E-Medihelp - Single Platform for All Medical Centers

A dissertation submitted for the Degree of Master of Information Technology

P.G. Rajitha Sampath

University of Colombo School of Computing

2019

Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name:

Registration Number:

Index Number:

____

Signature: Date:

This is to certify that this thesis is based on the work of

Mr./Ms.

under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name:

____

Signature: Date:

Acknowledgement

I would like to express my special thanks of gratitude for UCSC for providing me this golden opportunity to become a master level graduate in Information Technology. This will lead to uplift my career progression in the future. My sincere and grateful thanks to my supervisor Mr. Kapila Dias who encouraged and guided me to do this wonderful project on the topic E-Medihelp, which also helped me in doing a lot of research and I came to know about so many new things. I get this opportunity to thank all UCSC staff for their dedication towards the masters.

Also, I would like to thank Dr. Somasiri. Because this system development idea initiated by him. He was provided great support by spending his valuable time on my project discussion and offering more relevant contact for gathering information for the success of this project.

Then I would like to thank my parent for encouraging me and supporting me throughout the project development. Finally, I would like to thank my friends and everybody who helped me a lot in finalizing this project.

Abstract

A healthy society is a crucial fact for development of any country. Sri Lanka government hospitals and private sector hospitals facilitate health care services. The Sri Lanka hospital system is not enough to cater the each and every person. Hence need to establish small and medium size medical centers across the country. Most of the doctors open their own medical centers by identifying this fact. This project provides solution for many issues faced small and medium size medical centers by implementing universal web application for medical service industry. This system helpful for patient, doctor, as well as any one interacts with medical center.

From a doctor point of view doctors able to create their own medical center account. This account related to his medical center. All medical center activities can manage using that account. Like that any doctor can create medical center account and use it for his medical center apart of establishing an information system to his medical center. Doctor able to see patient medical history which includes patient diseases and treatments he had for that disease. Doctor able to see his appointment list before he comes to the medical center. Further doctor able to record patient disease details and the doctor suggested treatment for that disease. From a patient point of view patient able to use this system to access all small and medium size medical centers in a single place. Patient able to see the nearest medical center and doctor available time, current appointments. Patient profile can use to showcase his all existing medical records. It’s helpful to any doctor, he is going to contact getting an idea about his diseases. Patient able to make the appointment by checking available doctor.

The object-oriented programming concepts were used to implement this web-based information system. Waterfall model was used as a software development approach since the requirements are clear and vague. Bootstrap was used to system interface design since it is modern technique to build web pages easily. JEE was used as a programming language and WAMP server was used as a local database server. SQL use for backend database.

Table of content

List of figures

List of tables

1.0. Introduction

When population increase medicine is essential fact. Hence need small and medium size medical centers across the country. Currently having a number of medical centers. This application focused on connect each medical center and patient in a single place. And this system helpful to conduct treatments right time and facilitate a quick service.

1.1. Identification of the Problem

Unavailability of the right software solution for managing small and medium size medical centers face many problems. Further unavailable information system for connecting all small and medium size medical service providers.

Problems they face

- Patient appointment management problem
- Patient waiting time increase
- Difficult to maintain patient documents
- Previous patient records unavailability

Those are common problems to most of small and medium size medical centers. To address these issues, need a medical center information system. But spending much cost and implementing information system for small medical center not practical. Hence most of the medical centers reluctant to spend for that.

Identified Problem

Unavailability of computerized information system for small and medium size medical centers across the healthcare industry in Sri Lanka.

1.2. Motivation

My cousin's doctor has medical center. When I went there most of the times can see the number of patients waiting there. Sometimes the doctor takes too much time to per patient because of unavailability of patient disease history with him. I thought how to provide effective service to a patient in this kind of medical centers using information technology.

One day I got an opportunity to discuss with a doctor regarding this scenario. He told me actual problems he faces when operating his medical center. He suggests me to implement information systems to his medical center.

Then I looked from a patient point of view. Then I recognize currently Sri Lanka healthcare industry not having universal software for access all small and medium size medical centers in a single place.

By considering patient and doctors requirement and adding business value to new software I planned to develop a universal information system for small and medium size medical centers. It will be web based free application which provides facility to doctors create their own medical center account. This account related to his medical center. All medical center activities can manage using that account. Patient able to see the nearest medical center and doctor available time, current appointments.

Then got an opportunity to meet a doctor again and introduced my project proposal on the propose web-based system. During the meeting doctor advised to change some features and further added new changes regarding the functions of the medical center. Then doctor appreciated my effort and promised to give his support. Thereafter prepared a document related to discuss things and made changes accordingly.

1.3. Goals/Objectives

Developing free and universal application for the healthcare industry is a main objective of this project. It’s web based application for small and medium size medical centers. It’s totally free and doctor able to create his medical center account apart of establishing an information system to his medical center.

This project will aim to help two parties

- **Doctor**
    - Provide free web-based application to manage medical center.
    - Provide better understanding about his patient.
    - Increase efficiency of the medical center.
- **Patient**
    - Find nearest medical center
    - Reduce waiting time in the medical center

1.4. Scope

Proposed system target is a small and medium size medical center. Any adult person can create his patient account based on NIC number. This system focuses on the patient doctor relation only. There is no medical center other activity management, such as pharmacy. This system not address to large hospitals, but this can expand to entire healthcare industry in Sri Lanka. Medical center account creation only based on doctors.

- **Doctor**
  - Doctors able to create a medical center profile under his doctor identification number.
  - Doctor should be able to see patient medical history which includes patient diseases and treatments he had for that disease.
  - Doctor able to see his appointment list before he comes to the medical center. Which is helpful for getting to understand about patients he going to treat.
  - Record the patient details.
  - Doctor prescription is a system generated printout and doctor just need to sign and handover to the patient.
  - Record patient disease details and the doctor suggested treatment for that disease.

- **Patient**
  - Patient able to create his own profile based on his NIC number.
  - This profile can use to showcase his all existing medical records. It's helpful to any doctor, he is going to contact getting an idea about his diseases.
  - Patient able to see available doctors near to his current location and see the number of current appointments for that doctor.
  - Patient able to make the appointment by checking available doctor.

2.0. Analysis

2.1. Introduction

Different medical centers have different systems. But still don’t have this kind of common system for entire medical services industry. In this chapter will be discussed this system development approach.

2.2. Recommended hardware requirements

Recommended hardware requirement shown in table 2.1.

Table 2.1 :- Recommended hardware requirements

Processer	Intel Core i3 2.20 GHz or above
RAM	One GB RAM or above recommended
VGA	VGA graphics in 640x480x16 resolution or above
Hard Disk	Standard hard disc capacity is recommended (160 GB or above)
Monitor	15” monitor or 17” monitor recommended
Internet Connection	One Mbps download and zero point five Mbps upload speed

2.3. Recommended operating systems

To access this web-based application, need operating system with the web browser. An operating system plays an intermediate role between application programs and hardware devices. An operating system needed to run the web browser on the computer. Microsoft Windows-based operating system such as Windows 7 or above version is suitable for launch this web application.

2.4. Similar web systems related to medical centers

Sri Lanka private hospitals have their own web application which is able to operate their entire branch network. As an example, Nawaloka Hospital internal web application for managing their health care centers provide many services. Such as doctor channeling, payment handling, check doctor availability, employee management, maintain medicine stocks. Most of the other reputed private hospitals also have that kind of web applications.

The government hospital network also having their own web application to manage each and every hospital. It’s helpful manage hospitals remotely from the health ministry. It is not perfectly working with rural hospitals because of less IT infrastructure. However, their all major hospitals and laboratories work with web applications.

E-Channeling is the one of the best web application for channel doctors from any ware. It provides facilities to patient, such as find relevant doctors from registered hospitals, make appointments, etc.

2.5 Similarities between proposed web-based system and described websites

- Channel doctor
- Handle payment
- Find medical center
- Contact detail
- Customer inquiry

2.6 Advantages of the proposed web-based system when comparing with similar systems

Proposed web application caters doctors as well as the patient. This application may fill the gap of not having a single platform for the whole medical industry. Because this application provides facilities to any doctor create their own medical center account. Therefor any small and medium size medical center can use this application as their primary medical center application.

When consider patient point of view they able to search nearest medical center from their current location. They can see doctor availability and make an appointment through the system. If they have patent account, they can store all medical records of them. Then it will help full to continue treatment from any doctor. Before the treatment doctor can access patient’s, previous medical records and prepare for good treatment.

2.7 Analysis of the proposed web-based system

2.7.1 Requirement Gathering

Requirement gathering is the process of addressing the needs and conditions of the system. There are few requirements gathering techniques are available such as sampling, questionnaires, interviews, background reading, and observations.

Observation

Observation is look and feel how the things going on actual environment. It's basically studies of users in their day to day activities and performance of assigned tasks and duties. While observation can understand that process flows, awkward steps, pain points and opportunities for improvement.

Interview

The interview is one of the common way to data gathering. Before conducting the interview needs to prepare for it because within the limited time slot of interview need to gather everything. There are some common practices to follow when conducting interviews because it affects peoples' emotions.

2.7.2 Requirements gathering on this proposed web-based system

Interview and observation basically use for requirement gathering in this propose system. Visit to the cousin doctor's medical center after his approval to observe everything happen in there. Then noted how doctor interaction with the patient, how make patient appointment, how issue medicine to the patient. At the end of the day conducted interviews with doctor, pharmacist and admin staff. Then got an opportunity to talk with few patients and identify their problems of having treatment from small medical centers.

After that prepared document and at the next meeting prepared document was shown to the doctor and changes were done with the supervision of the doctor. Then doctor introduces me to his few friends, which they are conducting medical centers. It's become a big opportunity to me design, web application by comparing different medical centers conditions. Able to visit the other four medical centers. They are located in different areas in the country (Piliyandala, Haputhale, Beliaththa).

After these iterative meetings with doctors proper and precise requirements were gathered. After all of them finalized software requirement document with my cousin's doctor.

All requirements were gathered under three categories as user requirements, business requirements and system requirements. Requirement categories shown in table 2.2.

Table 2.2 :- Requirement category table

Type of the requirement	**Contents**
User requirements	User inputs and outputs, user point of view, user goals
Business requirements	Client point of view, scope of the project, business objects
System requirements	Functional and non-functional requirements

2.7.3 Requirement Analysis

Gathered requirements need to carefully analyze. Because software specification develops by using analyzed requirement. Therefor requirement analysis plays a major role in a software development process. Early days it was done as a requirement before the project begins.

Usually requirements analysis occupies capturing both functional and non-functional requirements. The successfulness of the project depends on the way of doing the requirement analysis.

When it comes to actual problem domain its problem analysis was done in a precise manner. Requirements analysis involves frequent communication with medical center staff, doctor and patient. Its need to determine specific features are expected, by removing the resolution of conflict or ambiguity in gathering requirements and ensuring documentation of all aspects of the project development process from start to finish. The ultimate goal of requirement analysis should be proposed system matches to client needs.

Functional requirements

A functional requirement in software engineering can be defined as functions of a software system or its component. Basically, function has three step input, process and output. In software system functional requirements are data manipulation, data processing, data calculation and other specific functionality that defines what a system is supposed to accomplish. Use case diagram uses to capture the functional requirements.

Functional requirements of the proposed web based system

- System Signup

There is a sign up for the patient to create their account and doctors to create their medical center account.

- System Login

This system has four login options.

  - Patient

There is a patient login for find medical center and make an appointment, Upload medical information.

- Doctor (Medical Center)

There is a doctor login by based on creating medical center account. He has all administrative privileges for his medical center account.

- Staff (Medical Center)

Staff login based on medical center account. Patient appointment managing and other medical center activities can manage using this login.

- Admin

A system administrator has full access of the system. Using his dashboard able to manage entire application.

- Search medical center

Patient able to search medical centers of his area and find doctor availability.

- Upload documents

Patient able to upload his medical reports to his account. Doctors and medical center admin staff able to upload medical center document and patient documents.

- Generate document

Medical center staff able to download prescription about patient treatment which is crate by using doctor entered details.

- Payment handling

In medical center account have to separate tab for handling patient payment.

Non-functional requirements

Nonfunctional requirements are common to the any software system. Because it defines the qualities of a system and they are essential. These requirements specify criteria that can be used to judge the operation of a software system, rather than specific behaviors.

Nonfunctional requirements of the propose web based system

- Usability

Currently users not having this kind of experience. Therefor after developing this system, it will be a new user experience. In Sri Lanka currently having well computer literacy. Therefor people can easily use this system. When the system is going complex, then people are not very confident in handling. The system needs to be developed with user convenience and provide the best accuracy.

- Security

Patient details and medical centers details are very confident. Hence system administrator has a higher responsibility to maintain the security and make sure that the information is not available for irrelevant parties. The system must capture all user's login details. According to the user system must be adjusted by delivering only relevant data. Authorized users can only access and modified system data.

- Availability

This web-based system is available at any time unless web server was down. 24 hours and 365 days accessibility guaranteed after the system is host to the web. In case of further development then web based system will be down by prior notification to their client in order to protect data integrity.

- Performance and Efficiency

Quick respond of the proposed system is essential. Because users expect to do the things at minimum time. Users waiting time must be reduced and the data must be available soon the request made. Number of clicks must reduce hence information requests from the system need to be readily accessible with a single click.

- Accuracy

Accuracy of single web request is essential. If not, it will be utter time and money waste of users with the wrong response of the system. Further, it may be badly affected on the patient's life because of wrong medical details.

- Risk Management

There is internal and external risk. Manage both of the risk is essential to provide guaranteed service to clients. Internal risks such as data loss, unexpected personnel changes, work flow challenges and process changes. External risks such as unauthorized system access, enter fraud data to the system. Need to provide system accessibility for only authorized parties. When having fraud activities with the system need to give authority to system administrators block the such kind of users.

2.7.4 Requirements specification

Software Development Life Cycle requirement stage final outcome is requirement specification. Conduct software designing and implementation base on requirement specification. Therefor requirement specification is one of the most important documents in SDLC. Because all client requirements and finalized evaluation include here. Requirements should be defined clearly to the success of design and implementation.

3.0. Design

3.1. Introduction

In this chapter discuss software development approach. Proposed web-based system development feasibility discusses. Using waterfall model software development process discuss. Further database design and user interface design part discuss.

3.2. Feasibility study

Feasibility study measures the feasibility of conducting this system by considering its strength, weaknesses, opportunities and threats (SWOT). Ultimately the feasibility study finds the ability of software system to fulfil the client need. Operational, economic and schedule feasibility are the main areas of feasibility study.

Operational feasibility

This study mainly focused on evaluating whether the proposed system operationally acceptable or not. In here measures how well the proposed system solves client need. In the system design and development phase must be ensured operational feasibility of the system. Each and every function need to analyze and confirm those functions work according to the client requirement. Because meet the user requirement and satisfying them is important when designing the software system.

Economic feasibility

This analysis uses to measure economical effectiveness of the new system. It’s basically focused on cost and benefit of the new system. An economically feasible system should have more benefits when compared to the cost. In here analyze existing medical center operations handling VS new system.

This analysis can identify manual operations of the medical center implies more time-wasting facts to the patient, doctor and medical center staff. However, new system will be able to address all of this drawback. But it may imply some initial cost to medical center such as staff training cost and hardware devices purchasing cost.

Schedule feasibility

In here consider the how project complete and deliver according to the schedule. The project will be failed if it takes long time to complete. This system may able to deliver the right time. Because this system uses a waterfall method and all requirements are fixed initially and system development process run as scheduled.

This proposed web application operationally, economically and schedule feasible.

- This web application faster and efficiently works when compared manual system. It will process data within 10 seconds.
- It has more user-friendly user interfaces. Therefor users can easily understand and using less effort they can learn the system.
- Handle data more effectively. This system works with confidential data about patient health. Therefor it maintains data availability, data accuracy and data integrity.
- Medical center operations such as patient appointment prescription generation and medicine issuing are able to handle less time period.
- According to the patient current location system suggest him to nearby medical centers. Therefore, from any ware in the country at emergency situation he can quickly find the nearest medical center.

3.3. Software development lifecycle (SDLC)

Using SDLC design, implement and deliver high quality software system to the client. Its guide the entire software development process from requirement gathering to deliver the final product. When using SDLC able to deliver final product at an agreed time. Because it tracks the state of the system development and move to the next step at the right time. Each stage of the SDLC provides incremental output, which uses as input on the next stage.

SDLC facilitates development plan for new software, enhance existing software or maintain developed software. SDLC provides methodologies for improving the quality of software and the overall development process.

The waterfall model

Most popular SDLC methodology is the waterfall method. It consists with number of sequential steps. When software requirement clear and defined already this methodology is suitable. Its sequential development method. Initial phase needs complete to start the next phase of the development process. Because the initial phase outcome becomes an input of another phase. Waterfall method doesn't suit for when the requirement is not clear and change time to time.

3.4. Prototyping

Prototyping is one of modern requirement gathering technique. Develop the working model with only basic requirements. Introduce it to the client and discover more requirement by considering it. Further development, conduct by considering gathered requirements. Throwaway prototype just uses to requirement gathering purpose only. In incremental prototype use prototype to further development and built final product. It's useful to satisfy client initially of the development process and identify exact requirement by using prototype.

3.5. Database design

One of the most important thing of software design phase is the database design. When designing the database need to consider logical design. Otherwise, it will not properly function and data redundancy become increase.

For this application creates database call "E-Medihelp". All tables needed include in there. There is a table for initial signup of patient and doctor.

- Registration pending patient data
- Registration pending doctor data

After the system administration team accepts move accepted patient and doctors to the registered user table. Patient login and medical center user login referrers that tables.

- Registered patient data
- Registered doctor data

Patient related all data stored in registered patent table. Medical center related all data stored in under registered doctor table. Because in this application doctor represent the medical center.

Further need tables for,

- Prescription data
- Patient appointment data
- Appointment transaction data

The database created using SQL language. All tables created with consistent and secured primary key. All related tables are connected using foreign keys.

3.6. Logical database design

Industry standard modeling language is a Unified Modeling Language (UML). UML use for visualizing, construct and document the artifact of software systems. There are different UML diagram uses to visualize artifacts in different ways. Such as use case diagram, sequence diagram, class diagram, ER diagram etc. UML visualized artifacts as a modeled element which are rendered as shapes or connectors and logically connected each other element.

To visualize the functions related to this web-based system use following diagrams.

1. Use case diagram

Use case diagram visualizes the interaction between actors and the use case. In this use case diagram can identify five actors including the system itself. Use case diagram shown in fig 3.1.

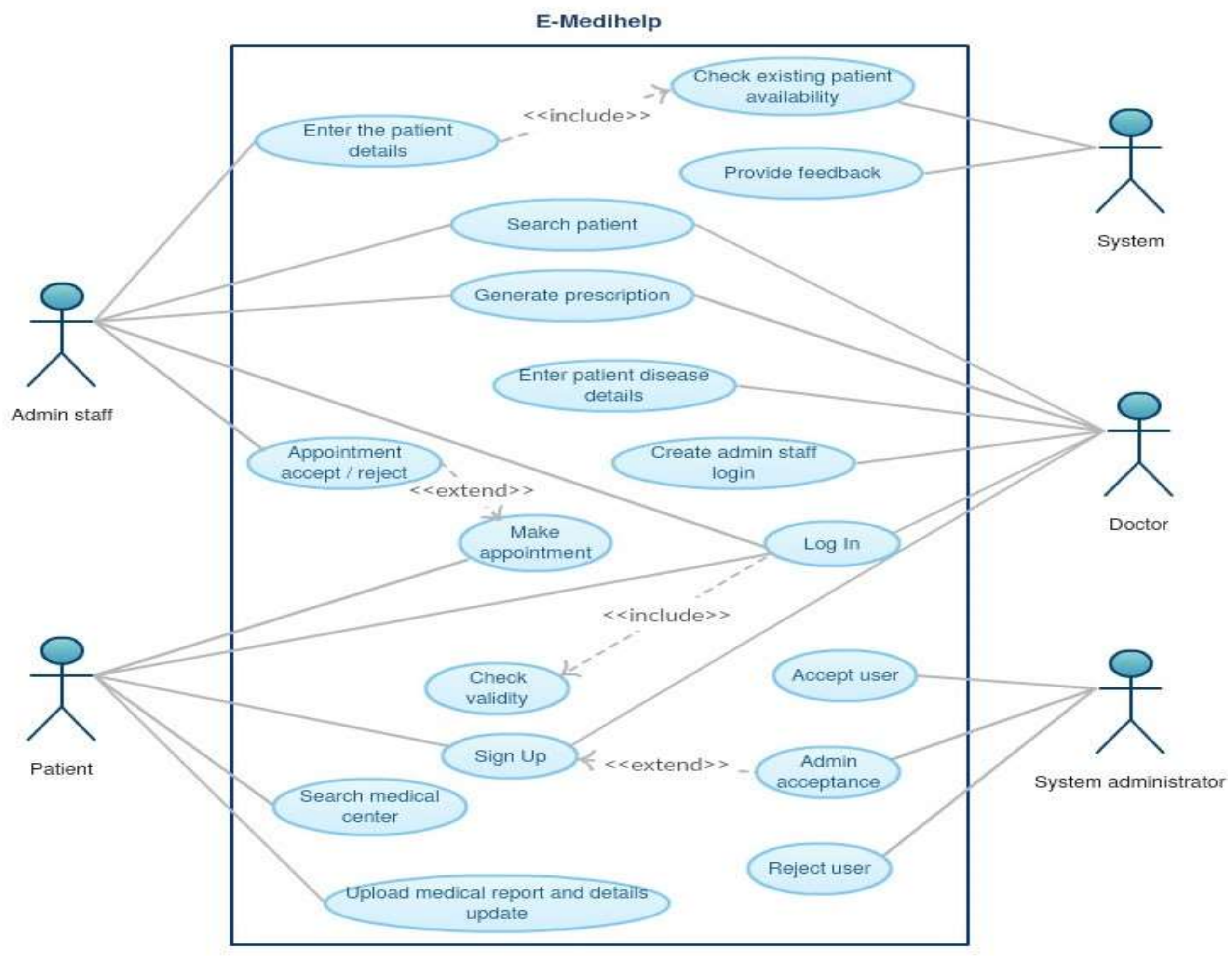


Figure 3.1 :- Use case diagram

2. Class diagram

Class diagram shown in figure 3.2.

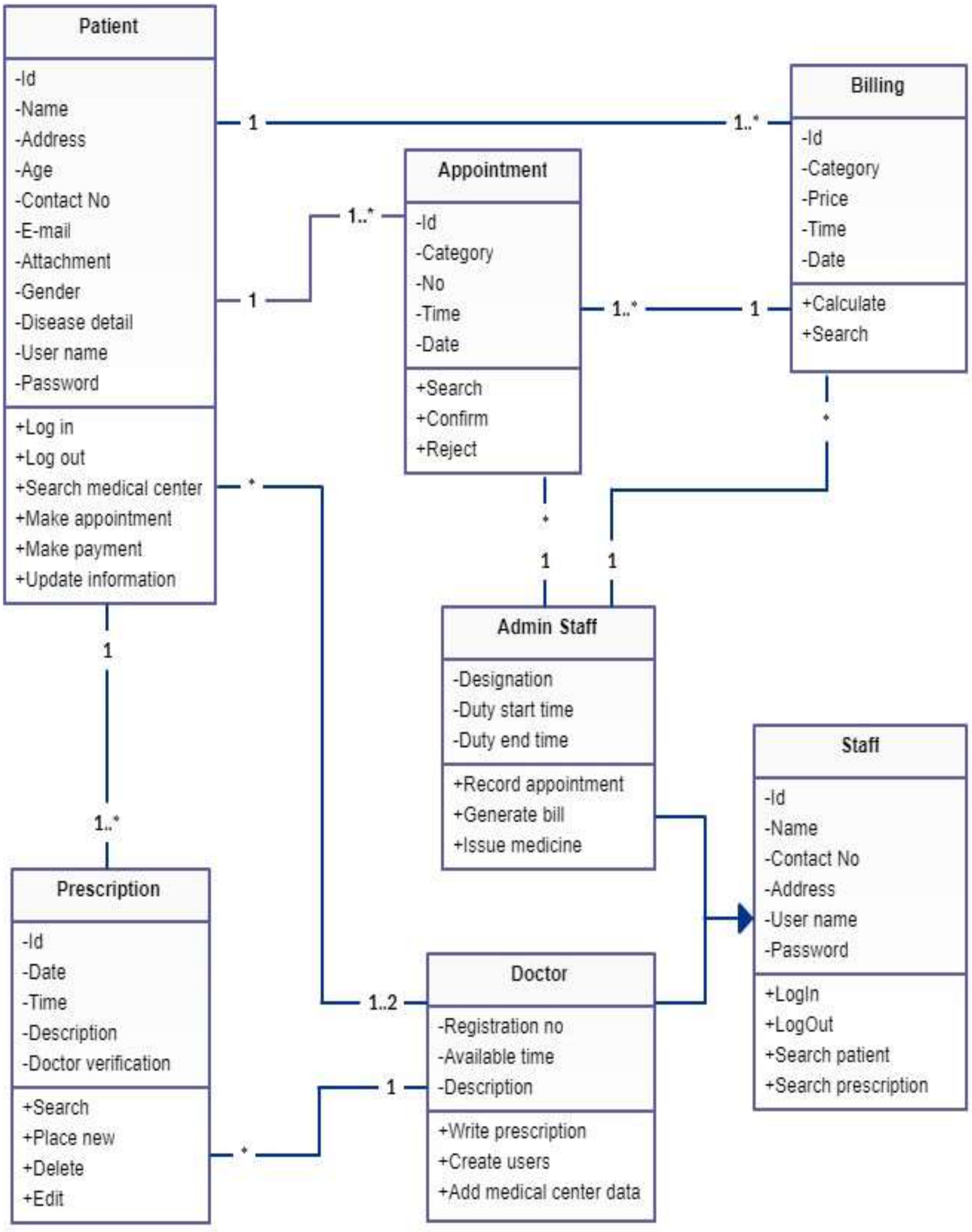


Figure 3.2 :- Class diagram

3. ER diagram

ER diagram shown in figure 3.3.

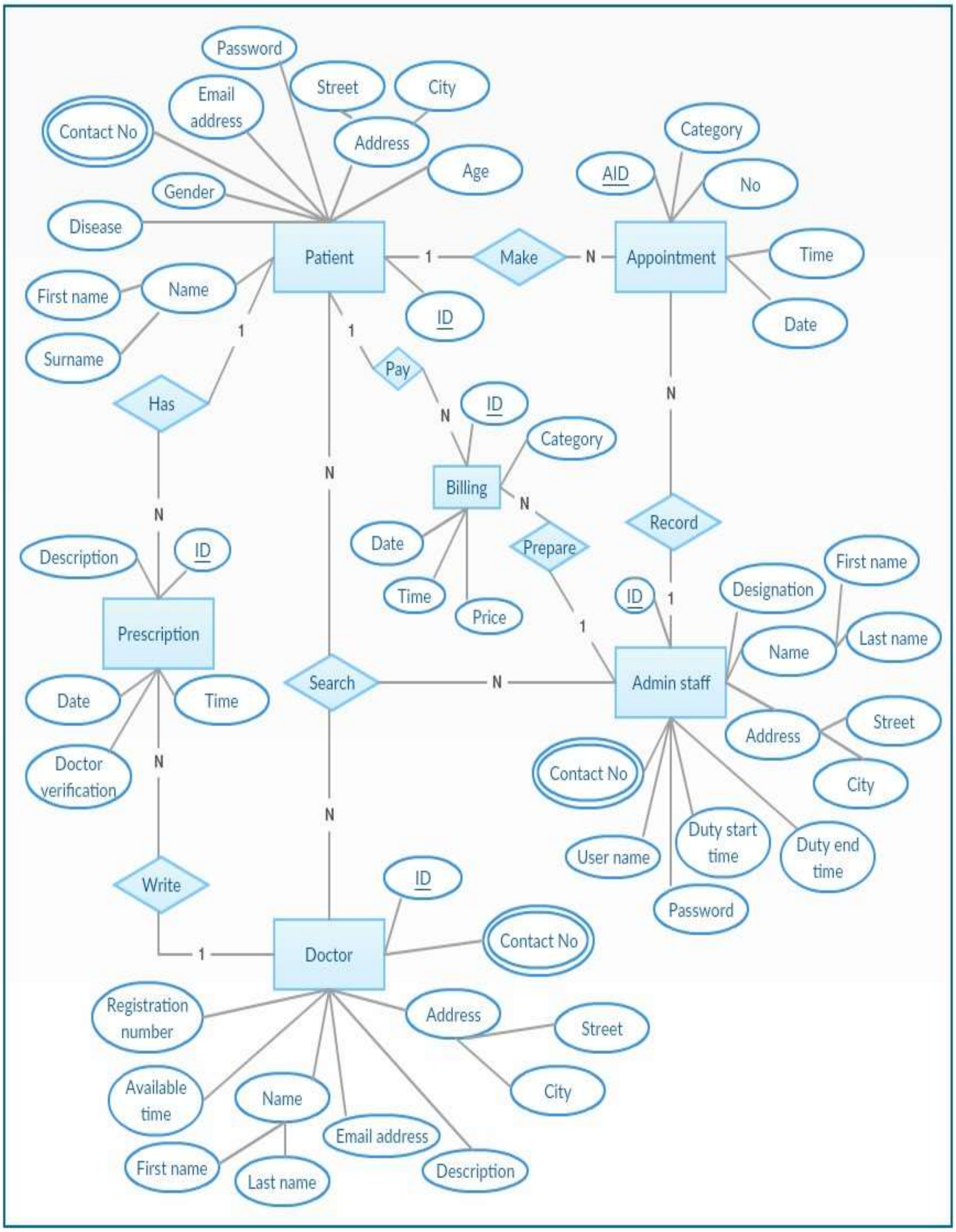


Figure 3.3 :- ER diagram

4. Activity diagram

Activity diagram shown in figure 3.4.

