****

**EGERTON UNIVERSITY**

**COMPUTER SCIENCE DEPARTMENT**

**COMP 402: SYSTEM DESCRIPTION DESIGN**

**MOBDOC MEDICAL APPOINTMENT SYSTEM**

**NAME: CRESCENT MBWIKA MUSYOKI**

**REG.NO: S13/20730/13**

**DATE:**

**Supervised by: MR. KORIR**

**Presented to:**

For the partial fulfillment of Degree in Computer Science of Egerton University

Table of Contents

1 INTRODUCTION 1

1.1 Purpose and Scope 1

1.2 Project Executive Summary 1

**1.2.1 System Overview** 1

**1.2.2 Design Constraints** 1

**1.2.3 Future Contingencies** 1

1.2.4 Document Organization 1

2 SYSTEM ARCHITECTURE 2

3 FILES AND DATABASE DESIGN 3

3.1 Database Management System Files   3

3.2 Non-Database Management System Files   7

4 HUMAN-MACHINE INTERFACES 8

4.1 Inputs 8

4.2 Outputs   9

5 DETAILED DESIGN 13

5.1 Hardware Detailed Design   13

5.2 Software Detailed Design   13

**5.2.1 Login module** 13

**5.2.2 Signup module** 13

**5.2.3 Practice module** 13

**5.2.4 User module** 13

**5.2.5 Specialist module** 13

**5.2.6 Appointment module** 13

**5.2.7 Admin module** 13

6 EXTERNAL INTERFACES 14

6.1 INTERFACE ARCHITECTURE 14

6.2 INTERFACE DETAILED DESIGN 14

7. SYSTEM INTEGRITY CONTROLS 15

# 1 INTRODUCTION

## 1.1 Purpose and Scope

The key purpose of the software design description is to provide the design description of MOBDOC Medical Appointment System in the form of object-oriented analysis. The document comprises use case, data flow diagrams and prior to that the context level diagram which shows the working model of the system.

The basic architecture of the web application is done with the help of PHP and bootstrap for design. The supervisor of the project would be in charge of this document to make any necessary changes related to the diagrammatic description and the contents of the index.

## 1.2 Project Executive Summary

### **1.2.1 System Overview**

There exist little support for medical services on-line. With the current condition, a user can only search for a specialist and place their appointment. The disadvantage of this is that one isn’t able to make a choice easily on which specialist to visit. This because there is no past user reviews, ratings or event patients’ comments. It also does not indicate which facility either of the specialists works for.

Another current situation is that most of the appointment systems a made and deployed specifically for a given facility. This denies users the chance to compare the services provided with the ones provided by other facilities.

### **1.2.2 Design Constraints**

The main trade-off expected in the proposed system design is the SMS integration, which will be depended on a third party system.

### **1.2.3 Future Contingencies**

Some of the foreseen contingencies that might arise during the development process include a change in the way data is displayed after being retrieved from the system. This will lead to a change in the layout design of the given affected module.

## 1.2.4 Document Organization

The document is organized in the following sections to ensure a better understanding of the system: system overview, system architecture, file and database design, human-machine interface, detailed system design, external interfaces and system integrity controls.

# 2 SYSTEM ARCHITECTURE

The project’s scope is developing a web-based system for medical scheduling with an aim of providing quality and reliable services. This system will mainly have a web interface where patients or users can get a variety of services. These include making bookings and reservations for appointment time-slots and post reviews from wherever they are across the country.

On the specialist side, the specialist will be able to respond to patients’ appointments, which will involve confirmation, updating or cancelling.

On the administrator side, the administrator will have a general overview of the whole system activities and processes. He/she will also be responsible for confirming any new registered specialist details to ensure that they are true as recorded in the Medical Board of Kenya.

This system will have 3 levels of login:

Administrator level, which will perform the following tasks:

* Manage system users.
* Confirming of newly registered specialists.

Specialist level, which will perform the following tasks:

* Add a schedule.
* Manage appointments.

