'] : MYCALENDARTYPE\_DAY;

$machid = isset($\_GET['machid']) ? $\_GET['machid'] : null;

$scheduleid = isset($\_GET['scheduleid']) ? $\_GET['scheduleid'] : null;

$calendar = new ResCalendar(Auth::getCurrentID(), $type, get\_calendar\_actual\_date(), $machid, $scheduleid);

$calendar->print\_calendar();

// End main table

$t->endMain();

list($e\_sec, $e\_msec) = explode(' ', microtime()); // End execution timer

$tot = ((float)$e\_sec + (float)$e\_msec) - ((float)$s\_sec + (float)$s\_msec);

echo '<!--Schedule printout time: ' . sprintf('%.16f', $tot) . ' seconds-->';

// Print HTML footer

$t->printHTMLFooter();

/\*\*

\* Sets the 'actualDate' field of the MyCalendar object

\* @param none

\* @return datestamp of the viewed date

\*/

function get\_calendar\_actual\_date() {

if (isset($\_GET['date'])) {

$date\_split = explode('-', $\_GET['date']);

}

else {

$date\_split = explode('-', date('m-d-Y', Time::getAdjustedTime(mktime(), date('H') \* 60)));

}

return mktime(0,0,0, $date\_split[0], $date\_split[1], $date\_split[2]);

}

?>

### 5.3.5 Making Reservation

<?php

include\_once('lib/Resource.class.php');

include\_once('lib/Template.class.php');

include\_once('lib/helpers/ReservationHelper.class.php');

include\_once('lib/Utility.class.php');

$timer = new Timer();

$timer->start();

$is\_blackout = (isset($\_GET['is\_blackout']) && ($\_GET['is\_blackout'] == '1'));

if ($is\_blackout) {

// Make sure user is logged in

if (!Auth::is\_logged\_in()) {

Auth::print\_login\_msg();

}

include\_once('lib/Blackout.class.php');

$Class = 'Blackout';

$\_POST['minres'] = $\_POST['maxRes'] = null;

}

else {

include\_once('lib/Reservation.class.php');

$Class = 'Reservation';

}

if ((!isset($\_GET['read\_only']) || !$\_GET['read\_only']) && $conf['app']['readOnlyDetails']) {

// Make sure user is logged in

if (!Auth::is\_logged\_in()) {

Auth::print\_login\_msg();

}

}

$t = new Template();

if (isset($\_POST['btnSubmit']) && strstr($\_SERVER['HTTP\_REFERER'], $\_SERVER['PHP\_SELF'])) {

$t->set\_title(translate("Processing $Class"));

$t->printHTMLHeader();

$t->startMain();

process\_reservation($\_POST['fn']);

}

else {

$res\_info = getResInfo();

$t->set\_title($res\_info['title']);

$t->printHTMLHeader();

$t->startMain();

present\_reservation($res\_info['resid']);

}

// End main table

$t->endMain();

$timer->stop();

$timer->print\_comment();

// Print HTML footer

$t->printHTMLFooter();

/\*\*

\* Processes a reservation request (add/del/edit)

\* @param string $fn function to perform

\*/

function process\_reservation($fn)

{

$success = false;

global $Class;

$is\_pending = (isset($\_POST['pending']) && $\_POST['pending']);

if (isset($\_POST['start\_date'])) // Parse the POST-ed starting and ending dates

{

$sd = explode(INTERNAL\_DATE\_SEPERATOR, $\_POST['start\_date']);

$ed = explode(INTERNAL\_DATE\_SEPERATOR, $\_POST['end\_date']);

$start\_date = mktime(0,0,0, $sd[0], $sd[1], $sd[2]);

$end\_date = mktime(0,0,0, $ed[0], $ed[1], $ed[2]);

}

if (isset($\_POST['resid']))

{

$res = new $Class($\_POST['resid'], false, $is\_pending);

}

else if (isset($\_GET['resid']))

{

$res = new $Class($\_GET['resid'], false, $is\_pending);

}

else

{

// New reservation

$res = new $Class(null, false, $is\_pending);

if ($\_POST['interval'] != 'none') // Check for reservation repeation

{

if ($start\_date == $end\_date)