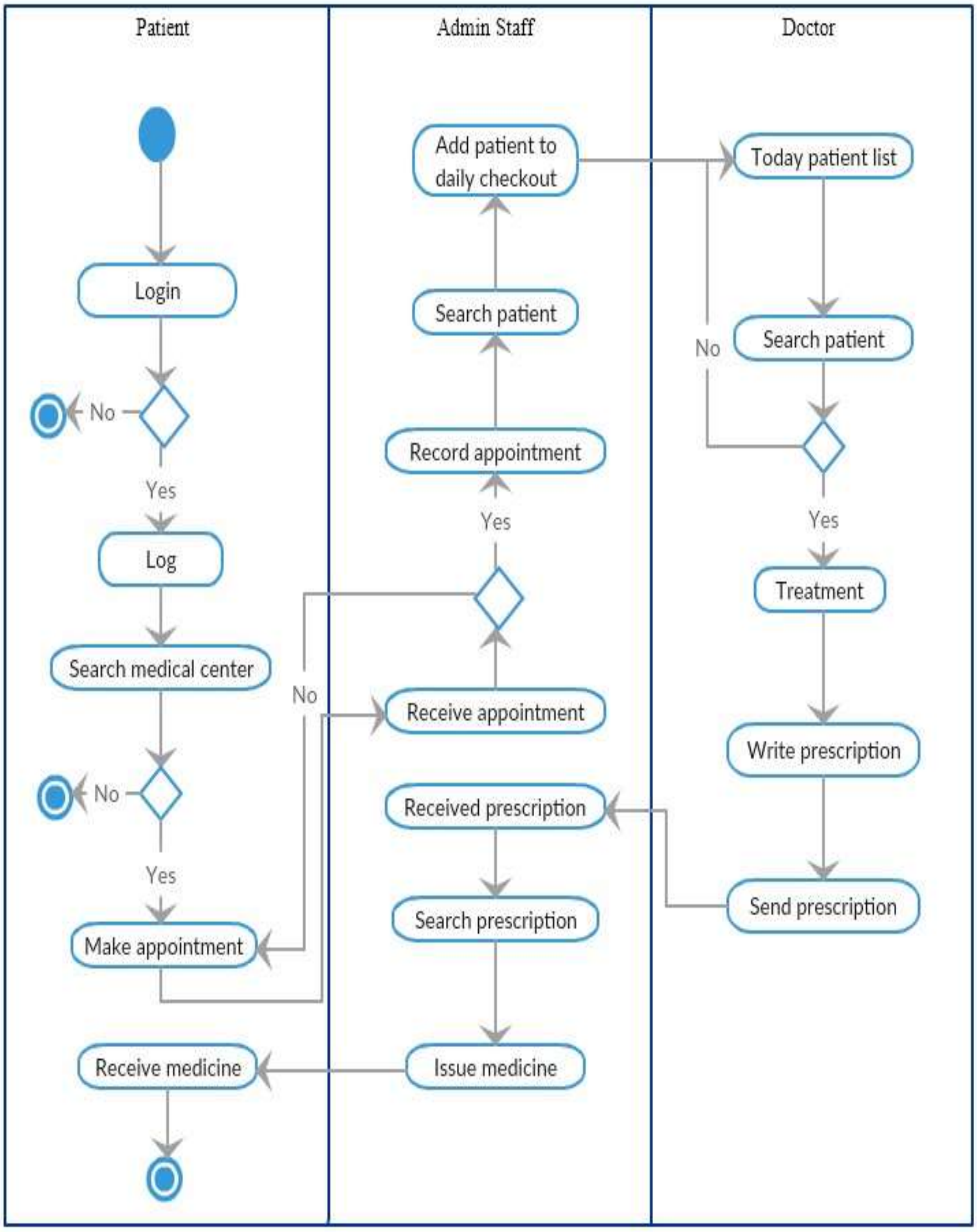


Figure 3.4 :- Activity diagram

3.7. User interface design

User interface designing is the most important for developing more usability software system. Because any software system ultimate goal is satisfying their users. Therefor need to provide more usability system to its users. Now a day most of software systems develop by considering its usability. They are provided more attractive user interfaces by using numerical and more graphical information. Users are interested such kind of interfaces because of,

- Attractiveness of the interfaces
- Better understandability
- Fast and ease of accessibility
- Give feedback appropriately
- Express everything relevant to user more effective way

Early stages of software industry only had command line interfaces. While evolving the software industry graphical interfaces developed. Graphical interfaces graphically provide actions to their users to complete the task. Therefor its easily understandable to the user what he has to do with the system. Graphical user interfaces have following characteristics.

- Menu / Tab menu /Drop down menu
- Tables
- Map
- Progress bars
- Graphs
- Multiple window
- Buttons / Ratio button / Check box
- Images / Icon / Media
- Text field / Text area
- Graphics. Etc.

This web application user interface developed using Bootstrap, CSS, JavaScript. Bootstrap predefine classes use appropriately. Further custom designs add to the interfaces using CSS. JavaScript is used to form validation and inform users what is going on the background of the system. Maintain predefine color theme in the entire application by considering the medical field to maintain the consistency of the system.

User interface design decision taken based on few facts.

- Reduce learning

Consistency limits the number of ways actions and operations are represented, ensuring that users do not have to learn new representations for each task.

- Eliminate confusion

Users tend to apply rules they've experienced outside of this system, bringing in a set of their own expectations. Hence need to be mindful of whether or not this system, causing confusion and alienation when deviate from design standards and conventions. Further, users should not have to spend time wondering whether different words, interactions, or actions actually mean the same thing within the context of this product.

User interfaces of the proposed system

- ➢ Patient login and signup interface shown in figure 3.5.

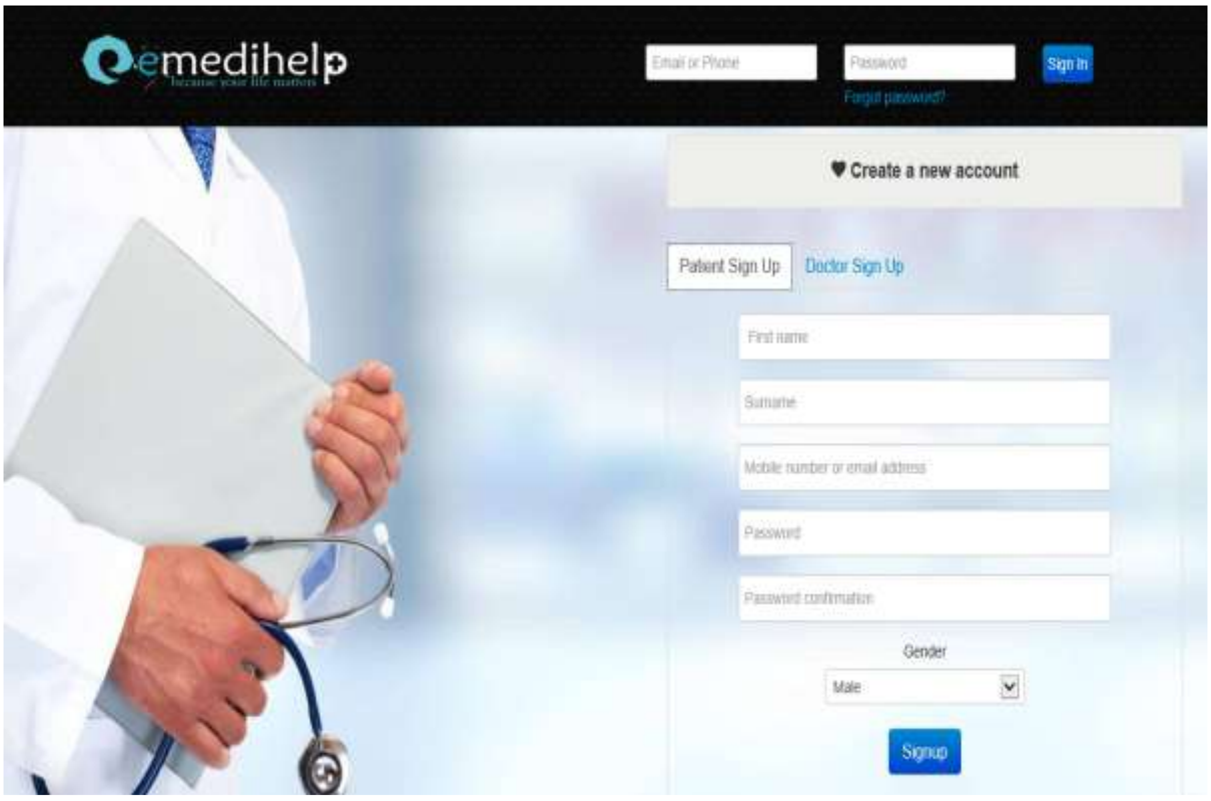


Figure 3.5 :- Patient Login and signup interface

➢ Doctor login and signup interface shown in figure 3.6 and 3.7.

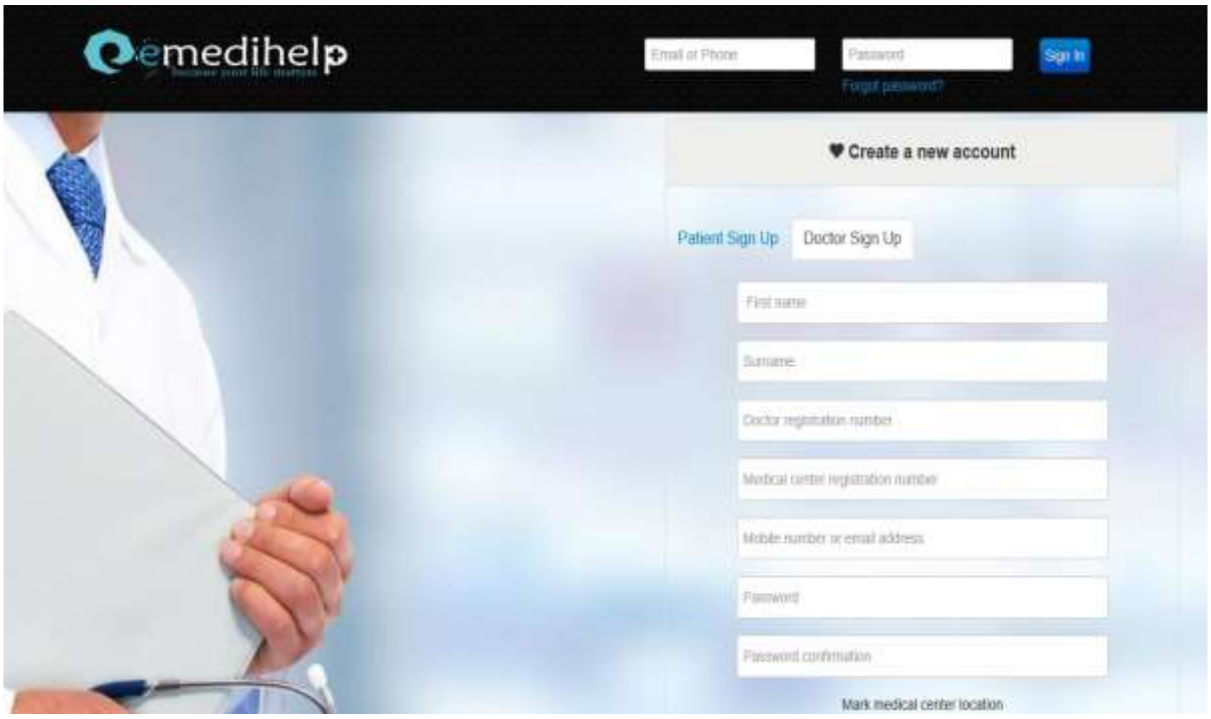


Figure 3.6 :- Doctor login and signup interface part 1

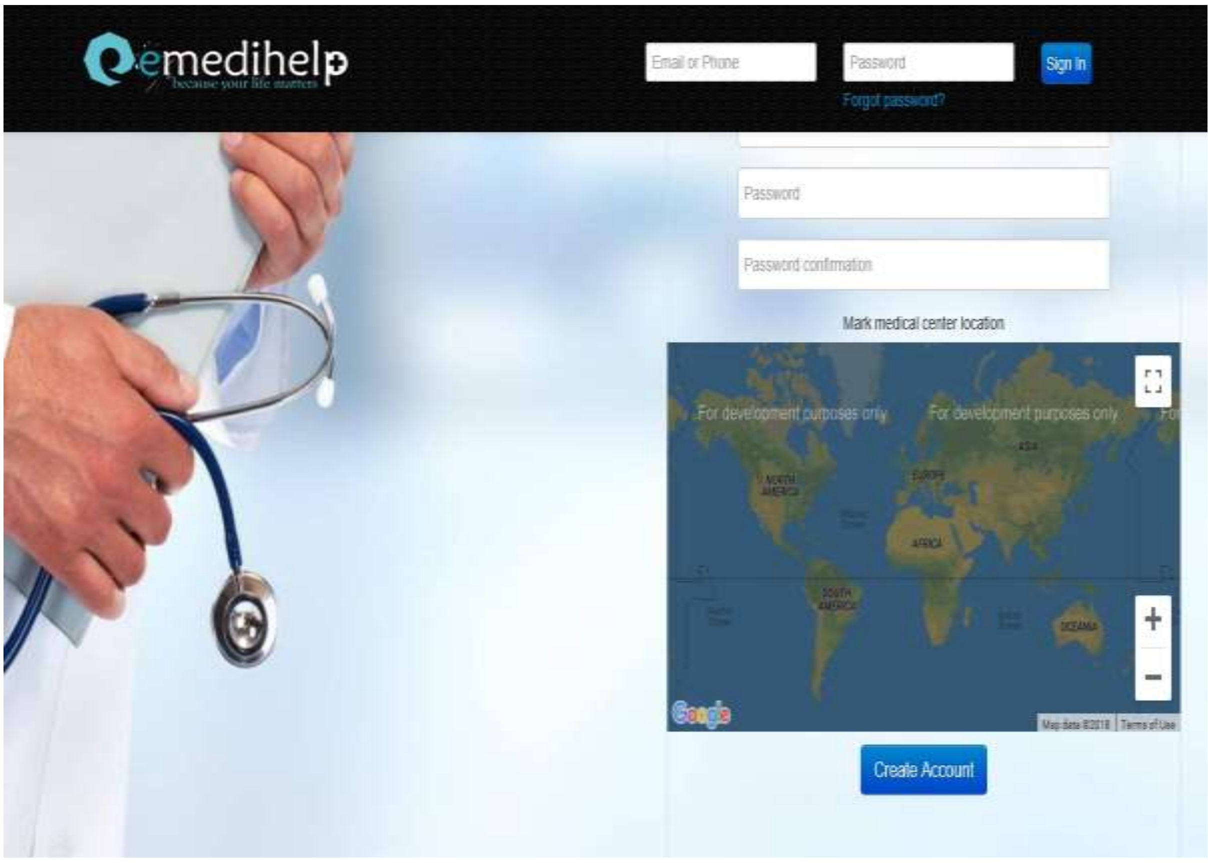
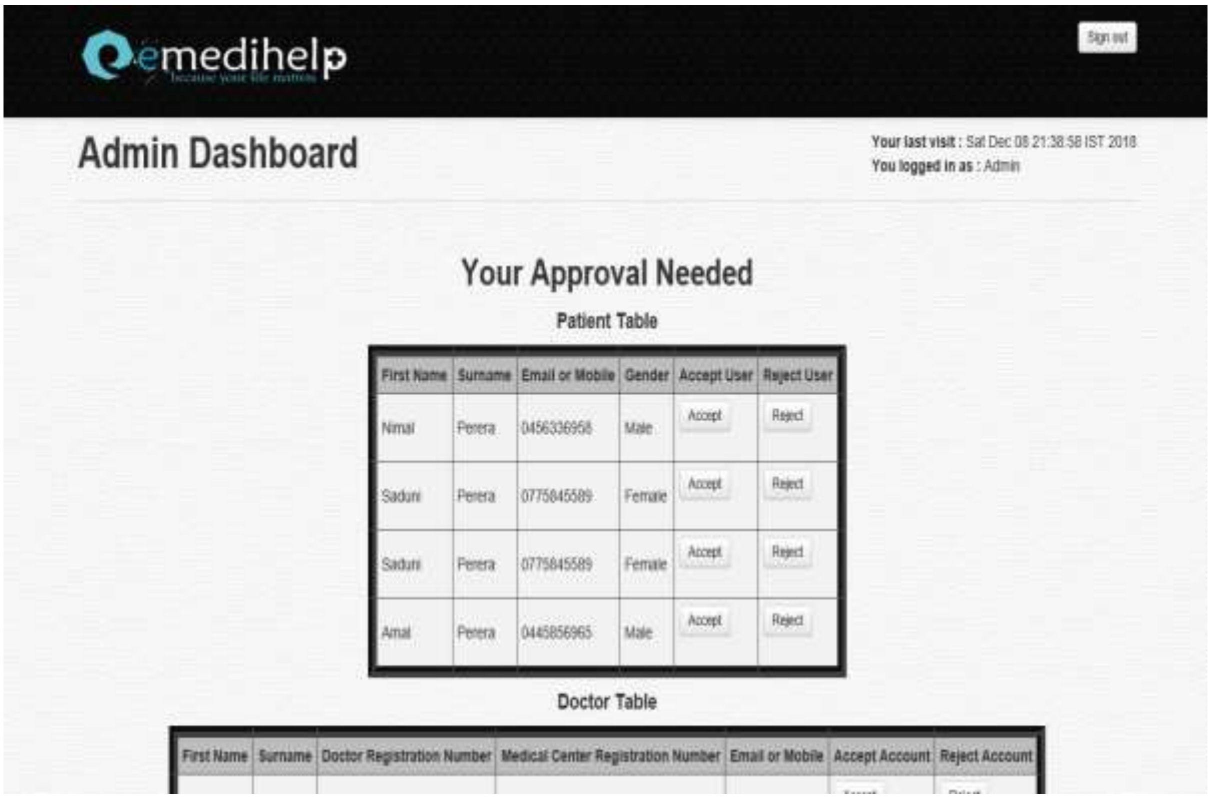

emedihelp
because your life matters
Email or Phone
Password
Sign In
Forgot password?
Password
Password confirmation
Mark medical center location
For development purposes only
For development purposes only
Create Account

Figure 3.7 :- Doctor login and signup interface part 2

➢ System administration team dashboard shown in figure 3.8.



Admin Dashboard

Your last visit : Sat Dec 08 21:38:58 IST 2018
You logged in as : Admin

Your Approval Needed

Patient Table

First Name	Surname	Email or Mobile	Gender	Accept User	Reject User
Nimal	Perera	0456336958	Male	Accept	Reject
Saduni	Perera	0775845589	Female	Accept	Reject
Saduni	Perera	0775845589	Female	Accept	Reject
Amal	Perera	0445856965	Male	Accept	Reject

Doctor Table

First Name	Surname	Doctor Registration Number	Medical Center Registration Number	Email or Mobile	Accept Account	Reject Account

Figure 3.8 :- System administrator dashboard

4.0. Implementation and Evaluation

4.1. Implementation

Project implementation aids will be discussed in this chapter. This chapter consists with detailed of the development environment of the hardware and software, code structure with reused code and development tools used and security of the system.

Major code structure

This project implements used J2EE framework (Java Enterprise Edition). The J2EE framework structure consists with three layers.

Presentation web content

- JSP and regular Web content (HTML, style sheets, images, etc.)

Coordinate presentation layer with data model

- Servlets

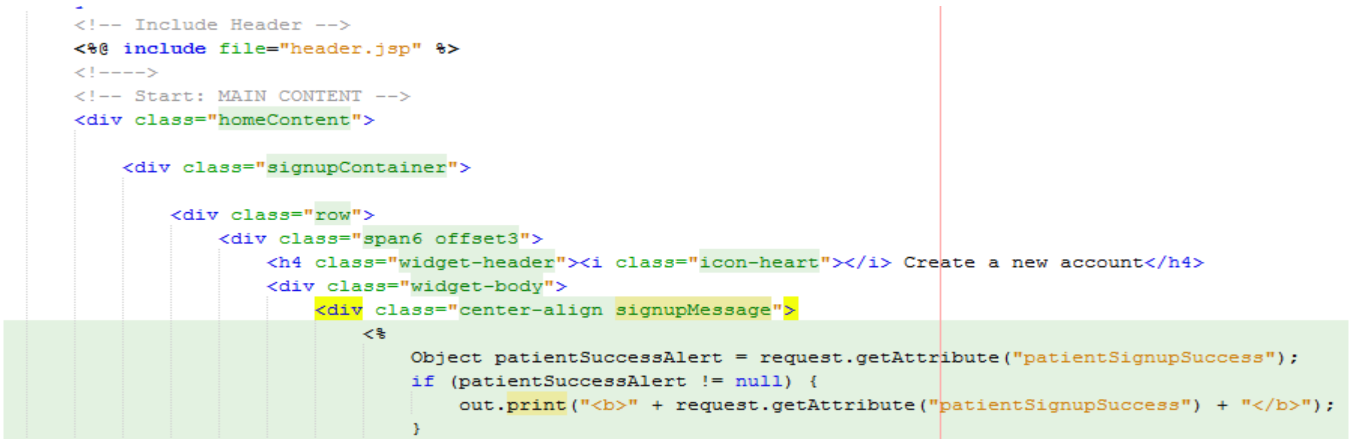
Data model for communicating with the database

- Java bean classes

This implementation based on MVC architecture. Views develop using JSP pages. It consists HTML, CSS, Java Script and JSP directives to communicate with controllers and models. Java Bean classes use to write the data model and communicate with databases. Servlet work as controller of the system, it accepts requests from views, bind data with model and result redirect to view.

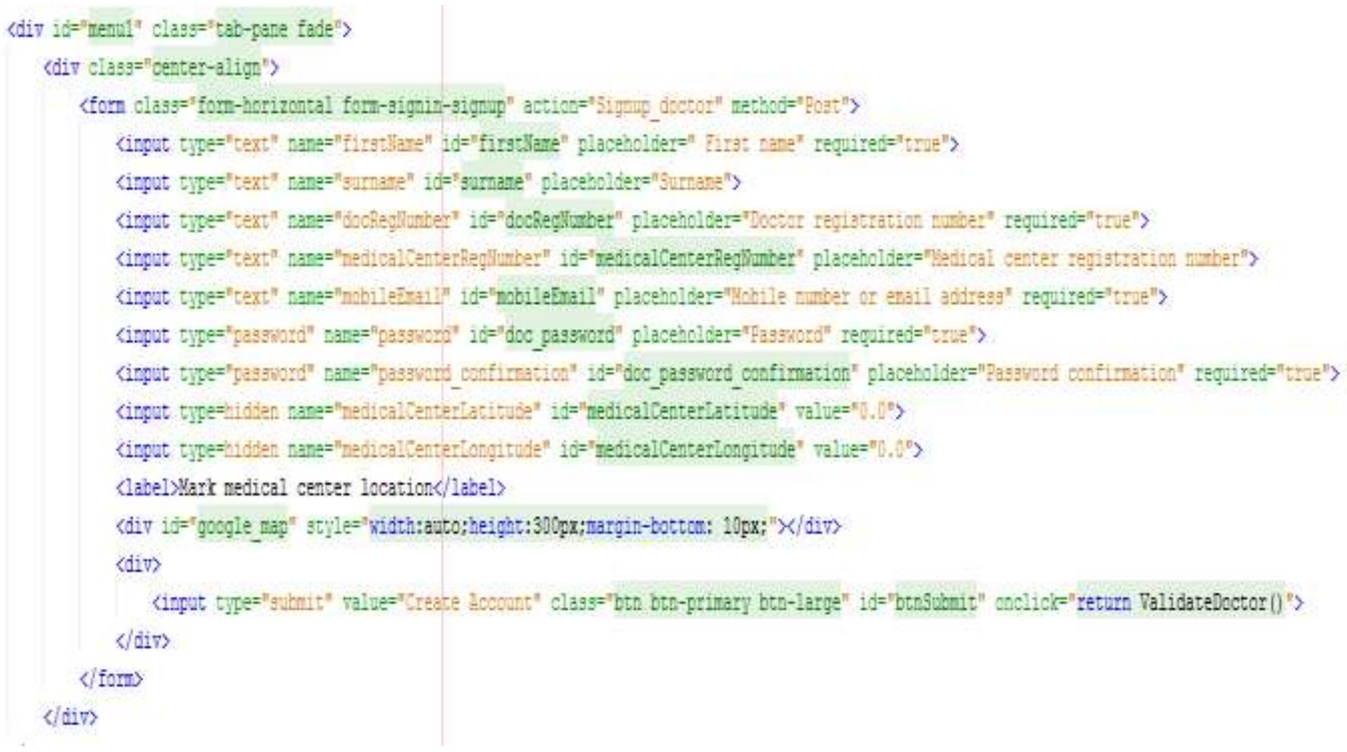
JSP content

JSP content shown in figure 4.1, figure 4.2 and figure 4.3.



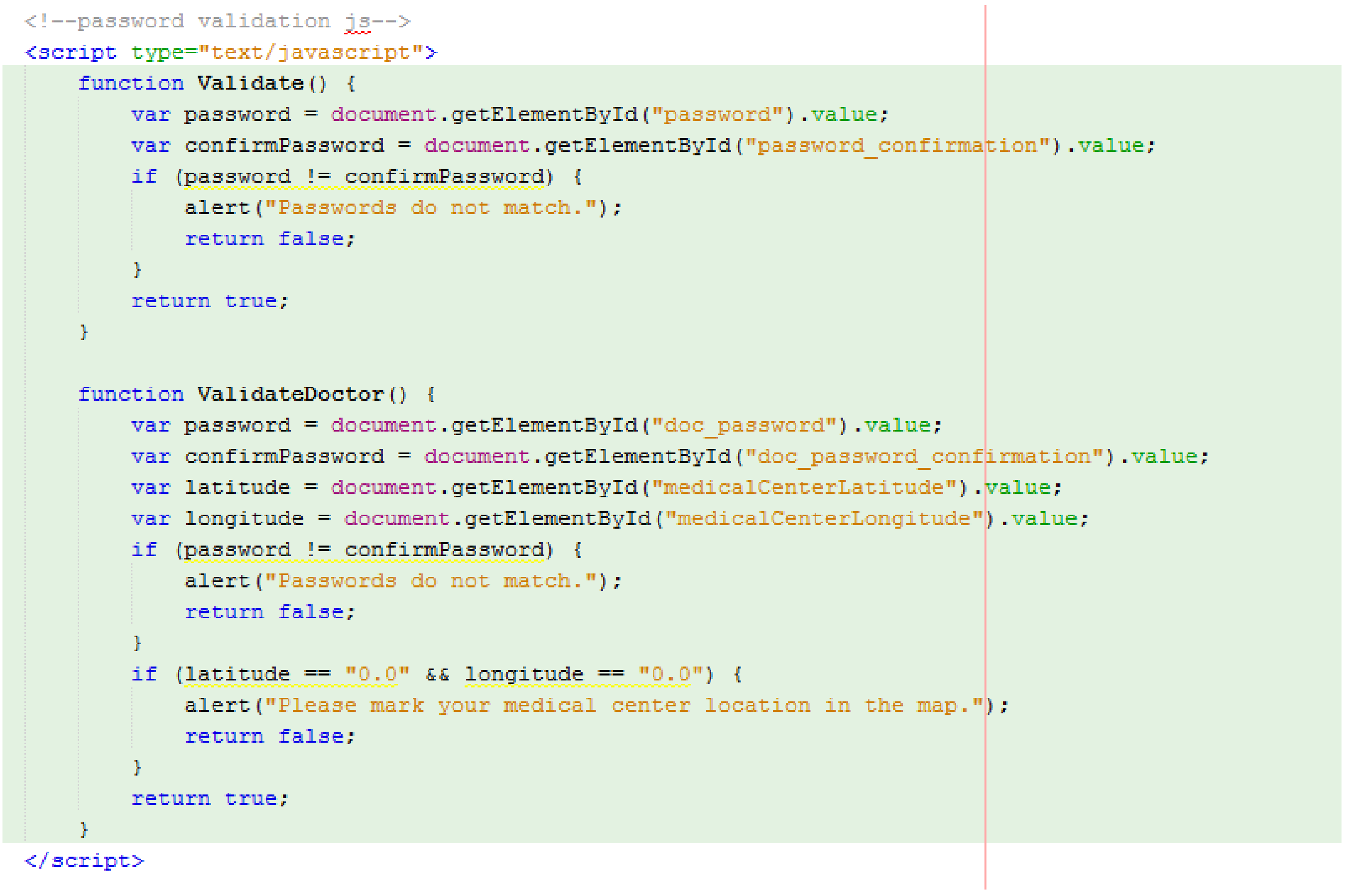
```
<!-- Include Header -->
<%@ include file="header.jsp" %>
<!---->
<!-- Start: MAIN CONTENT -->
<div class="homeContent">

    <div class="signupContainer">

        <div class="row">
            <div class="span6 offset3">
                <h4 class="widget-header"><i class="icon-heart"></i> Create a new account</h4>
                <div class="widget-body">
                    <div class="center-align signupMessage">
                        <%
                            Object patientSuccessAlert = request.getAttribute("patientSignupSuccess");
                            if (patientSuccessAlert != null) {
                                out.print("<b>" + request.getAttribute("patientSignupSuccess") + "</b>");
                            }
```

Figure 4.1 :- JSP content 1



```
<div id="menu1" class="tab-pane fade">
    <div class="center-align">
        <form class="form-horizontal form-signin-signup" action="Signup_doctor" method="Post">
            <input type="text" name="firstName" id="firstName" placeholder=" First name" required="true">
            <input type="text" name="surname" id="surname" placeholder="Surname">
            <input type="text" name="docRegNumber" id="docRegNumber" placeholder="Doctor registration number" required="true">
            <input type="text" name="medicalCenterRegNumber" id="medicalCenterRegNumber" placeholder="Medical center registration number">
            <input type="text" name="mobileEmail" id="mobileEmail" placeholder="Mobile number or email address" required="true">
            <input type="password" name="password" id="doc_password" placeholder="Password" required="true">
            <input type="password" name="password_confirmation" id="doc_password_confirmation" placeholder="Password confirmation" required="true">
            <input type=hidden name="medicalCenterLatitude" id="medicalCenterLatitude" value="0.0">
            <input type=hidden name="medicalCenterLongitude" id="medicalCenterLongitude" value="0.0">
            <label>Mark medical center location</label>
            <div id="google_map" style="width:auto;height:300px;margin-bottom: 10px;"></div>
            <div>
                <input type="submit" value="Create Account" class="btn btn-primary btn-large" id="btnSubmit" onclick="return ValidateDoctor()">
            </div>
        </form>
    </div>
```

Figure 4.2 :- JSP content 2



```
<!--password validation js-->
<script type="text/javascript">
    function Validate() {
        var password = document.getElementById("password").value;
        var confirmPassword = document.getElementById("password_confirmation").value;
        if (password != confirmPassword) {
            alert("Passwords do not match.");
            return false;
        }
        return true;
    }

    function ValidateDoctor() {
        var password = document.getElementById("doc_password").value;
        var confirmPassword = document.getElementById("doc_password_confirmation").value;
        var latitude = document.getElementById("medicalCenterLatitude").value;
        var longitude = document.getElementById("medicalCenterLongitude").value;
        if (password != confirmPassword) {
            alert("Passwords do not match.");
            return false;
        }
        if (latitude == "0.0" && longitude == "0.0") {
            alert("Please mark your medical center location in the map.");
            return false;
        }
        return true;
    }
</script>
```

Figure 4.3 :- JSP content 3

Java bean content

Java bean content shown in figure 4.4.

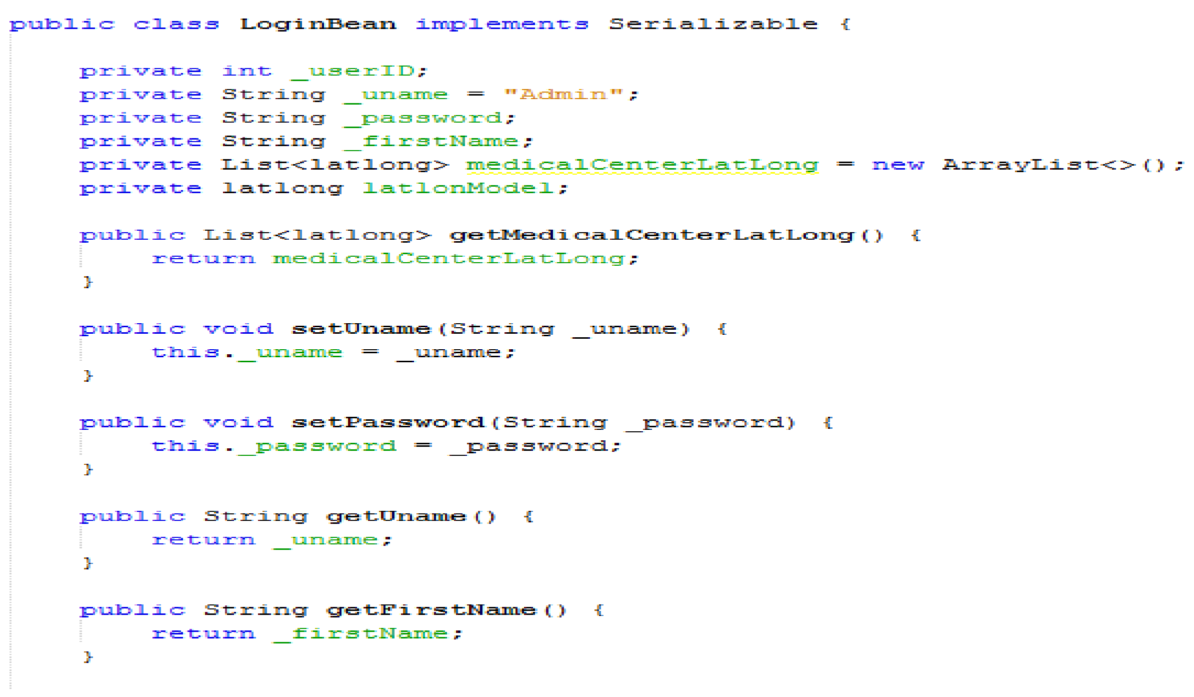

```
public class LoginBean implements Serializable {

    private int _userID;
    private String _uname = "Admin";
    private String _password;
    private String _firstName;
    private List<latlong> medicalCenterLatLong = new ArrayList<>();
    private latlong latlonModel;

    public List<latlong> getMedicalCenterLatLong() {
        return medicalCenterLatLong;
    }

    public void setUname(String _uname) {
        this._uname = _uname;
    }

    public void setPassword(String _password) {
        this._password = _password;
    }

    public String getUname() {
        return _uname;
    }

    public String getFirstName() {
        return _firstName;
    }
```

Figure 4.4 :- Java bean content

Servlet content

Servlet content shown in figure 4.5.

