CHAPTER – 1

INTRODUCTION

INTRODUCTION

The Mediface Administration System is a comprehensive software application designed to streamline and automate various administrative and operational tasks within a healthcare facility. This system aims to enhance the efficiency, accuracy, and overall effectiveness of managing hospital operations, including patient registration, appointment scheduling, billing and payments, electronic health records, pharmacy management, and inventory control. In today's fast-paced healthcare environment, hospitals face numerous challenges in managing the complex workflow and vast amount of information associated with patient care. Manual processes and paper-based systems often result in inefficiencies, errors, and delays. The Mediface Administration System addresses these challenges by providing a centralized, integrated platform that simplifies and automates key tasks, enabling healthcare professionals to focus more on delivering quality patient care.

The Mediface Administration System project seeks to address these challenges by implementing a robust and user-friendly system that automates and streamlines key hospital operations. The primary goal is to improve the overall efficiency, accuracy, and effectiveness of managing hospital functions, resulting in better patient care, optimized resource utilization, and improved administrative processes.

The project encompasses a range of functionalities, including patient registration and information management. It allows for seamless and accurate recording of patient data, such as personal details, medical history, and contact information. This information is securely stored and can be easily accessed, updated, and shared among authorized healthcare professionals, enhancing communication and coordination in patient care.

Appointment scheduling is another crucial aspect of the Mediface Administration System . The system provides an intuitive interface for patients to schedule appointments with doctors based on availability, ensuring efficient time management and reducing scheduling conflicts. Automated reminders and notifications are incorporated to minimize no-shows and improve patient attendance.

The Mediface Administration System project is developed using a combination of HTML, Java, and MYSQL technologies. This ensures a robust, scalable, and secure system that can handle the complexities of hospital operations. The system provides a user-friendly interface

for easy navigation and access to various functionalities, promoting efficient utilization by healthcare professionals and staff.

In conclusion, the Mediface Administration System project aims to revolutionize hospital operations by automating key tasks, streamlining processes, and improving overall efficiency and patient care. By implementing this comprehensive system, hospitals can enhance their administrative capabilities, optimize resource allocation, and ultimately provide better healthcare services to patients.

Objectives:

The objectives of the Mediface Administration System project are to:

1. **Improve Patient Registration and Information Management:** The system allows for easy and accurate patient registration, maintaining essential demographic details, medical history, and contact information. This information can be securely accessed and updated by authorized personnel, ensuring quick retrieval and facilitating better communication.

2. **Streamline Appointment Scheduling:** The system provides a user-friendly interface for scheduling appointments, allowing patients to book appointments with doctors based on availability. It enables efficient management of appointment slots, reduces scheduling conflicts, and sends automated reminders to patients to minimize no-shows.

3. **Efficient Electronic Health Records (EHR) Management:** The system incorporates electronic health records, securely storing and managing patients' medical records. It allows authorized healthcare professionals to access patient information, including diagnoses, treatment plans, medications, and test results. This promotes efficient and coordinated care delivery while maintaining data privacy and security.

4. Provide a centralized platform to manage patient records, appointments, and medical history.

5. Facilitate seamless communication between doctors and patients.

6. Enhance the efficiency of medical procedures.

7. Ensure data security, confidentiality, and integrity of patient information

The Mediface Administration System is designed to meet the specific needs of healthcare facilities, improve operational efficiency, enhance patient care, and ensure data integrity and security. By leveraging the power of technology, this system empowers healthcare professionals to focus on their core responsibilities while providing better and more accessible healthcare services to patients.

Problem Statement:

In this busy world we don't have the time to wait in infamously long hospital queues. The problem is, queuing at hospital is often managed manually by administrative staff, then take a token there and then wait for our turn then ask for the doctor and the most frustrating thing - I went there by traveling a long distance and then I come to knowthe doctor is on leave or the doctor can't take appointments.

MAS will help us overcome all these problems because now patients can book their appointments at home, they can check whether the doctor they want to meet is available or not. Doctors can also confirm or decline appointments, this help bothpatient and the doctor because if the doctor declines' appointment then patient will know this in advance and patient will visit hospital only when the doctor confirms' the appointment this will save time and money of the patient.

MAS is essential for all healthcare establishments, be it hospitals, nursing homes, health clinics, rehabilitation centers, dispensaries, or clinics. The main goal is to computerize allthe details regarding the patient and the hospital. The installation of this healthcare software results in improvement in administrative functions and hence better patient care, which is the prime focus of any healthcare unit.

Benefits of implementing a Mediface Administration System:

1. **Appointment booking**
    a. Helps patients cut the long queue and saves their time
    b. Is equipped with features like automated email and text messagereminders
2. **Role-Based Access Control**
    a. Allows employees to access only the necessary information to effectively

perform their job duties

b. Increases data security and integrity

3. **Overall cost reduction**
    a. Cuts down paper costs as all the data are computerized
    b. No separate costs for setting up physical servers

4. **Data accuracy**
    a. Removes human errors

5. **Data security**
    a. Helps to keep patients records private
    b. Restricts access through role-based access control

6. **Revenue management**
    a. Makes daily auditing simple
    a. Helps with statistics and other financial aspects

CHAPTER – 2

MODULES

ADMIN MODULE:

The admin module in the Mediface Administration System includes functionality to add and manage doctors and their specialties. This allows administrators to maintain an updated and organized list of doctors within the system. Here is an explanation of how the admin module handles the addition of doctors and their specialties:

1. Doctor Management:

- Add Doctor: Administrators can add new doctors to the system by entering their relevant details, such as name, contact information, and qualifications. This information helps in identifying and differentiating doctors within the system.
- Edit Doctor: Administrators have the capability to modify and update doctor information as needed. This includes updating contact details, qualifications, or any other relevant details that may change over time.
- Delete Doctor: In case a doctor leaves the hospital or is no longer associated with the system, administrators can remove their profile and data from the system.

2. Specialist Management:

- Add Specialist: Administrators can assign specialties to doctors based on their area of expertise. This involves selecting from a predefined list of specialties or creating new specialties if needed. The administrator assigns the appropriate specialty to each doctor to ensure accurate categorization and easy search within the system.
- Edit Specialist: Administrators have the flexibility to modify or update the list of specialties as needed. They can add new specialties, modify existing ones, or remove specialties that are no longer relevant.
- Assign Specialists to Doctors: Administrators can associate specialists with doctors based on their respective areas of expertise. This ensures that doctors are appropriately categorized within the system and makes it easier for patients to find doctors based on their specific medical needs.

In the admin module of the Mediface Administration System, administrators have access to important statistics and counts related to users, doctors, specialists, and appointments. These counts provide a comprehensive overview of the system's activities and help administrators monitor the overall performance of the hospital.

Here is an explanation of the total count of users, doctors, specialists, and appointments in the admin module:

3. Total Count:

a. Total Count of Users:

- Administrators can view the total count of registered users within the system. This count represents the number of individuals who have user accounts and other authorized personnel.
- It provides an overview of the user base and the number of individuals who can access and utilize the system's functionalities.

b. Total Count of Doctors:

- Administrators can access the total count of doctors registered in the system. This count represents the number of active doctors associated with the hospital or healthcare facility.
- It includes doctors from various specialties and departments within the organization.
- The count helps administrators track the number of doctors available in the system and manage the distribution of workload and patient care responsibilities.

c. Total Count of Specialists:

- Administrators can view the total count of specialists registered in the system. This count represents the number of doctors who have been assigned specific specialties or areas of expertise.
- It helps administrators identify the distribution of specialists across different medical fields and ensures accurate categorization of doctors within the system.

d. Total Count of Appointments:

- Administrators can access the total count of appointments scheduled within a specific time period, such as daily, weekly, or monthly.
- This count includes both upcoming appointments and past appointments within the specified time frame.
- The count helps administrators evaluate the appointment volume, track the utilization of healthcare services, and monitor the overall efficiency of the appointment scheduling process.

By having access to these total counts, administrators can gather valuable insights about the hospital's operations and user base. These statistics enable administrators to make informed decisions regarding resource allocation, scheduling, and system management. They help administrators monitor the growth of the user base, track the number of active doctors and specialists, and assess the demand for appointments. This information assists in ensuring that the Mediface Administration System operates effectively, meeting the needs of both healthcare providers and patients.

The addition and management of doctors and specialists within the admin module contribute to maintaining an up-to-date and organized database of medical professionals in the Mediface Administration System. It allows for efficient search and retrieval of doctors based on their specialties, simplifying the appointment scheduling process and improving patient care.

DOCTOR MODULE:

The doctor module in the Mediface Administration System provides a range of functionalities to doctors, including the ability to log in and out, view appointments, edit their profile, change their password, and comment on patient records. Here is an explanation of each feature within the doctor module:

1. Login and Logout:

a. Doctors can log in to the system using their unique credentials, such as a username and password.
b. The login functionality verifies the doctor's identity and grants access to the doctor module.
c. When the doctor is done accessing the system, they can log out to ensure the security of their account and patient data.

2. View Appointments:

a. Once logged in, doctors can access their appointment schedule.

- The system displays a list of upcoming appointments, including details such as date, time, patient name, and purpose of the appointment.

b. Doctors can easily navigate through the schedule and view additional appointment information.

3. Edit Profile:

a. The doctor module allows doctors to update their profile information.
b. This includes personal details like name, contact information, qualifications, and areas of expertise.
c. Doctors can make necessary changes to ensure their profile is accurate and up to date.

4. Change Password:

a. To maintain account security, doctors have the option to change their password.
b. The system provides password change functionality, requiring the doctor to enter their current password and then set a new password.
c. This ensures that doctors have control over their account access and can update their passwords periodically.

5. Comment on Patient Records:

a. The doctor module allows doctors to provide comments or notes on patient records.
b. After an appointment, doctors can add relevant comments or observations to the patient's electronic health record (EHR).
c. These comments can include diagnoses, treatment plans, progress notes, or any other relevant information for ongoing patient care.

The doctor module ensures that doctors have a secure and personalized experience within the Mediface Administration System. It provides convenient access to appointments, the ability to update personal information, change passwords, and add comments to patient records. These functionalities contribute to efficient and effective communication, streamline workflows, and enhance the quality of patient care.

USER MODULE:

The user module in the Mediface Administration System provides functionalities for users, such as patients, to register, login, and logout. It also includes features to change passwords, book appointments, view the status of appointments, and access information about doctors and specialists. Here is an explanation of each feature within the user module

1. Register:

a. Users can create an account by registering in the system.
b. The registration process typically involves providing necessary details such as name, contact information, date of birth, and any other required information.
c. Upon successful registration, users are assigned a unique username and password for future login.

2. Login and Logout:

a. Users can log in to the system using their registered username and password.
b. The login functionality verifies the user's credentials and grants access to the user module.
c. After accessing the system, users can perform various tasks. When they are done, they can choose to log out, ensuring the security and privacy of their account.

3. Change Password:

a. Users have the ability to change their password for enhanced security.
b. The system provides password change functionality, requiring users to enter their current password and then set a new password.
c. This enables users to update their passwords periodically and maintain control over their account access.

4. Book an Appointment:

a. The user module allows users to book appointments with doctors based on availability.
b. Users can view the list of available doctors and specialists, along with their schedules and specialties.
c. Users can select a preferred doctor, choose an available time slot, and book an appointment based on their medical needs and preferences.

5. Status of Appointment:

a. After booking an appointment, users can view the status of their appointment.
b. The system provides information such as appointment date, time, doctor's name, and confirmation status.
c. Users can check the status of their appointment to ensure its confirmation or make any necessary changes.

6. View Doctors and Specialists:

a. The user module allows users to access information about doctors and specialists within the hospital.
b. Users can view a list of doctors and their respective specialties, qualifications, and contact information.
c. This feature helps users in making informed decisions while booking appointments or seeking medical advice.

The user module provides a user-friendly interface for users to register, log in, and log out. It facilitates secure access to the system and enables users to perform essential tasks such as changing passwords, booking appointments, checking appointment status, and accessing information about doctors and specialists. These functionalities enhance the user experience, streamline the appointment scheduling process, and promote efficient communication between users and healthcare providers.

APPOINTMENT MODULE:

The appointment module in the Mediface Administration System includes functionalities for registered users to book appointments, view appointment status, and for both users and administrators to access appointment-related information. It also allows doctors to comment on appointments after treatment. Here is an explanation of each feature within the appointment module

1. Book Appointment:

a. Registered users can log in to the system and access the appointment module to book appointments.
b. Users can select a preferred doctor or specialist based on their specialty, availability, and other relevant criteria.
c. After selecting the appointment details, users can confirm and book the appointment.

2. View Appointment Status:

a. Registered users can access the appointment module to view the status of their booked appointments.
b. Users can check the status of their appointments to ensure they are confirmed or make any necessary changes.

3. View Appointment Status by Admin:

a. Administrators, in the admin module, have access to view the appointment status of all appointments within the system.
b. The admin module provides an overview of appointment statuses, including completed, pending appointments.
c. Administrators can track the overall appointment schedule and manage any changes or updates as needed.

4. Comment on Appointments by Doctors:

a. After treating a patient, doctors can access the appointment details and add comments or notes regarding the treatment provided.
b. The comments added by doctors serve as valuable records for future reference and continuity of patient care.

The appointment module facilitates a streamlined process for registered users to book appointments with doctors, view appointment status, and access relevant information. It allows administrators to monitor and manage the overall appointment schedule, ensuring effective appointment management within the system. Additionally, doctors can contribute to appointment records by adding comments and treatment-related information, promoting comprehensive patient care and documentation.

ADVANTAGES:

1. User-Friendly Interface: Our Website provides a clean and intuitive user interface, making it easy for visitors to navigate and find the information they need. This enhances the overall user experience and encourages users to engage with our Website.

2. Comprehensive Information: Our Website offers comprehensive and up-to-date information about our hospital, including services, departments, doctors, specialties, and facilities. Visitors can easily access the information they need to make informed decisions about their healthcare.

3. Online Appointment Booking: Our Website provides a convenient online appointment booking feature. Patients can easily schedule appointments with their preferred doctors, select suitable time slots, and receive confirmation instantly. This saves time and effort for both patients and hospital staff.

4. Secure Patient Portal: we offer a secure patient portal on our Website, allowing registered patients to access their personal health records, test results, and medical history. This empowers patients to take an active role in managing their healthcare and enables seamless communication with healthcare providers.

5. Educational Resources: Our Website includes educational resources such as articles, blogs, and FAQs to educate visitors about various health conditions, preventive care, and wellness tips. This fosters a culture of health awareness and promotes the well-being of our community.