{

$res->is\_repeat = true;

$days = isset($\_POST['repeat\_day']) ? $\_POST['repeat\_day'] : NULL;

$week\_num = isset($\_POST['week\_number']) ? $\_POST['week\_number'] : NULL;

$repeat = CmnFns::get\_repeat\_dates($start\_date, $\_POST['interval'], $days, $\_POST['repeat\_until'], $\_POST['frequency'], $week\_num);

}

else

{

// Cannot repeat multi-day reservations

$repeat = array($start\_date);

$res->is\_repeat = false;

}

}

else

{

$repeat = array($start\_date);

$res->is\_repeat = false;

}

}

$cur\_user = new User(Auth::getCurrentID());

$res->adminMode = Auth::isAdmin() || $cur\_user->get\_isadmin() || ($fn != 'create' && $cur\_user->is\_group\_admin($res->user->get\_groupids()));

if (Auth::isAdmin() || $cur\_user->get\_isadmin())

{

$res->is\_pending = false;

}

if ($fn == 'create' || $fn == 'modify')

{

$helper = new ReservationHelper();

$util = new Utility();

$orig = (isset($\_POST['orig\_invited\_users']) && count($\_POST['orig\_invited\_users']) > 0) ? $\_POST['orig\_invited\_users'] : array();

$invited = (isset($\_POST['invited\_users'])) ? $\_POST['invited\_users'] : array();

$removed = (isset($\_POST['removed\_users'])) ? $\_POST['removed\_users'] : array();

$participating = (isset($\_POST['participating\_users'])) ? $\_POST['participating\_users'] : array();

$users\_to\_remove = $helper->getRowsForRemoval($orig, $removed, $invited);

$users\_to\_invite = $helper->getRowsForInvitation($orig, $invited);

$unchanged\_users = $helper->getUnchangedUsers($orig, $invited, $participating);

$orig\_resources = (isset($\_POST['orig\_resources']) && count($\_POST['orig\_resources']) > 0) ? $\_POST['orig\_resources'] : array();

$selected\_resources = (isset($\_POST['selected\_resources']) && count($\_POST['selected\_resources']) > 0) ? $\_POST['selected\_resources'] : array();

$resources\_to\_add = $util->getAddedItems($orig\_resources, $selected\_resources);

$resources\_to\_remove = $util->getRemovedItems($orig\_resources, $selected\_resources);

$res->user = new User($\_POST['memberid']);

$res->start\_date= $start\_date;

$res->end\_date = $end\_date;

$res->start = $\_POST['starttime'];

$res->end = $\_POST['endtime'];

$res->summary = stripslashes($\_POST['summary']);

$res->allow\_participation = (int)isset($\_POST['allow\_participation']);

$res->allow\_anon\_participation = (int)isset($\_POST['allow\_anon\_participation']);

$res->reminderid = isset($\_POST['reminderid']) ? $\_POST['reminderid'] : null;

$res->reminder\_minutes\_prior = isset($\_POST['reminder\_minutes\_prior']) ? intval($\_POST['reminder\_minutes\_prior']) : 0;

}

if ($fn == 'create')

{

$res->resource = new Resource($\_POST['machid']);

$res->scheduleid= $\_POST['scheduleid'];

$res->repeat = $repeat;

$res->add\_res($users\_to\_invite, $resources\_to\_add);

}

else if ($fn == 'modify')

{

$res->summary = str\_replace("\n", '', $\_POST['summary']);

$res->mod\_res($users\_to\_invite, $users\_to\_remove, $unchanged\_users, $resources\_to\_add, $resources\_to\_remove, isset($\_POST['del']), isset($\_POST['mod\_recur']));

}

else if ($fn == 'delete')

{

$res->del\_res(isset($\_POST['mod\_recur']));

}

else if ($fn == 'approve')

{

$res->approve\_res(isset($\_POST['mod\_recur']));

}

}

/\*\*

\* Prints out reservation info depending on what parameters

\* were passed in through the query string

\* @param none

\*/

function present\_reservation($resid) {

global $Class;

// Get info about this reservation

$res = new $Class($resid, false, false, $\_GET['scheduleid']);