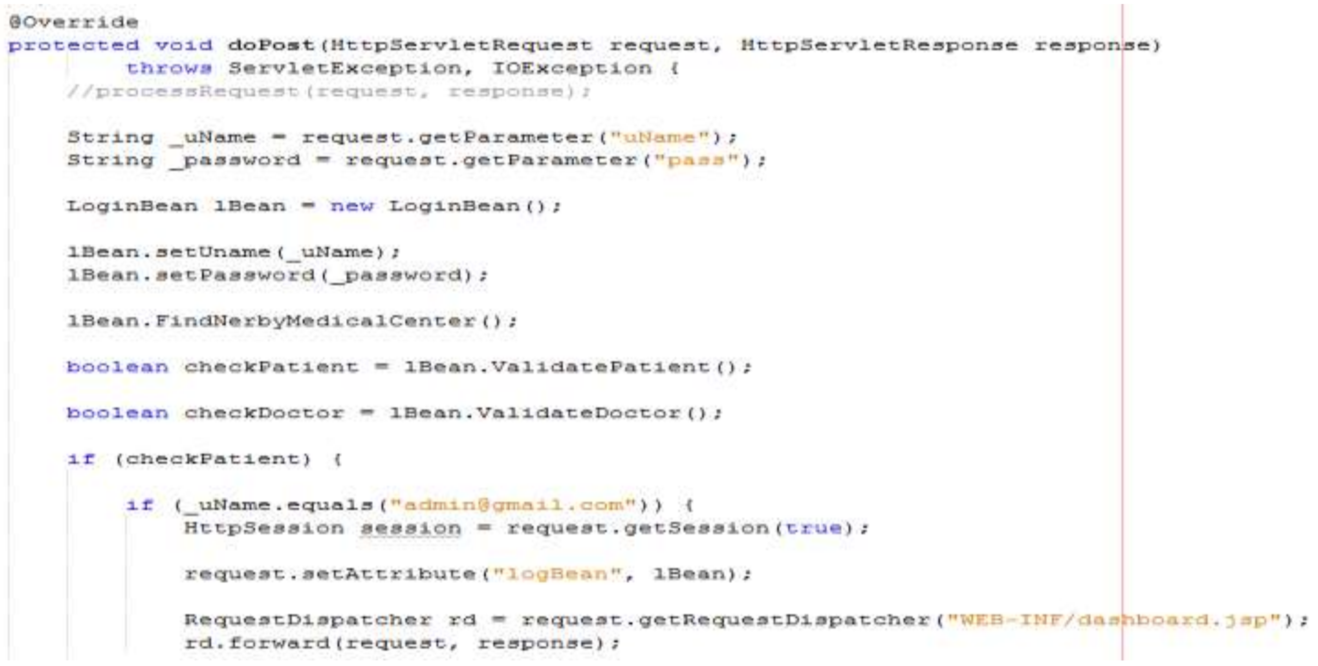

```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    //processRequest(request, response);

    String _uName = request.getParameter("uName");
    String _password = request.getParameter("pass");

    LoginBean lBean = new LoginBean();

    lBean.setUname(_uName);
    lBean.setPassword(_password);

    lBean.FindNerbyMedicalCenter();

    boolean checkPatient = lBean.ValidatePatient();

    boolean checkDoctor = lBean.ValidateDoctor();

    if (checkPatient) {

        if (_uName.equals("admin@gmail.com")) {
            HttpSession session = request.getSession(true);

            request.setAttribute("logBean", lBean);

            RequestDispatcher rd = request.getRequestDispatcher("WEB-INF/dashboard.jsp");
            rd.forward(request, response);
```

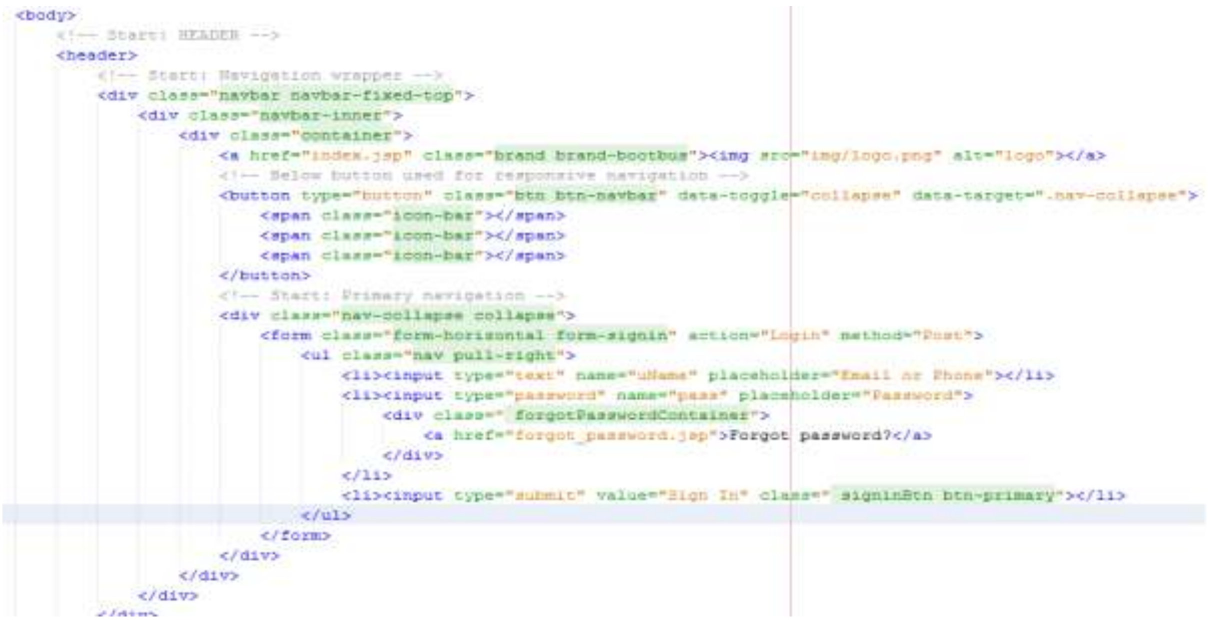
Figure 4.5 :- Servlet content

Reused codes

There is common code structure reused in many pages. Page headers and footers are commonly use all pages. Headers include to the JSP page top and footers include to the JSP page bottom.

Header

Header shown in figure 4.6.

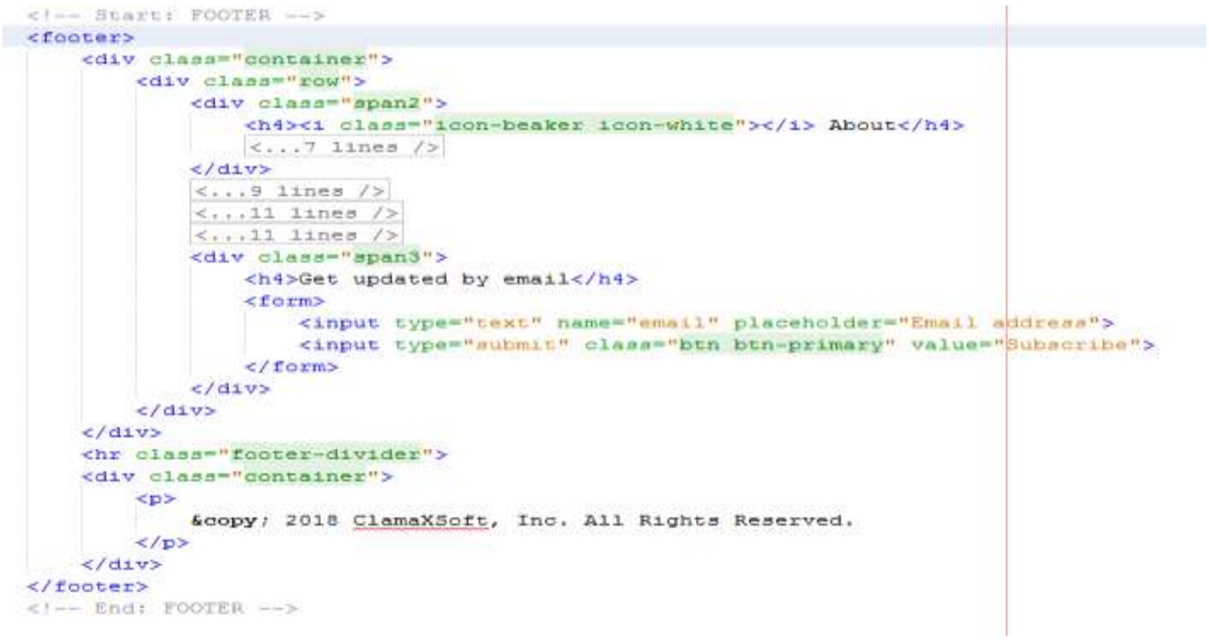


```
<body>
    <!-- Start: HEADER -->
    <header>
        <!-- Start: Navigation wrapper -->
        <div class="navbar navbar-fixed-top">
            <div class="navbar-inner">
                <div class="container">
                    <a href="index.jsp" class="brand brand-bootbus"><img src="img/logo.png" alt="logo"></a>
                    <!-- Below button used for responsive navigation -->
                    <button type="button" class="btn btn-navbar" data-toggle="collapse" data-target=".nav-collapse">
                        <span class="icon-bar"></span>
                        <span class="icon-bar"></span>
                        <span class="icon-bar"></span>
                    </button>
                    <!-- Start: Primary navigation -->
                    <div class="nav-collapse collapse">
                        <form class="form-horizontal form-signin" action="Login" method="Post">
                            <ul class="nav pull-right">
                                <li><input type="text" name="uName" placeholder="Email or Phone"></li>
                                <li><input type="password" name="pass" placeholder="Password">
                                    <div class=" forgotPasswordContainer">
                                        <a href="forgot_password.jsp">Forgot password?</a>
                                    </div>
                                </li>
                                <li><input type="submit" value="Sign In" class=" signinBtn btn-primary"></li>
                            </ul>
                        </form>
                    </div>
                </div>
            </div>
        </div>
```

Figure 4.6 :- Header content

Footer

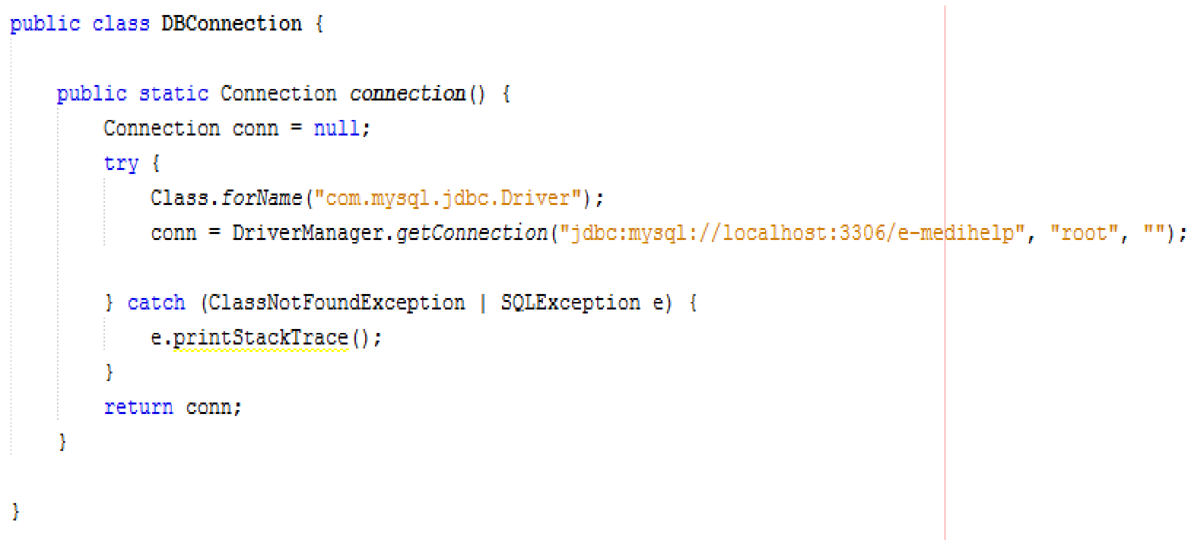
Footer shown in figure 4.7.



```
<!-- Start: FOOTER -->
<footer>
    <div class="container">
        <div class="row">
            <div class="span2">
                <h4><i class="icon-beaker icon-white"></i> About</h4>
                <...7 lines />
            </div>
            <...9 lines />
            <...11 lines />
            <...11 lines />
            <div class="span3">
                <h4>Get updated by email</h4>
                <form>
                    <input type="text" name="email" placeholder="Email address">
                    <input type="submit" class="btn btn-primary" value="Subscribe">
                </form>
            </div>
        </div>
    </div>
    <hr class="footer-divider">
    <div class="container">
        <p>
            &copy; 2018 ClamaXSoft, Inc. All Rights Reserved.
        </p>
    </div>
</footer>
<!-- End: FOOTER -->
```

Figure 4.7 :- Footer content

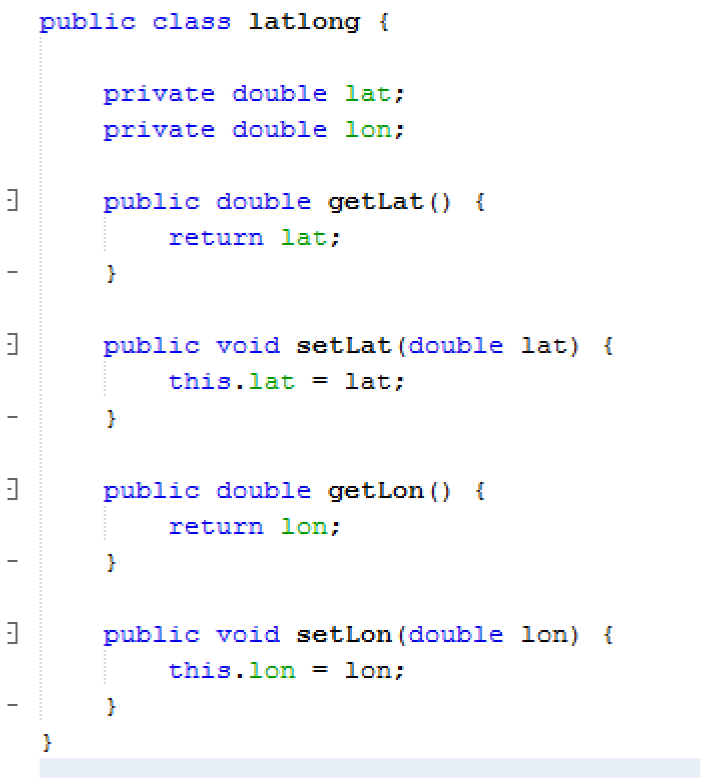
Database connection class is reused in all places which is needed to make a connection with the database. Database connection content shown in figure 4.8.



```
public class DBConnection {

    public static Connection connection() {
        Connection conn = null;
        try {
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/e-medihelp", "root", "");

        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
        }
        return conn;
    }

}
```

Figure 4.8 :- Database connection content

Models use different places in code structure when required to bind data and create objects. Data binding model content shown in figure 4.9.



```
public class latlong {

    private double lat;
    private double lon;

    public double getLat() {
        return lat;
    }

    public void setLat(double lat) {
        this.lat = lat;
    }

    public double getLon() {
        return lon;
    }

    public void setLon(double lon) {
        this.lon = lon;
    }
}
```

Figure 4.9 :- Data binding model content

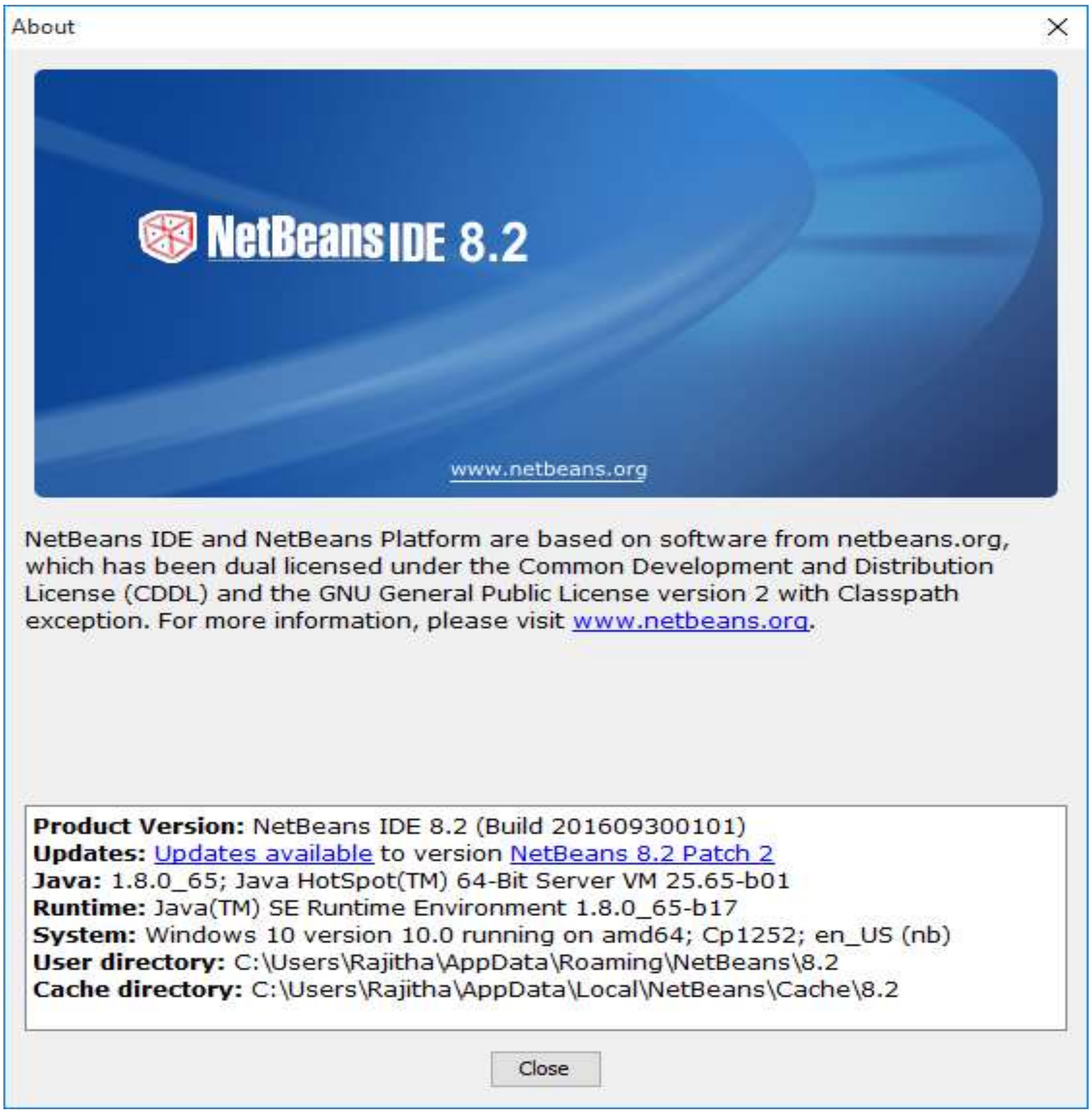
Development environment

To develop this system, use windows (windows 10) based operating system as software development environment. Development environment shown in figure 4.10.

Figure 4.10 :- Development environment

This system development environment consists with Intel Core i5 2.60 GHz Processor, 4 GB RAM and 160 GB HD as hardware features.

This system based on J2EE therefor to develop this Java project uses the NetBeans 8.2 as an Integrated Development Environment (IDE). IDE shown in figure 4.11.



NetBeans IDE and NetBeans Platform are based on software from netbeans.org, which has been dual licensed under the Common Development and Distribution License (CDDL) and the GNU General Public License version 2 with Classpath exception. For more information, please visit www.netbeans.org.

Product Version: NetBeans IDE 8.2 (Build 201609300101)
Updates: Updates available to version NetBeans 8.2 Patch 2
Java: 1.8.0_65; Java HotSpot(TM) 64-Bit Server VM 25.65-b01
Runtime: Java(TM) SE Runtime Environment 1.8.0_65-b17
System: Windows 10 version 10.0 running on amd64; Cp1252; en_US (nb)
User directory: C:\Users\Rajitha\AppData\Roaming\NetBeans\8.2
Cache directory: C:\Users\Rajitha\AppData\Local\NetBeans\Cache\8.2

Figure 4.11 :- IDE

Glassfish 4.1.1 server use as a web server in the NetBeans development environment. It’s able to handle HTTP requests from web browser and servlet content. Server detail shown in figure 4.12.

Oracle GlassFish Server
Oracle GlassFish Server is the world's first implementation of the Java Platform, Enterprise Edition (Java EE) 6 specification. Built using the GlassFish Server Open Source Edition, Oracle GlassFish Server delivers a flexible, lightweight, and production-ready Java EE 6 application server.

Figure 4.12 :- Server used

XAMPP Version: 7.2.12 use as windows web development environment. It allows to create web application with Apache2, MySQL database. In this project use XAMPP to create MySQL database using phpMyAdmin. XAMPP 64-bit version was installed and create a MySQL database.

4.1.1. System Security

Proposed web-based system expects a highly secure environment. Because this web application contains patient personal details with their medical condition, no need to expose other parties. As well as medical center accounts contain their own business details and patient details no need to expose other parties. The system administrator has authority to manage all kinds of users of this system. In this system use different security tools.

- This system allows to create patient or medical center account if he has relevant details. But he can only create an account. A system administrator has full authority to accept or reject account by considering his identity.
- This system not allowed to any one look at the system inner functionalities without having a user account.
- System able to identify a particular user individually and last access date and time of the system.
- Users are secure when they forgot password. A system able to send email to the user along with the new password.
- When user logout and try to redirect to the previous page the system will not allow. Because system maintains session for each user login.
- When create user account there is required fields for enhancing security of the user account such as password strength.
- Medical center accounts new user creating privilege has only medical center account owner (Doctor).

4.2. Evaluation

In this chapter describes the testing phase of this system. Testing is an activity to check whether the actual result matches the expected results and to ensure that the software system is defect free. Client satisfaction of the software will be based on the result of the evaluation phase.

4.2.1. Test Plan

A test plan is the document. It describes the scope of the test plan, approaches used to test the system, available resources to conduct the test and the schedule of test activities. It identifies the features to be tested from among all of them, who will be conducting the test, any risk they will face and contingency planning. Test plan shown in figure 4.13.

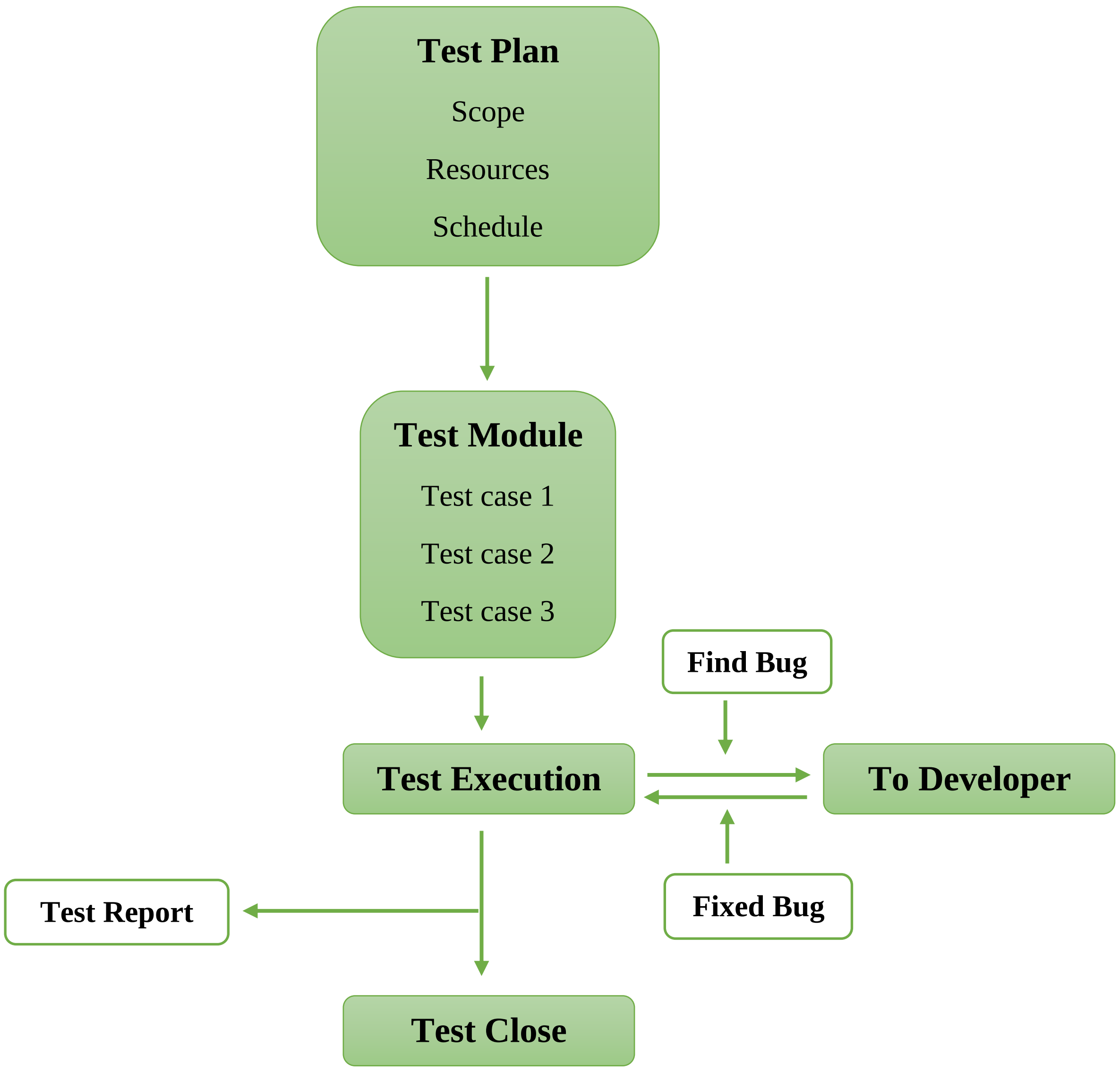


Figure 4.13 :- Test plan

Testing Scope

Testing scope describe which features need to be tested and what will not to be tested. Further describe which areas need to be tested in particular feature. That is including performance, security and globalization test, etc. To manage the project in initial stage test scope definition is important. Clearly defined scope will helpful to everyone to identify what is the tested and what is not.

Test Scheduling

Scheduling the testing activities depends on development completion date and delivery dates of the modules to test. Test plan prepared at this stage. It’s showing testing activities with an estimation.

Test Cases

The test case is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly. The process of developing test cases can also help find problems in the requirements or design of an application.

4.2.2. System Testing

System testing is the black box testing of all the system modules. There’s integrate all the modules in order to verify if the system works as expected or not. System testing is done after integration testing. This plays an important role in delivering a high-quality product.

This system test by module vise. Each module test individually and finally tests integrating all modules together. When a system error occurred went through the particular module each code segment and rectify the issue. Following modules test and module vise collect the issues.

- Tested doctor signup module.
- Tested patient signup module.
- Tested user login module.
- Tested user logout module.
- Tested user forgets password module.
- Tested system administrator dashboard module.
- Tested patient landing page module.
- Tested find the nearest medical center module.
- Tested medical center dashboard module.

- Tested medical center landing page module.
- Tested logged user dynamic information.

Above all modules tested using different test information and verified successfully work above all of the modules. User acceptance test conducted with one of medical center doctors appointed employee. Few test cases and its results are following. User registration test shown in table 4.1. User login test shown in table 4.2. User logout test shown in table 4.3.

User Registration

Table 4.1 :- User registration test

Test Case	Expected Result	Actual Result	Status
Select signup tab.	Signup modal should display for patient and doctor.	Signup modal will display for patient and doctor	“Pass”
Select medical center location from the map, enter details and press signup button.	Signup details should be sent to the system and display message weather the doctor account create or not.	Doctor signup detail will successfully be sent to the system and display registration signup message. Otherwise signup fail message display.	“Pass”
Enter patient signup detail and press signup button.	If details are correctly entered signup success message should display otherwise signup fail message should display.	Signup success message will display if entered correct data. Otherwise will display signup fail message.	“Pass”
After signup user going to log into the system.	The system should restrict the login attempt because that user not verified yet.	The system will restrict the logging attempt.	“Pass”

User login to the system

Table 4.2 :- User login test

Test Case	Expected Result	Actual Result	Status
User enter invalid username and password.	An alert should be displayed indicating invalid username or password.	An alert will be display indicating invalid username or password.	“Pass”
User enter correct username and password.	According to the user level user should log into the system.	According to the user level user will log into the system.	“Pass”

User logout from the system

Table 4.3 :- User logout test

Test Case	Expected Result	Actual Result	Status
Sign out button available for logged user.	If user is in logged state he should see the log out button.	If user is in logged state he will see the log out button.	“Pass”
User press logout button.	If user press logout button user should navigate to system home page.	If user press logout button user will navigate to system home page.	“Pass”

4.2.3. Test Evaluation

Test evaluation need to ensure testing process is in line. Software test exit criteria evaluation, we assess the test execution against the defined and agreed exit criteria for a particular test level. Based on this evaluation we can decide if we have actually done enough testing for that level to mark it officially complete.

4.2.4. User Evaluation of the Proposed System

System functionalities evaluate using client response about this system. User satisfaction evaluation questionnaire given to some doctors and patient to get the user feedback. The questioner was cover following functionalities.

- Response time
- Easy of learn
- Readability
- Usability
- Readiness
- Visibility

4.2.5. User Evaluation Sample Questionnaire

User feedback about the system gathered by using a questionnaire. It consists close ended question related to system evaluation. Therefor user can answer the all questions within a short time period. User response rate also high in this method. Ten questionnaires are given to ten users and gathered their response. User feedback questioner shown in table 4.4.

Table 4.4 :- User feedback questioner

E – Medihelp User Feedback Questionnaire						
Please mark ✓ for your choice.						
Q No	**Question**	**Strong agree**	**Agree**	**Intermediate**	**Disagree**	**Strong disagree**
1	The system responds with short time period.					
2	User motivate to use the system.					
3	A system able to handle easily without more pre-learning.					
4	Entire system appearance good.					
5	System components and contents are clear.					
6	Used color them is clear.					
7	Easy to learn system.					

User response related to each question shown in table 4.5.

Table 4.5 :- User feedback questioner result

Q No	**Response (Out of ten users)**				
	Strong agree	**Agree**	**Intermediate**	**Disagree**	**Strong Disagree**
1	6	2	2	0	0
2	5	3	2	0	0
3	7	1	2	0	0
4	7	3	0	0	0
5	5	4	1	0	0
6	5	2	3	0	0
7	8	1	1	0	0

The user response category's rate shown in table 4.6.

Table 4.6 :- User response rate

Response	Rate
Strong agree	10.0
Agree	7.5
Intermediate	5.0
Disagree	2.5
Strong disagree	0.0

According the above result final rate of each question shown in table 4.7.