6. Browser-Friendly Design: Our Website is designed to be responsive and browser-friendly, ensuring optimal viewing and functionality across different devices and screen sizes. Visitors can access our Website and book appointments conveniently from their PC or tablets.

9. News and Updates: we provide regular news and updates about our hospital, including new services, medical advancements, and community initiatives. This keeps visitors informed and engaged with our healthcare offerings.

10. Secure Online Communication: Our Website incorporates secure communication channels, such as encrypted forms or chatbots, to ensure privacy and confidentiality when patients reach out to us with inquiries or feedback.

These advantages of our Website contribute to an enhanced online presence, improved patient experience, and efficient healthcare services. Our Website serves as a valuable platform for information dissemination, appointment management, and fostering meaningful patient-provider interactions.

DISADVANTAGES:

1. Technical Issues: Websites can encounter technical problems, such as server downtime, slow loading times, broken links, or compatibility issues with certain browsers or devices. These issues can negatively impact the user experience and discourage visitors from accessing the Website.

2. Dependence on Internet Connectivity: Websites rely on a stable internet connection to function properly. If users or visitors have limited or no internet access, they may not be able to access the Website, hindering their ability to obtain information or use online services.

3. Information Overload or Inaccuracy: Websites with excessive information or poorly organized content can overwhelm users and make it challenging to find the specific information they are seeking. Additionally, outdated or inaccurate information on the Website can mislead users and negatively impact their experience.

It is crucial to address these disadvantages by implementing robust security measures, conducting regular Website maintenance, providing accurate and up-to-date information, and offering alternative communication channels for users who prefer direct interaction. By mitigating these challenges, the Website can provide a more seamless and satisfying user experience.

CHAPTER – 3

SYSTEM REQUIREMENTS

SYSTEM INTERFACES

User Interfaces

a. This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.
b. The protocol used shall be HTTP.
c. The Port number used will be 8080.
d. There shall be logical address of the system in IPv4 format.

Hardware Interfaces

a. Laptop/Desktop PC-Purpose of this is to give information when Patients ask information about doctors, medicine available lab tests etc. To perform such Action it need very efficient computer otherwise due to that reason patients have to wait for a long time to get what they ask for.
b. Laser Printer (B/W) - This device is for printing patients' info etc.
c. Wi-Fi router - Wi-Fi router is used to for internetwork operations inside of a hospital and simply data transmission from pc's to sever.

Software Interfaces

a. JDK 1.8 – JAVA development kit consists of all the libraries and utilities to develop an application
b. MYSQL server - Database connectivity and management
c. OS Windows 7/8/10/11- Very user friendly and common OS
d. JRE 1.8 – JAVA run time environment to run the program.

SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS		
Processor	**RAM**	**Disk space**
1.6 GHz or faster processor	1GB of RAM (1.5 GB if runningon a virtual machine)	5400 RPM hard drive and 5 GB of available hard disk space

Table (1): Hardware Requirement

SOFTWARE REQUIREMENTS		
Operating System	**Database**	**User Interfacedesign**
Windows 7, Windows 8, Windows10, Windows 11	MYSQL work bench 5.1	Eclipse IDE 2022-2023,Visual studio 2023,Chrome/MS edge

Table (2): Software Requirement

TECHNOLOGIES USED:

Front end Technologies:

a. HTML

i. **HTML** or **Hypertext Markup Language** is the standard markup language used to create Web pages.

ii. HTML is written in the form of HTML elements consisting of tags enclosed in angle brackets (like <html>). HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent empty elements and so are unpaired, for example <img>. The first tag in a pair is the start tag, and the second tag is the end tag (they are also called opening tags and closing tags). Though not always necessary, it is best practice to append a slash to tags which are not paired with a closing tag.

iii. The purpose of a Web browser is to read HTML documents and compose them into visible or audible Web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page. HTML describes the structure of a Website semantically along with cues for presentation, making it a markup language rather than a programming language.

iv. HTML elements form the building blocks of all Websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML Web pages.

b. CSS

i. **CASCADING STYLE SHEETS (CSS)** is used to style the WebPages.

ii. It is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to style WebPages and interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. CSS is a cornerstone specification of the Web and almost all Web pages use CSS style sheets to describe their presentation.

iii. CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts.[1] This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content .

iv. CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. It can also be used to allow the Web page to display differently depending on the screen size or device on which it is being viewed. While the author of a document typically links that document to a CSS file, readers can use a different style sheet, perhaps one on their own computer, to override the one the author has specified. However if the author or the reader did not link the document to a specific style sheet the default style of the browser will be applied.

c. BOOTSTRAP

Bootstrap is a popular open-source front-end framework used for building responsive and mobile-first Web applications. It provides a collection of CSS and JavaScript components and utilities that simplify the process of creating consistent, visually appealing, and user-friendly interfaces.

Here are key features and benefits of using Bootstrap:

i. Responsive Design: Bootstrap offers a responsive grid system and predefined CSS classes that automatically adapt to different screen sizes and devices. This ensures that Web applications built with Bootstrap are mobile-friendly and provide a consistent user experience across various devices.

ii. Pre-styled Components: Bootstrap provides a wide range of pre-styled components such as buttons, forms, navigation bars, modals, carousels, and more. These components are customizable and can be easily incorporated into Web applications, saving development time and effort.

iii. Typography and CSS Utilities: Bootstrap includes a set of typography styles and CSS utilities that simplify text formatting, alignment, spacing, and other common styling tasks. This allows developers to achieve consistent and visually appealing designs with ease.

iv. Browser Compatibility: Bootstrap is designed to be compatible with modern Web browsers, ensuring consistent rendering and functionality across different browsers. It handles cross-browser inconsistencies, reducing the need for manual CSS adjustments.

v. Extensive Documentation and Community Support: Bootstrap has comprehensive documentation with examples, guidelines, and explanations of each component and feature. Additionally, being an open-source framework, Bootstrap has a large and active community that provides support, resources and regular updates.

vi. Customization Options: Bootstrap can be customized to match the specific design requirements of a project. Developers can modify the default theme, create custom styles, and selectively include only the necessary components to optimize the size and performance of the application.

vii. Integration with JavaScript Libraries: Bootstrap includes JavaScript plugins that enhance the functionality of components, such as dropdowns, modals, tooltips, and more. It seamlessly integrates with popular JavaScript libraries like jQuery, allowing developers to enhance user interactions and add dynamic behavior to Web applications.

Middleware Technologies:

a. JAVASCRIPT

WHY TO USE JAVASCRIPT?

JavaScript is one of the 3 languages all Web developers must learn:

i. HTML to define the content of Web pages
ii. CSS to specify the layout of Web pages
iii. JavaScript to specify the behaviour of Web pages

Example

```
x = document.getElementById("demo");  //Find the HTML element with id="demo"
    x.innerHTML = "Hello JavaScript";  //Change the content of the HTML element
```

document.getElementById() is one of the most commonly used HTML DOM methods.

OTHER USES OF JAVASCRIPT:

- Delete HTML elements
- Create new HTML elements
- Copy HTML elements
- In HTML, JavaScript is a sequence of statements that can be executed by the Web browser.

JAVASCRIPT STATEMENTS:

- JavaScript statements are "commands" to the browser.
- The purpose of the statements is to tell the browser what to do.
- This JavaScript statement tells the browser to write "Hello Dolly" inside an HTML element with id="demo":

  Semicolon;

- Semicolon separates JavaScript statements.
- Normally you add a semicolon at the end of each executable statement.

- Using semicolons also makes it possible to write many statements on one line.

JAVASCRIPT CODE:

- JavaScript code (or just JavaScript) is a sequence of JavaScript statements.
- Each statement is executed by the browser in the sequence they are written.
- This example will manipulate two HTML elements:
- Example
- document.getElementById("demo").innerHTML="Hello Dolly";
  document.getElementById("ourDIV").innerHTML="How are you?";

JAVASCRIPT PROPERTIES:

- Properties are the values associated with a JavaScript object.
- A JavaScript object is a collection of unordered properties.
- Properties can usually be changed, added, and deleted, but some are read only.

JavaScript is a powerful language for Web development, providing the ability to create interactive, dynamic, and user-friendly Web applications. Its versatility, wide adoption, and vast ecosystem of tools and libraries make it a fundamental language for modern Web development.

b. JSP files

A JSP (JavaServer Pages) file is a text-based document that contains a combination of HTML, XML, and Java code. It is used in I b development with Java to dynamically generate Web pages. JSP files are processed by a JSP engine on the server side, which interprets the Java code and generates HTML output that is sent to the client's I b browser.

Here are some key points about JSP files:

i. File Extension: JSP files have the ".jsp" file extension, which helps identify them as JavaServer Pages.

ii. HTML Markup: JSP files can include HTML markup, allowing developers to structure the web page layout and define elements such as headers, paragraphs, tables, forms, and more. HTML tags and attributes are used just like in regular HTML files.

iii. JSP Tags: JSP files contain special tags, known as JSP tags or scriptlets, which are used to embed Java code within the HTML markup. The Java code is enclosed within "<%" and "%>" delimiters. These tags allow developers to perform dynamic operations, retrieve data from databases, iterate over collections, and manipulate the page content based on business logic.

iv. Expression Language (EL): JSP files can use Expression Language, denoted by "${}", to access and display Java variables or values within the HTML markup. EL simplifies the retrieval and presentation of data from Java objects, making the code more concise and readable.

v. Directives: JSP files may include directives that provide instructions to the JSP container. Common directives include the "page" directive, which sets page-specific attributes like error handling, language, and content type, and the "include" directive, which includes external files into the JSP file.

vi. Standard Actions: JSP files can utilize standard actions, such as "jsp:include" or "jsp:forward", to include or forward requests to other JSP pages or servlets. These

actions help in modularizing the code and reusing components across multiple pages.

vii. Scriptlets and Declarations: JSP files allow the declaration of variables and methods using the "<%! %>" tag, which is typically placed outside the HTML markup. This section is used for defining reusable code snippets and utility functions that can be accessed throughout the JSP file.

viii. Outputting Dynamic Content: JSP files can generate dynamic content by using the "out" implicit object or the JSP expression syntax. This allows for the dynamic rendering of data retrieved from Java objects or calculated within the JSP file.

ix. Integration with Java Beans and Servlets: JSP files can interact with Java Beans and servlets, accessing their methods and properties to fetch and process data. This enables the separation of business logic from presentation, promoting modular and maintainable code.

JSP files offer a powerful way to combine HTML markup, Java code, and dynamic content generation. They provide a flexible and efficient approach for developing dynamic Web pages within Java-based Web applications.

Backend Technologies:

a. JAVA (SpringMVC, Hibernate)

i. WHAT IS JAVA?

- JAVA is a widely-used, open source scripting language
- JAVA codes are executed on the server
- JAVA costs nothing, it is free to download and use

ii. WHAT IS JAVA FILE?

- JAVA code are executed on the server, and the result is returned to the browser as plain HTML
- JAVA files have extension ".java
- It also supports HTML and CSS codes with the help of JSP.

iii. WHY JAVA?

Java is a widely used programming language that offers several advantages, making it a popular choice for various software development projects. Here are some key reasons why Java is preferred in many contexts:

1. Platform Independence: One of the primary advantages of Java is its platform independence. Java programs can run on any platform with a Java Virtual Machine (JVM), including Windows, macOS, Linux, and more. This "write once, run anywhere" capability enables developers to create applications that can be easily deployed and executed on different operating systems.

2. Robust and Reliable: Java is known for its robustness and reliability. It includes features like automatic memory management, exception handling, and strong type checking, which help in building stable and secure applications. Java's strict compile-time checking catches errors before runtime, reducing the occurrence of bugs and improving application stability.

3. Object-Oriented Programming (OOP): Java is based on the object-oriented programming paradigm, which promotes modular, reusable, and maintainable code. It allows developers to create classes, objects, and interfaces, facilitating code organization and promoting code reusability.

4. Vast Ecosystem and Libraries: Java has a rich ecosystem of libraries, frameworks, and tools that provide ready-made components and functionalities for various application requirements. These libraries, such as Spring, Hibernate, and Apache Commons, simplify development tasks, speed up development, and enhance productivity.

5. Community and Support: Java has a large and active developer community worldwide. This community provides support, resources, and updates, making it easier to find solutions to programming challenges and stay updated with the latest advancements in the Java ecosystem. Additionally, the availability of extensive documentation and online forums further contributes to community support.

6. Scalability: Java is I ll-suited for building scalable applications. Its threading model supports concurrent programming, allowing developers to take advantage

of multi-core processors and build applications that can handle high volumes of simultaneous users and data processing.

7. Security: Java includes built-in security features, such as a robust security manager, bytecode verification, and encryption support. These features help in creating secure applications and protect against vulnerabilities and threats.

8. Enterprise Application Development: Java is widely used for enterprise application development. Its mature enterprise frameworks, like Java EE (Enterprise Edition), provide features for distributed computing, transaction management, messaging, and web services, making it suitable for building large-scale, robust, and scalable enterprise applications.

9. Longevity and Backward Compatibility: Java has been in use for over two decades and has maintained backward compatibility throughout its evolution. This means that applications developed in older versions of Java can continue to run on newer versions without major modifications, ensuring the longevity and sustainability of Java-based applications.

Considering these advantages, Java has established itself as a versatile, reliable, and widely adopted programming language, making it a preferred choice for a range of applications, from Web and mobile development to enterprise systems and beyond.

iv. WHAT CAN JAVA DO?

- JAVA can generate dynamic page content
- JAVA can create, open, read, write, delete, and close files on the server
- JAVA can collect form data
- JAVA can send and receive cookies
- JAVA can add, delete, modify data in your database
- JAVA can restrict users to access some pages on your website
- JAVA can encrypt data

DATABASE

a. MYSQL

MYSQL is developed, distributed, and supported by Oracle Corporation. MYSQL is a database system used on the web it runs on a server. MYSQL is ideal for both small and large applications. It is very fast, reliable, and easy to use. It supports standard SQL. MYSQL can be compiled on a number of platforms.

The data in MYSQL is stored in tables. A table is a collection of related data, and it consists of columns and rows. Databases are useful when storing information categorically.

FEATURES OF MYSQL:

Internals and portability:

a. Written in C and C++.
b. Tested with a broad range of different compilers.
c. Works on many different platforms.
d. Tested with Purify (a commercial memory leakage detector) as well as with Val grind, a GPL tool.
e. Uses multi-layered server design with independent modules.

Security:

f. A privilege and password system that is very flexible and secure, and that enables host-based verification.
g. Password security by encryption of all password traffic when you connect to a server.