// Load the properties

if ($resid == null) {

$res->resource = new Resource($\_GET['machid']);

$res->start\_date = $\_GET['start\_date'];

$res->end\_date = $\_GET['start\_date'];

$res->user = new User(Auth::getCurrentID());

$res->is\_pending = $\_GET['pending'];

$res->start = $\_GET['starttime'];

$res->end = $\_GET['endtime'];

}

$cur\_user = new User(Auth::getCurrentID());

$res->adminMode = Auth::isAdmin() || $cur\_user->get\_isadmin() || $cur\_user->is\_group\_admin($res->user->get\_groupids() );

if (Auth::isAdmin() || $cur\_user->get\_isadmin())

{

$res->is\_pending = false;

}

$res->set\_type($\_GET['type']);

$res->print\_res();

}

/\*\*

\* Return array of data from query string about this reservation

\* or about a new reservation being created

\* @param none

\*/

function getResInfo() {

$res\_info = array();

global $Class;

// Determine title and set needed variables

$res\_info['type'] = $\_GET['type'];

switch($res\_info['type']) {

case RES\_TYPE\_ADD :

$res\_info['title'] = "New $Class";

$res\_info['resid'] = null;

break;

case RES\_TYPE\_MODIFY :

$res\_info['title'] = "Modify $Class";

$res\_info['resid'] = $\_GET['resid'];

break;

case RES\_TYPE\_DELETE :

$res\_info['title'] = "Delete $Class";

$res\_info['resid'] = $\_GET['resid'];

break;

case RES\_TYPE\_APPROVE :

$res\_info['title'] = "Approve $Class";

$res\_info['resid'] = $\_GET['resid'];

break;

default : $res\_info['title'] = "View $Class";

$res\_info['resid'] = $\_GET['resid'];

break;

}

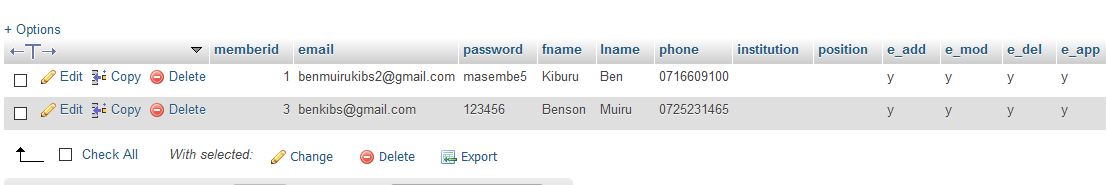
return $res\_info;

}

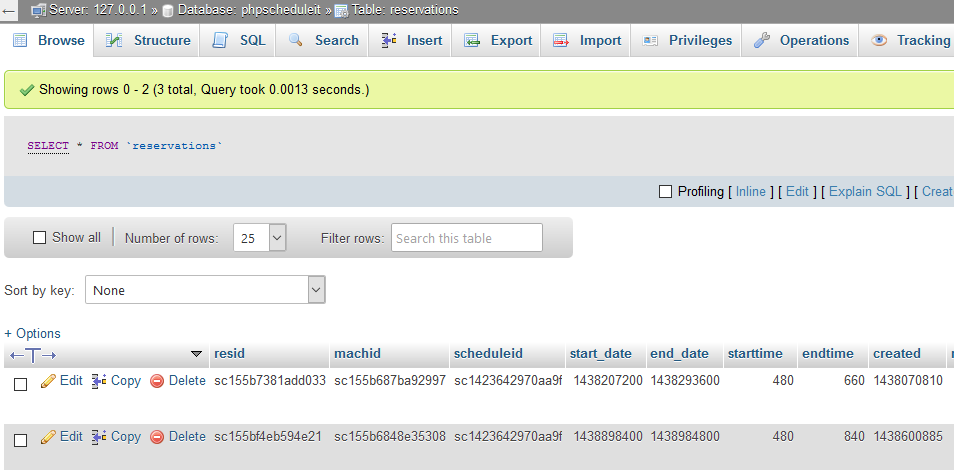
?>

## Appendix 2: Test data

### Test data for Login



### Test data for Reservation



### Test data for Resources

