

Developing a Cloud-Based Application

Salam yaser ahmed Hammad 220201765
Hanan naser abdallah Abu Shaban 220191395
Malak ghassan basem Abu Sharar 220203511
Aya yaser Alkahoot 220200236
Software Development Department
Faculty of Information Technology
Islamic University of Gaza

A Requirement for the Course Advanced Software Engineering (SDEV 4304)
Instructor: Dr. Rebhi S. Baraka

Abstract

We developed a cloud-based healthcare management system (HMS) that is designed to streamline and automate the management of healthcare services. HMS offers a user-friendly interface for administrators, doctors, and patients to efficiently manage and track healthcare activities. It facilitates patient registration, uploading medical records, appointment scheduling, and interactive communication tools. With features like unique patient ID generation, appointment conflict resolution, and secure cloud-based data management, HMS ensures scalability and accessibility for healthcare providers of all sizes.

The system also incorporates advanced functionalities tailored to the roles of each user. Patients benefit from easy access to their medical history and seamless communication with healthcare providers. Doctors are equipped with tools to manage patient appointments, issue prescriptions, and send notifications, ensuring an efficient workflow. Administrators oversee the system's operations, including account management, billing, and secure data updates. Built on a robust SQL database structure, the HMS leverages modern cloud computing methodologies to offer a scalable, secure, and efficient solution. With integrated analytics, user activity tracking, and a well-documented deployment process, the system represents a comprehensive approach to improving healthcare management.

1. Introduction

The cloud-based Healthcare Management System (HMS) is a web application designed to streamline and automate the management of healthcare services. It serves as a centralized platform for patients, doctors, and administrators to collaborate efficiently, manage medical records, and track healthcare activities. HMS provides essential features such as patient registration, appointment scheduling with conflict resolution, uploading medical documents (e.g., lab reports and CT scans), and secure communication tools like notifications and emails. The system is tailored to address the needs of all user roles, ensuring an integrated and user-friendly experience.

To develop HMS, we followed a cloud-oriented software methodology, leveraging agile practices for iterative development, frequent feedback, and continuous collaboration. This approach enabled us to adapt to evolving system requirements while ensuring timely delivery of key features. The methodology focused on meeting the specified requirements, including secure cloud-based data storage, appointment management, and user activity tracking. Additionally, the deployment of HMS on a scalable and secure cloud platform ensures high availability, flexibility, and accessibility. By adhering to these practices, HMS effectively meets the demands of modern healthcare management, providing a reliable and comprehensive solution for administrators, doctors, and patients alike.

2. Cloud Software Application/Service Requirements

The cloud-based Healthcare Management System (HMS) is a cutting-edge web application designed to enhance the efficiency, scalability, and accessibility of healthcare services. The system serves as a centralized platform that streamlines processes for patients, doctors, and administrators. By leveraging secure cloud-based infrastructure, HMS ensures seamless collaboration among all users and provides robust tools for managing healthcare activities.

The application addresses key functionalities, including patient registration, secure storage of medical documents (e.g., lab reports and CT scans), conflict-free appointment scheduling, and real-time communication through notifications and emails. Each feature has been meticulously designed to meet the specific requirements outlined in the project specifications. For example, unique patient ID generation ensures a secure and streamlined identification process, while the administrator's exclusive data update rights guarantee data integrity and security. HMS also tracks all user activities to maintain transparency and provides analytics to support informed decision-making for healthcare providers.

HMS's User Stories:

For Patients:

1. Patient Registration

As a patient, I want to register on the platform and provide my medical details to access healthcare services.

Acceptance Criteria:

- The registration process should be user-friendly and intuitive.
- Patients should be able to provide necessary personal details and upload medical documents (e.g., lab reports, CT scans).
- Upon successful registration, patients should receive a unique ID and a confirmation notification.

2. Appointment Scheduling

As a patient, I want to schedule appointments with doctors without conflicts to receive timely medical care.

Acceptance Criteria:

- Patients should be able to select doctors based on specialty and availability.
- The system should resolve scheduling conflicts automatically to avoid overlapping appointments.
- Patients should receive notifications confirming their appointments.