Scalability and Limits:

h. Support for large databases. we use MYSQL Server with databases that contain 50 million records. we also know of users who use MYSQL Server with 200,000 tables and about 5,000,000,000 rows.
i. Support for up to 64 indexes per table (32 before MYSQL 4.1.2). Each index may consist of 1 to 16 columns or parts of columns. The maximum index width is 767

bytes for InnoDB tables, or 1000 for MYISAM; before MYSQL 4.1.2, the limit is 500 bytes. An index may use a prefix of a column for CHAR, VARCHAR, BLOB, or TEXT column types.

CONNECTIVITY:

Clients can connect to MYSQL Server using several protocols:

- Clients can connect using TCP/IP sockets on any platform.
- On Windows systems in the NT family (NT, 2000, XP, 2003, or Vista), clients can connect using named pipes if the server is started with the enable-named-pipe option. In MYSQL 4.1 and higher, Windows servers also support shared-memory connections if started with the shared-memory option. Clients can connect through shared memory by using the protocol=memory option.
- On UNIX systems, clients can connect using Unix domain socket files.

LOCALIZATION:

a. The server can provide error messages to clients in many languages.

b. All data is saved in the chosen character set.

CLIENTS AND TOOLS:

c. MYSQL includes several client and utility programs. These include both command-line programs such as MYSQLdump and MYSQLadmin, and graphical programs such as MYSQL Workbench.

d. MYSQL Server has built-in support for SQL statements to check, optimize and repair tables. These statements are available from the command line through the MYSQL check client

e. MYSQL programs can be invoked with the --help or -? option to obtain online assistance.

WHY TO USE MYSQL:

i. Leading open source RDBMS
ii. Ease of use – No frills
iii. Fast
iv. Robust
v. Security
vi. Multiple OS support
vii. Free
viii. Technical support
ix. Support large database– upto 50 million rows, file size limit upto 8 Million TB

Project Management Tool

a. MAVEN

Maven is a popular build automation and dependency management tool used primarily in Java-based projects. It provides a standardized way to manage project dependencies, compile source code, package applications, and facilitate project management tasks. Here are key aspects and features of Maven:

1. Project Object Model (POM): Maven uses a Project Object Model (POM) file, typically named pom.xml, to define the project's configuration, dependencies, build instructions, and other project-related information. The POM serves as the central configuration file for Maven-based projects.

2. Dependency Management: Maven simplifies dependency management by allowing developers to declare project dependencies within the POM file. It automatically resolves and downloads the required dependencies from remote repositories, simplifying the process of managing external libraries and frameworks.

3. Build Lifecycle: Maven provides a predefined set of build phases and goals, collectively known as the build lifecycle. These build phases (e.g., compile, test, package, install) define the order in which build goals are executed. Developers can customize the build process by binding plug-ins to specific build phases.

4. Plugin Architecture: Maven is highly extensible through its plugin architecture. Plugins provide additional functionality for tasks such as compiling code, running tests, generating reports, deploying artifacts, and more. Maven includes a rich ecosystem of plugins that can be easily integrated into the build process.

8. Integration with IDEs: Maven integrates well with popular Integrated Development Environments (IDEs) such as Eclipse, IntelliJ IDEA, and NetBeans. IDEs can leverage Maven's configuration and project structure to enhance development productivity and provide seamless build and dependency management.

Maven simplifies the development and management of Java projects by providing a standardized build process, dependency management, and project structure. It helps ensure consistency, improves productivity, and facilitates collaboration among developers working on the same project.

SERVER

a. Apache Tomcat (9.0)

Apache Tomcat, often referred to simply as Tomcat, is an open-source web server and Java Servlet container developed by the Apache Software Foundation. It provides a platform for running Java-based web applications, specifically those built using Java Servlet, JavaServer Pages (JSP), and Java Expression Language (EL) technologies. Here are key aspects and features of Apache Tomcat:

1. Java Servlet Container: Tomcat is primarily known as a Java Servlet container, which means it provides an environment for executing Java Servlets, handling HTTP requests, and generating dynamic web content. It follows the Java Servlet API specifications, allowing developers to build web applications conforming to this standard.

2. Web Server: In addition to serving as a Servlet container, Tomcat can also function as a standalone web server. It supports HTTP and HTTPS protocols, allowing it to handle client requests, serve static content, and host web applications without the need for an additional web server.

3. Platform Independence: Tomcat is designed to be platform-independent and can run on various operating systems, including Windows, macOS, Linux, and others, as long as the Java Runtime Environment (JRE) or Java Development Kit (JDK) is installed.

4. Community and Support: Tomcat benefits from a large and active open-source community. This community provides support, documentation, tutorials, and frequent updates, ensuring that Tomcat remains stable, secure, and up-to-date.

Apache Tomcat is widely used in the Java web development community due to its simplicity, versatility, and compatibility with Java Servlet-based applications. It serves as a reliable and efficient platform for deploying and running web applications, making it a popular choice for developers.

CHAPTER – 4

ANALYSIS & DESIGN

ANALYSIS

EXISTING SYSTEM:

Hospitals currently use a manual system for the management and maintenance of critical information. The current system requires numerous paper forms, with data stores spread throughout the hospital management infrastructure. Often information is incomplete or does not follow management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital information is lost. Multiple copies of the same information exist in the hospital and may lead to inconsistencies in data in various data stores.

PROPOSED SYSTEM:

The Mediface administration system is designed for any hospital to replace their existing manual paper based system. The new system is to control the information of patients. Appointment bookings. These services are to be provided in an efficient, cost effective manner, with the goal of reducing the time and resources currently required for such tasks.

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are:

i. Economic Feasibility

This study is carried out to check the economic impact will have on the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customised products have to be purchased.

ii. Technical Feasibility

This study is carried out to check the technical feasibility that is the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes for the implementing this system.

iii. Operational Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism.

SYSTEM DESIGN:

INTRODUCTION TO UML:

UML Design

The Unified Modeling Language (UML) is a standard language for specifying, visualizing, constructing, and documenting the software system and its components. It is a graphical language ,which provides a vocabulary and set of semantics and rules. The UML focuses on the conceptual and physical representation of the system. It captures the decisions and understandings about systems that must be constructed. It is used to understand, design, configure, maintain, and control information about the systems.

The UML is a language for:

i. Visualizing

Through UML we see or visualize an existing system and ultimately we visualize how the system is going to be after implementation. Unless we think, we cannot implement. UML helps to visualize, how the components of the system communicate and interact with each other.

ii. Specifying

Specifying means building, models that are precise, unambiguous and complete UML addresses the specification of all the important analysis design, implementation decisions that must be made in developing and deploying a software system.

iii. Constructing

UML models can be directly connected to a variety of programming language through mapping a model from UML to a programming language like JAVA or C++ or VB. Forward Engineering and Reverse Engineering is possible through UML.

iv. Documenting

The Deliverables of a project apart from coding are some Artifacts, which are critical in controlling, measuring and communicating about a system during its developing requirements, architecture, desire, source code, project plans, tests, prototypes releasers, etc...

UML Approach

UML Diagram

A diagram is the graphical presentation of a set of elements, most often rendered as a connected graph of vertices and arcs. you draw diagram to visualize a system from different perspective, so a diagram is a projection into a system. For all but most trivial systems, a diagram represents an elided view of the elements that make up a system. The same element may appear in all diagrams, only a few diagrams , or in no diagrams at all. In theory, a diagram may contain any combination of things and relationships. In practice, however, a small number of common combinations arise, which are consistent with the five most useful views that comprise the architecture of a software-intensive system. For this reason, the UML includes nine such diagrams:

1. Class diagram
2. Object diagram
3. Use case diagram
4. Sequence diagram
5. Activity diagram
6. Flow chart

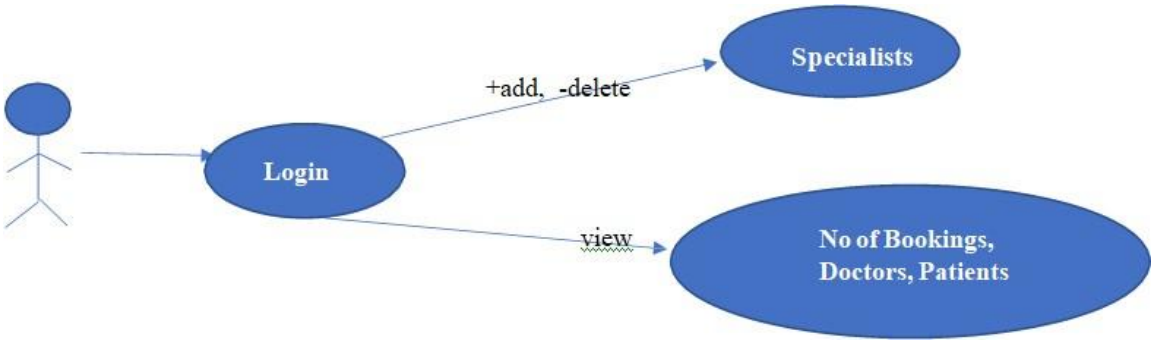
USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases),and any dependencies between those use cases.

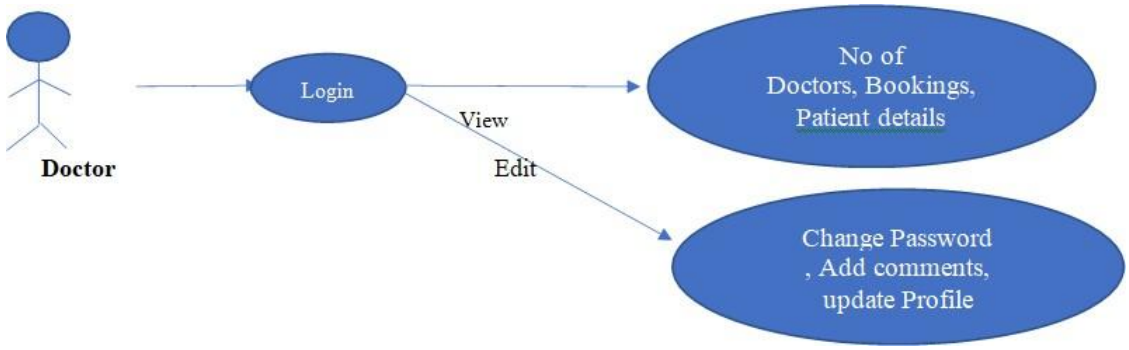
Use case diagrams are formally included in two modeling languages defined by the unified modeling language (UML) and the systems modeling language(sysML)

Use case diagram of our project:

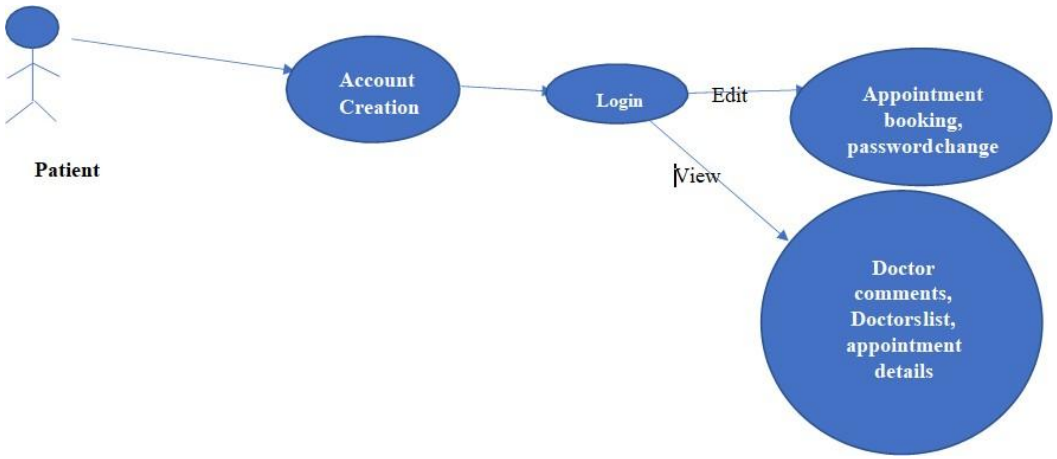
i. Admin



ii. Doctor

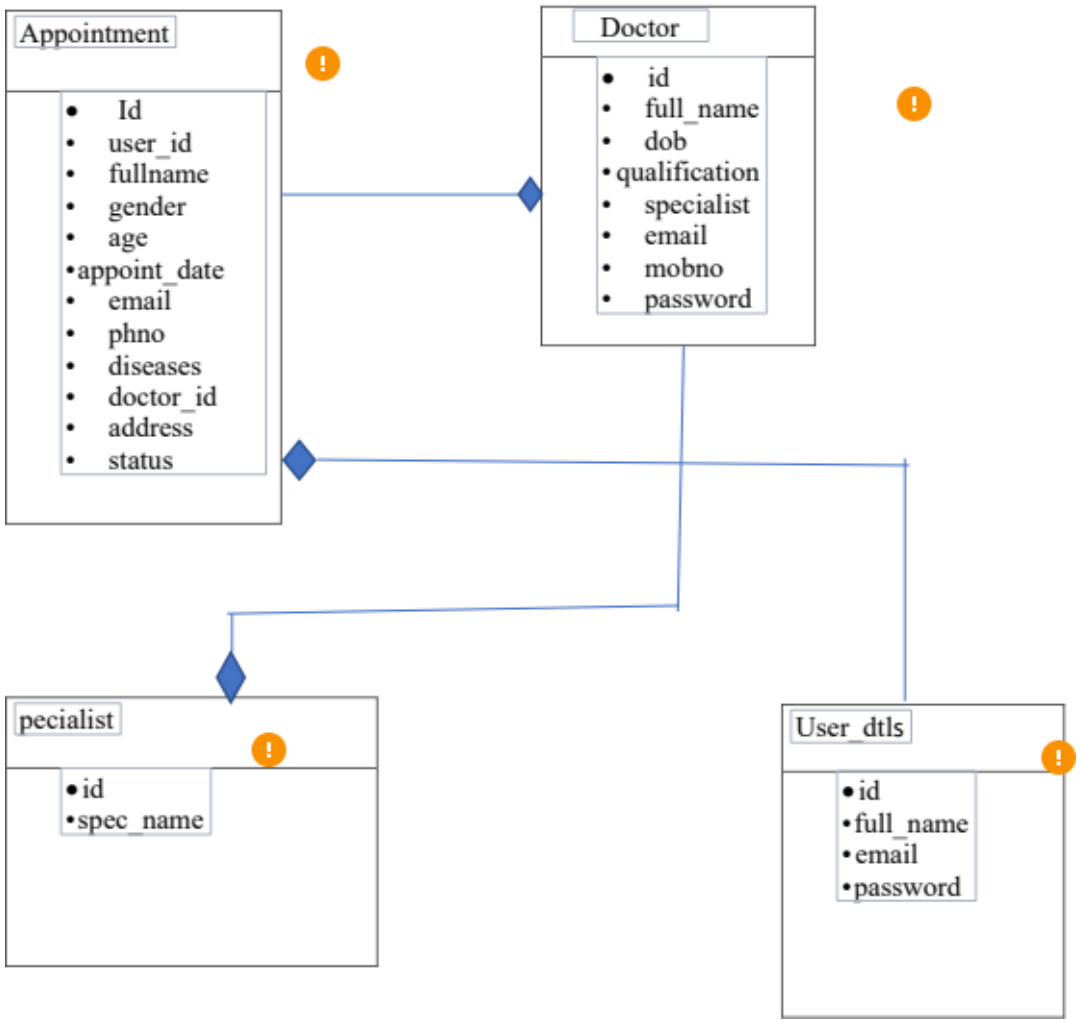


iii. Patient



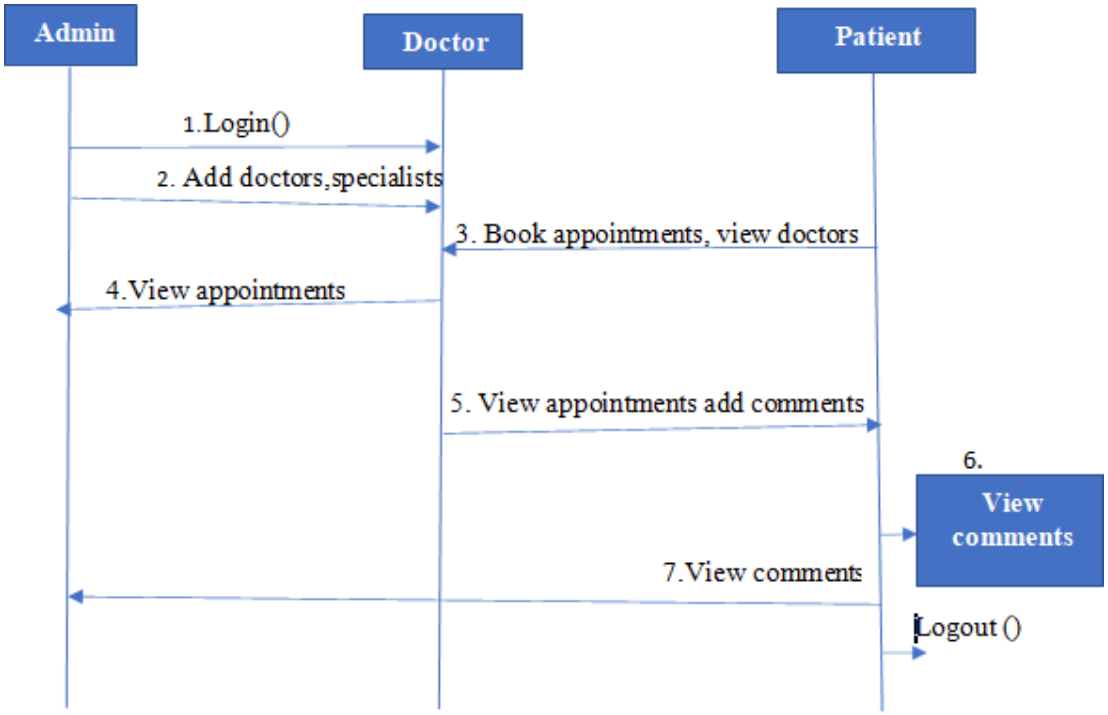
CLASS DIAGRAM:

A Class is a category or group of things that has similar attributes and common behaviour. A Rectangle is the icon that represents the class it is divided into three areas. The upper most area contains the name, the middle; area contains the attributes and the lowest areas show the operations. Class diagrams provides the representation that developers work from. Class diagrams help on the analysis side, too.



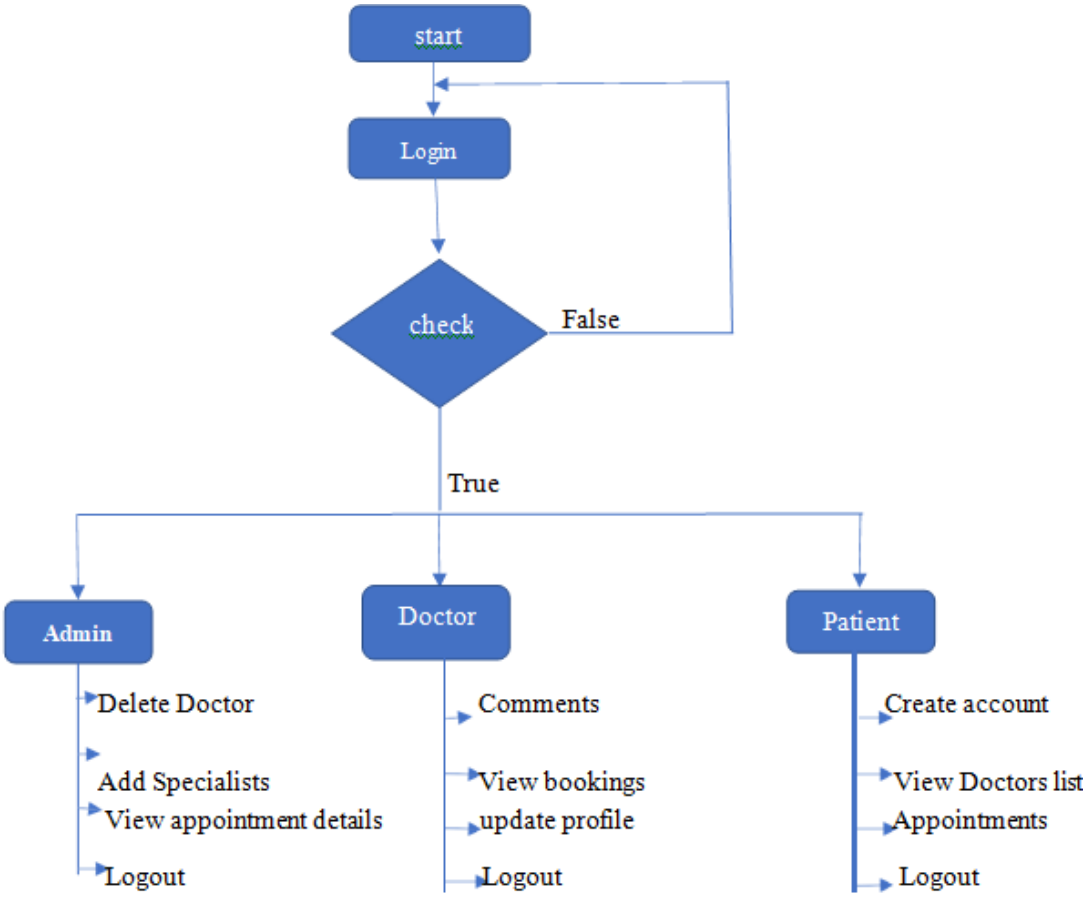
SEQUENCE DIAGRAM:

A **Sequence Diagram** is an interaction diagram that emphasis the time ordering of messages; a collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other.



FLOW CHART:

A flowchart is a diagram that illustrates the steps, sequences, and decisions of a process or workflow. While there are many different types of flowcharts, a basic flowchart is the simplest form of a process map. It's a powerful tool that can be used in multiple fields for planning, visualizing, documenting, and improving processes.

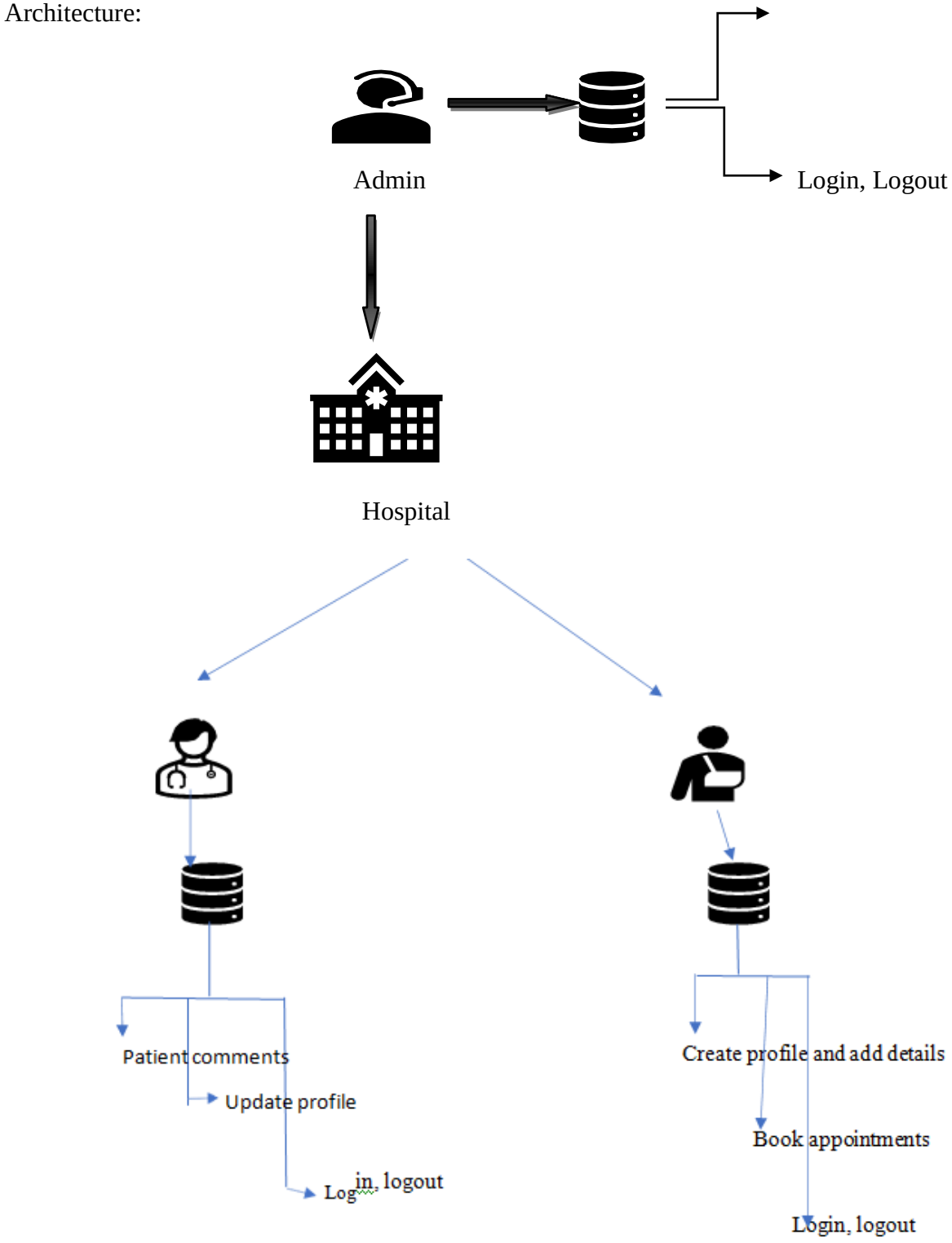


Architecture:

Architecture is a critical aspect of designing a system, as it sets the foundation for how the system will function and be built. It is the process of making high-level decisions about the organization of a system, including the selection of hardware and software components, the design of interfaces, and the overall system structure.

Doctors, Specialist

Architecture:



IMPLEMENTATION:

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

CHAPTER – 5

SAMPLE CODES & SCREENSHOTS

Sample code

JSP code

User register.jsp

```
<%@page import="com.entity.Doctor"%>

<%@page import="com.dao.DoctorDao"%>

<%@page import="com.entity.Specalist"%>

<%@page import="java.util.List"%>

<%@page import="com.db.DBConnect"%>

<%@page import="com.dao.SpecialistDao"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

<%@include file="../component/allcss.jsp"%>

<style type="text/css">

.paint-card { box-shadow: 0 0 10px 0 rgba(0, 0, 0, 0.3); } </style>

</head>

<body>

<%@include file="navbar.jsp"%>

<div class="container-fluid p-3">
```

Add Doctor

<c:if test="${not empty errorMsg}">

${errorMsg} <c:remove var="errorMsg" scope="session" /> </c:if>

<c:if test="${not empty succMsg}">

${succMsg}

<c:remove var="succMsg" scope="session" /> </c:if>

<form action="../addDoctor" method="post">

 <label class="form-label">Full Name</label> <input type="text" required name="fullname" class="form-control">

<label class="form-label">DOB</label>

<input type="date" required name="dob" class="form-control">

<label class="form-label">Qualification</label>

<input required name="qualification" type="text" class="form-control">

<label class="form-label">Specialist</label>

<select name="spec" required class="form-control">

<option>--select--</option>

```
<% SpecialistDao dao = new SpecialistDao(DBConnect.getConn()); List<Specalist>
list = dao.getAllSpecialist(); for (Specalist s : list) { %>
<option><%=s.getSpecialistName()%>

</option> <% } %> </select>

</div>

<div class="mb-3">

<label class="form-label">Email</label>

<input type="text" required name="email" class="form-control"> </div>

<div class="mb-3">

<label class="form-label">Mob No</label> <input type="text" required
name="mobno" class="form-control">

</div>

<div class="mb-3">

<label class="form-label">Password</label>

<input required name="password" type="password" class="form-control">

</div>

<button type="submit" class="btn btn-primary">Submit</button> </form>

</div> </div>

</div> </div>

</div>

</body>

</html>
```

JAVA code

i. DTO (Data Transfer Object)

Doctor.java

```
package com.entity;

public class Doctor

{

private int id;

private String fullName;

private String dob;

private String qualification;

private String specialist;

private String email;

private String mobNo;

private String password;

public Doctor()

{

super(); // TODO Auto-generated constructor stub

}

public Doctor(String fullName, String dob, String qualification, String specialist,
String email, String mobNo, String password)

{

super();

this.fullName = fullName;
```

```
this.dob = dob;
this.qualification = qualification;
this.specialist = specialist;
this.email = email;
this.mobNo = mobNo;
this.password = password;
 }
public Doctor(int id, String fullName, String dob, String qualification, String specialist, String email, String mobNo, String password)
{
super();
this.id = id;
this.fullName = fullName;
this.dob = dob;
this.qualification = qualification;
this.specialist = specialist;
this.email = email;
this.mobNo = mobNo;
this.password = password;
 }
public int getId()
{
return id;
```

```
} public void setId(int id)
{
this.id = id;
 }
public String getFullName()
{
return fullName;
 }
public void setFullName(String fullName)
{
this.fullName = fullName;
}
public String getDob()
{
return dob;
} public void setDob(String dob)
{
this.dob = dob;
} public String getQualification()
{
return qualification;
}
public void setQualification(String qualification)
```

```
{
this.qualification = qualification;
}
 public String getSpecialist()
{
return specialist;
} public void setSpecialist(String specialist)
 { this.specialist = specialist;
} public String getEmail()
{
return email;
} public void setEmail(String email)
{ this.email = email;
 } public String getMobNo() {
 return mobNo;
} public void setMobNo(String mobNo) {
this.mobNo = mobNo;
} public String getPassword() {
return password;
 } public void setPassword(String password)
{ this.password = password;
}
}
```

ii. DAO (Data Access Object)