Table 4.7 :- User response final rate

Question No	Response out of eight users	
	Final rate out of hundred	
1	(6*10+2*7.5+2*5)%	85.0%
2	(5*10+3*7.5+2*5)%	82.5%
3	(7*10+1*7.5+2*5)%	87.5%
4	(7*10+3*7.5)%	92.5%
5	(5*10+4*7.5+1*5)%	85.0%
6	(5*10+2*7.5+3*5)%	80.0%
7	(8*10+1*7.5+1*5)%	92.5%
	Average	86.0%

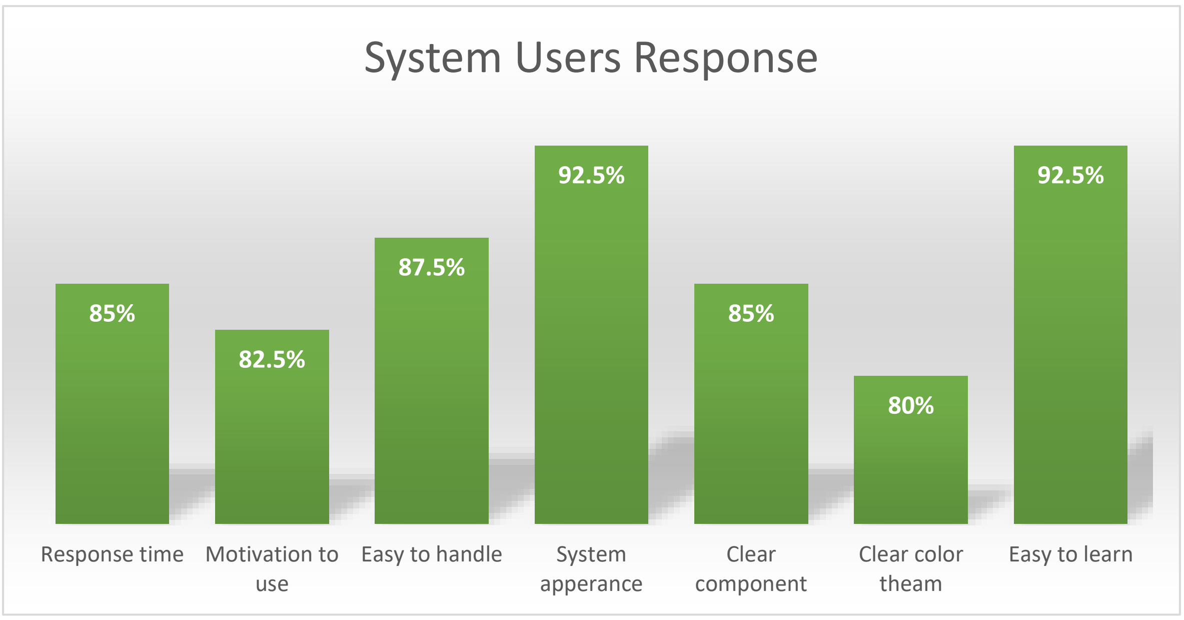


Figure 4.14 :- System user response

5.0. Conclusion

In this chapter discuss obstacles faced in this system implementation. How learnt things using encountered problems. Critical evaluation of this project and further improvement of this project.

5.1. Problem Encountered and Lessons Learnt

Initial stage of this project discussed with my cousin’s doctor about this project idea. In this stage he implies a smaller number of requirements. Using identified requirements prepared manual document. During the project idea discussion stage met few doctors and identify their true requirements. According to gathered requirement change the manually prepared document. After changes made to meet the clients and verify the captured requirement and rest of the mistakes are resolved and comes to the final requirements and start to the development process.

Requirement gathering stage faced some difficulties.

- Finding clients from different area in the country.
- Difficult to meet the client regularly.
- Different clients had different ideas about particular requirement from their perspective.
- Unable to capture client complete requirements at first phrase.
- Doctor’s corporation was less at the beginning.

Requirement gathering stage client and developer communication is essential to clearly identify client requirements. Clients’ confidence about the software can make when they met and discuss the software idea.

Software requirement analysis part is essential to this project. Hence this project idea was vague. Because this was not a single client project. This implementation affects to many clients in the medical service industry.

Software development stage faced many problems. Some of them are as follows.

- Most of development techniques, learn through the self-studies. Hence had to spend a lot of time for that.
- Development environment had to establish home laptop as well as office laptop. Because less time for project development.

- When having two development environments need to update both of them with newest code segment.
- Free times of some office hours also had to code to complete the project on time. Hence had to daily basis update office work environment and code the project. Evening update home work environment and code the rest. This process has continued until project implementation complete.

During the project development stage lot of technical things able to learn.

- Learn session management in JEE.
- Learn servlet deeply.
- JSP implementation in web page deeply learns.
- JavaScript learns deeply since google map data gathered using JavaScript.
- Throughout this project implementation learn a lot of things about JEE and MySQL.
- Learn MVC implementation thoroughly.
- Beginning of the project Bootstrap, XAMP knowledge was poor since implementation process learn a lot of things.

5.2. Project Critical Evaluation

This web-based application has a positive user (doctors and patient) feedback. 86% users have positive feedback about this system. User feedback about the system shown in table 5.1.

Table 5.1 :- User feedback about the system

Component	**User's comment**
System security	This system has well managed its users. No one can't access the system until system administrator's approval. Therefor clients satisfied about the system.
Maintainability	Clients expect developer support while the system in a live environment. Because system failures may happen in a live environment. Clients agreed to further new feature development with payments. Hence client satisfied about system maintenance.
Usability	Users able to easily learn the system and interact with it. Therefor no need much training about this. Therefor clients satisfied about software usability.

Data accuracy	The client expects 100% data accuracy. It’s difficult to measure until some user experience with the system. Client confidence about the data accuracy will develop while using the system.
User interface	Clients satisfied about the clear and attractive user interfaces.

Proposed system compares with the similar kind of systems in the industry. System comparison shown in table 5.2.

Table 5.2 :- System comparison

	Proposed System	**Similar Systems**
Clear information	Clearly display information using standard font size and fonts.	It is a different system to system. Some systems complicate since some are simple.
User attraction	Present information in a more attractive way using web modals and dynamic headers and footers.	Most of the systems consist with new web elements which can use to increase attraction.
Multimedia usage	Images use.	Most of the systems consist with images, videos, and graphics.
System login	Single login for any kind of users.	Most of the system have.
System signup	Specific signup according to the user.	Don’t have.
System dashboard	Have	May or may not have.
Online users	Haven’t	May or may not have.
Location base search	Have	Don’t have.
System user management	Have	Don’t know.
Localization	Don’t have.	Most of the system have.
Patient detail maintain	Have	May or may not have.
Promotion	Registered medical center promotion available.	Most of the system have promotions.

Client ratings	Don’t have.	Don’t know.
Medical center account	It is the unique feature of this system.	Don’t have.
Sub user creation	Medical center administrator able to create their own users.	Don’t have.
User feedback	System users able to send their feedback.	May or may not have.

5.3. Future Improvements

- Mobile application for E-Medihelp is the next large step of this product. Because it is the most convenient way to access the system for patient and doctors.
- Doctor voice recognition and prescription write through doctor voice command will be value added features for this system. Voice command can integrate with Google home.
- Online health assistant service can introduce to their registered patients. It’s helpful to enhance emergency treatment.

6.0. References

[1] Community, "Apache Friends," Apache, May 2002. [Online]. Available: https://www.apachefriends.org. [Accessed 3 August 2018].

[2] Seyed M.M. Tahaghoghi, Hugh E. Williams, Learning MySQL, Sebastopol: Reilly Media, 2006.

[3] O. Corporation, "Apache NetBeans," Oracle Corporation , 2015. [Online]. Available: https://netbeans.org/. [Accessed 1 May 2018].

[4] S. Ground, "Site Ground," 2018. [Online]. Available: https://www.siteground.com/tutorials/phpmyadmin/database-management/. [Accessed 5 June 2018].

[5] Stephanie Bodoff, Dale Green, Kim Haase, Eric Jendrock, Monica Pawlan, Beth Stearns, "J2EE Tutorial," March 2002. [Online]. Available: http://people.cs.ksu.edu/~pra4444/j2ee-1_3-doc-tutorial-draft5.pdf. [Accessed 10 June 2018].

[6] R. Data, "W3Schools," W3Schools, 1998. [Online]. Available: https://www.w3schools.com/. [Accessed 10 May 2018].

[7] Community, "Bootstrap," Bootstrap, 19 August 2011. [Online]. Available: https://getbootstrap.com/. [Accessed 8 June 2018].

[8] M. o. Health, "Ministry of Health, Nutrition & Indigenous Medicine," Government, 2016. [Online]. Available: http://www.health.gov.lk/moh_final/english/. [Accessed 12 August 2018].

[9] M. IT, "E Channelling," e-Channelling PLC, 2001. [Online]. Available: https://www.echannelling.com. [Accessed 25 December 2018].

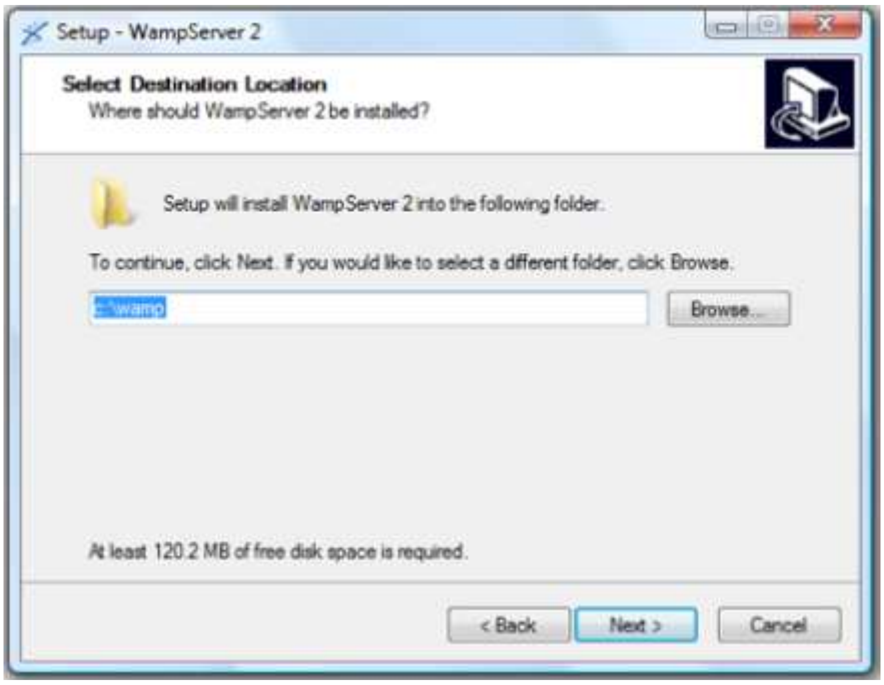
[10] I. I. Systems, "Doc990," Digital Health Private Limited , 2016. [Online]. Available: https://www.doc.lk/. [Accessed 29 December 2018].

[11] Keogh, James Edward, J2EE : the complete reference, London: McGraw-Hill , 2002.

Appendix A – System documentation

❖ WAMP server installation

Initially need to download WAMP server .exe file. Then run the installation file. After running few windows appear and prompt installation folder selection window.



Setup - WampServer 2

Select Destination Location

Where should WampServer 2 be installed?

Setup will install WampServer 2 into the following folder.

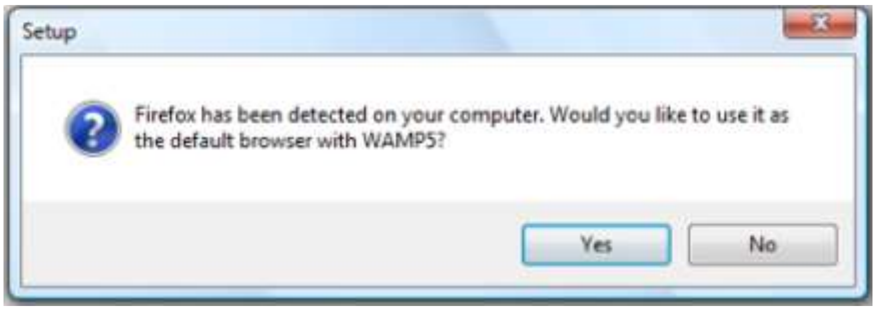
To continue, click Next. If you would like to select a different folder, click Browse.

c:\wamp

At least 120.2 MB of free disk space is required.

Figure A.1 :- WAMP server setup 1

Once it completes the setup process will begin and WAMP ask the default browser want to use.

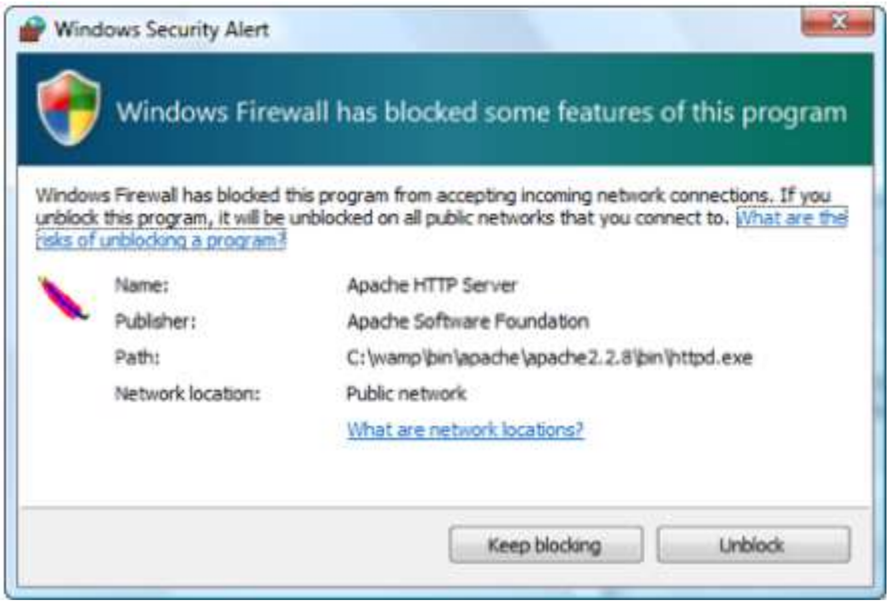


Setup

Firefox has been detected on your computer. Would you like to use it as the default browser with WAMP5?

Figure A.2 :- WAMP server setup 2

In this step if software Firewall was installed it will probably pop up and warn that Apache wants to accept incoming connections. Depending on the Firewall the “button” to click may be different, but it will be something similar to unblock. Press unblock button to proceed.



Windows Security Alert

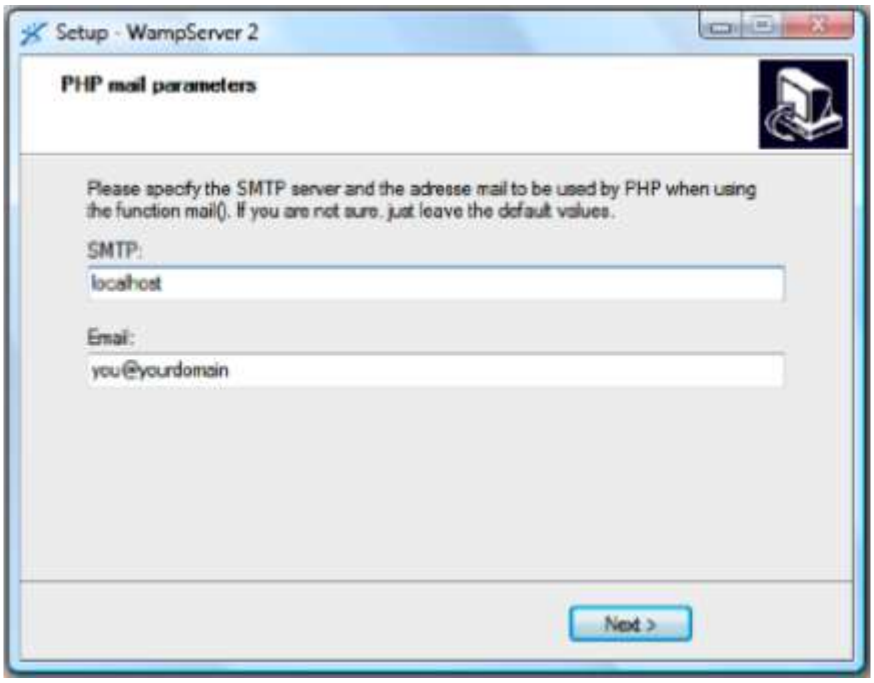
Windows Firewall has blocked some features of this program

Windows Firewall has blocked this program from accepting incoming network connections. If you unblock this program, it will be unblocked on all public networks that you connect to. What are the risks of unblocking a program?

Name:	Apache HTTP Server
Publisher:	Apache Software Foundation
Path:	C:\wamp\bin\apache\apache2.2.8\bin\httpd.exe
Network location:	Public network
	What are network locations?

Figure A.3 :- WAMP server setup 3

Leave the SMTP: server set as localhost and change the email address and click next.



Setup - WampServer 2

PHP mail parameters

Please specify the SMTP server and the adresse mail to be used by PHP when using the function mail(). If you are not sure, just leave the default values.

SMTP: localhost

Email: you@yourdomain

Figure A.4 :- WAMP server setup 4

WAMP server installation complete.

❖ WAMP server and phpMyAdmin configuration

Using one of the icons created on start menu or desktop, open the WAMP server. Once opened it appears in the lower right-hand corner of the screen.

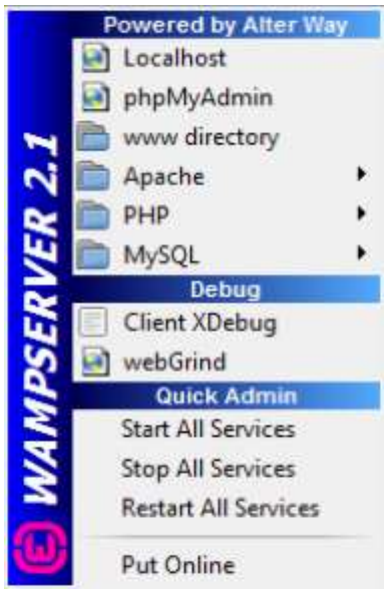


Figure A.5 :- WAMP server configuration 1

If WAMP server not started need to look small green icon in the toolbar. It indicates WAMP server status. Its read mean server is stopped. Green means everything is running smoothly. Orange mean some services are running.

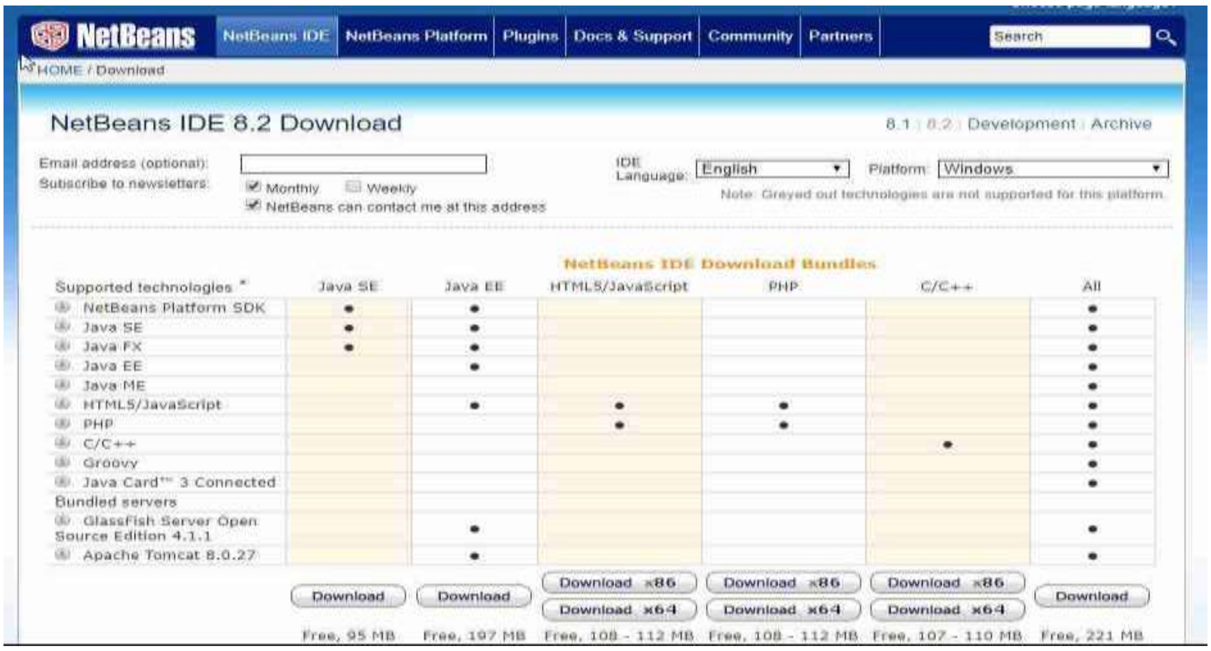
If everything was installed correctly following screen will appear when click the localhost link in WAMP management console.

Figure A.6 :- WAMP server configuration 2

The default phpMyAdmin username is root and no password.

❖ Installation of NetBeans

Go to the NetBeans IDE website and find the relevant version to download.


NetBeans IDE 8.2 Download

8.1 | 8.2 | Development | Archive

Email address (optional):
Subscribe to newsletters: ☑ Monthly ☐ Weekly
☑ NetBeans can contact me at this address

IDE Language: English
Platform: Windows

Note: Greyed out technologies are not supported for this platform.

NetBeans IDE Download Bundles

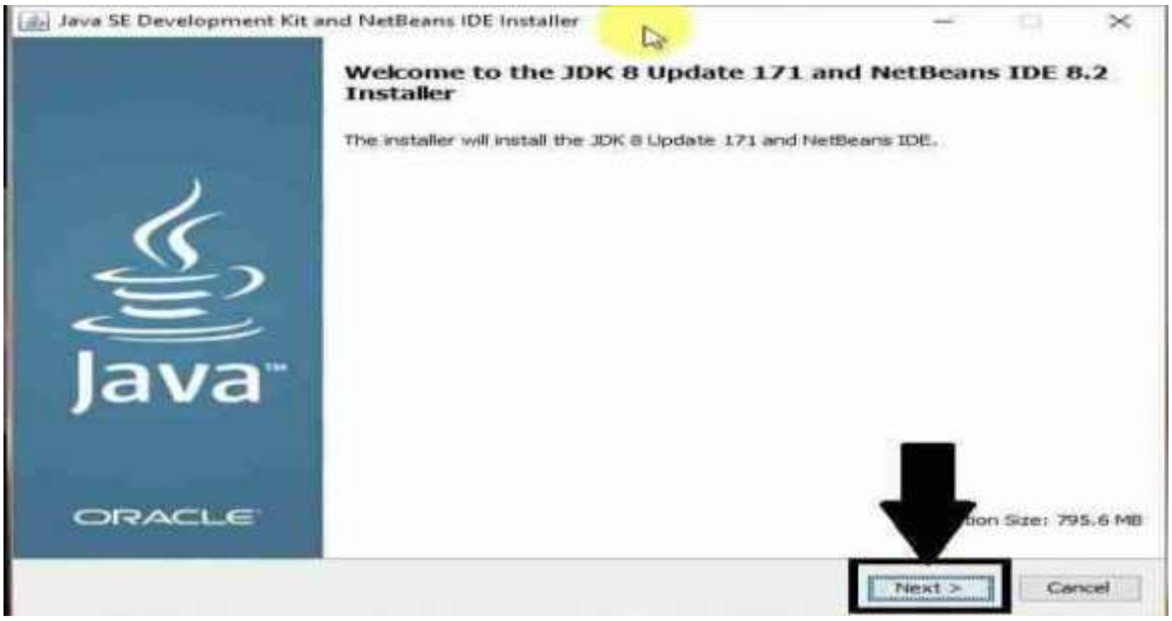
Supported technologies *	Java SE	Java EE	HTML5/JavaScript	PHP	C/C++	All
NetBeans Platform SDK	●	●				●
Java SE	●	●				●
Java FX	●	●				●
Java EE		●				●
Java ME						●
HTML5/JavaScript		●	●	●		●
PHP			●	●		●
C/C++					●	●
Groovy						●
Java Card™ 3 Connected						●
Bundled servers						
GlassFish Server Open Source Edition 4.1.1		●				●
Apache Tomcat 8.0.27		●				●
	Download	Download	Download x86 Download x64	Download x86 Download x64	Download x86 Download x64	Download
	Free, 95 MB	Free, 197 MB	Free, 108 - 112 MB	Free, 108 - 112 MB	Free, 107 - 110 MB	Free, 221 MB

Figure A.7 :- NetBeans installation

Click on the ‘Download’ button which belongs to relevant version. Some of them have x86 versions and x64 versions. Need to download the appropriate one for the system.

JDK 8 or more is necessary for NetBeans therefor need to installed it to Windows.

Open the downloaded NetBeans installer .exe file. Then prompt the following screen. After that, a new Window will appear, then need to click on ‘Next’.

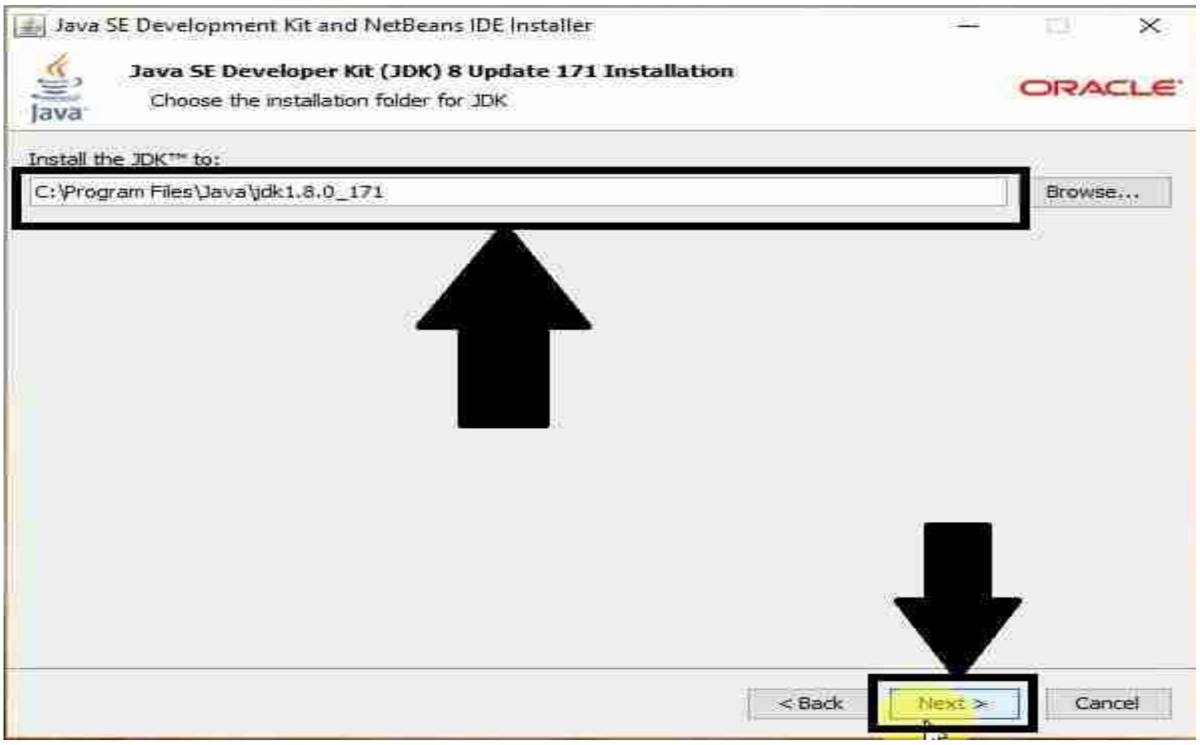

Java SE Development Kit and NetBeans IDE Installer

Welcome to the JDK 8 Update 171 and NetBeans IDE 8.2 Installer

The installer will install the JDK 8 Update 171 and NetBeans IDE.

Figure A.8 :- NetBeans setup 1

Now need to choose the location, where the JDK will exist and click on ‘Next’.



Java SE Development Kit and NetBeans IDE Installer

Java SE Developer Kit (JDK) 8 Update 171 Installation

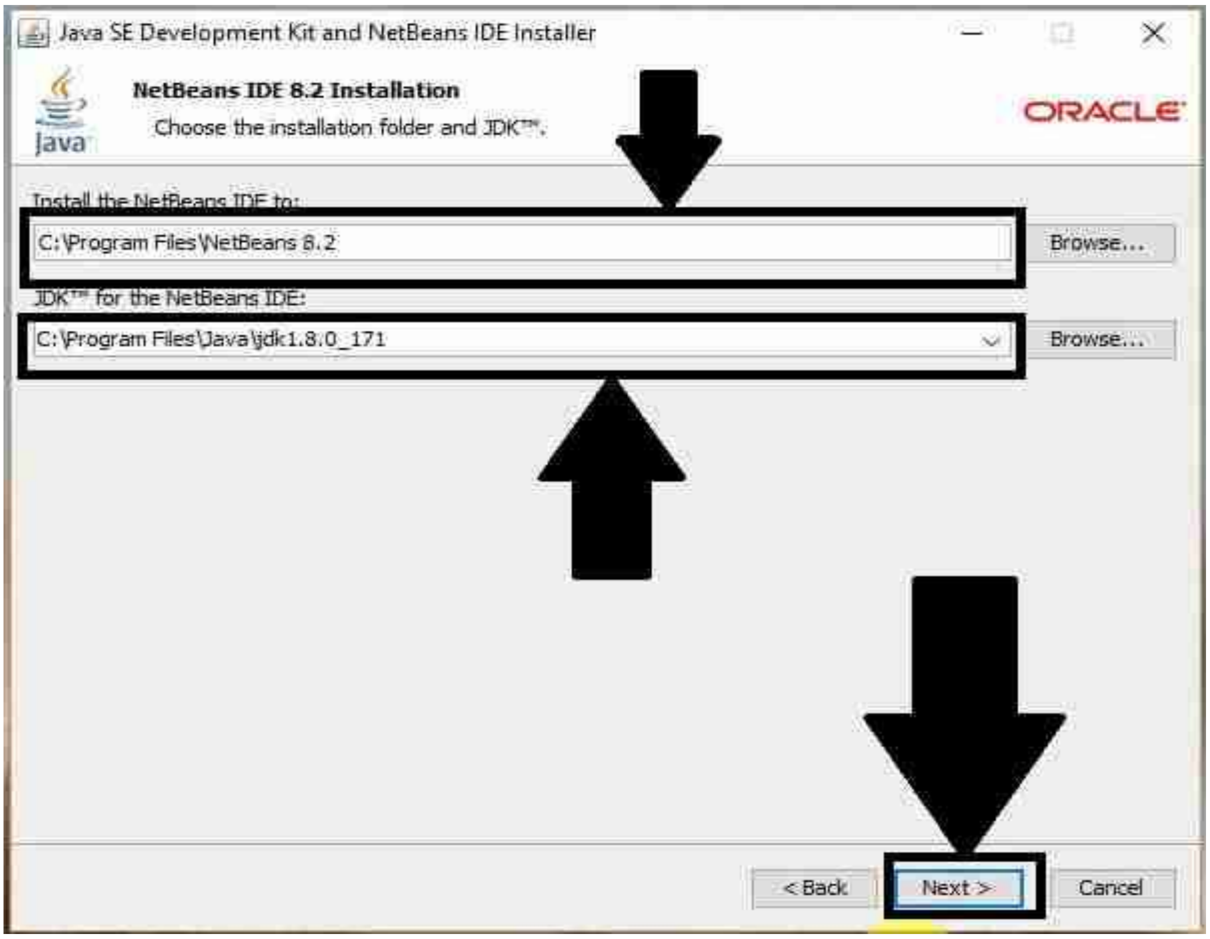
Choose the installation folder for JDK

Install the JDK™ to:

C:\Program Files\Java\jdk1.8.0_171

Figure A.9 :- NetBeans setup 2

Now need to choose the location to install NetBeans IDE, and the JDK for the NetBeans IDE.