Patient level, which will perform the following tasks:

* Make an appointment.
* Write a specialist review.

# 3 FILES AND DATABASE DESIGN

## 3.1 Database Management System Files

Data Storage layer will contain a database where all the data is stored. For the proposed system, the database layer will be in a form of a MYSQL database, which will be hosted on the online server. The following outlines the various database tables, their structure and content.

**Structure for appointment table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| appointmentId | int(11) | No |  |
| appointmentBookTime | timestamp | No | CURRENT\_TIMESTAMP |
| appointmentStatus | tinyint(1) | No |  |
| patientId | int(11) | No |  |
| specialistId | int(11) | No |  |
| timeslotId | int(11) | No |  |
| patientName | varchar(20) | No |  |
| patientMobile | varchar(15) | No |  |
| visitingReason | text | No |  |

*Table 1*

**Structure for facilities table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| facilityId | int(11) | No |  |
| registrationNumber | varchar(20) | No |  |
| facilityName | varchar(20) | No |  |
| createdBy | int(11) | No |  |
| addedOn | timestamp | No | CURRENT\_TIMESTAMP |
| facilityStatus | tinyint(1) | No | 0 |

*Table 2*

**Structure for insurance table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| insuranceId | int(11) | No |  |
| insuranceName | varchar(20) | No |  |
| insuranceTag | varchar20) | No |  |
| addedOn | timestamp | No | CURRENT\_TIMESTAMP |
| insuranceStatus | tinyint(1) | No | 1 |

*Table 3*

**Structure for review table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| reviewId | int(11) | No |  |
| reviewPublisher | int(11) | No |  |
| reviewTarget | int(11) | No |  |
| reviewContent | text | No |  |
| reviewDate | timestamp | No | CURRENT\_TIMESTAMP |
| reviewStatus | tinyint(1) | No | 0 |

*Table 4*

**Structure for specialist table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| specialistId | int(11) | No |  |
| userId | int(11) | No |  |
| qualification | text | No |  |
| displayName | Varchar(20) | No |  |
| registrationNumber | varchar(20) | No |  |
| registrationNumber | varchar(20) | No |  |
| facilityId | int(11) | Yes | 0 |
| specialistStatus | int(11) | No | 0 |

*Table 5*

**Structure for specialist\_insurance table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| specInsId | int(11) | No |  |
| specialistId | int(11) | No |  |
| insuranceId | int(11) | No |  |
| specInsStatus | int(11) | No | 1 |

*Table 6*

**Structure for specialty table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| specialtyId | int(11) | No |  |
| specialtyName | varchar(20) | No |  |
| specialtyStatus | tinyint(1) | No | 1 |

*Table 7*

**Structure for timeslot table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| timeslotId | int(11) | No |  |
| timeslotDate | date | No |  |
| timeslotTime | time | No |  |
| specialistId | int(11) | No |  |
| timeslotStatus | tinyint(1) | No | 1 |