DoctorDao.java

```
package com.dao;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;
import com.entity.Doctor;
public class DoctorDao {
private Connection conn;
public DoctorDao(Connection conn) {
super();
this.conn = conn;
}
public boolean registerDoctor(Doctor d) {
boolean f = false;
try {
String sql="insert into doctor(full_name,dob,qualification,specialist,email,mobno,password) values(?,?,?,?,?,?,?)";
PreparedStatement ps = conn.prepareStatement(sql); ps.setString(1, d.getFullName()); ps.setString(2, d.getDob());
ps.setString(3, d.getQualification()); ps.setString(4, d.getSpecialist());
ps.setString(5, d.getEmail());
ps.setString(6, d.getMobNo());
ps.setString(7, d.getPassword());
int i = ps.executeUpdate();
if (i == 1) {
f = true; }
```

```
} catch (Exception e) {
e.printStackTrace();
} return f;
} public List<Doctor> getAllDoctor() {
List<Doctor> list = new ArrayList<Doctor>();
Doctor d = null;
try { String sql = "select * from doctor order by id desc"; PreparedStatement ps =
conn.prepareStatement(sql); ResultSet rs = ps.executeQuery();
while (rs.next()) {
d = new Doctor();
d.setId(rs.getInt(1));
d.setFullName(rs.getString(2));
d.setDob(rs.getString(3));                    d.setQualification(rs.getString(4));
d.setSpecialist(rs.getString(5)); d.setEmail(rs.getString(6));
d.setMobNo(rs.getString(7));
d.setPassword(rs.getString(8));
list.add(d);
} } catch (Exception e)
{ e.printStackTrace();
} return list;
} public Doctor getDoctorById(int id) {
Doctor d = null;
try {
String sql = "select * from doctor where id=?"; PreparedStatement ps =
conn.prepareStatement(sql); ps.setInt(1, id);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
d = new Doctor();
d.setId(rs.getInt(1));
d.setFullName(rs.getString(2));
```

```
d.setDob(rs.getString(3));
d.setQualification(rs.getString(4));
d.setSpecialist(rs.getString(5));
d.setEmail(rs.getString(6));
d.setMobNo(rs.getString(7));
d.setPassword(rs.getString(8));
}
} catch (Exception e) {
e.printStackTrace();
} return d;
} public boolean updateDoctor(Doctor d) {
boolean f = false;
try {
String sql = "update doctor set full_name=?,dob=?,qualification=?,specialist=?,email=?,mobno=?,password=? where id=?";
PreparedStatement ps = conn.prepareStatement(sql); ps.setString(1, d.getFullName());
ps.setString(2, d.getDob());
ps.setString(3, d.getQualification());
ps.setString(4, d.getSpecialist());
ps.setString(5, d.getEmail());
ps.setString(6,d.getMobNo());
ps.setString(7, d.getPassword());
ps.setInt(8, d.getId());
int i = ps.executeUpdate();
 if (i == 1) {
f = true; }
} catch (Exception e)
{ e.printStackTrace();
```

```
}
return f;
} public boolean deleteDoctor(int id) {
boolean f = false;
try {
String sql = "delete from doctor where id=?"; PreparedStatement ps =
conn.prepareStatement(sql); ps.setInt(1, id);
int i = ps.executeUpdate();
if (i == 1) {
f = true;
} } catch (Exception e) { e.printStackTrace();
} return f;
}
public Doctor login(String email, String psw)
{ Doctor d = null;
try {
String sql = "select * from doctor where email=? and password=?";
PreparedStatement ps = conn.prepareStatement(sql); ps.setString(1, email);
ps.setString(2, psw);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
d = new Doctor();
d = new Doctor();
d.setId(rs.getInt(1));
d.setFullName(rs.getString(2));
d.setDob(rs.getString(3));                    d.setQualification(rs.getString(4));
d.setSpecialist(rs.getString(5)); d.setEmail(rs.getString(6));
d.setMobNo(rs.getString(7));
d.setPassword(rs.getString(8));
}
```

```
} catch (Exception e) {
e.printStackTrace();
} return d;
} public int countDoctor() {
int i = 0;
try {
String sql = "select * from doctor";
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
i++;
}
} catch (Exception e) {
e.printStackTrace();
} return i;
}
public int countAppointment() {
int i = 0;
try {
String sql = "select * from appointment";
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
i++;
}
} catch (Exception e) {
e.printStackTrace();
} return i;
} public int countAppointmentByDocotrId(int did) {
int i = 0;
```

```
try {
String sql = "select * from appointment where doctor_id=?"; PreparedStatement ps
= conn.prepareStatement(sql); ps.setInt(1, did);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
i++;
}
} catch (Exception e) {
e.printStackTrace();
} re turn i;
} public int countUSer() {
int i = 0;
try {
String sql = "select * from user_dtls";
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) { i++; }
} catch (Exception e) { e.printStackTrace();
 } return i;
 } public int countSpecialist() {
 int i = 0;
try {
String sql = "select * from specialist";
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) { i++;
} }
catch (Exception e) { e.printStackTrace();
} return i;
}
```

```
 public boolean checkOldPassword(int userid, String oldPassword) {
boolean f = false;
try {
String sql = "select * from doctor where id=? and password=?";
PreparedStatement ps = conn.prepareStatement(sql); ps.setInt(1, userid);
ps.setString(2, oldPassword);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
f = true;
} }
catch (Exception e) { e.printStackTrace();
} return f;
} public boolean changePassword(int userid, String newPassword) {
boolean f = false;
try {
String sql = "update doctor set password=? where id=?";
PreparedStatement ps = conn.prepareStatement(sql);
ps.setString(1, newPassword);
ps.setInt(2, userid);
int i = ps.executeUpdate();
if (i == 1) {
f = true; }
} catch (Exception e) {
e.printStackTrace();
} return f;
} public boolean editDoctorProfile(Doctor d) {
boolean f = false;
try {
String sql = "update doctor set
full_name=?,dob=?,qualification=?,specialist=?,email=?,mobno=? where id=?";
```

```
PreparedStatement ps = conn.prepareStatement(sql); ps.setString(1,
d.getFullName());
ps.setString(2, d.getDob());
ps.setString(3, d.getQualification());
ps.setString(4, d.getSpecialist());
ps.setString(5, d.getEmail());
ps.setString(6, d.getMobNo());
ps.setInt(7, d.getId());
int i = ps.executeUpdate();
if (i == 1) {
f = true;
} } catch (Exception e) {
e.printStackTrace();
} return f;
} }
```

iii. **Controller**

AppointmentServlet.java

```
package com.user.servlet;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import com.dao.AppointmentDAO;
import com.db.DBConnect;
import com.entity.Appointment;
```

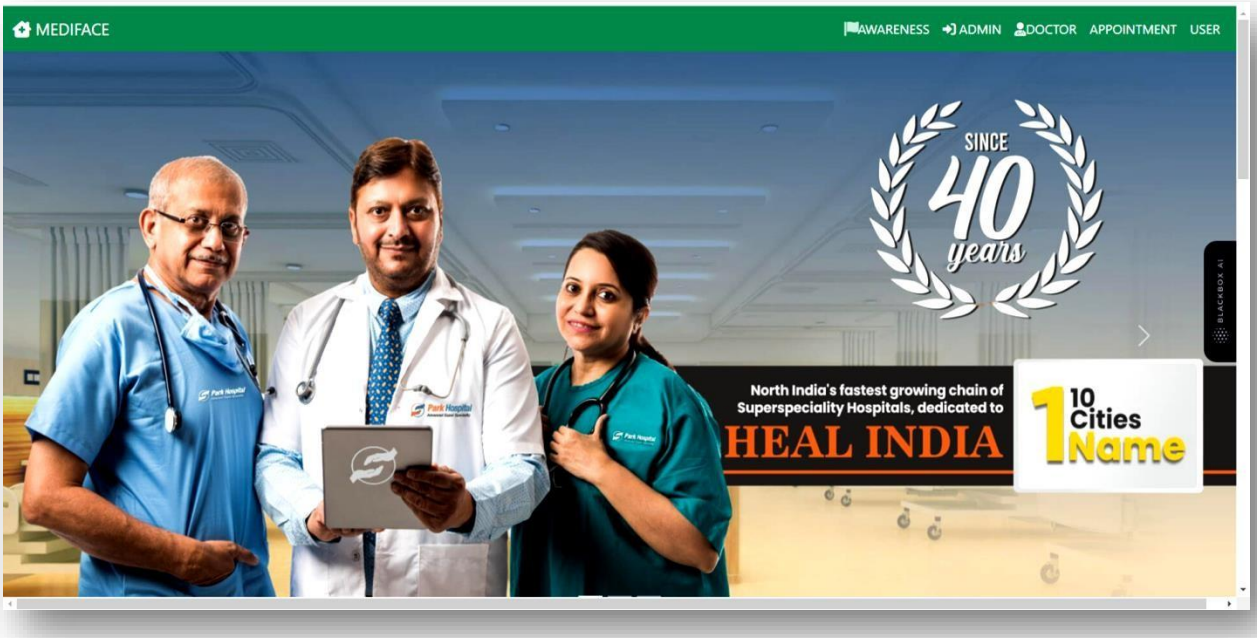
```
@WebServlet("/appAppointment")
public class AppointmentServlet extends HttpServlet { @Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
int userId = Integer.parseInt(req.getParameter("userid")); String fullname =
req.getParameter("fullname");
String gender = req.getParameter("gender");
String age = req.getParameter("age");
String appoint_date = req.getParameter("appoint_date"); String email =
req.getParameter("email");
String phno = req.getParameter("phno");
String diseases = req.getParameter("diseases");
int doctor_id = Integer.parseInt(req.getParameter("doct")); String address =
req.getParameter("address");
Appointment ap = new Appointment(userId, fullname, gender, age, appoint_date,
email, phno, diseases, doctor_id, address, "Pending");
AppointmentDAO dao = new AppointmentDAO(DBConnect.getConn());
HttpSession session = req.getSession();
if (dao.addAppointment(ap)) {
session.setAttribute("succMsg", "Appointment_Sucessful");
resp.sendRedirect("user_appointment.jsp");
 } else { session.setAttribute("errorMsg", "Something wrong on server");
resp.sendRedirect("user_appointment.jsp");
    }
  }
}
```

Database Connection

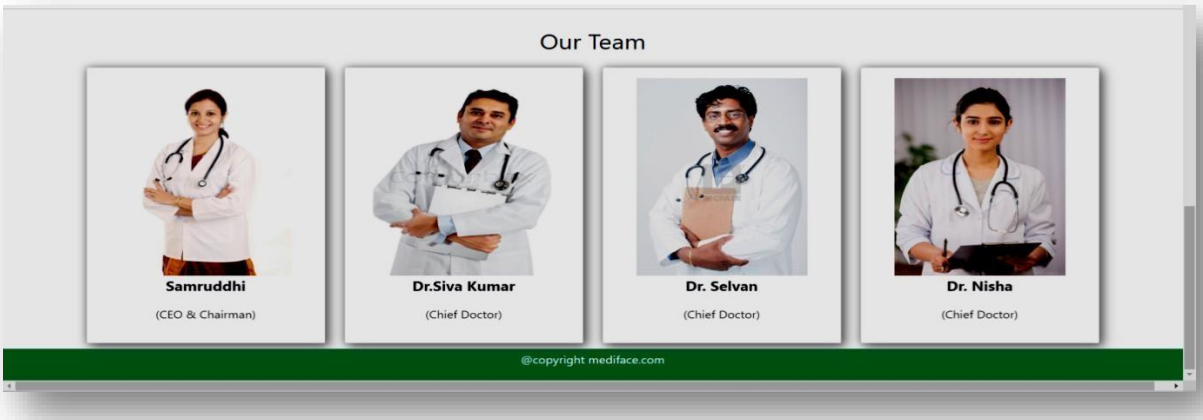
DBConnect.java

```
package com.db;

import java.sql.Connection;

import java.sql.DriverManager;

public class DBConnect {

private static Connection conn;

public static Connection getConn() {

try {

Class.forName("com.MYSQL.cj.jdbc.Driver");

conn=DriverManager.getConnection("jdbc:MYSQL://localhost:3306/hospital_2",
"root", "password");

 }

catch (Exception e)

{

e.printStackTrace();

} return conn;

}

}
```

HOMEPAGE

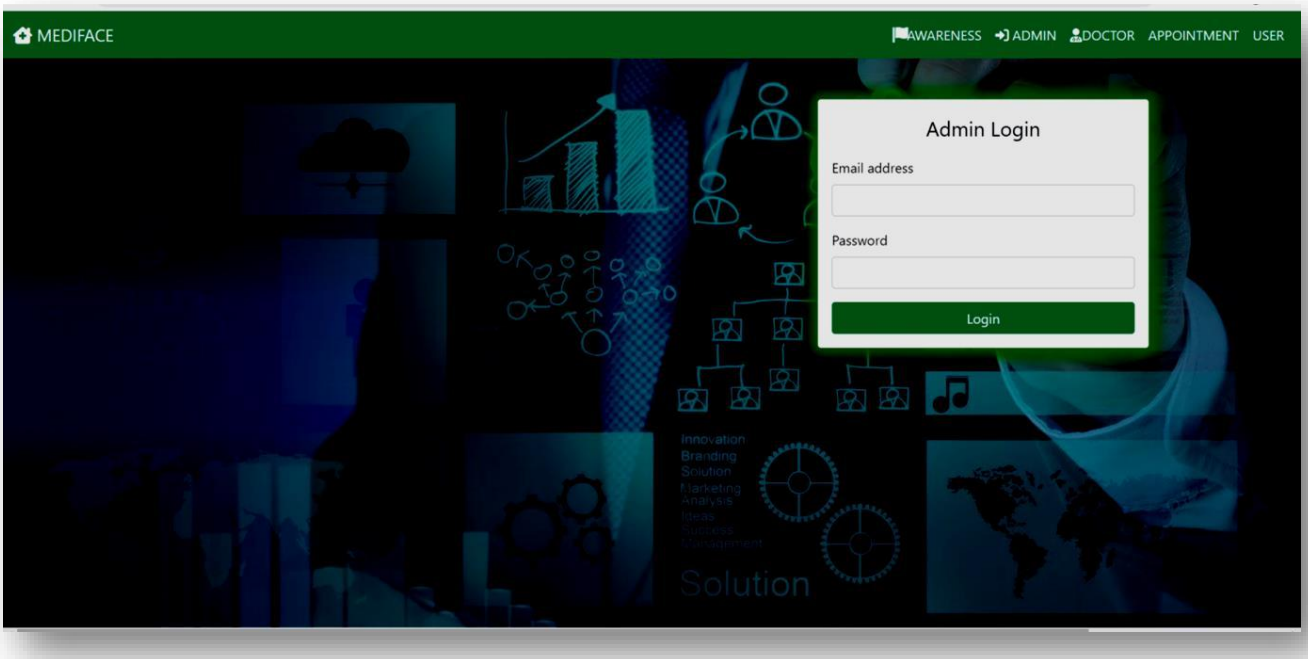

MEDIFACE
AWARENESS
ADMIN
DOCTOR
APPOINTMENT
USER
SINCE
40
years
North India's fastest growing chain of
Superspeciality Hospitals, dedicated to
HEAL INDIA
10
Cities
Name