Java SE Development Kit and NetBeans IDE Installer

NetBeans IDE 8.2 Installation

Choose the installation folder and JDK™.

Install the NetBeans IDE to:

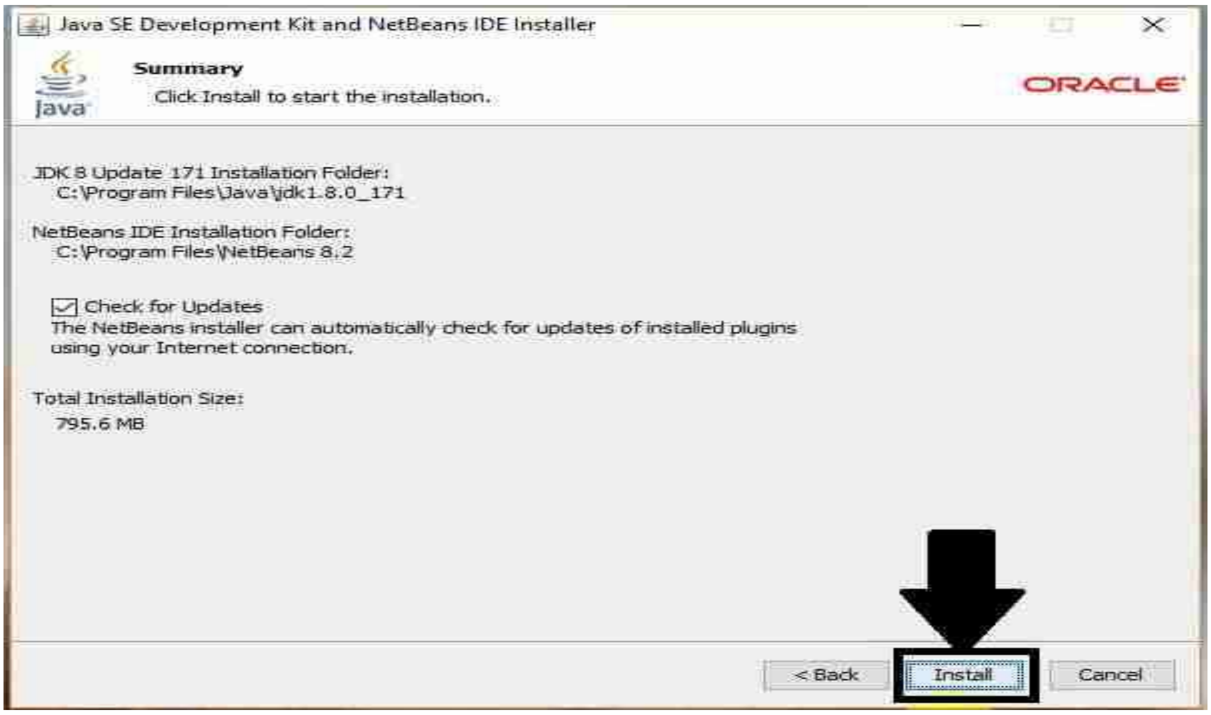
C:\Program Files\NetBeans 8.2

JDK™ for the NetBeans IDE:

C:\Program Files\Java\jdk1.8.0_171

Figure A.10 :- NetBeans setup 3

Check the small checkbox to automatically download and install updates. Hit the ‘Install’ button to start installing.


Java SE Development Kit and NetBeans IDE Installer

Summary
Click Install to start the installation.

JDK 8 Update 171 Installation Folder:
C:\Program Files\Java\jdk1.8.0_171

NetBeans IDE Installation Folder:
C:\Program Files\NetBeans 8.2

☑ Check for Updates
The NetBeans installer can automatically check for updates of installed plugins using your Internet connection.

Total Installation Size:
795.6 MB

Figure A.11 :- NetBeans setup 4

Once the installation complete need to click on the ‘Finish’ button to exit the installer.

The installation is complete. Now, NetBeans can access on Windows from the Start menu, and from the Desktop. On opening NetBeans IDE following screen will appear.

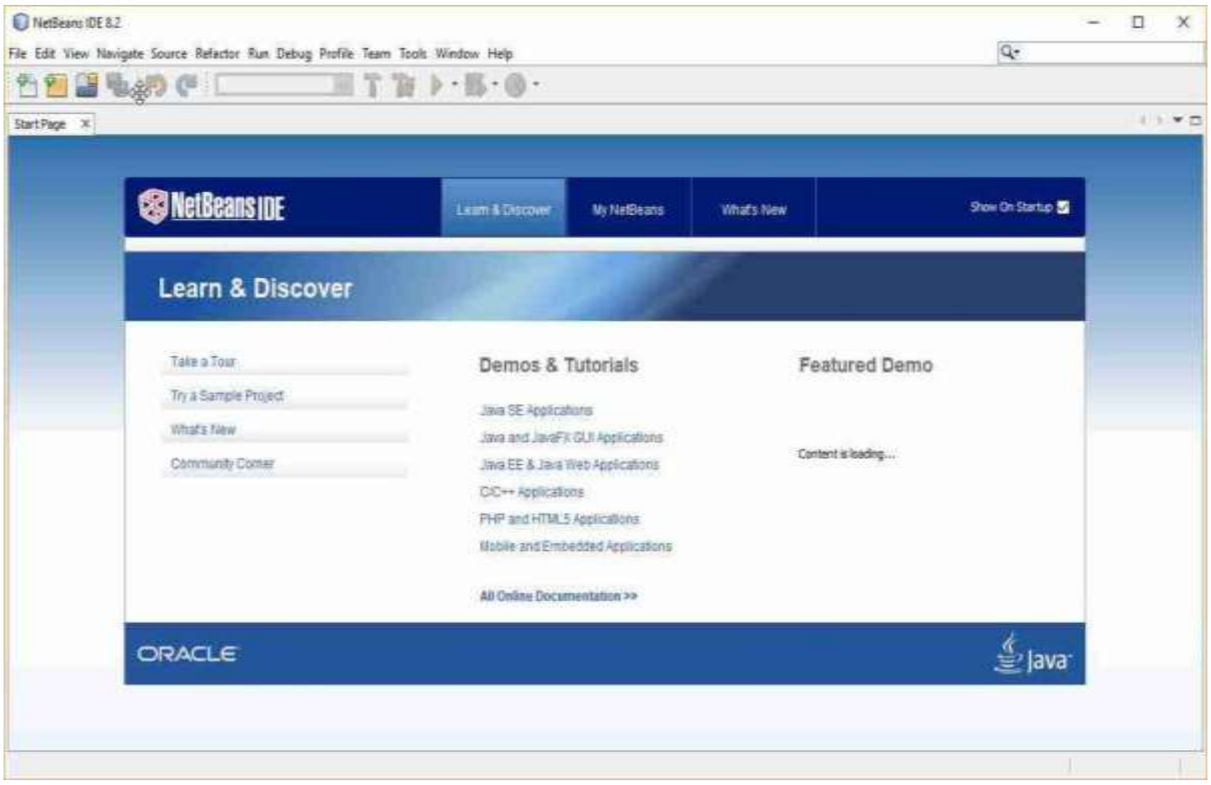
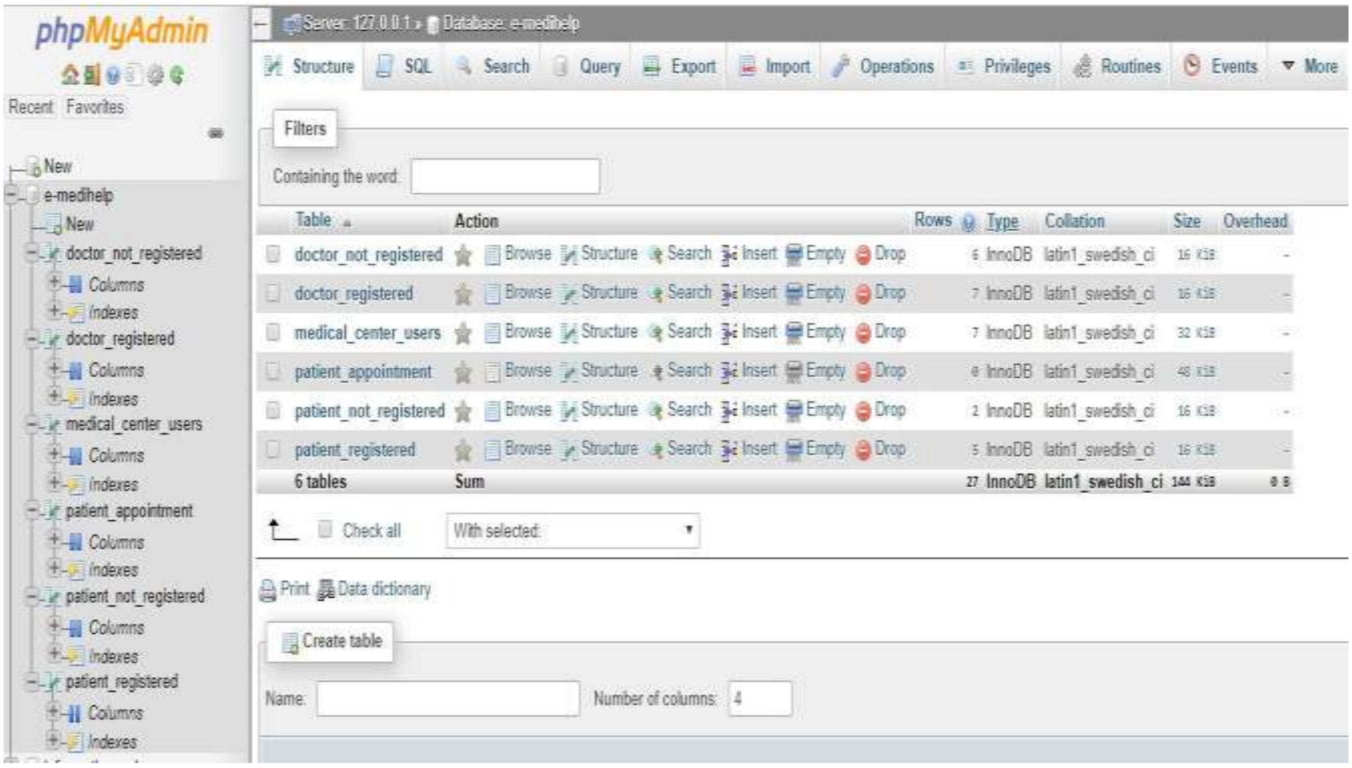


Figure A.12 :- NetBeans setup 5

Appendix B – Design documentation

A database was created as “e-medihelp” and tables are included in there. Table structure of each table is listed below.

- E-medihelp database

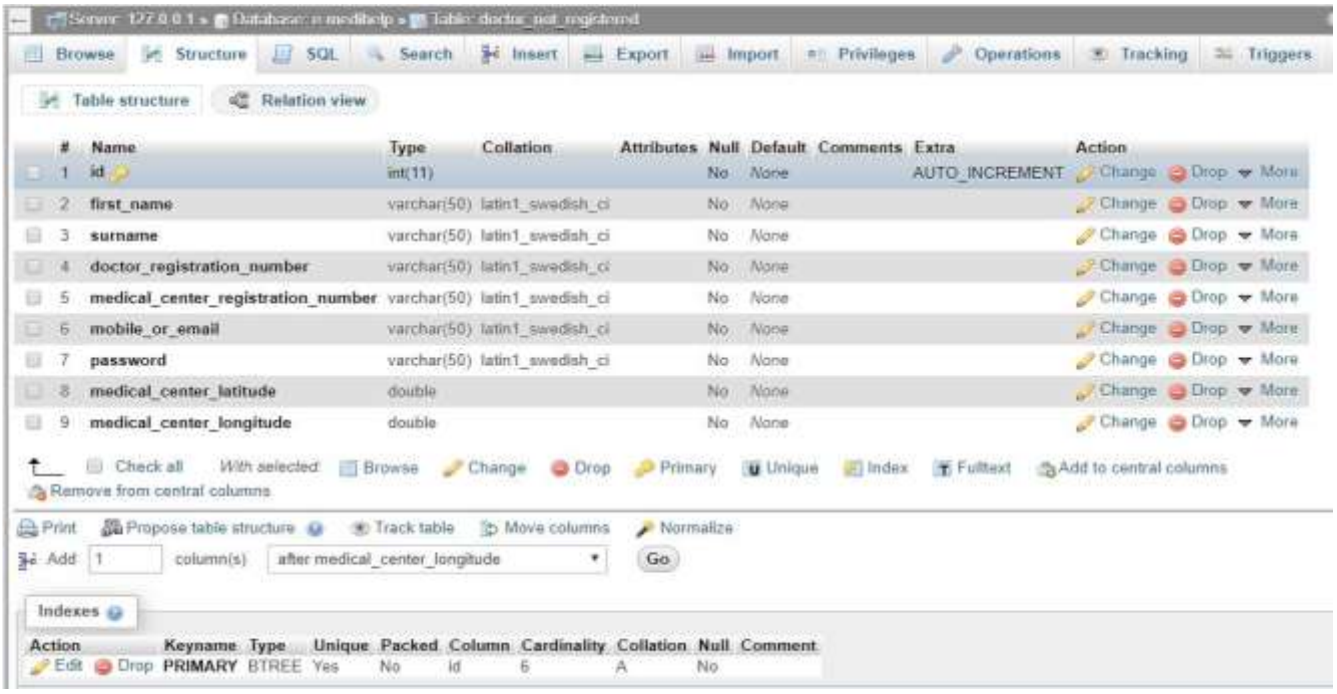


Server: 127.0.0.1 » Database: e-medihelp

Table	Action	Rows	Type	Collation	Size	Overhead
doctor_not_registered	Browse Structure Search Insert Empty Drop	6	InnoDB	latin1_swedish_ci	16 KiB	-
doctor_registered	Browse Structure Search Insert Empty Drop	7	InnoDB	latin1_swedish_ci	16 KiB	-
medical_center_users	Browse Structure Search Insert Empty Drop	7	InnoDB	latin1_swedish_ci	32 KiB	-
patient_appointment	Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	48 KiB	-
patient_not_registered	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	16 KiB	-
patient_registered	Browse Structure Search Insert Empty Drop	5	InnoDB	latin1_swedish_ci	16 KiB	-
6 tables	Sum	27	InnoDB	latin1_swedish_ci	144 KiB	0 B

Figure B.1 :- E-Medihelp database

- Signup doctor’s table



Server: 127.0.0.1 » Database: e-medihelp » Table: doctor_not_registered

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	first_name	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
3	surname	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
4	doctor_registration_number	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
5	medical_center_registration_number	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
6	mobile_or_email	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
7	password	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
8	medical_center_latitude	double			No	None			Change Drop More
9	medical_center_longitude	double			No	None			Change Drop More

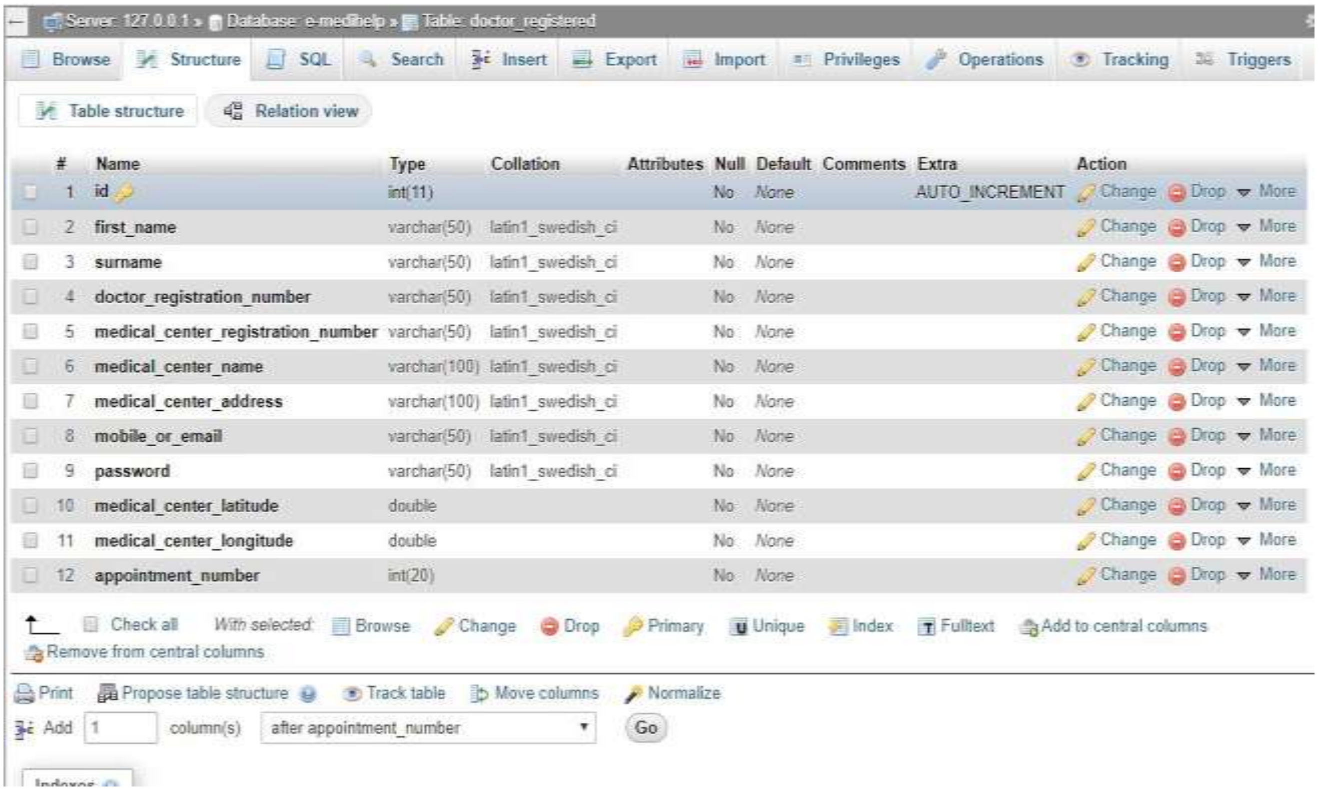
Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	id	6	A	No	

Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers
Table structure Relation view
Check all With selected: Browse Change Drop Primary Unique Index Fulltext Add to central columns
Remove from central columns
Print Propose table structure Track table Move columns Normalize
Add 1 column(s) after medical_center_longitude Go

Figure B.2 :- Signup doctor table

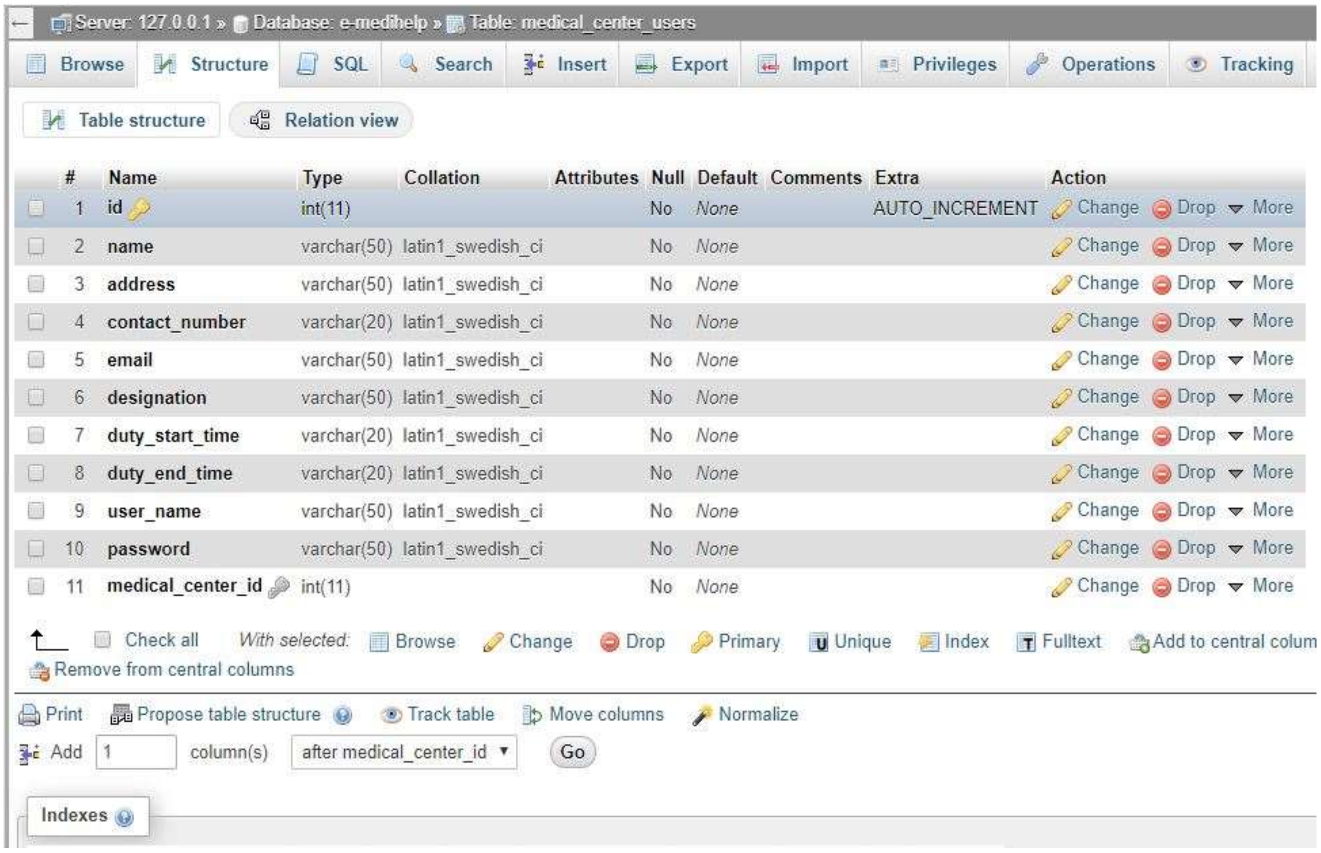
- Registered doctors table



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	first_name	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
3	surname	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
4	doctor_registration_number	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
5	medical_center_registration_number	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
6	medical_center_name	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
7	medical_center_address	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
8	mobile_or_email	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
9	password	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
10	medical_center_latitude	double			No	None			Change Drop More
11	medical_center_longitude	double			No	None			Change Drop More
12	appointment_number	int(20)			No	None			Change Drop More

Figure B.3 :- Registered doctor table

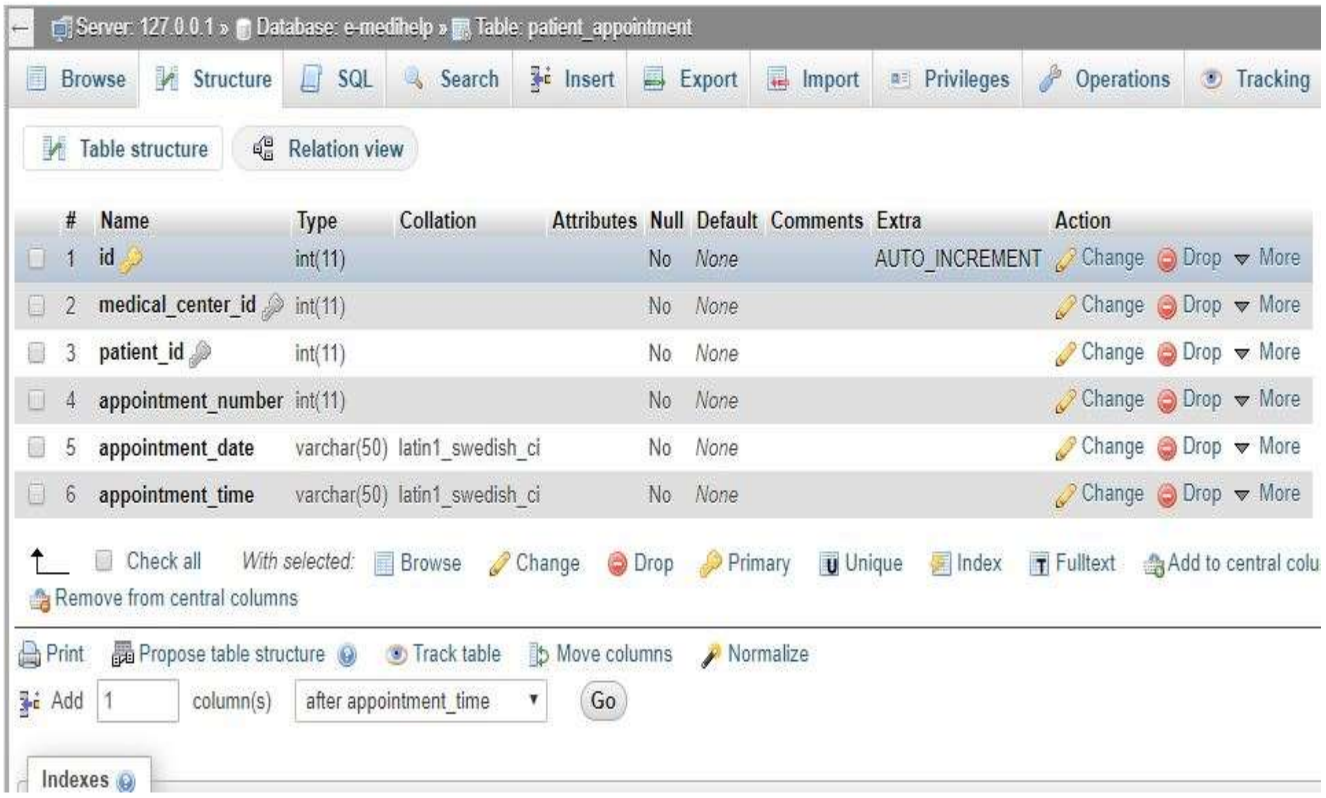
- Medical center all user table



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
3	address	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
4	contact_number	varchar(20)	latin1_swedish_ci		No	None			Change Drop More
5	email	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
6	designation	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
7	duty_start_time	varchar(20)	latin1_swedish_ci		No	None			Change Drop More
8	duty_end_time	varchar(20)	latin1_swedish_ci		No	None			Change Drop More
9	user_name	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
10	password	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
11	medical_center_id	int(11)			No	None			Change Drop More

Figure B.4 :- Medical center all user table

- Patient appointment table

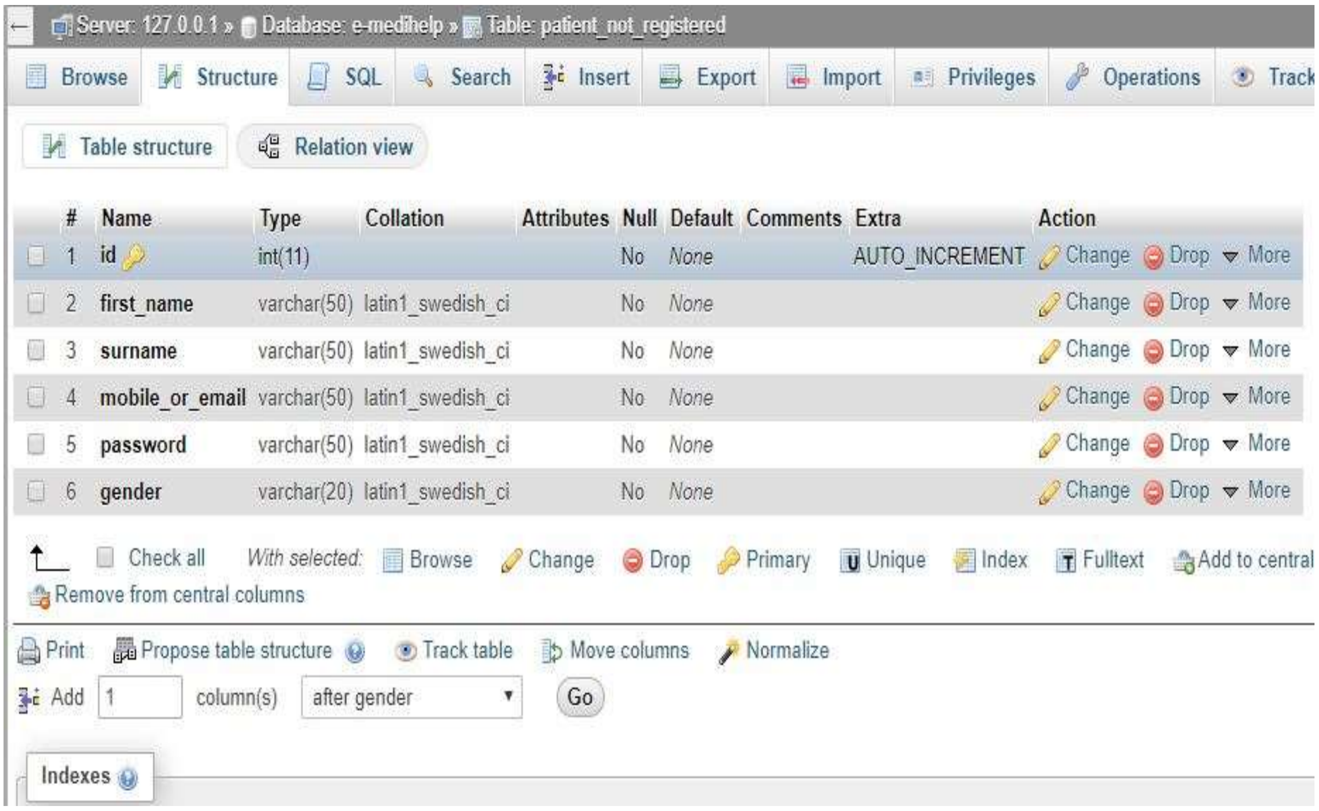

Server: 127.0.0.1 » Database: e-medihelp » Table: patient_appointment
Browse Structure SQL Search Insert Export Import Privileges Operations Tracking
Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	medical_center_id	int(11)			No	None			Change Drop More
3	patient_id	int(11)			No	None			Change Drop More
4	appointment_number	int(11)			No	None			Change Drop More
5	appointment_date	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
6	appointment_time	varchar(50)	latin1_swedish_ci		No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Fulltext Add to central colu
Remove from central columns
Print Propose table structure Track table Move columns Normalize
Add 1 column(s) after appointment_time Go
Indexes