*Table 8*

**Structure for user table**

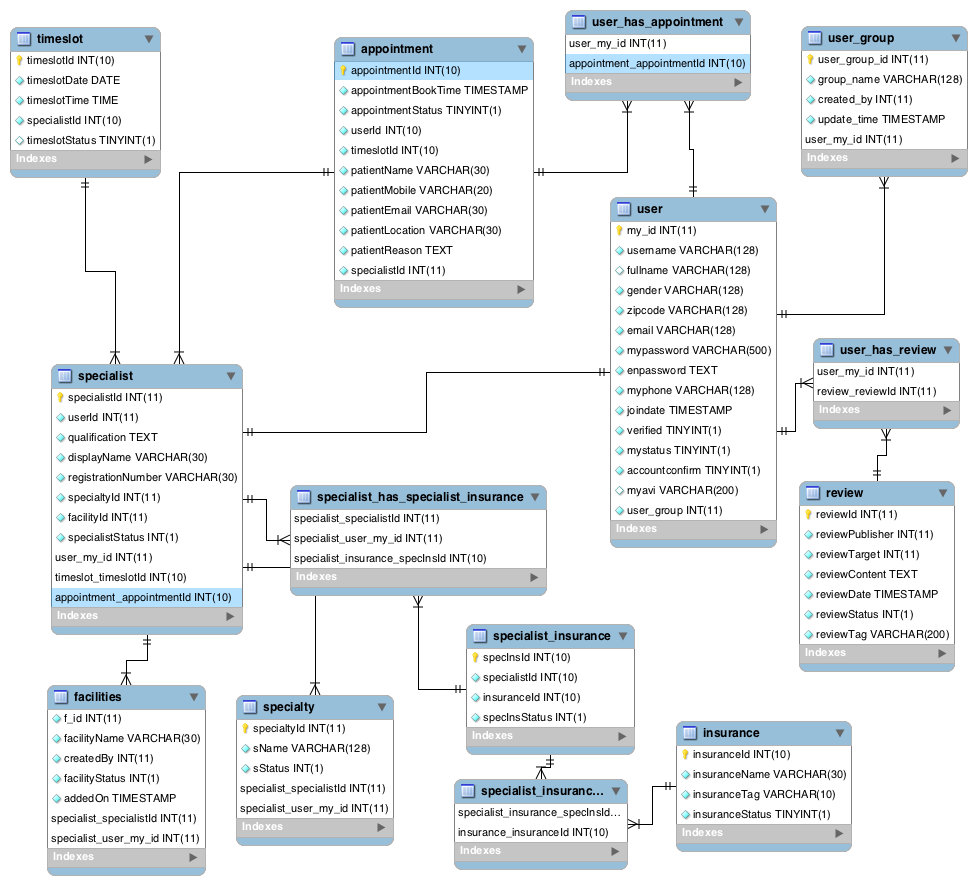
|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| my\_id | int(11) | No |  |
| Username | varchar(20) | No |  |
| Fullname | varchar(20) | No |  |
| Gender | char(1) | No |  |
| Zipcode | varchar (20) | No |  |
| Email | varchar(20) | No |  |
| mypassword | varchar (200) | No |  |
| enpassword | text | No |  |
| Joindate | timestamp | No | CURRENT\_TIMESTAMP |
| Mystatus | tinyint (1) | No | 0 |
| Mystatus | tinyint (1) | No | 0 |
| user\_group | int (22) | No |  |

*Table 9*

**Structure for user\_group table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Default |
| user\_group\_id | int(11) | No |  |
| group\_name | varchar (30) | No |  |
| created\_by | int(11) | No |  |
| update\_time | timestamp | No | CURRENT\_TIMESTAMP |

Below is a diagram to show the relationship between the different tables.



*Fig 1*

## 3.2 Non-Database Management System Files

The proposed system will also involve the file system of storage. This will be implemented in storing user profile pictures, which will be stored in a given specified directory in the online web server.

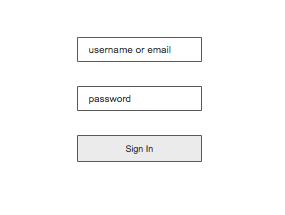
# 4 HUMAN-MACHINE INTERFACES

## 4.1 Inputs

The proposed system will use html forms as the main mode to get data from the user. Some of the forms include:

**4.1.1 Login**

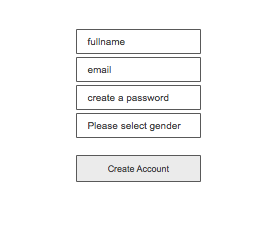
This form will collect an email address and an account password from the user, as they are stored in the database.



*Fig 2*

**4.1.2 Create account**

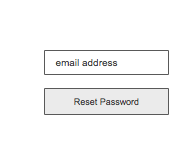
This form will collect a user’s full name, suggested account password, an email address and gender to be stored in the database for authentication.