Key Features of our Hospital
100% Safety
Staff keep checking any possible risks to your safety. For example, they look out for signs that your health could be getting worse and then take steps to improve it. Your care records are complete and accurate, and they are stored securely to keep them confidential.
Clean Environment
The hospital is clean and staff make sure that the risk of infection is as low as possible. Added information on certain activities that can increase risk of COVID-19 infection with references to ventilation guidance by CDC and WHO
Friendly Staff
Your care, treatment and health condition are explained to you in a way that you can understand. Staff are compassionate when you need help and think ahead to plan what you might need.
Appointments at ease
The appointments system is easy for everyone to use. The hospital tries to avoid long waiting times, delays and cancellations.Staff tell you about any changes or disruptions that may affect your treatment.

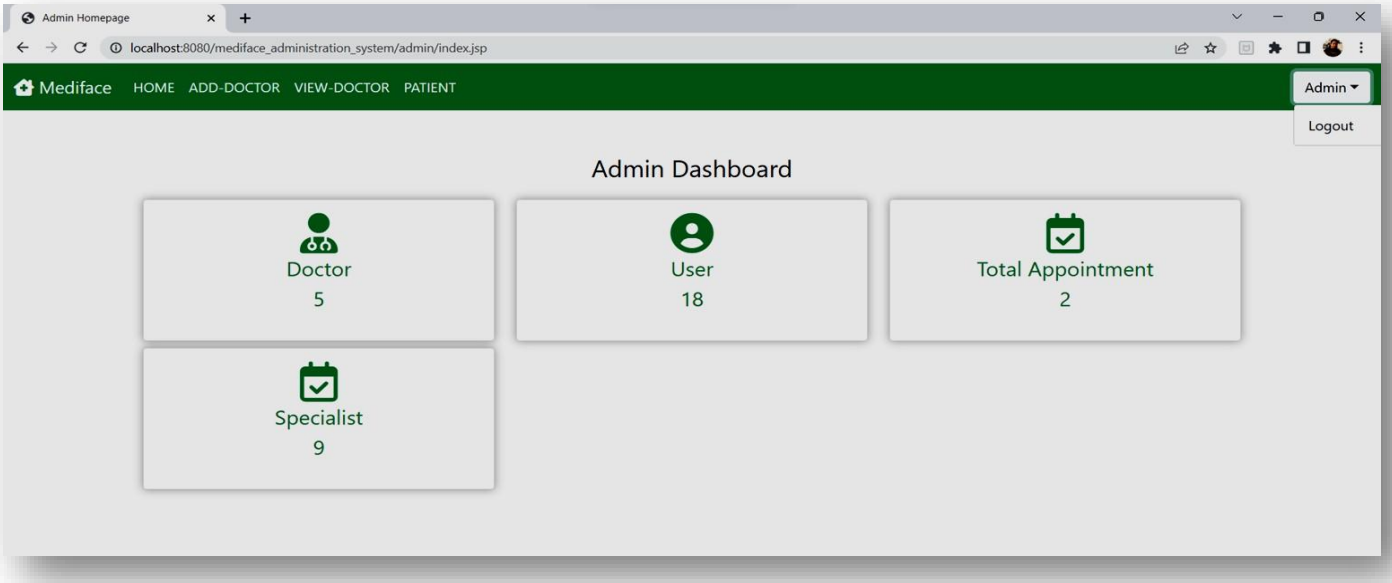

Our Team
Samruddhi
(CEO & Chairman)
Dr.Siva Kumar
(Chief Doctor)
Dr. Selvan
(Chief Doctor)
Dr. Nisha
(Chief Doctor)
@copyright mediface.com