Figure B.5 :- Patient appointment table

- Signup patient table

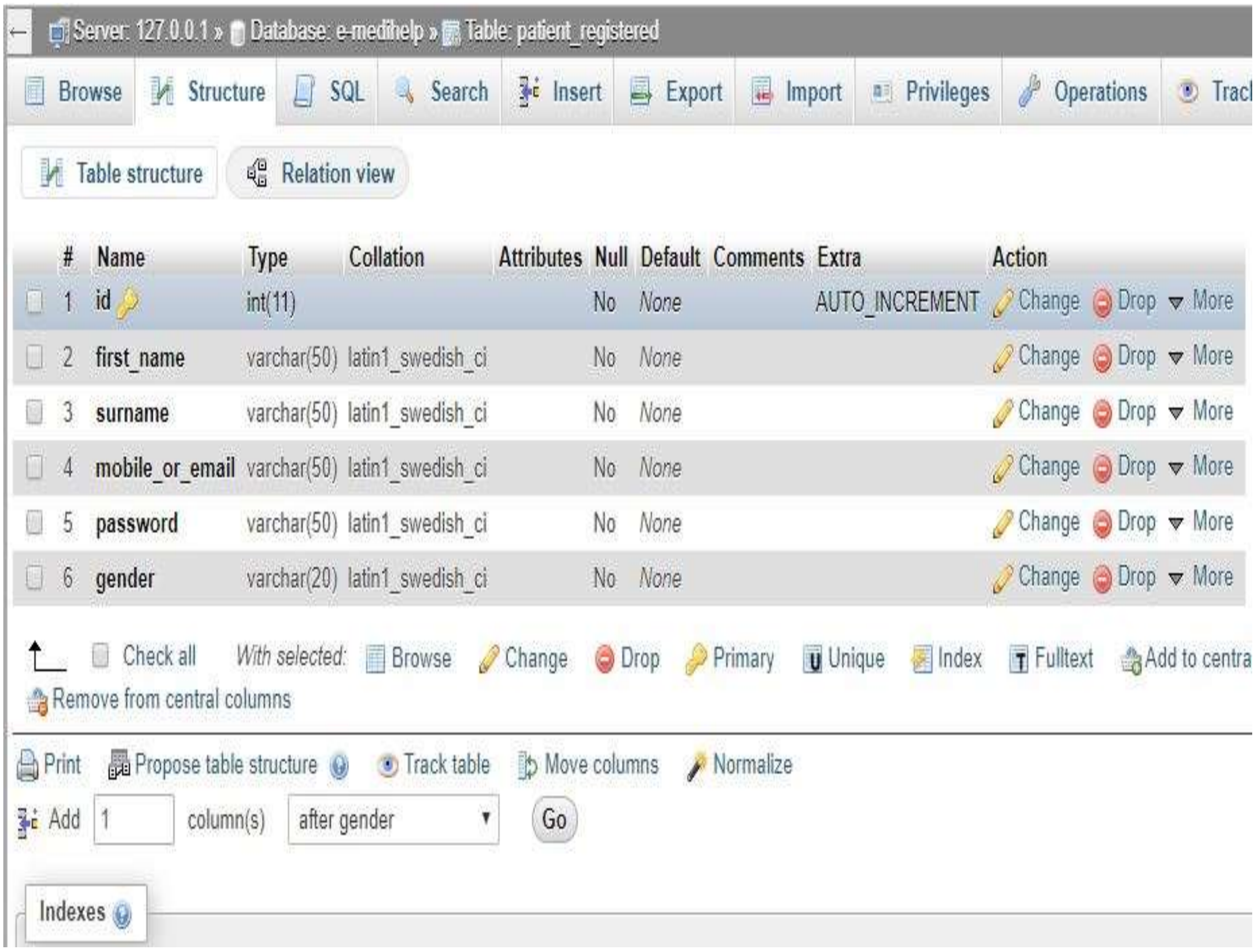

Server: 127.0.0.1 » Database: e-medihelp » Table: patient_not_registered
Browse Structure SQL Search Insert Export Import Privileges Operations Track
Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	first_name	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
3	surname	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
4	mobile_or_email	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
5	password	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
6	gender	varchar(20)	latin1_swedish_ci		No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Fulltext Add to central
Remove from central columns
Print Propose table structure Track table Move columns Normalize
Add 1 column(s) after gender Go
Indexes

Figure B.6 :- Signup patient table

- Registered patient table



Server: 127.0.0.1 » Database: e-medihelp » Table: patient_registered

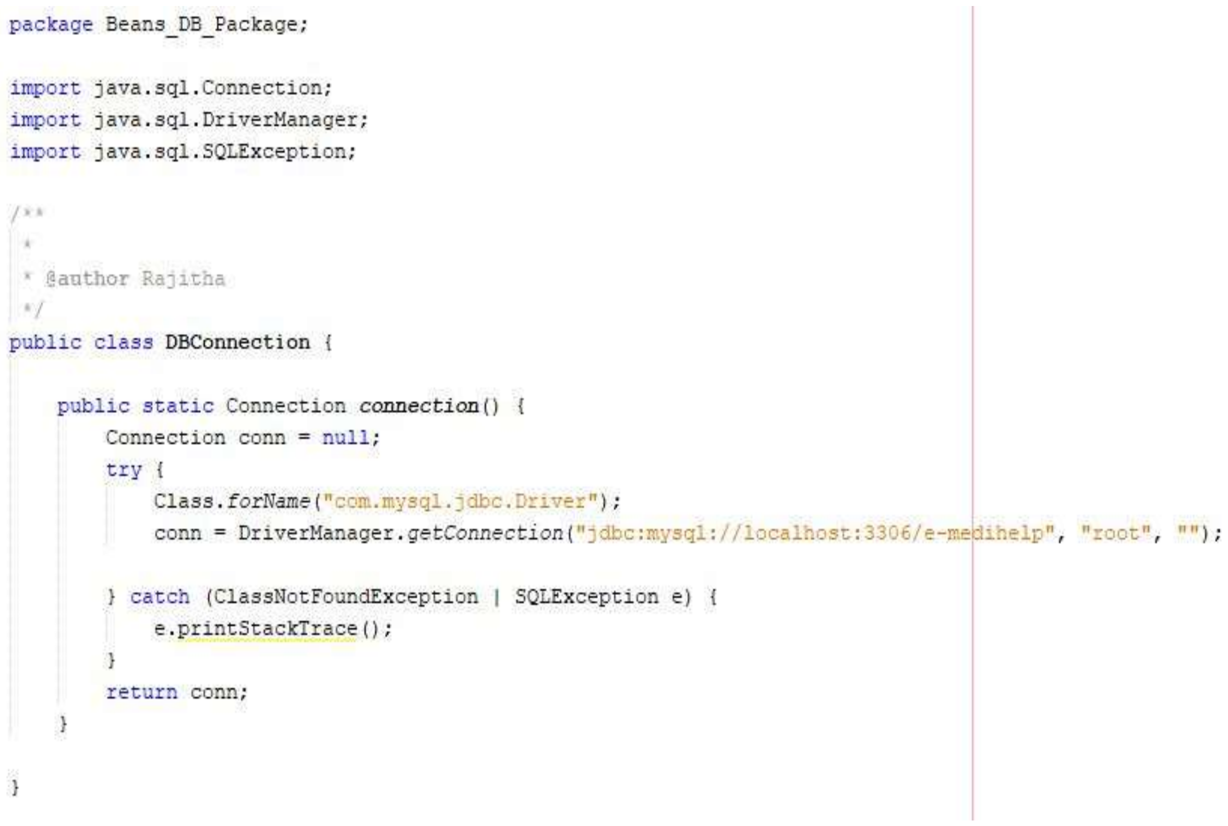
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	first_name	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
3	surname	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
4	mobile_or_email	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
5	password	varchar(50)	latin1_swedish_ci		No	None			Change Drop More
6	gender	varchar(20)	latin1_swedish_ci		No	None			Change Drop More

Figure B.7 :- Registered patient table

Appendix C – Coding

Main code structures listed below.

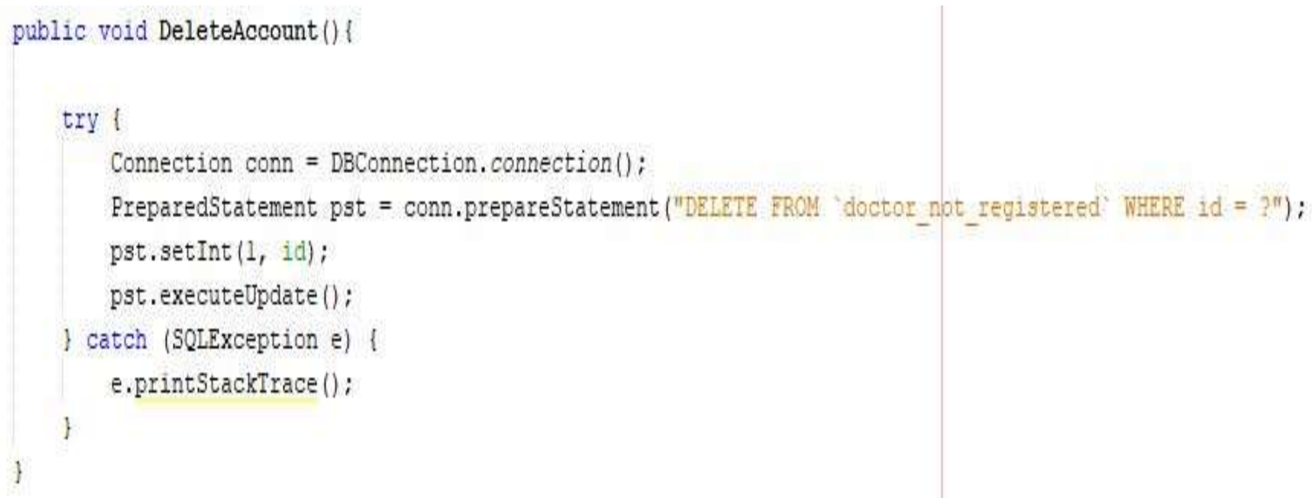
- Database connection script



```
package Beans_DB_Package;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

/**
 *
 * @author Rajitha
 */
public class DBConnection {

    public static Connection connection() {
        Connection conn = null;
        try {
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/e-medihelp", "root", "");

        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
        }
        return conn;
    }

}
```

Figure C.1 :- Database connection script

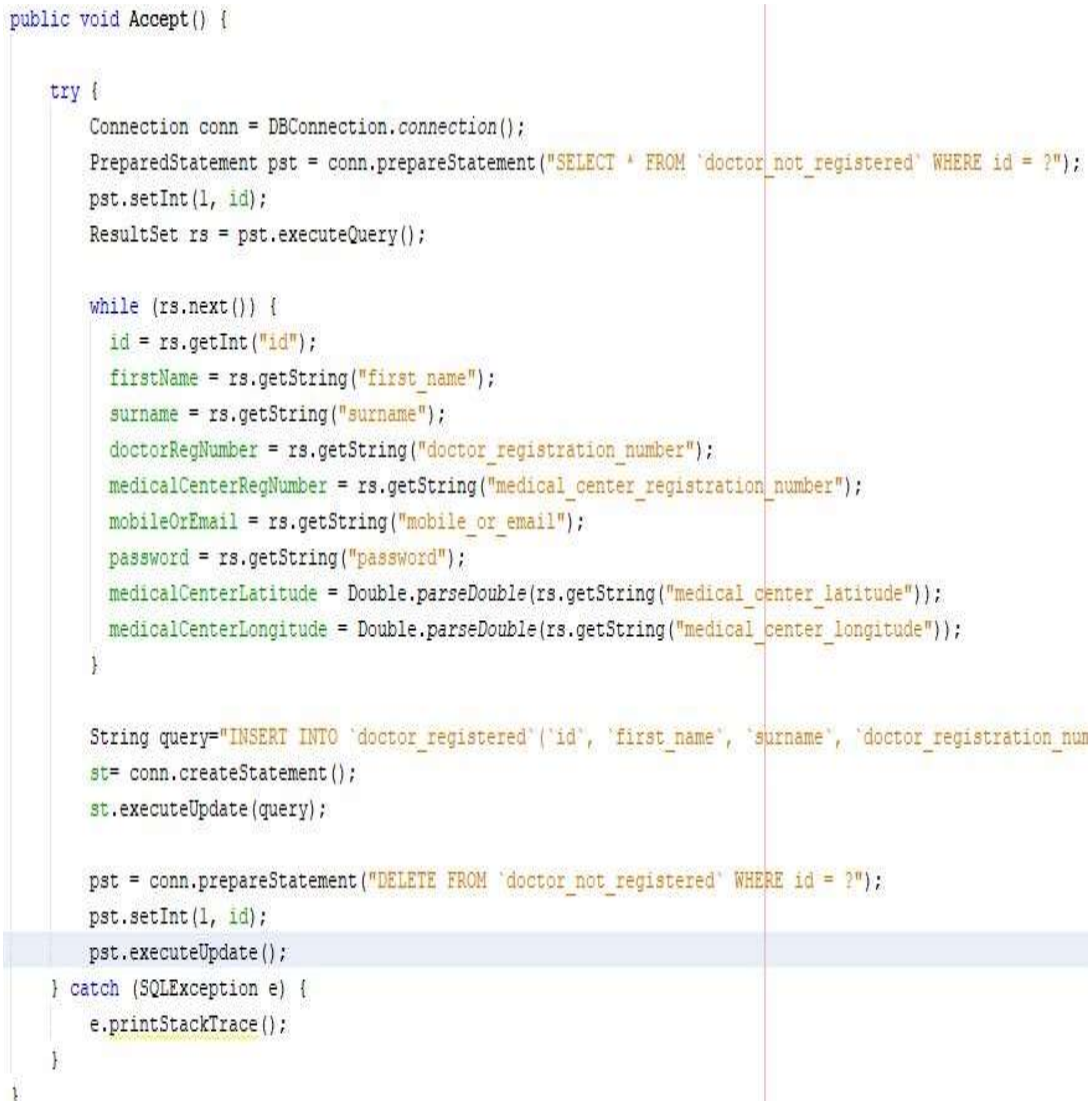
- New medical center request deletes script



```
public void DeleteAccount(){

    try {
        Connection conn = DBConnection.connection();
        PreparedStatement pst = conn.prepareStatement("DELETE FROM `doctor_not_registered` WHERE id = ?");
        pst.setInt(1, id);
        pst.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

Figure C.2 :- New medical center request delete script

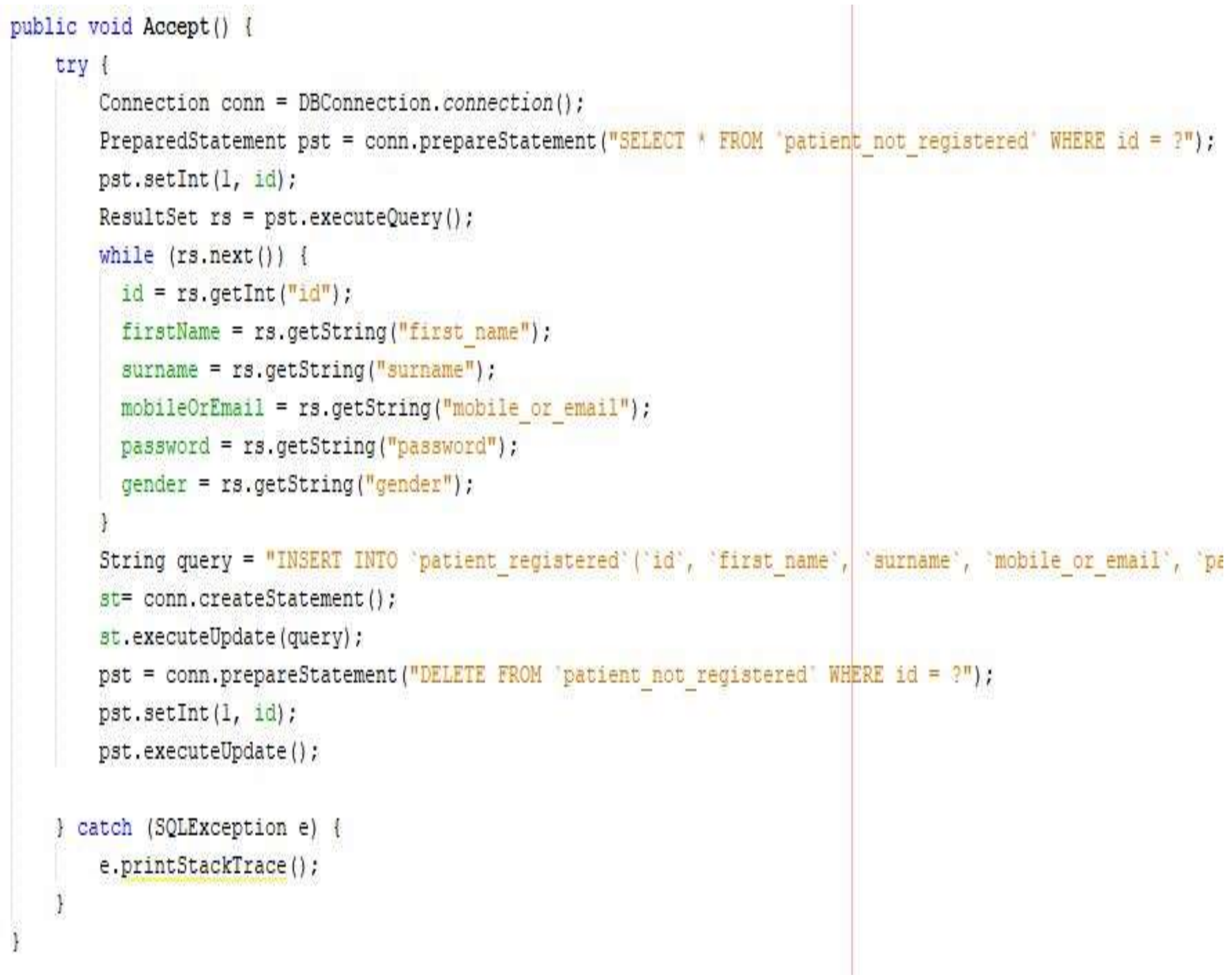
- New medical center request acceptance script



```
public void Accept() {

    try {
        Connection conn = DBConnection.connection();
        PreparedStatement pst = conn.prepareStatement("SELECT * FROM `doctor_not_registered` WHERE id = ?");
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();

        while (rs.next()) {
          id = rs.getInt("id");
          firstName = rs.getString("first_name");
          surname = rs.getString("surname");
          doctorRegNumber = rs.getString("doctor_registration_number");
          medicalCenterRegNumber = rs.getString("medical_center_registration_number");
          mobileOrEmail = rs.getString("mobile_or_email");
          password = rs.getString("password");
          medicalCenterLatitude = Double.parseDouble(rs.getString("medical_center_latitude"));
          medicalCenterLongitude = Double.parseDouble(rs.getString("medical_center_longitude"));
        }

        String query="INSERT INTO `doctor_registered`(`id`, `first_name`, `surname`, `doctor_registration_num
        st= conn.createStatement();
        st.executeUpdate(query);

        pst = conn.prepareStatement("DELETE FROM `doctor_not_registered` WHERE id = ?");
        pst.setInt(1, id);
        pst.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

Figure C.3 :- New medical center request accept script

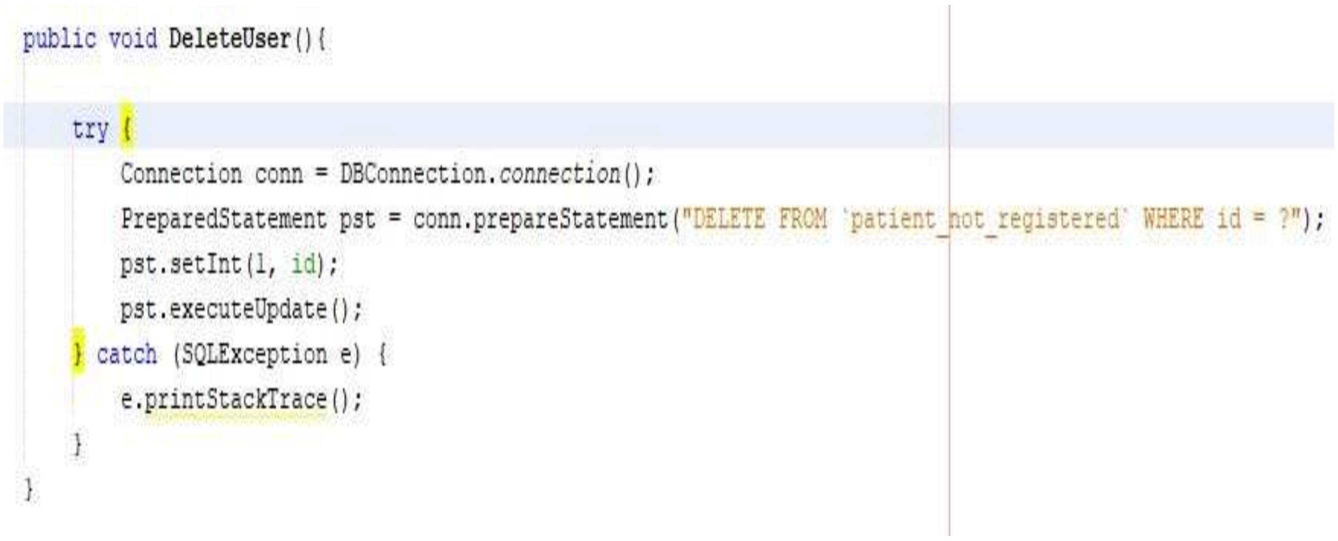
- E-Medihelp user registration request accepting script



```
public void Accept() {
    try {
        Connection conn = DBConnection.connection();
        PreparedStatement pst = conn.prepareStatement("SELECT * FROM `patient_not_registered` WHERE id = ?");
        pst.setInt(1, id);
        ResultSet rs = pst.executeQuery();
        while (rs.next()) {
          id = rs.getInt("id");
          firstName = rs.getString("first_name");
          surname = rs.getString("surname");
          mobileOrEmail = rs.getString("mobile_or_email");
          password = rs.getString("password");
          gender = rs.getString("gender");
        }
        String query = "INSERT INTO `patient_registered`(`id`, `first_name`, `surname`, `mobile_or_email`, `pa
        st= conn.createStatement();
        st.executeUpdate(query);
        pst = conn.prepareStatement("DELETE FROM `patient_not_registered` WHERE id = ?");
        pst.setInt(1, id);
        pst.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

Figure C.4 :- E-Medihelp user registration request accept script

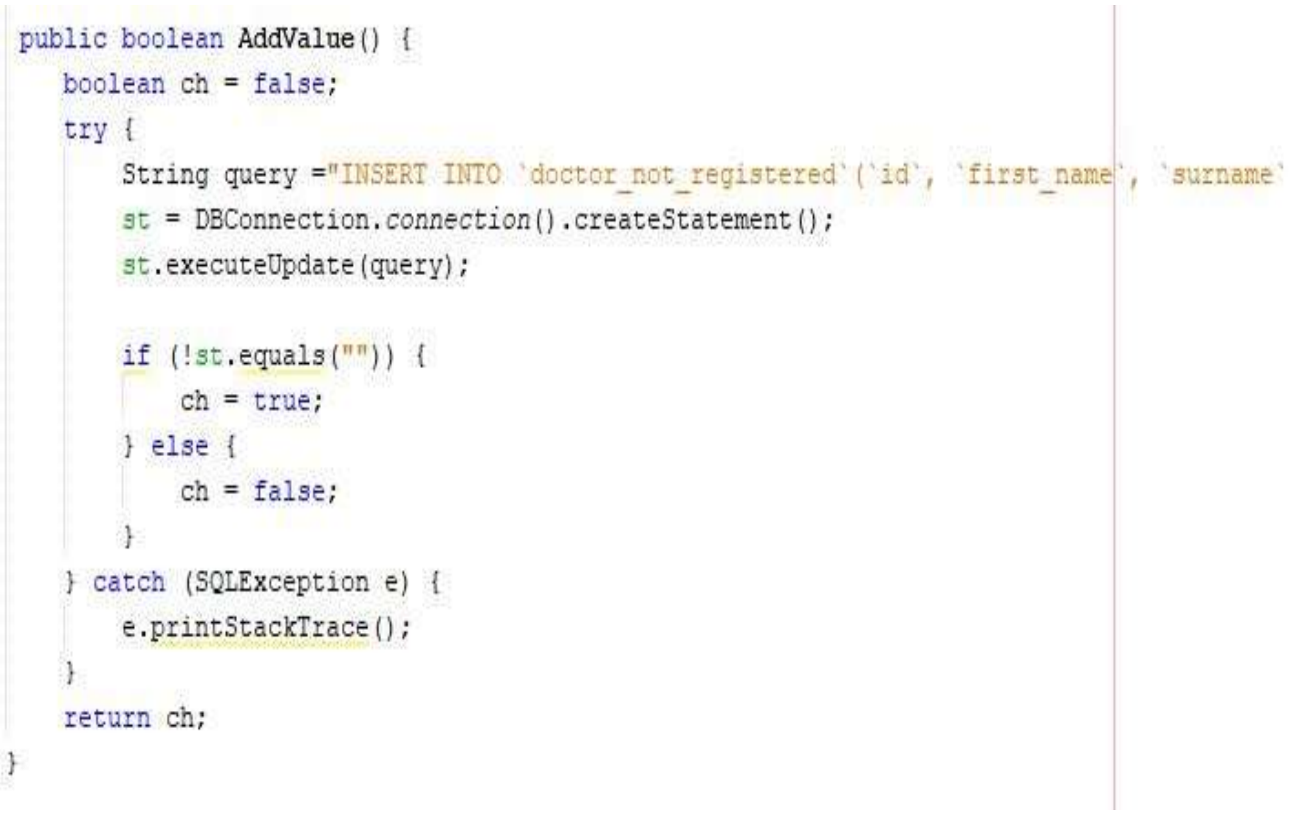
- E-Medihelp user registration request delete script



```
public void DeleteUser(){

    try {
        Connection conn = DBConnection.connection();
        PreparedStatement pst = conn.prepareStatement("DELETE FROM `patient_not_registered` WHERE id = ?");
        pst.setInt(1, id);
        pst.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

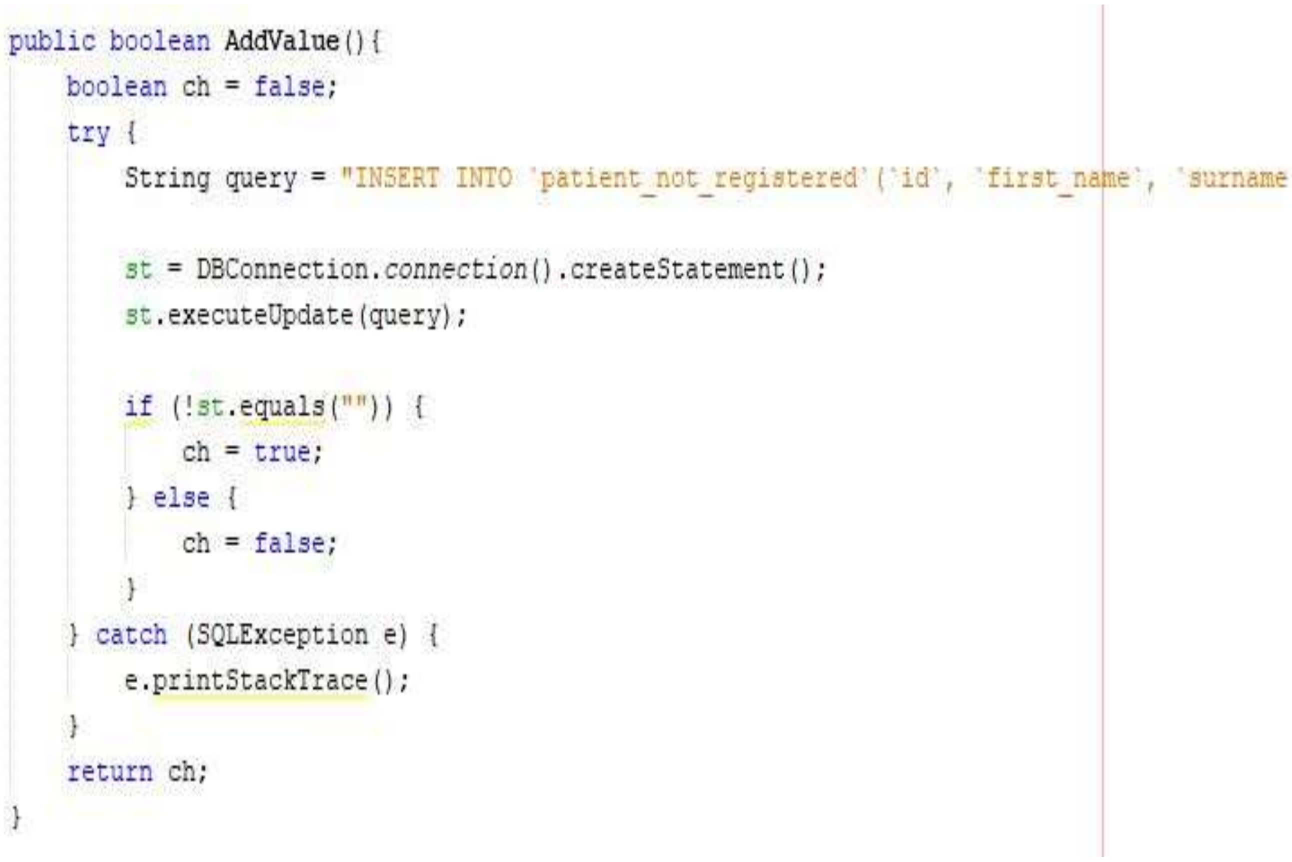
Figure C.5 :- E-Medihelp user registration request delete script

- Signup script



```
public boolean AddValue() {
    boolean ch = false;
    try {
        String query ="INSERT INTO `doctor_not_registered`(`id`, `first_name`, `surname`
        st = DBConnection.connection().createStatement();
        st.executeUpdate(query);

        if (!st.equals("")) {
            ch = true;
        } else {
            ch = false;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return ch;
}
```

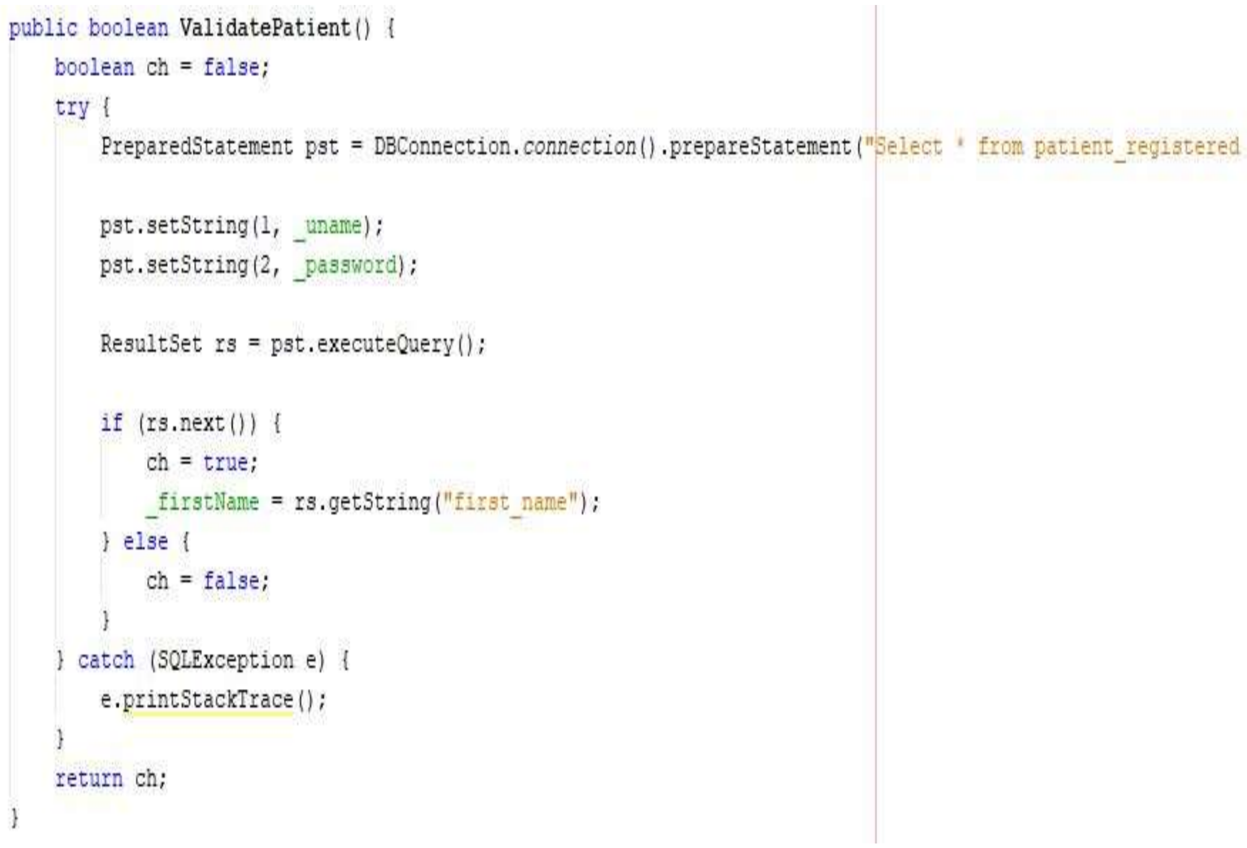
Figure C.6 :- Signup script one



```
public boolean AddValue(){
    boolean ch = false;
    try {
        String query = "INSERT INTO `patient_not_registered`(`id`, `first_name`, `surname

        st = DBConnection.connection().createStatement();
        st.executeUpdate(query);

        if (!st.equals("")) {
            ch = true;
        } else {
            ch = false;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return ch;
}
```