1

*Fig 3*

**4.1.3 Reset password**

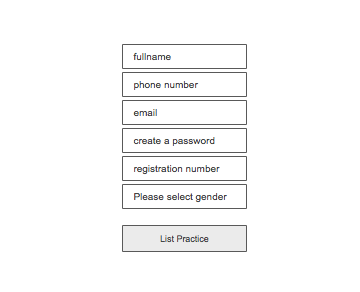
The reset password form will collect the user’s email address in order to send the account recovery details.



*Fig 4*

**4.1.4 List practice**

This form will be used to collect a specialist’s information for registration purposes. These will include full name, phone number, email address, a password, registration number and their area of specialty.



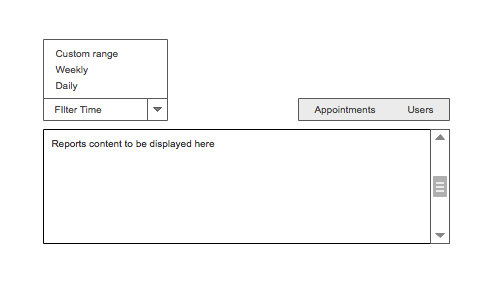
*Fig 5*

## 4.2 Outputs

The main system outputs include reports, which will have both the system users and the appointments, which have been made all through.

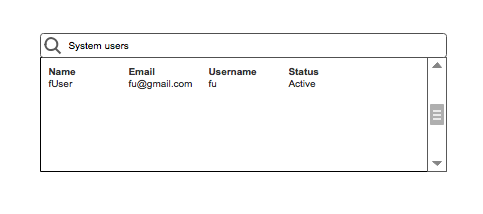
A demonstration is as shown bellow:

**4.2.1 Reports output**



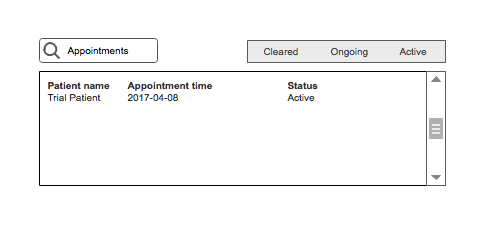
*Fig 6*

**4.2.2 Users output**



*Fig 7*

**4.2.3 Appointments output**



*Fig 8*

Below is an illustration of how the data will flow through the main system processes.