ADMIN

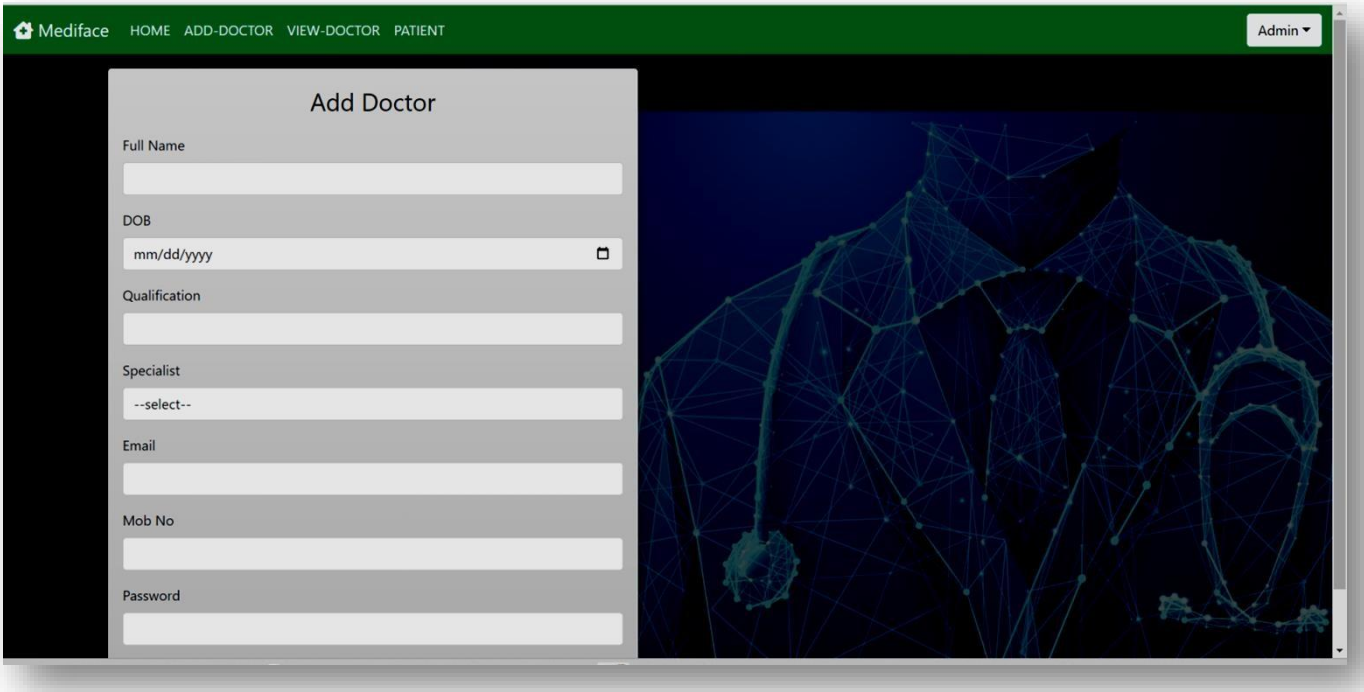
i. Login page



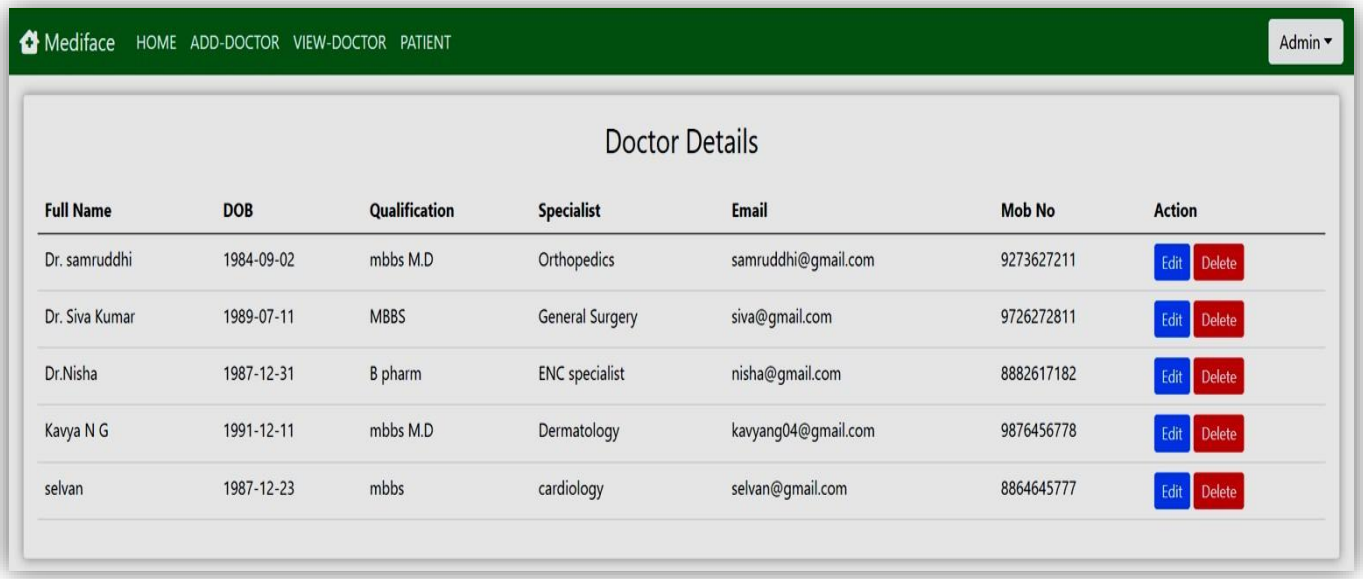
ii. Admin Homepage



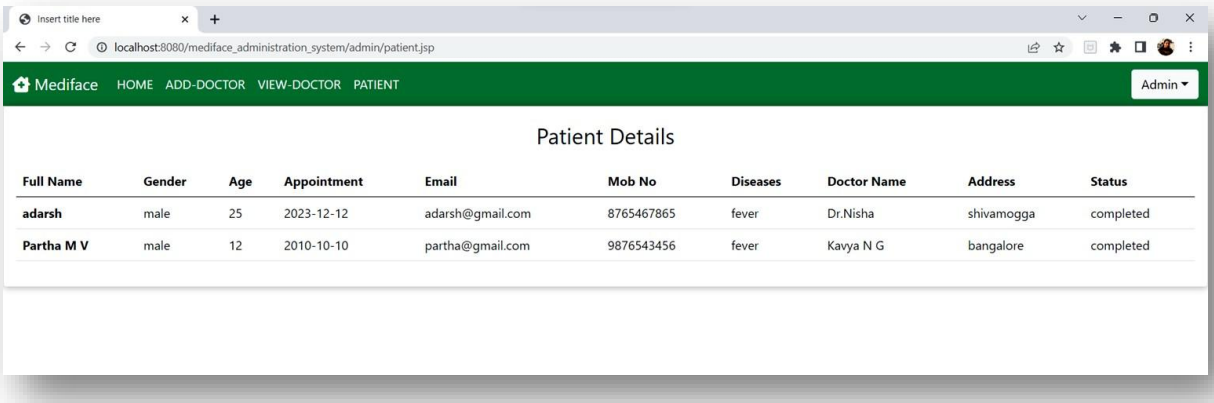
iii. Add Doctor



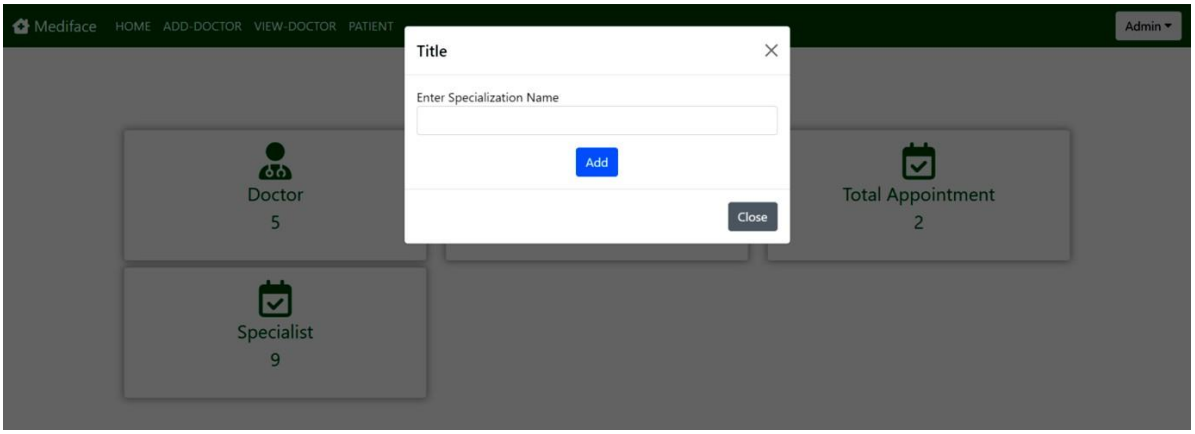
iv. View Doctors



v. View Patient details

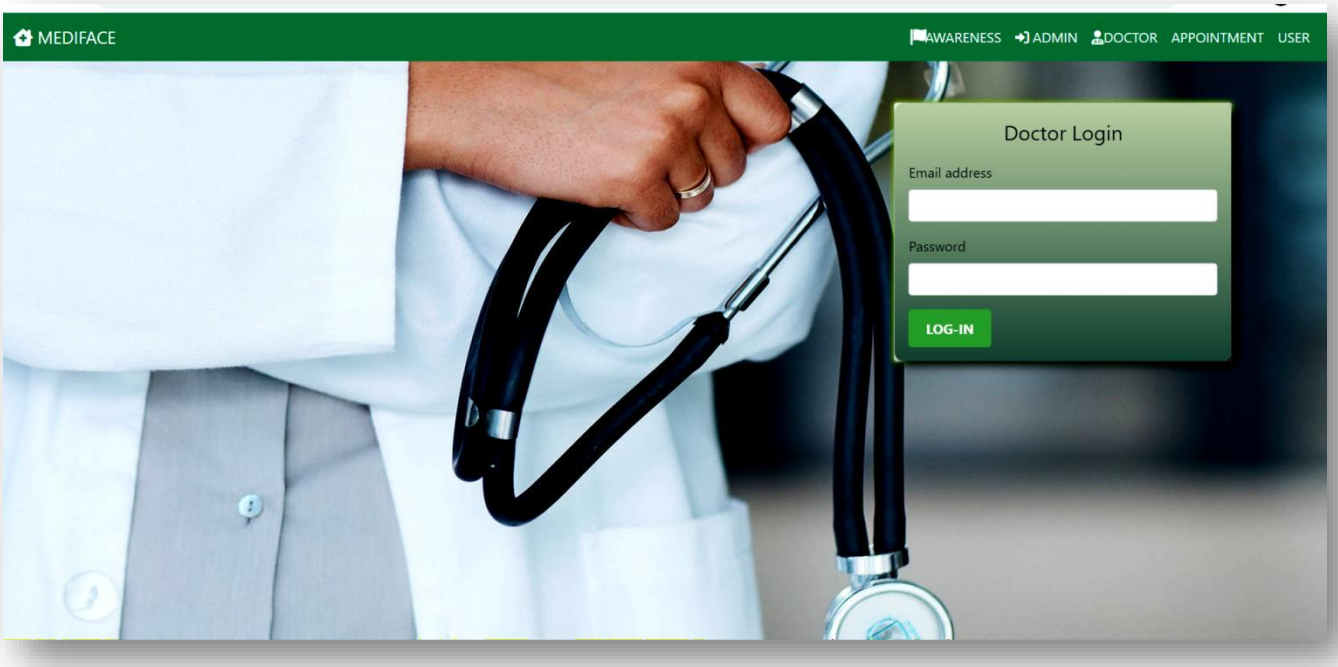


vi. Add specialist

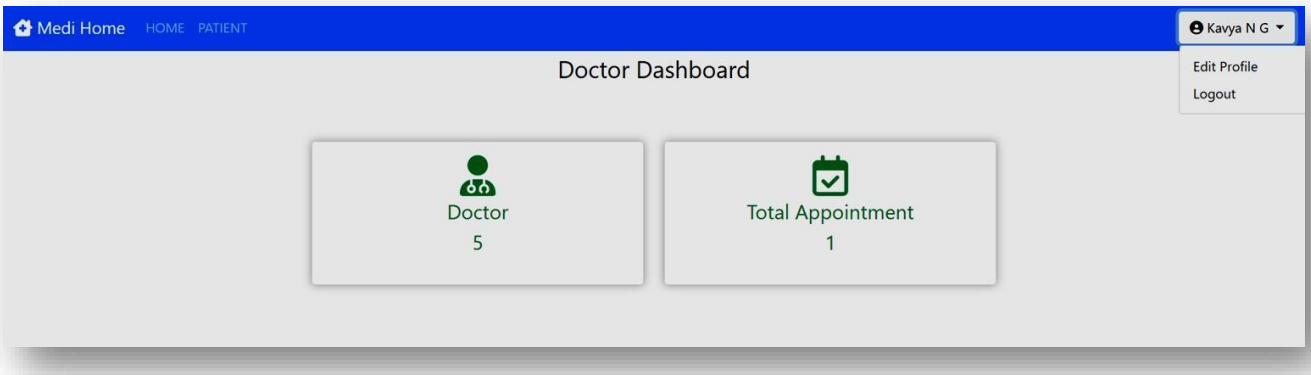


DOCTOR

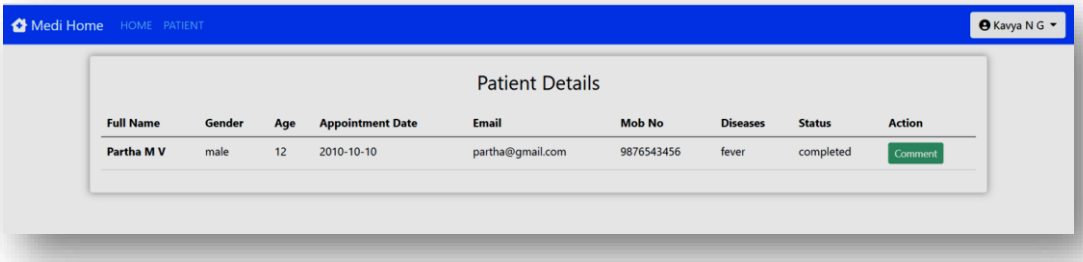
i. Login



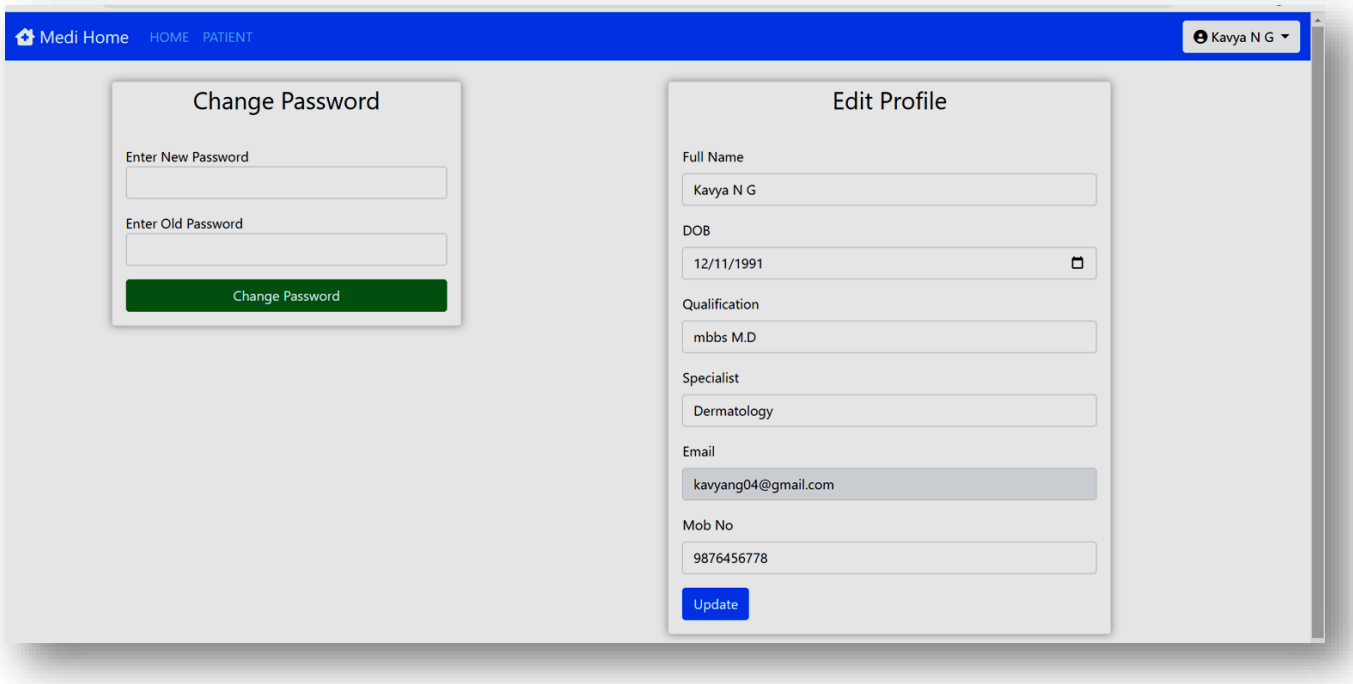
ii. Doctor Homepage



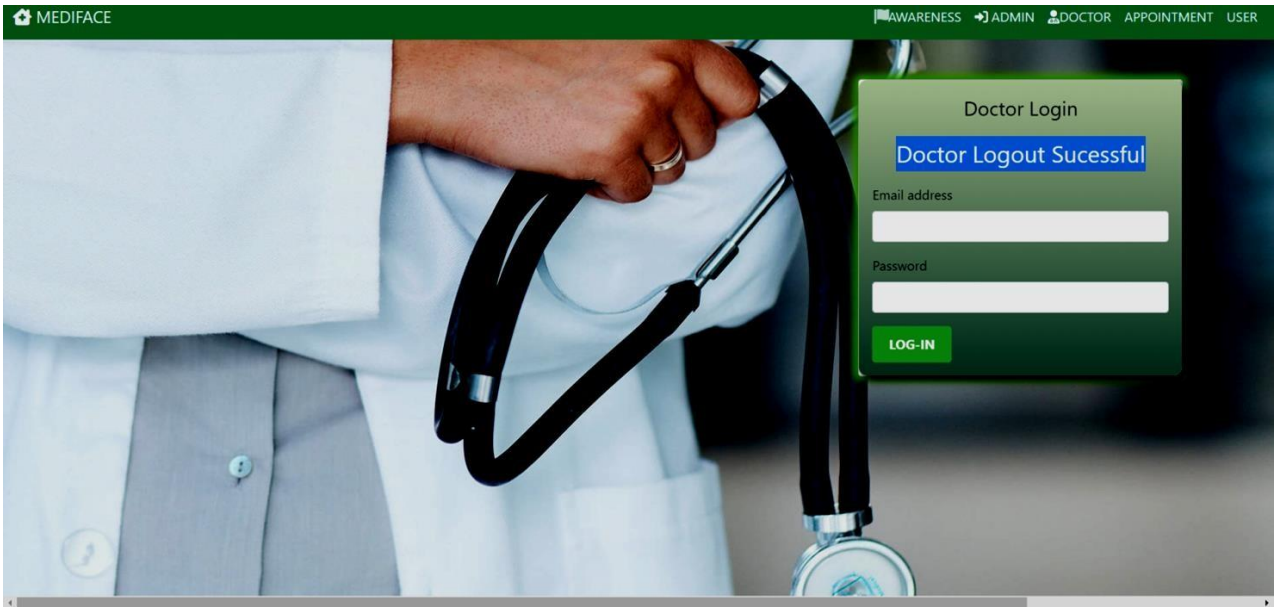
iii. View Appointment



iv. Edit profile and change password

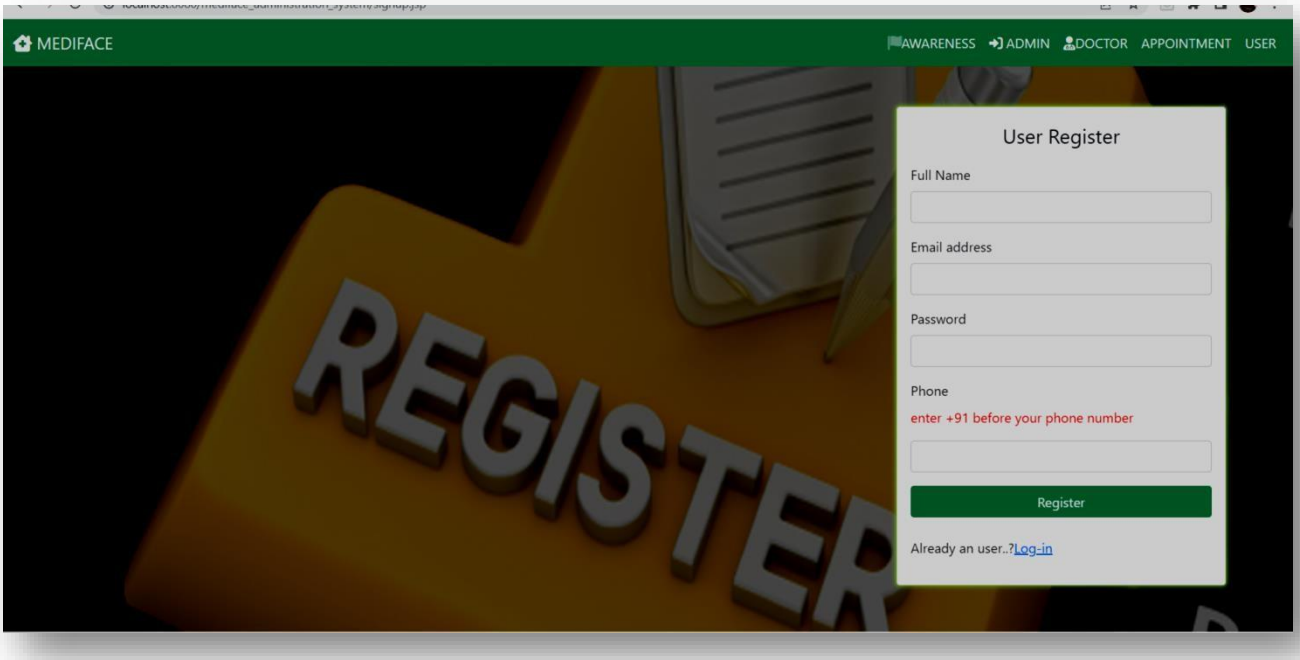


v. Log-out

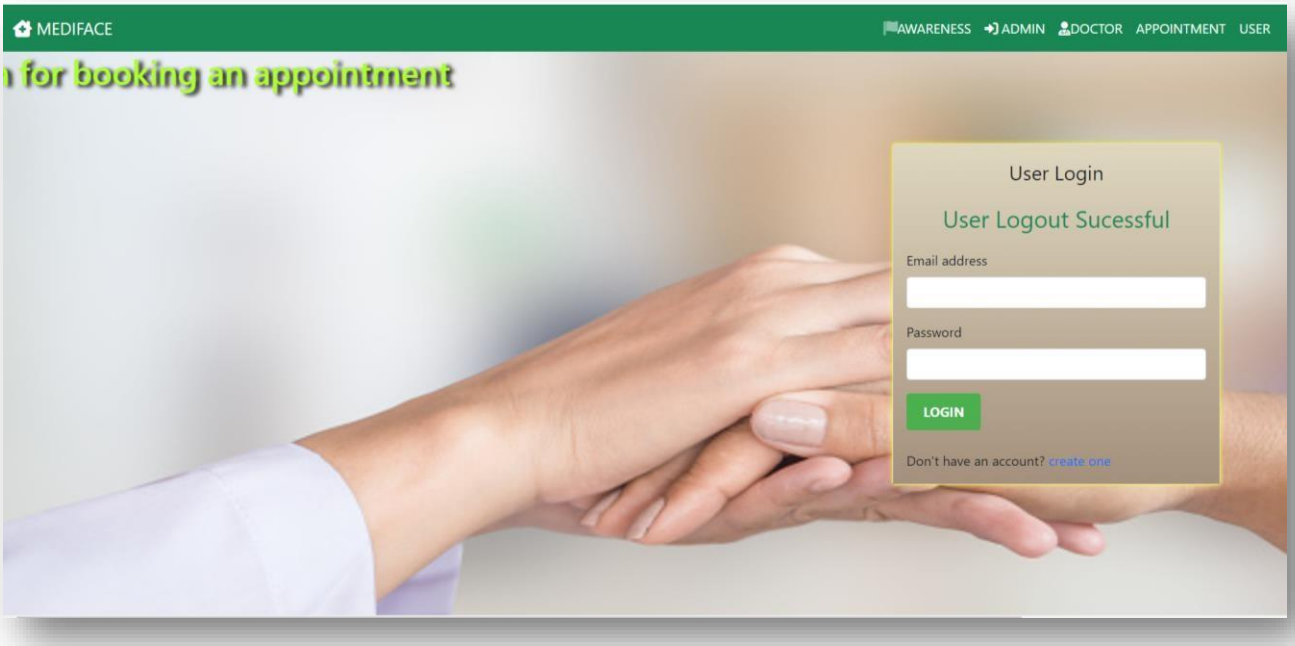


USER

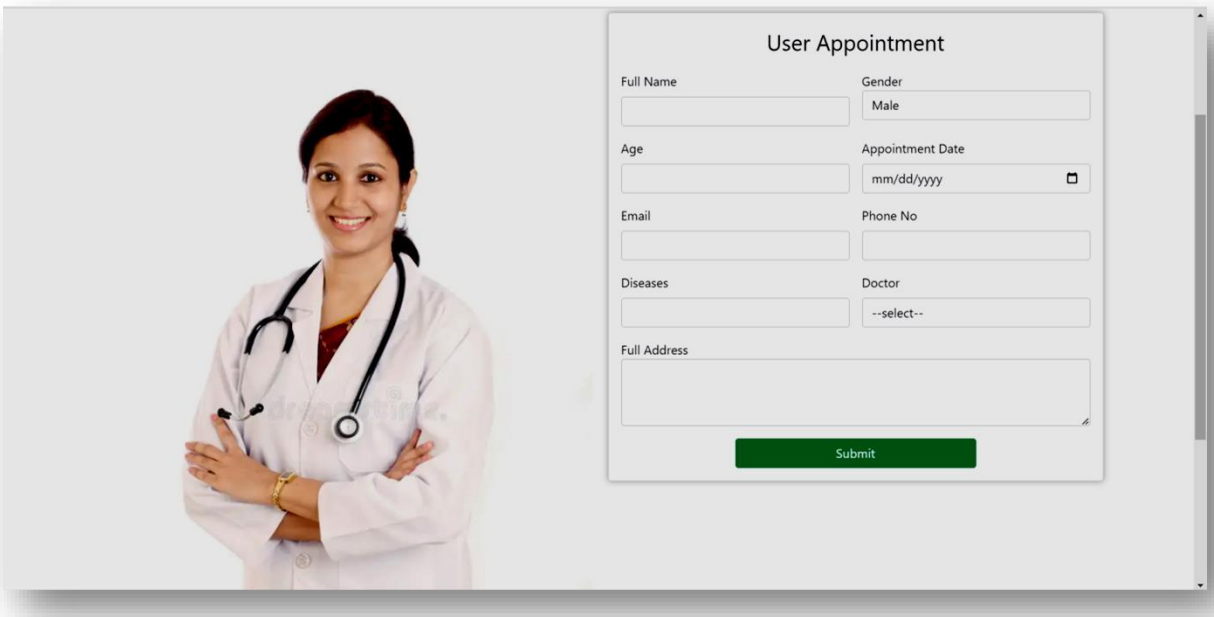
i. Register



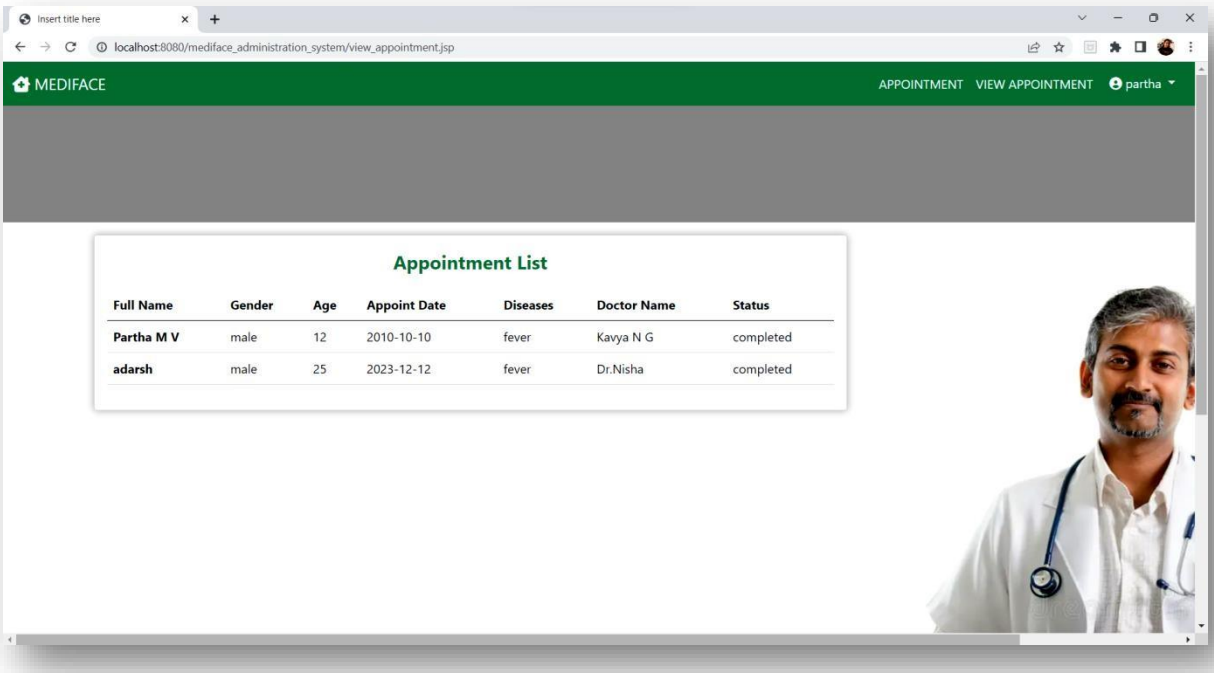
ii. Log-out



vii. Book an appointment

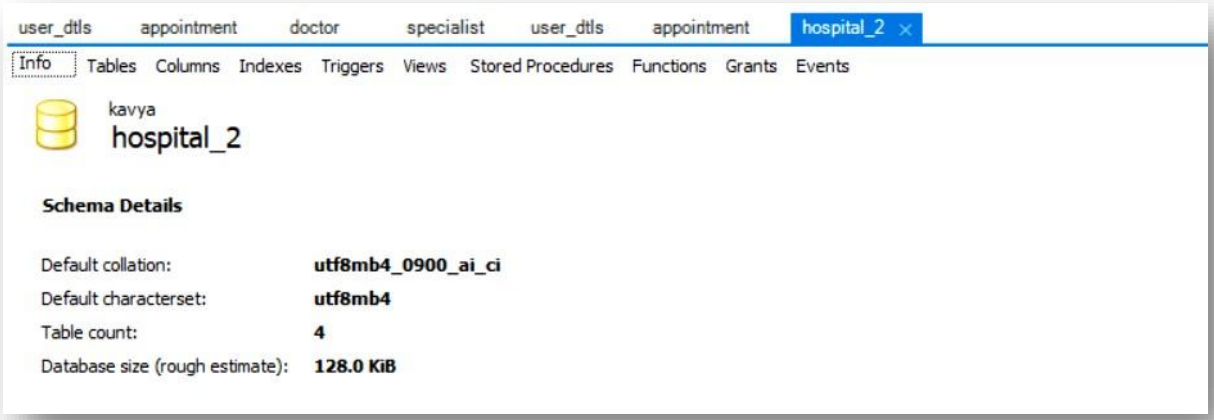


viii. View appointment

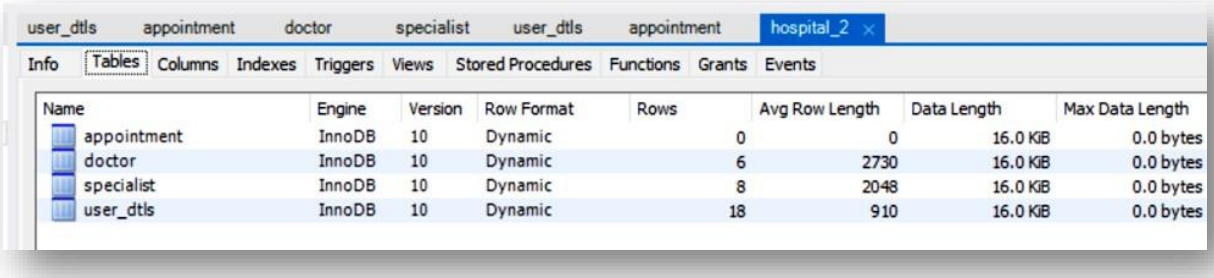


DATABASE

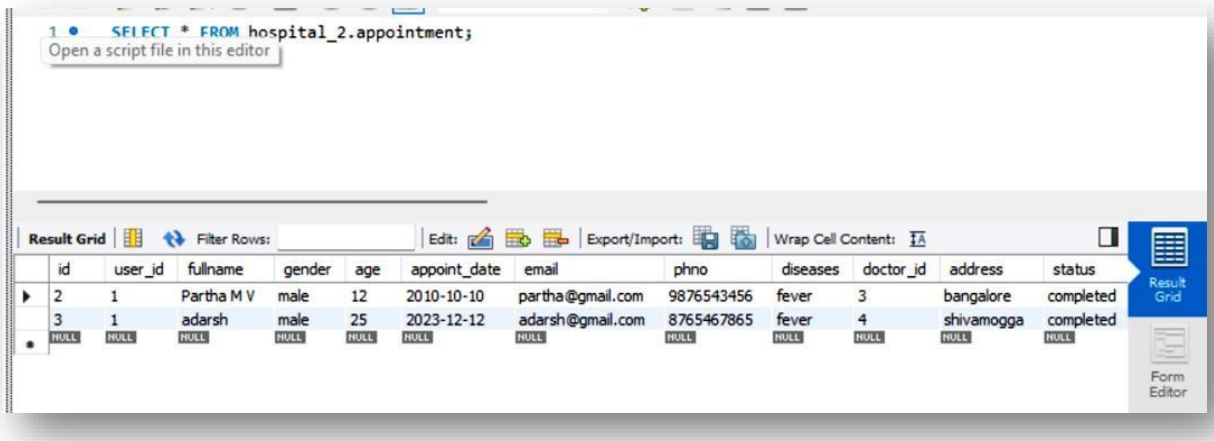
i. Database



ii. Tables



iii. Table details



CHAPTER – 6

FUTURESCOPE & CONCLUSION

FUTURE SCOPE

By adding an eye donation awareness module and prescription module to the Hospital Management System, you can extend its functionality and address additional aspects of patient care and community engagement. Here are some potential future scopes for these modules:

i. Eye Donation Awareness Module:

Educational Resources: Develop a comprehensive section dedicated to eye donation awareness, including information about the process, benefits, and myths associated with eye donation. Provide resources such as articles, videos, and FAQs to educate users and raise awareness. Interactive Campaigns: Conduct interactive campaigns and initiatives to promote eye donation. This can include quizzes, contests, and social media campaigns to engage users and encourage them to become eye donors.

Donor Pledge Tracking: Implement a feature where users can pledge to donate their eyes. Maintain a database of pledged donors, allowing users to track their pledges and receive updates and reminders about eye donation activities.

ii. Prescription Module:

Electronic Prescription Management: Enhance the prescription module to include features like e-prescriptions, where doctors can digitally generate and send prescriptions to pharmacies or patients. This can streamline the prescription process and reduce paper usage.

Integration with Pharmacy Systems: Integrate the prescription module with pharmacy systems to enable seamless transmission of prescriptions and facilitate accurate medication dispensing. This integration can help avoid errors in medication dosage and improve patient safety.