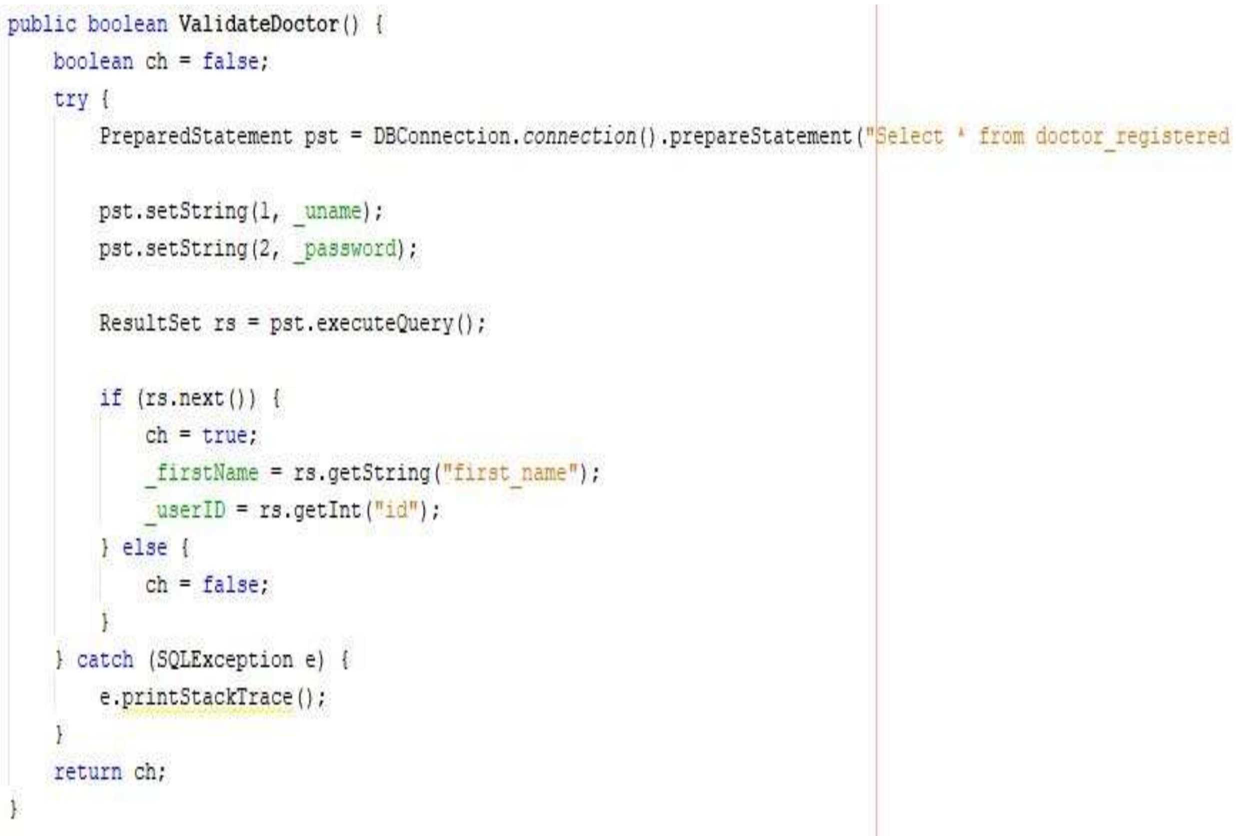
Figure C.7 :- Signup script two

- Login script



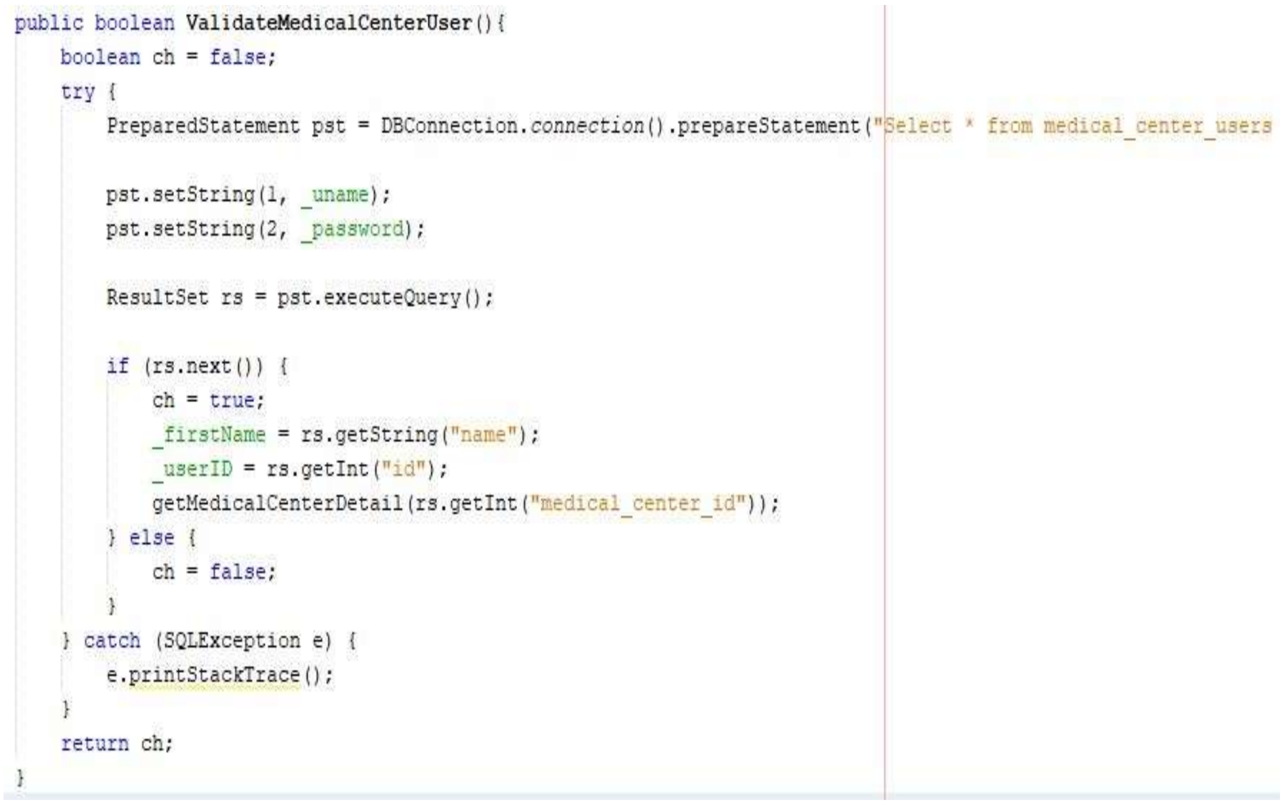
```
public boolean ValidatePatient() {
    boolean ch = false;
    try {
        PreparedStatement pst = DBConnection.connection().prepareStatement("Select * from patient_registered

        pst.setString(1, _uname);
        pst.setString(2, _password);

        ResultSet rs = pst.executeQuery();

        if (rs.next()) {
            ch = true;
            _firstName = rs.getString("first_name");
        } else {
            ch = false;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return ch;
}
```

Figure C.8 :- Login script one



```
public boolean ValidateDoctor() {
    boolean ch = false;
    try {
        PreparedStatement pst = DBConnection.connection().prepareStatement("Select * from doctor_registered

        pst.setString(1, _uname);
        pst.setString(2, _password);

        ResultSet rs = pst.executeQuery();

        if (rs.next()) {
            ch = true;
            _firstName = rs.getString("first_name");
            _userID = rs.getInt("id");
        } else {
            ch = false;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return ch;
}
```

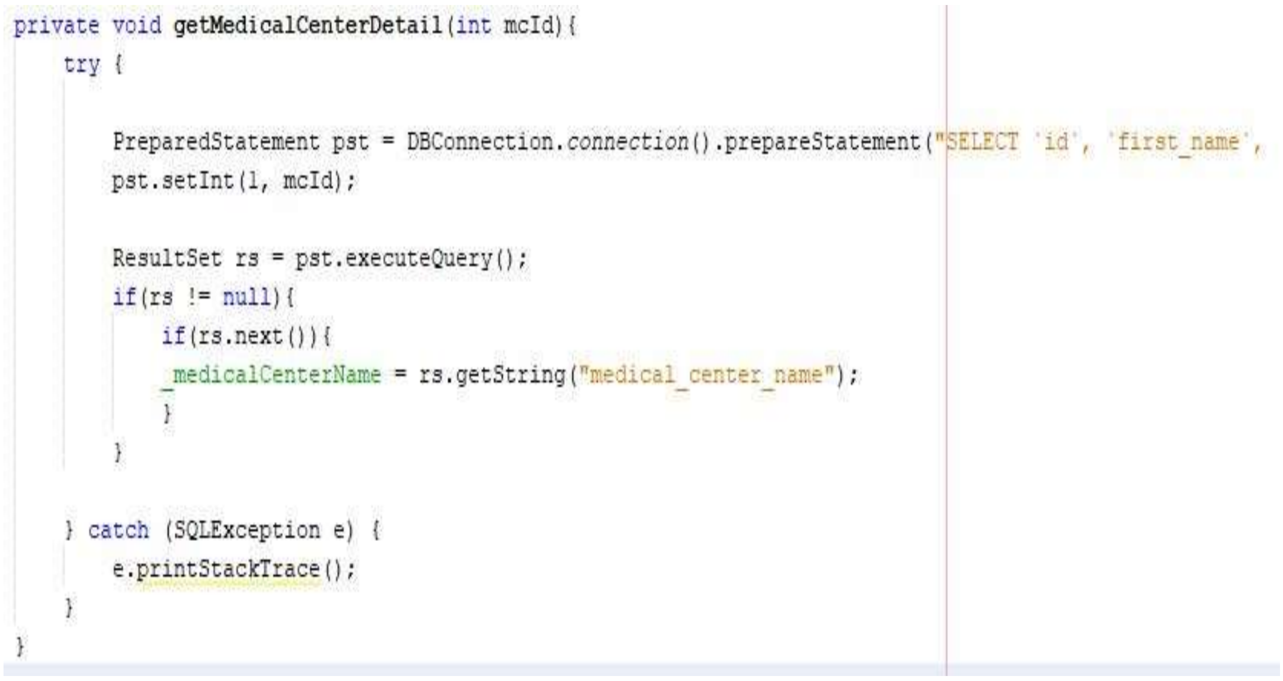
Figure C.9 :- Login script two



```
public boolean ValidateMedicalCenterUser(){
    boolean ch = false;
    try {
        PreparedStatement pst = DBConnection.connection().prepareStatement("Select * from medical_center_users

        pst.setString(1, _uname);
        pst.setString(2, _password);

        ResultSet rs = pst.executeQuery();

        if (rs.next()) {
            ch = true;
            _firstName = rs.getString("name");
            _userID = rs.getInt("id");
            getMedicalCenterDetail(rs.getInt("medical_center_id"));
        } else {
            ch = false;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return ch;
}
```

Figure C.10 :- Login script three

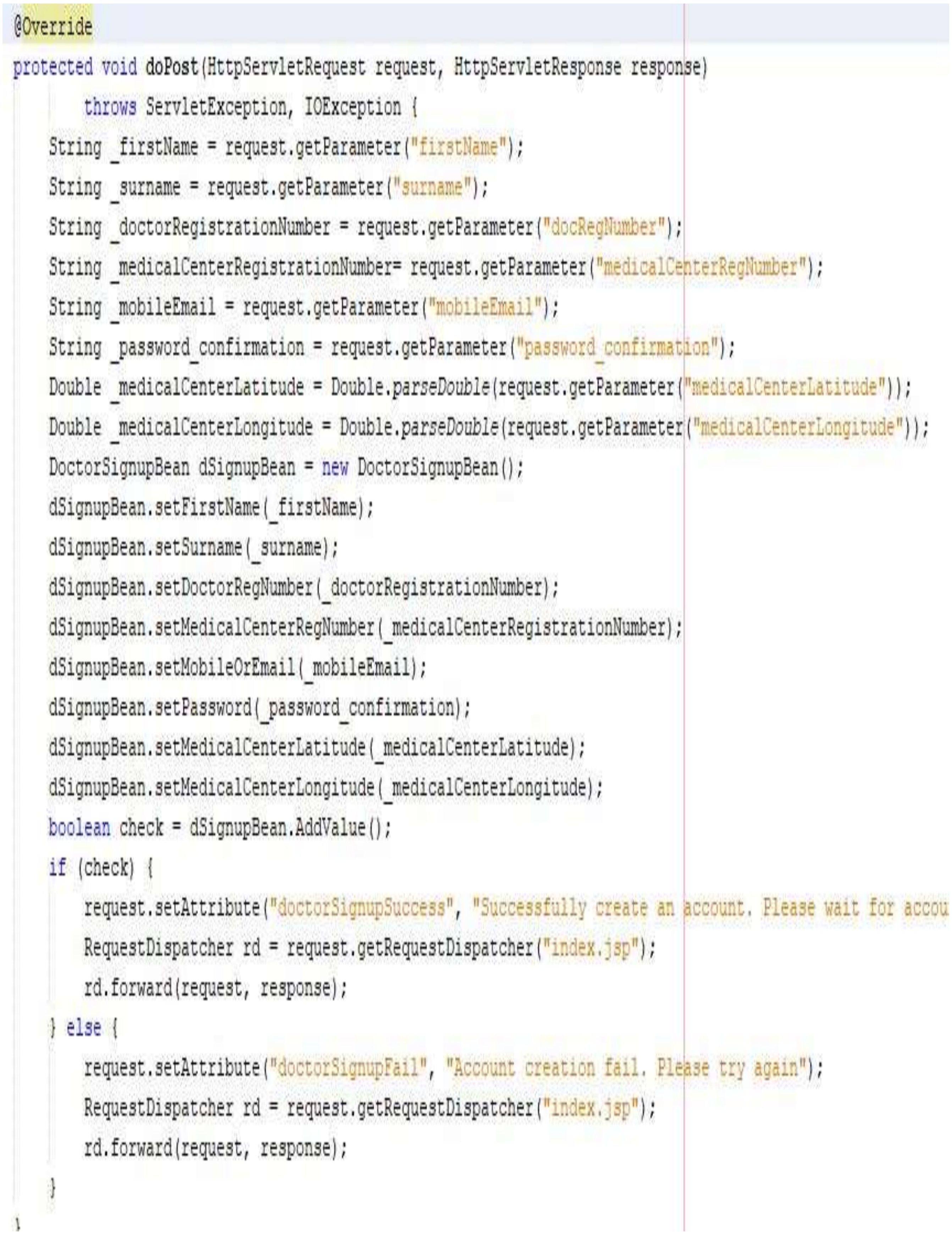
- Medical center information reading script



```
private void getMedicalCenterDetail(int mcId){
    try {

        PreparedStatement pst = DBConnection.connection().prepareStatement("SELECT `id`, `first_name`,
        pst.setInt(1, mcId);

        ResultSet rs = pst.executeQuery();
        if(rs != null){
            if(rs.next()){
            _medicalCenterName = rs.getString("medical_center_name");
            }
        }

    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

Figure C.11 :- Medical center information reading script

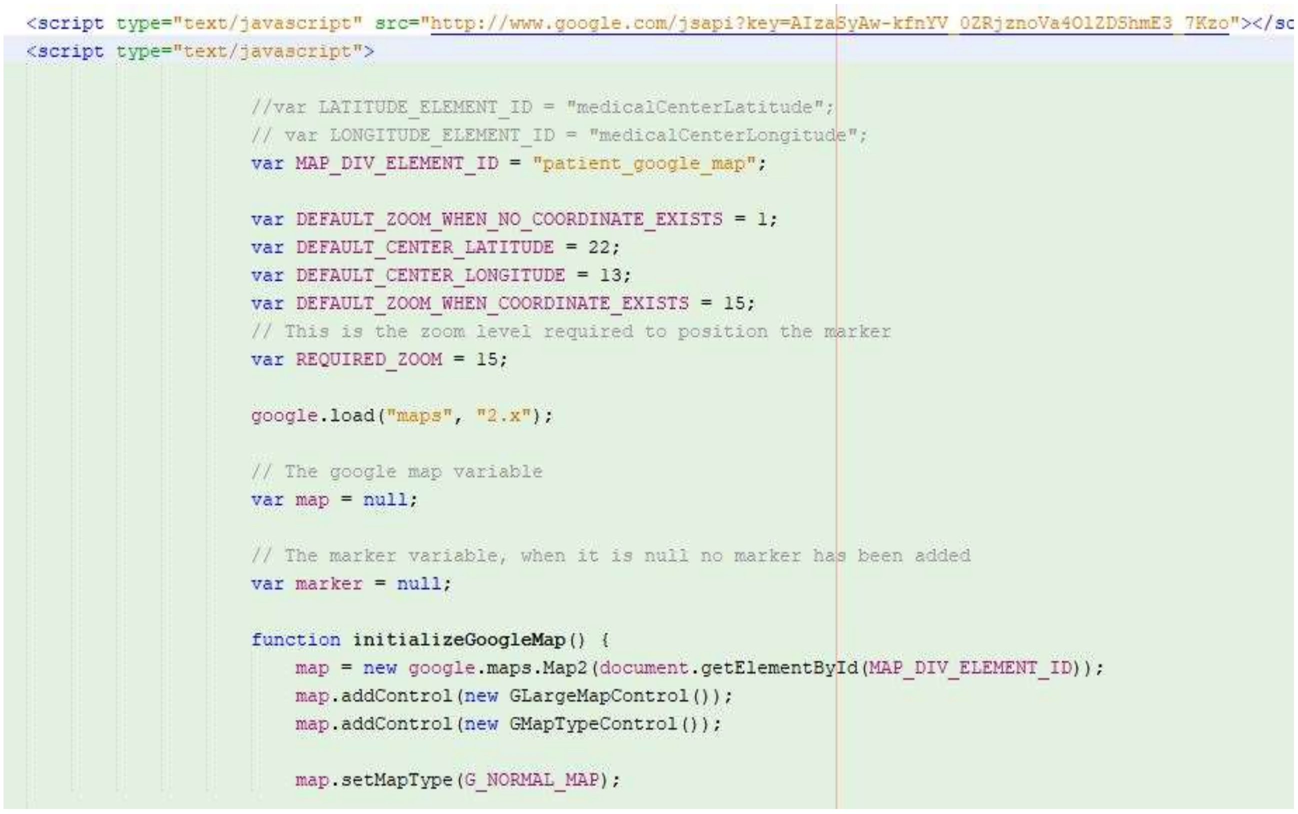
- User signup form request handling script



```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    String _firstName = request.getParameter("firstName");
    String _surname = request.getParameter("surname");
    String _doctorRegistrationNumber = request.getParameter("docRegNumber");
    String _medicalCenterRegistrationNumber= request.getParameter("medicalCenterRegNumber");
    String _mobileEmail = request.getParameter("mobileEmail");
    String _password_confirmation = request.getParameter("password_confirmation");
    Double _medicalCenterLatitude = Double.parseDouble(request.getParameter("medicalCenterLatitude"));
    Double _medicalCenterLongitude = Double.parseDouble(request.getParameter("medicalCenterLongitude"));
    DoctorSignupBean dSignupBean = new DoctorSignupBean();
    dSignupBean.setFirstName(_firstName);
    dSignupBean.setSurname(_surname);
    dSignupBean.setDoctorRegNumber(_doctorRegistrationNumber);
    dSignupBean.setMedicalCenterRegNumber(_medicalCenterRegistrationNumber);
    dSignupBean.setMobileOrEmail(_mobileEmail);
    dSignupBean.setPassword(_password_confirmation);
    dSignupBean.setMedicalCenterLatitude(_medicalCenterLatitude);
    dSignupBean.setMedicalCenterLongitude(_medicalCenterLongitude);
    boolean check = dSignupBean.AddValue();
    if (check) {
        request.setAttribute("doctorSignupSuccess", "Successfully create an account. Please wait for accou
        RequestDispatcher rd = request.getRequestDispatcher("index.jsp");
        rd.forward(request, response);
    } else {
        request.setAttribute("doctorSignupFail", "Account creation fail. Please try again");
        RequestDispatcher rd = request.getRequestDispatcher("index.jsp");
        rd.forward(request, response);
    }
}
```

Figure C.12 :- User signup form request handling script

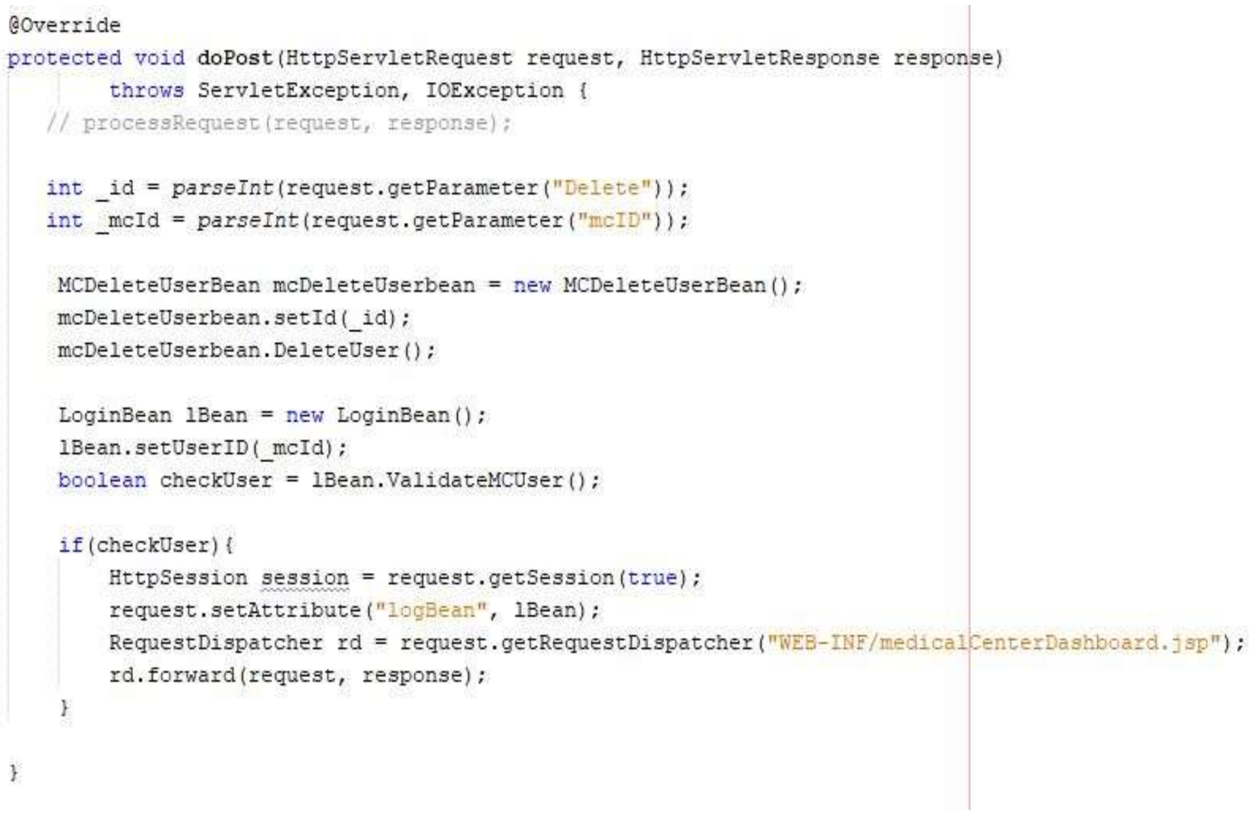
- User location displaying google map script



```
<script type="text/javascript" src="http://www.google.com/jsapi?key=AIzaSyAw-kfnYV_0ZRjznoVa4O1ZD5hmE3_7Kzo"></sc
<script type="text/javascript">

                //var LATITUDE_ELEMENT_ID = "medicalCenterLatitude";
                // var LONGITUDE_ELEMENT_ID = "medicalCenterLongitude";
                var MAP_DIV_ELEMENT_ID = "patient_google_map";

                var DEFAULT_ZOOM_WHEN_NO_COORDINATE_EXISTS = 1;
                var DEFAULT_CENTER_LATITUDE = 22;
                var DEFAULT_CENTER_LONGITUDE = 13;
                var DEFAULT_ZOOM_WHEN_COORDINATE_EXISTS = 15;
                // This is the zoom level required to position the marker
                var REQUIRED_ZOOM = 15;

                google.load("maps", "2.x");

                // The google map variable
                var map = null;

                // The marker variable, when it is null no marker has been added
                var marker = null;

                function initializeGoogleMap() {
                    map = new google.maps.Map2(document.getElementById(MAP_DIV_ELEMENT_ID));
                    map.addControl(new GLargeMapControl());
                    map.addControl(new GMapTypeControl());

                    map.setMapType(G_NORMAL_MAP);
```

Figure C.13 :- User location display google map script

- Users delete request handling script



```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
   // processRequest(request, response);

   int _id = parseInt(request.getParameter("Delete"));
   int _mcId = parseInt(request.getParameter("mcID"));

    MCDeleteUserBean mcDeleteUserbean = new MCDeleteUserBean();
    mcDeleteUserbean.setId(_id);
    mcDeleteUserbean.DeleteUser();

    LoginBean lBean = new LoginBean();
    lBean.setUserID(_mcId);
    boolean checkUser = lBean.ValidateMCUser();

    if(checkUser){
        HttpSession session = request.getSession(true);
        request.setAttribute("logBean", lBean);
        RequestDispatcher rd = request.getRequestDispatcher("WEB-INF/medicalCenterDashboard.jsp");
        rd.forward(request, response);
    }