3. Medical Document Upload

As a patient, I want to upload my medical documents so that doctors can access them for consultations.

Acceptance Criteria:

- Patients should be able to upload documents in various formats (e.g., PDF, images, videos).
- The system should securely store documents in the patient's profile and notify them of successful upload.

4. Notifications

As a patient, I want to receive notifications about appointment confirmations, approvals, and updates to stay informed.

Acceptance Criteria:

- Notifications should be sent via email and in-app messaging for all system events.

For Doctors:

1. Subscription and Registration

As a doctor, I want to subscribe and register by providing my necessary information to access the system's features.

Acceptance Criteria:

- Doctors must provide personal and professional details (e.g., license number, specialty).
- Registration is subject to administrator approval.

2. Specialty-Based Classification

As a doctor, I want my specialty to be recorded during registration so that patients can find the right doctor.

Acceptance Criteria:

- Doctors select their specialty (e.g., Dermatologist, Nephrologist) during registration.
- The system stores and displays specialties for patient searches and filtering.

3. Generate Prescriptions

As a doctor, I want to generate and share prescriptions for my patients to ensure they receive proper treatment.

Acceptance Criteria:

- Doctors can create prescriptions directly in the system.
- Prescriptions are securely stored and shared with patients.

4. Patient Management

As a doctor, I want to view and manage my assigned patients to provide effective medical care.

Acceptance Criteria:

- Doctors should have access to a dashboard displaying detailed patient profiles and medical histories.
- Tools should be available for doctors to add notes, prescribe treatments, and manage follow-ups.
- The system should allow doctors to track patient progress and update treatment plans.

4. Notifications to Patients

As a doctor, I want to send notifications to my patients to provide updates, reminders, or prescriptions.

Acceptance Criteria:

- Doctors should be able to compose and send notifications directly through the system.
- Notifications should be delivered securely via email .

5. Appointment Management and History

As a doctor, I want to view and manage appointments with my patients to organize my schedule efficiently and I want to access the history of appointments with my patients for better treatment planning.

Acceptance Criteria:

- Doctors should be able to view scheduled appointments and reschedule them if necessary.
- Notifications should be sent to patients about any changes in appointments.
- The system displays a log of past appointments with notes and outcomes.
- Appointment history is searchable and linked to patient profiles.

6. Cancel Appointment

As a doctor, I want to cancel appointments if necessary to manage my schedule effectively.

Acceptance Criteria:

- Doctors can cancel appointments for patients and send a notification that the appointment has been cancelled.

For Administrators:

1. Unique Patient ID Generation

As an administrator, I want to generate unique IDs for patients to ensure secure identification.

Acceptance Criteria:

- Unique patient IDs should be automatically generated upon successful registration.
- The IDs should be securely stored and linked to the corresponding patient profiles.

2. Activity Tracking

As an administrator, I want to track and log all user activities to ensure transparency and maintain system integrity.

Acceptance Criteria:

- The system should log user activities, such as registrations, appointments, and document uploads, with timestamps.
- Administrators should be able to access activity logs for auditing purposes.

3. Data Management

As an administrator, I want to manage and update data securely in the cloud to maintain system accuracy and security.

Acceptance Criteria:

- Only administrators should have the rights to modify or update cloud-stored data.
- Updates should be logged and verified for integrity.

4. Billing Management

As an administrator, I want to handle patient billing to ensure accurate payment processing.

Acceptance Criteria:

- Administrators should be able to generate invoices and track payments.
- Billing information should be securely stored and accessible for auditing.