1. Administrator

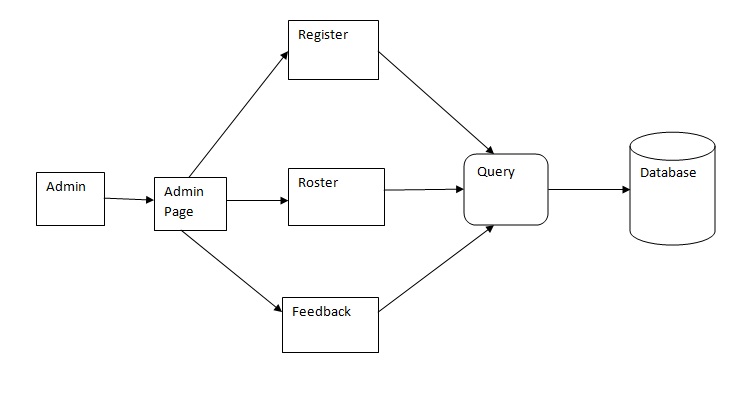


Fig 9

1. Login for both user and administrator

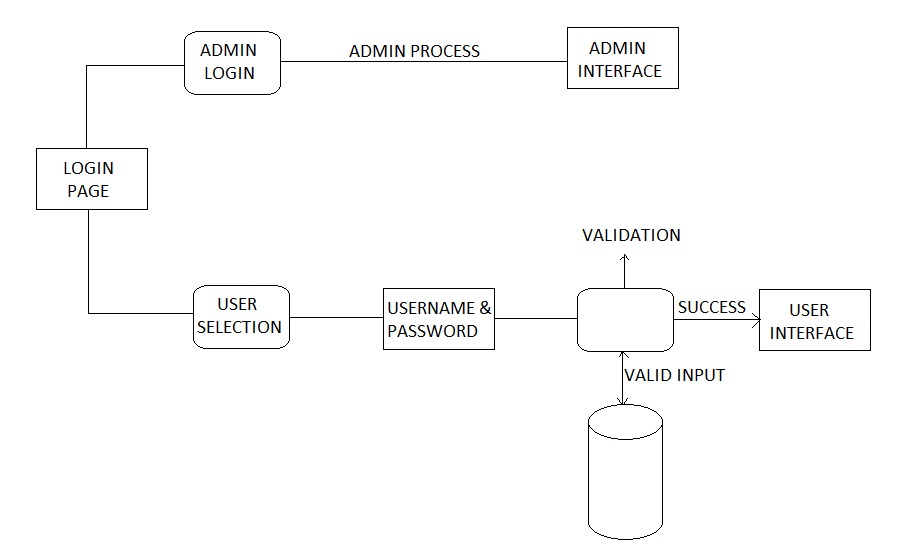


Fig 10

1. Patient/User Registration

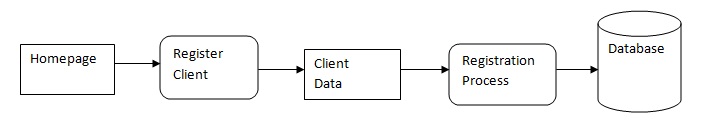
a

Fig 10

# 5 DETAILED DESIGN

## 5.1 Hardware Detailed Design

The system will be accessed through a web server. The main requirement will be any device with a web browser and Internet connection.

## 5.2 Software Detailed Design

The system will have several modules to ensure it’s full functionality as expected. The modules include:

### **5.2.1 Login module**

This module deals with the authentication of system user that is administrator, specialist and patients. It will require the users to have first been registered into the system. It will include a username or email address and a password.

### **5.2.2 Signup module**

This module handles patient registration into the system.

### **5.2.3 Practice module**

This module handles the registration of specialists into the system

### **5.2.4 User module**

This module deals with updating of personal profile and system settings.

### **5.2.5 Specialist module**

This will deal with setting up and updating specialist’s daily schedules and availability

### **5.2.6 Appointment module**

This module will handle the process of patient booking and modifying the appointment details. It will also deal with the confirmation and rescheduling on the side of the specialist.

### **5.2.7 Admin module**

This module allows the administrator to manage the general functionality of the system. This module allows the admin to generate report on the preferred duration i.e. daily, weekly, monthly or annually.

# 

# 6 EXTERNAL INTERFACES

## 6.1 INTERFACE ARCHITECTURE

External interfacing will facilitate the communication between the application and the external systems. Communication will be through TCP/IP since all the communications will be done through the Internet.

Specific network architectures implemented for the client, server and the database PCs does not matter, all that is needed is an active link of communication between these components and any other external systems that the application will be exchanging information with.

## 6.2 INTERFACE DETAILED DESIGN

Since communication is through TCP/IP, there is no need to reformat data before it is transmitted or received over the interface between two different communicating modules.

Data exchange over the interface will be purely regulated by the protocol. Errors arising from such exchange will be handled dynamically by the protocol.

Errors arising will be handled and a brief error description given to the user in HTML format.

# 7. SYSTEM INTEGRITY CONTROLS

To ensure a strong system base security, the proposed system will implement data encryption before any piece of critical and confidential information or data is submitted and stored in the database. This will be done through the following:

1. Authorization control. This ensures that users can only access data that belongs to their user groups based on user roles.
2. User-data access control. Access to the data in the database will be restricted to various user groups based on the user roles.
3. System authenticity. This ensures that only authorized users are allowed to access the system.
4. Data and information encryption. This is a security technique that encrypts the raw data that the user submits before it’s stored in the database.