Prescription History and Alerts: Provide patients and healthcare providers with access to prescription history, enabling better medication management and adherence. Implement alerts for prescription renewals and reminders to take medications, enhancing patient engagement and medication compliance.

iii. Integration with Eye Banks and Organizations:

Collaborate with eye banks and organizations involved in eye donation: Establish integration with eye banks and organizations to facilitate the seamless exchange of information about eye donors, donation requests, and eye transplantation processes. This integration can enhance the efficiency of eye donation initiatives and promote a higher success rate.

These future scopes enhance the Mediface Administration System by incorporating eye donation awareness and prescription management functionalities. They contribute to improved patient care, community engagement, and the promotion of a positive impact on society.

CONCLUSION

Since we are entering details of the patients electronically in the "Mediface administration system", data will be secured. Using this application I can retrieve patient's history with a single click. Thus processing information will be faster. It guarantees accurate maintenance of Patient details. It easily reduces the book keeping task and thus reduces the human effort and increases accuracy speed.

The project was successfully completed after a lot of efforts and work hours. This project underwent number of compiling, debugging, removing errors, making it bug free, adding more facilities in Mediface Administration System and interactivity making it more reliable and useful.

There are also few features which can be integrated with this system to make it more flexible. Below list shows the future points to be consider:

1. Getting the current status of patient.
2. Including a different module for pharmacy, LAB, Bed Allotment and many more.
3. Including a eye donation Section and FAQ's.

Finally, I like to conclude that I had put all our efforts throughout the development of my project and tried to fulfill most of the requirements of the user.

REFERENCES

1. **HTML &CSS** https://www.w3schools.com/
2. **Bootstrap** http://getbootstrap.com/
3. https://stackoverflow.com
4. **MYSQL** https://www.javatpoint.com/MYSQL-tutorial
5. Abraham Silberschatz, Henry F. Korth and S. Sudarshan “Sixth Edition Database System Concepts released
6. https://www.geeksforgeeks.org/java-database-connectivity-with-MYSQL/
7. https://www.edx.org/course/cs50s-introduction-to-computer-science
8. https://www.freecodecamp.org/
9. https://www.codeconquest.com/