HMS's UML Diagram:

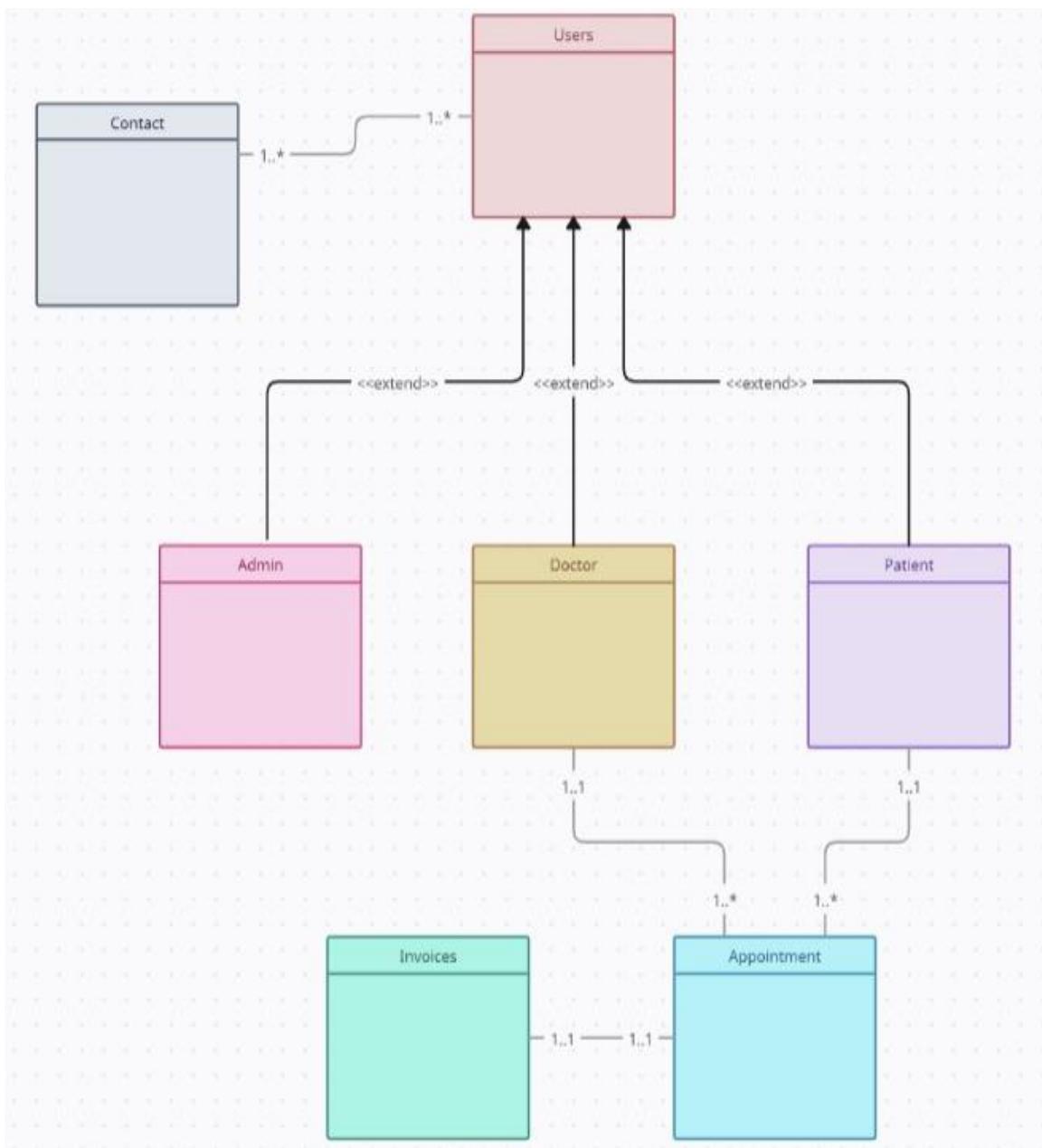


Figure 1- HMS's Componenr UML Diagram

HMS's Use Cases Scenario:

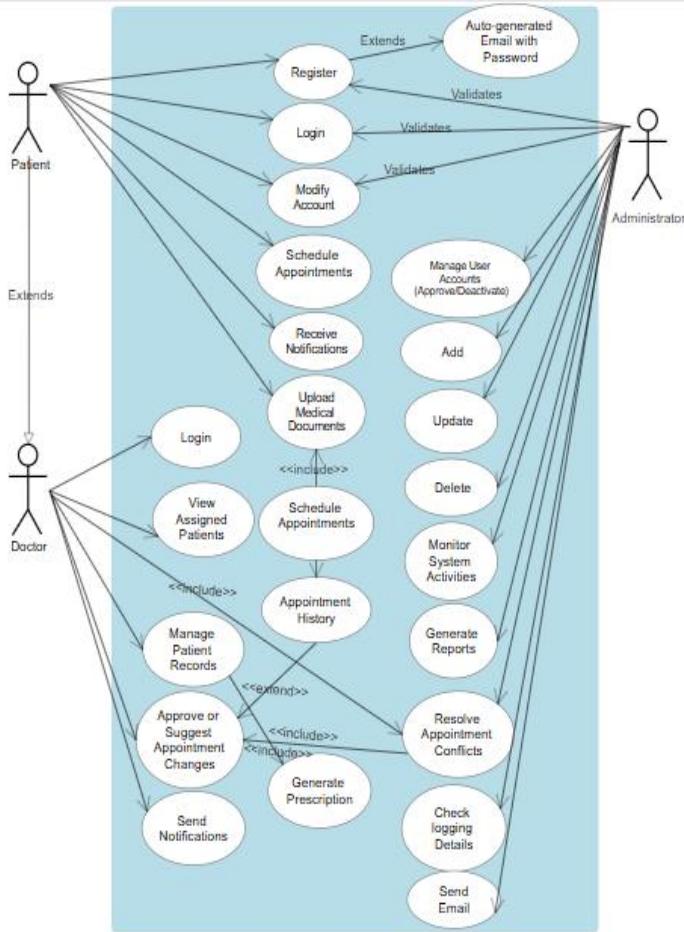


Figure 2- HMS's Use Case Scenario

Use Case 1: Patient Registration

Identifier	P01
Actors	Patient, Administrator, System
Goal	To enable patients to register and subscribe to the application.
Precondition	The patient accesses the application's registration page.
Postcondition	The patient is registered with a unique ID and can log in.
Main Scenario	<ol style="list-style-type: none"> 1. Patient navigates to the registration page. 2. Patient enters personal details and uploads necessary documents. 3. System validates the entered information. 4. Administrator verifies the documents. 5. Administrator approves the registration. 6. System generates a unique patient ID. 7. Patient receives a confirmation email with login credentials.

Use Case 2: Upload Medical Documents

Identifier	P02
Actors	Patient, System
Goal	To allow patients to upload medical files for doctor access.
Precondition	The patient is logged into the system.
Postcondition	Files are uploaded and associated with the patient's profile.
Main Scenario	<ol style="list-style-type: none"> 1. Patient navigates to the upload section. 2. Patient selects files to upload (e.g., CT scans, lab results). 3. System validates file formats and sizes. 4. System stores files in the cloud database. 5. Patient receives confirmation of a successful upload.

Use Case 3: Schedule Appointment

Identifier	P03
Actors	Patient, Doctor, System
Goal	To facilitate appointment scheduling between patients and doctors..
Precondition	The patient is logged in and has selected a doctor.
Postcondition	The appointment is scheduled without conflicts.
Main Scenario	<ol style="list-style-type: none"> 1. Patient navigates to the appointment section. 2. Patient selects a doctor and preferred time. 3. System checks for scheduling conflicts. 4. System confirms the availability. 5. Appointment details are stored and notifications are sent to both parties.

Use Case 4: Administrator Account Management

Identifier	A01
Actors	Administrator, System
Goal	To manage patient and doctor accounts.
Precondition	The administrator is logged in.
Postcondition	Account data is updated securely.
Main Scenario	<ol style="list-style-type: none"> 1. Administrator accesses the account management dashboard. 2. Administrator views the list of users (patients/doctors). 3. Administrator selects a user account to update. 4. Administrator edits account details and saves changes. 5. System updates the data in the cloud database.

Use Case 5: Doctor Subscription

Identifier	D01
Actors	Doctor, Administrator, System
Goal	To enable doctors to register and access the application.
Precondition	The doctor accesses the log in page.
Postcondition	The doctor is registered and can log in to the application.
Main Scenario	<ol style="list-style-type: none"> 1. Doctor navigates to the log in page. 2. Doctor enters personal details and specialization. 3. System validates the entered information. 4. Administrator verifies the details provided. 5. Administrator approves the registration.