}
```

Figure C.14 :- Users delete request handling script

Appendix D – User manual

- E-Medihelp home page looks like this. It consists everything needed for any kind of user’s initial usage.

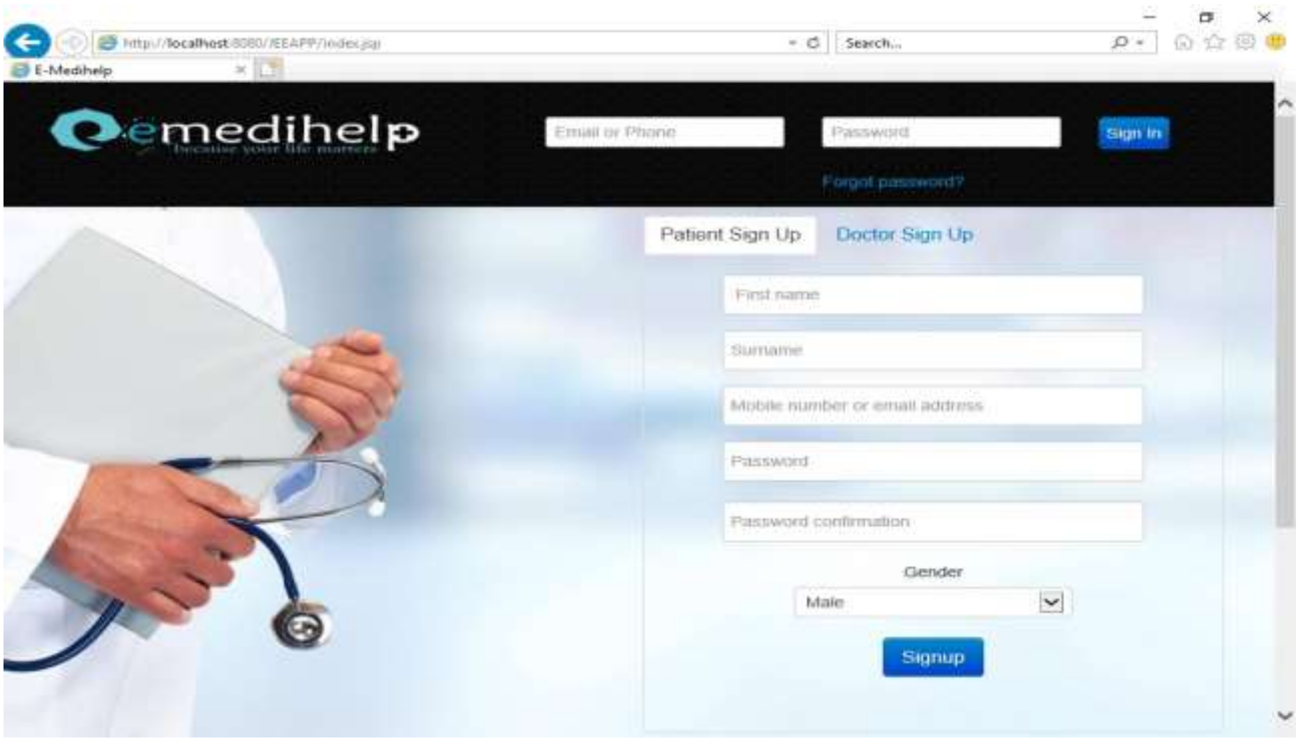


Figure D.1 :- E-Medihelp home page

- Patient signup form display in home page. The patient needs to fill their basic detail for creates an account.

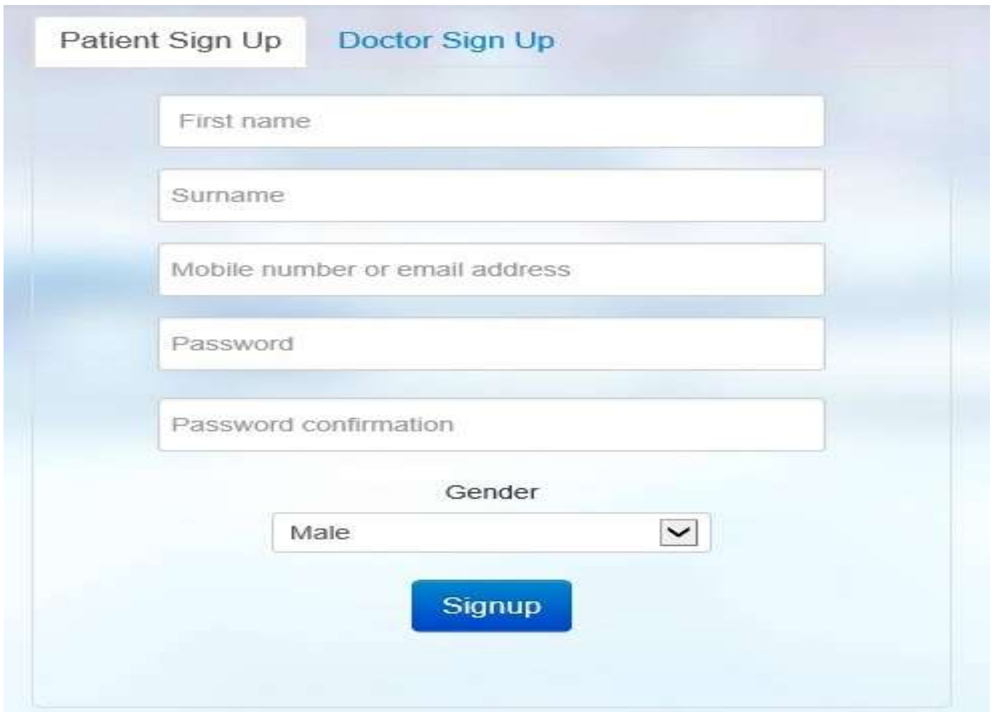
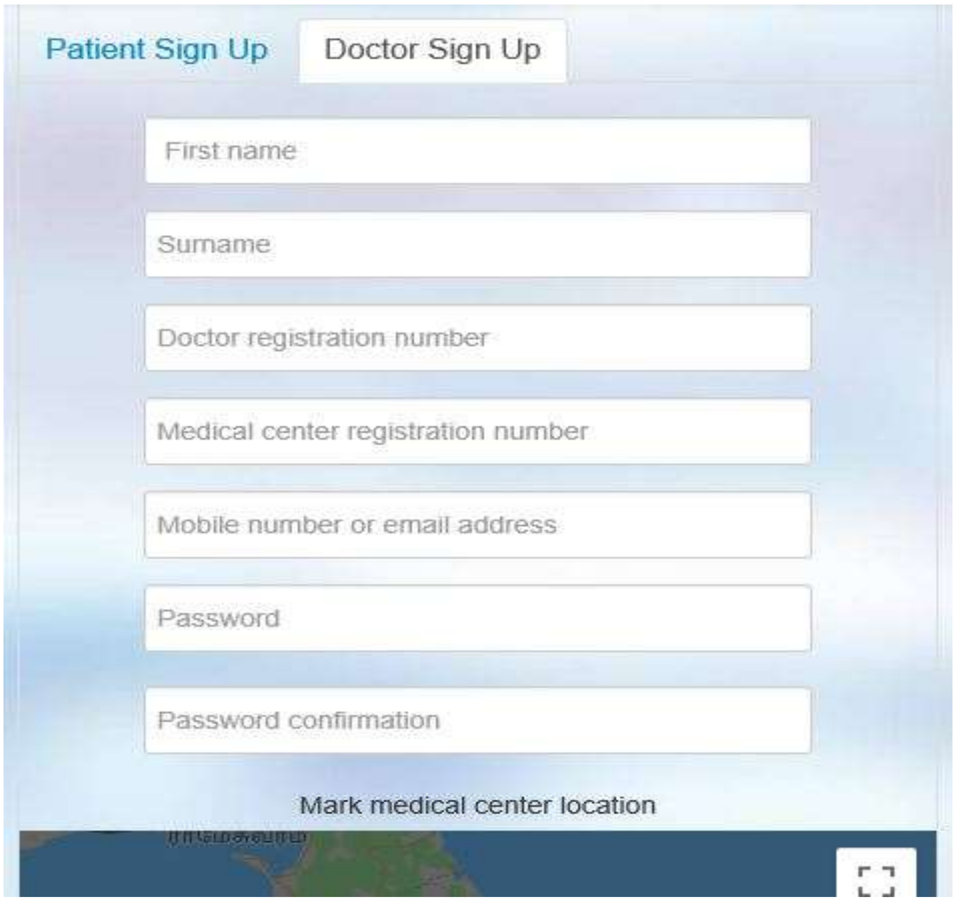


Figure D.2 :- Patient signup form

❖ Signup form second tab includes doctor signup form. Its use to create medical center account.



Patient Sign Up | Doctor Sign Up

First name

Surname

Doctor registration number

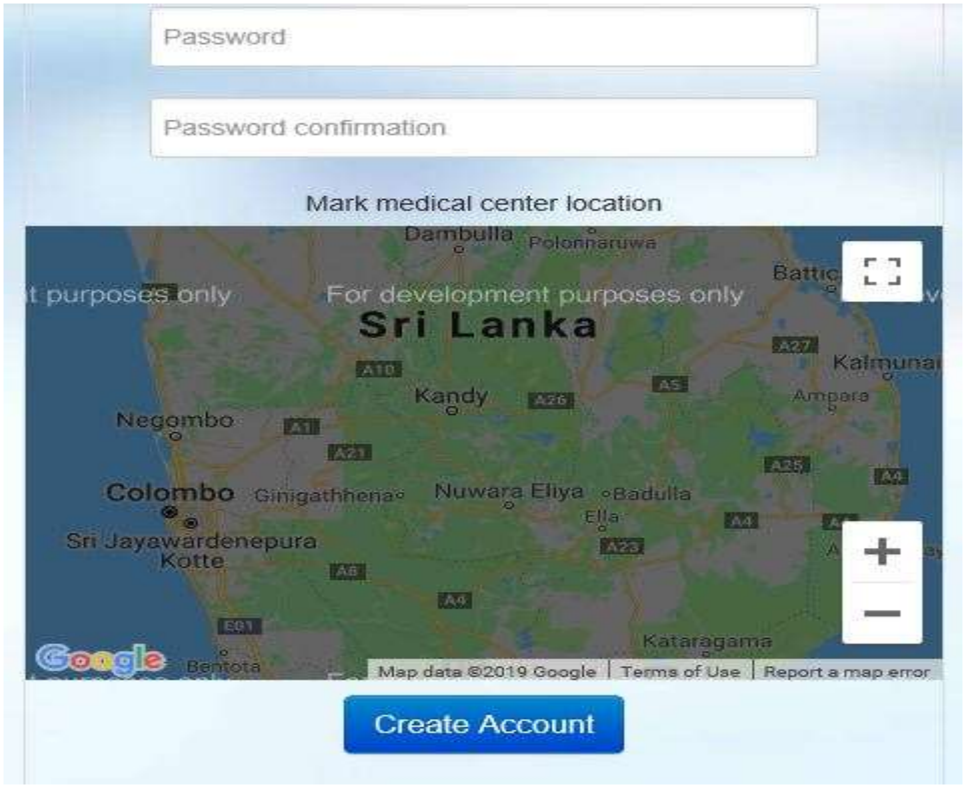
Medical center registration number

Mobile number or email address

Password

Password confirmation

Mark medical center location



Password

Password confirmation

Mark medical center location

Create Account

Figure D.3 :- Doctor signup form

❖ Use common login form for all kinds of users of the system.

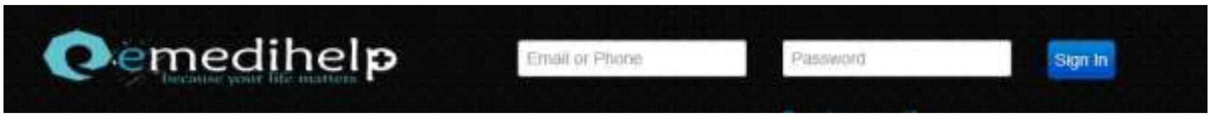


Figure D.4 :- User login form

❖ If the user's password forgot he can recover it using this link.

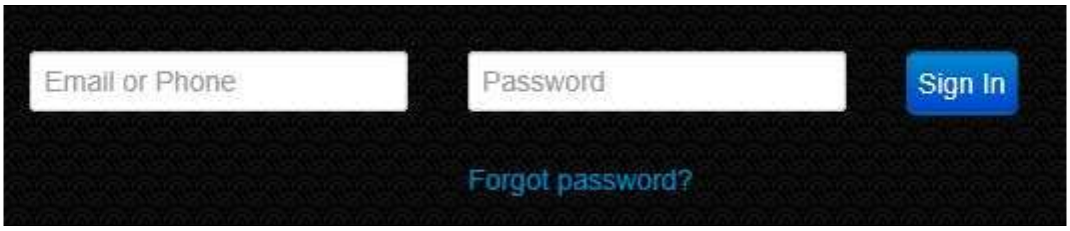
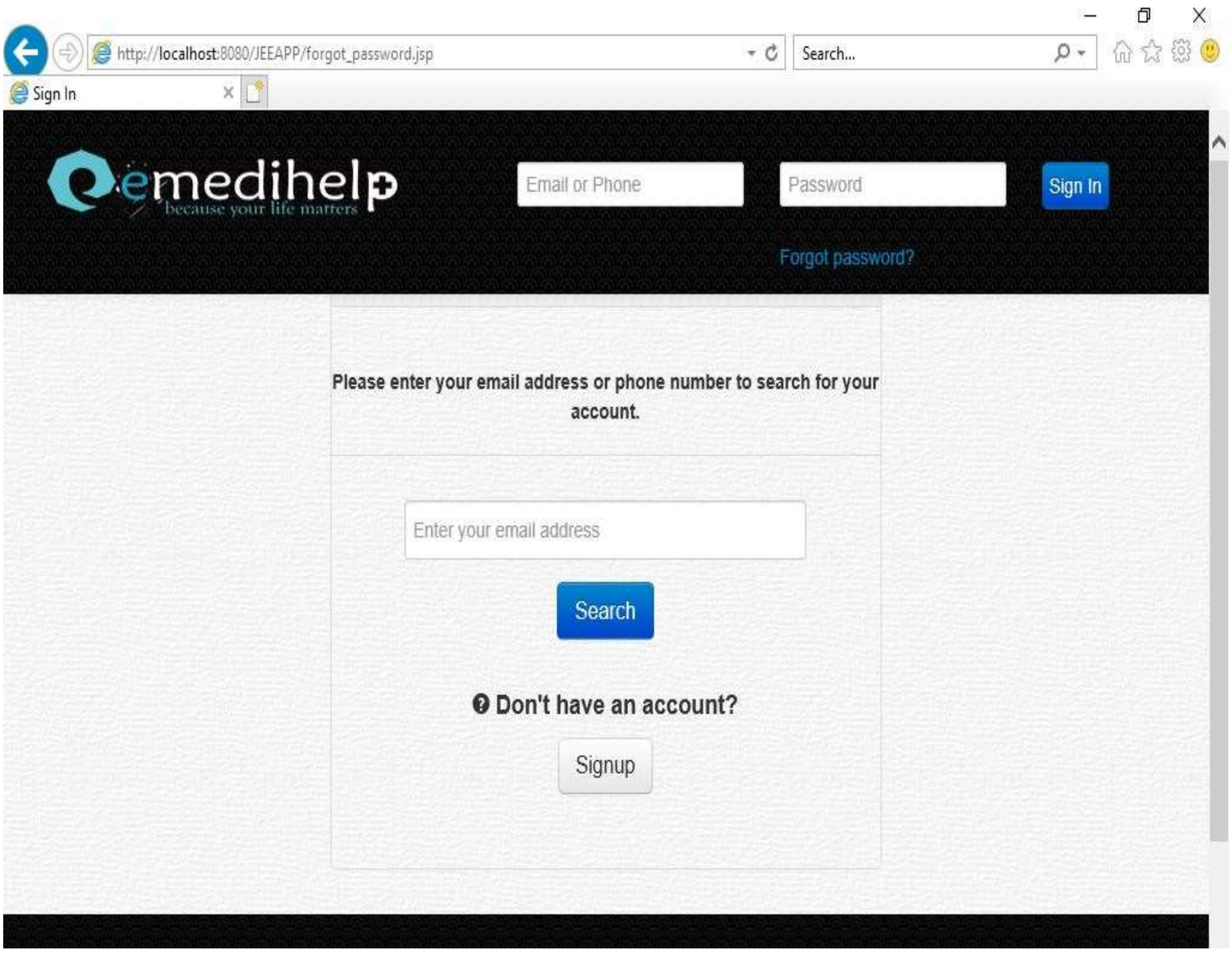


Figure D.5 :- Forgot password view

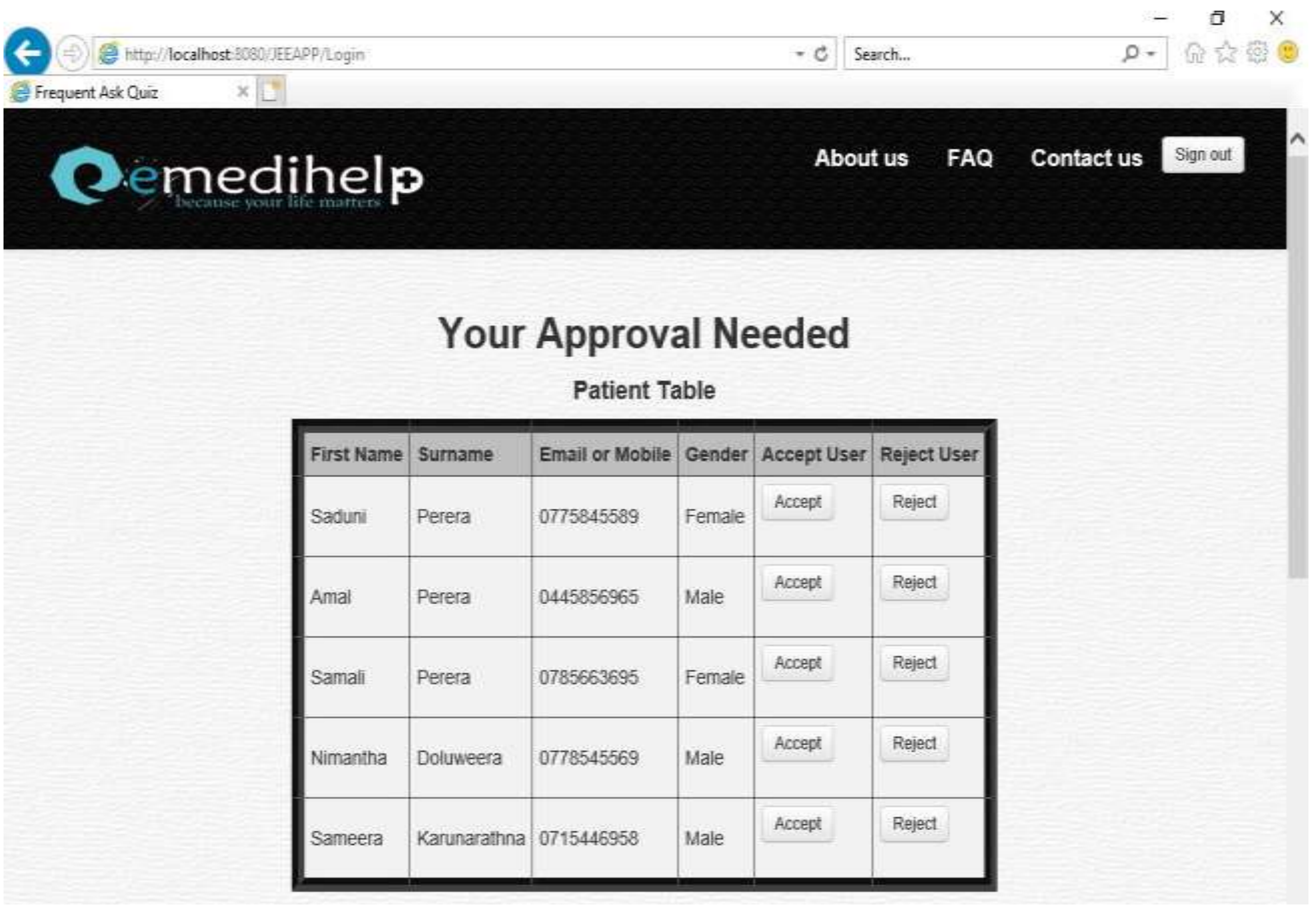
❖ Forgot password page users just need to enter his email. If an email match with the system database details user will receive new password options.


http://localhost:8080/JEEAPP/forgot_password.jsp

Please enter your email address or phone number to search for your account.

Figure D.6 :- Forgot password page

❖ System administrator able to accept or reject any kind of user registration request using admin panel. It includes separate tables for each user type with the accept and reject action buttons.

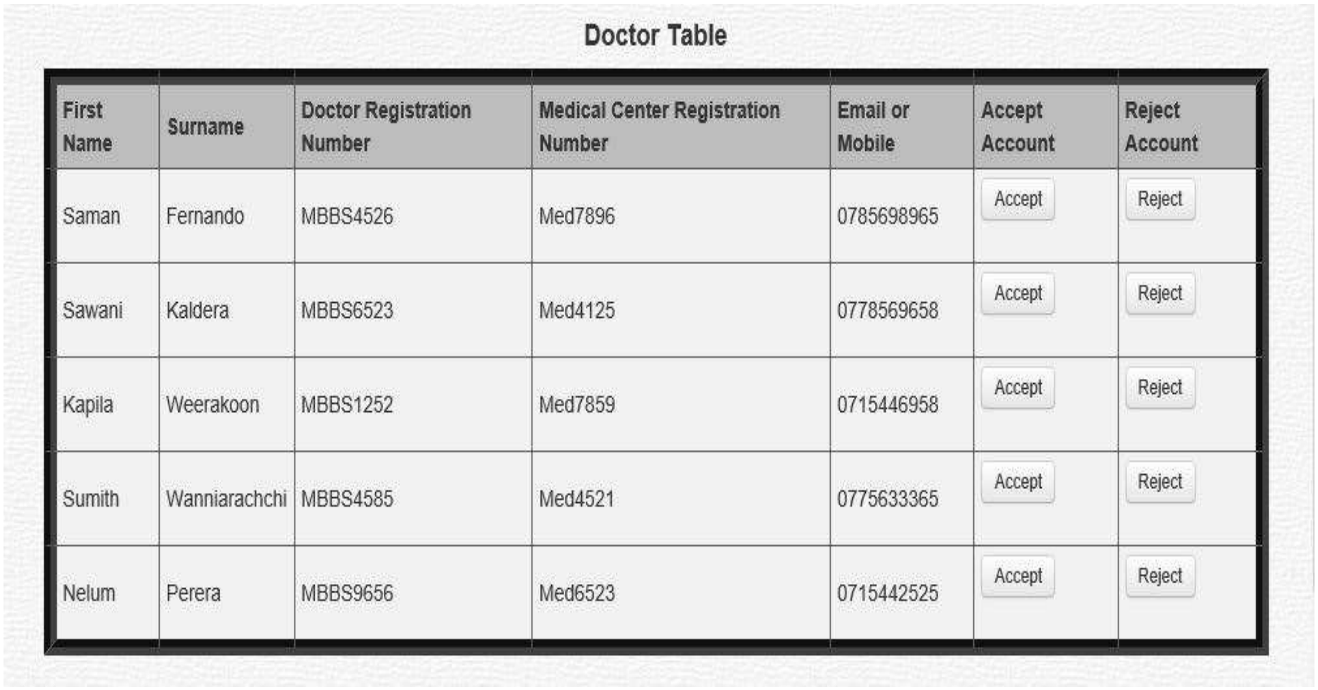


Your Approval Needed

Patient Table

First Name	Surname	Email or Mobile	Gender	Accept User	Reject User
Saduni	Perera	0775845589	Female	Accept	Reject
Amal	Perera	0445856965	Male	Accept	Reject
Samali	Perera	0785663695	Female	Accept	Reject
Nimantha	Doluweera	0778545569	Male	Accept	Reject
Sameera	Karunarathna	0715446958	Male	Accept	Reject

Figure D.7 :- Admin page view one

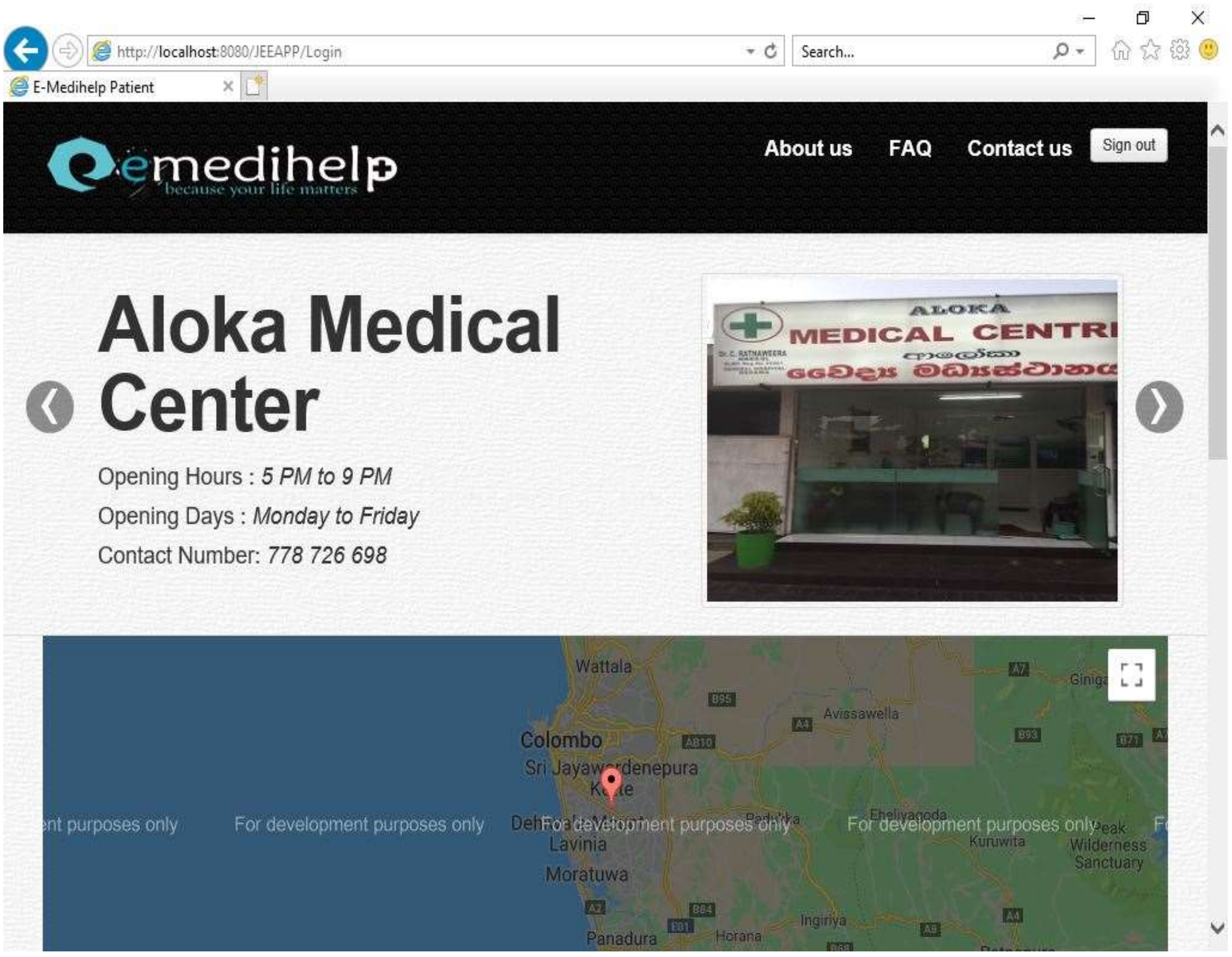


Doctor Table

First Name	Surname	Doctor Registration Number	Medical Center Registration Number	Email or Mobile	Accept Account	Reject Account
Saman	Fernando	MBBS4526	Med7896	0785698965	Accept	Reject
Sawani	Kaldera	MBBS6523	Med4125	0778569658	Accept	Reject
Kapila	Weerakoon	MBBS1252	Med7859	0715446958	Accept	Reject
Sumith	Wanniarachchi	MBBS4585	Med4521	0775633365	Accept	Reject
Nelum	Perera	MBBS9656	Med6523	0715442525	Accept	Reject

Figure D.8 :- Admin page view two

❖ The patient landing page looks like this. When login to the system he can see registered medical centers.



Aloka Medical Center

Opening Hours : *5 PM to 9 PM*

Opening Days : *Monday to Friday*

Contact Number: *778 726 698*

Figure D.9 :- Patient landing page

❖ Patient can find nearby medical centers using his map.

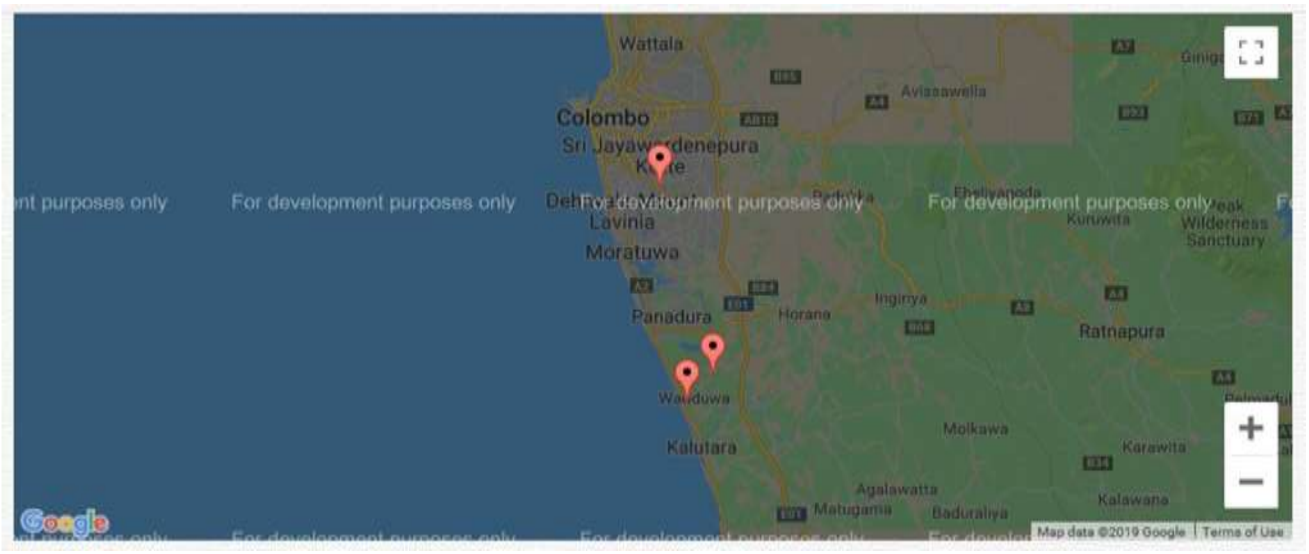
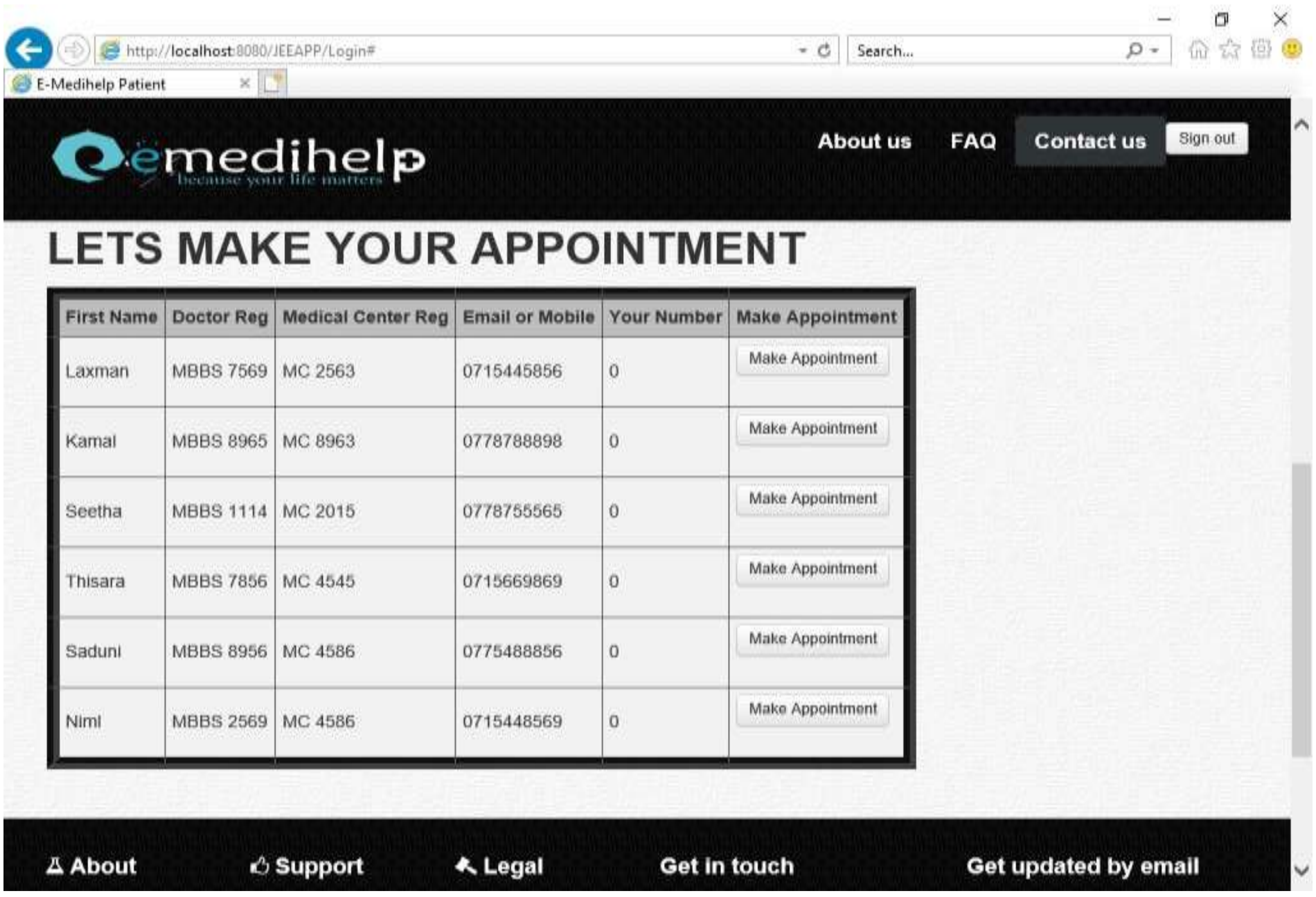


Figure D.10 :- Nearby medical centers map

❖ Patient able to make an appointment for selected medical center using his nearby medical centers list. It populates a current appointment number and relevant details to take a decision.

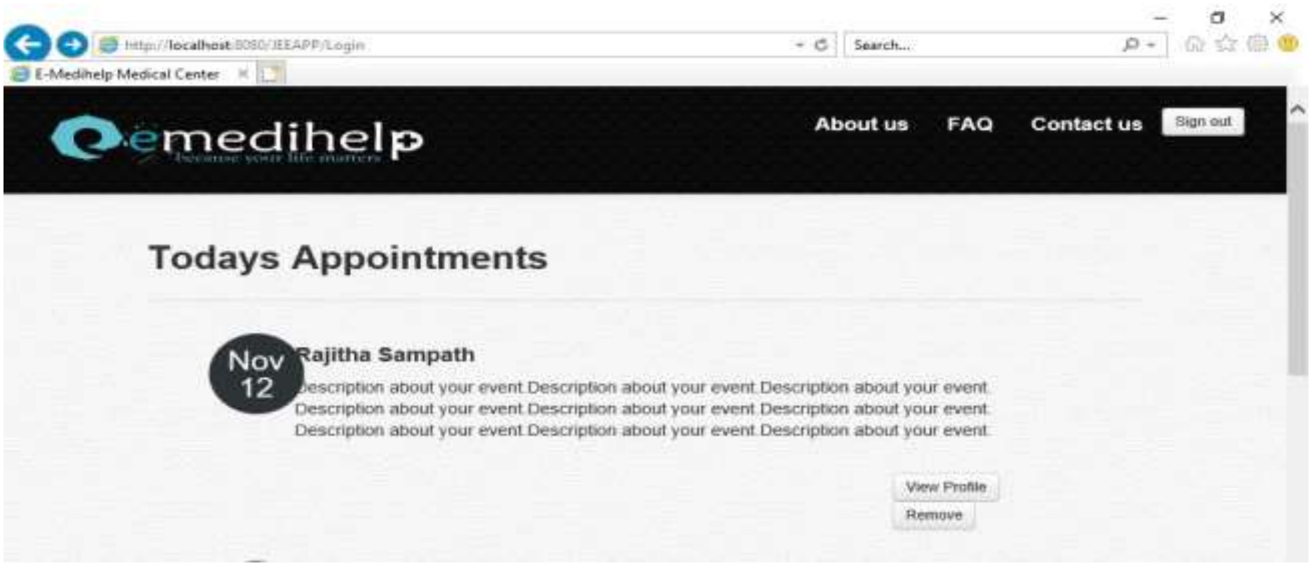


LETS MAKE YOUR APPOINTMENT

First Name	Doctor Reg	Medical Center Reg	Email or Mobile	Your Number	Make Appointment
Laxman	MBBS 7569	MC 2563	0715445856	0	Make Appointment
Kamal	MBBS 8965	MC 8963	0778788898	0	Make Appointment
Seetha	MBBS 1114	MC 2015	0778755565	0	Make Appointment
Thisara	MBBS 7856	MC 4545	0715669869	0	Make Appointment
Saduni	MBBS 8956	MC 4586	0775488856	0	Make Appointment
Niml	MBBS 2569	MC 4586	0715448569	0	Make Appointment

Figure D.11 :- Nearby medical centers list

❖ Doctors able to see his todays appointment list by login to medical center account. It leads to view particular patent detail page. Doctor can accept or reject patient appointment.



Todays Appointments

Rajitha Sampath

Description about your event Description about your event Description about your event Description about your event Description about your event Description about your event Description about your event Description about your event Description about your event

Figure D.12 :- Medical center appointment list