Use Case 6: Manage Patient Records

Identifier	D02
Actors	Doctor, System
Goal	To allow doctors to view and manage their patient records.
Precondition	The doctor is logged into the system.
Postcondition	The patient records are accessed and updated as necessary.
Main Scenario	<ol style="list-style-type: none"> 1. Doctor logs into the system. 2. Doctor navigates to their patient list. 3. Doctor selects a patient's record. 4. Doctor reviews the medical history and uploads new diagnoses. 5. System updates the patient's record in the database.

Use Case 7: Cloud Data Backup and Security

Identifier	S01
Actors	System
Goal	To ensure data security and periodic backups.
Precondition	The system is operational.
Postcondition	Data integrity is maintained, and backups are securely stored.
Main Scenario	<ol style="list-style-type: none"> 1. System runs a scheduled backup task. 2. Data is encrypted and uploaded to secure storage. 3. Backup logs are generated and stored. 4. Administrator is notified of successful backups.

Use Case 8: Notification Management

Identifier	S02
Actors	System, Patient, Doctor
Goal	To send timely notifications to users.
Precondition	A trigger event occurs (e.g., appointment creation).
Postcondition	Notifications are sent and acknowledged by users.
Main Scenario	<ol style="list-style-type: none">1. Event occurs (e.g., appointment scheduled).2. System generates a notification message.3. System sends notifications via email and in-app.4. Users acknowledge the notification.

Requirements specifications for Healthcare Management System (HMS) :

Patient Functionality:

- Patients should be able to register on the platform and provide necessary personal and medical details.
- Patients should be able to upload required medical documents (e.g., lab reports, CT scans, prescriptions).
- Patients should be able to schedule appointments with doctors based on specialty and availability.
- Patients should receive notifications regarding their appointment status, document uploads, and updates from doctors.

Doctor Functionality:

- Doctors should be able to register and provide necessary information for accessing the system.
- Doctors should be able to view and manage assigned patients, including access to their medical records and uploaded documents.
- Doctors should manage patient care by prescribing treatments, updating progress, and scheduling follow-ups.
- Doctors should use a dashboard to manage appointments, reschedule if needed, and send notifications or emails to patients.

Administrator Functionality:

- Administrators should have access to a dashboard for reviewing and approving patient registrations and able to generate unique IDs for each registered patient.
- Administrators should be able to update and manage patient and doctor data stored in the system while tracking and logging all user activities for auditing and reporting purposes.
- Administrators should handle billing processes, including invoice generation and payment tracking.

System Functionality:

- The system should store and retrieve user data securely using a combination of SQL databases.
- The system should provide appointment management features for patients and doctors, including conflict resolution.
- The system should send notifications and emails to users based on specific events (e.g., registration approval, appointment confirmations).
- The system should record all user activities.
- The system should ensure data integrity and allow only authorized administrators to update records.

Performance and Security:

- The application should provide a seamless and responsive user experience across all devices.
- The system should be scalable to handle a large number of users and data efficiently.
- Appropriate security measures (e.g., encryption, access control) should be implemented to protect user data and ensure privacy.
- Regular data backups should be performed to prevent data loss.

Healthcare Management System (HMS) Works at Various Levels of Abstraction:

1. User Interaction Layer (Frontend):

The frontend provides the interface through which patients, doctors, and administrators interact with the Healthcare Management System (HMS). It is designed to be simple, responsive, and user-friendly to ensure a seamless experience for all users.

- **Patients:**

Patients access the system through a web interface where they can register, schedule appointments, upload medical documents (e.g., lab reports, CT scans), and receive notifications.

- **Doctors:**

Doctors use dashboards to view and manage patient records, track appointments, prescribe treatments, and send notifications.

- **Administrators:**

Administrators manage user accounts, monitor system activities and handle billing. They have access to specialized administrative panels.

Technology Stack:

- HTML5 and CSS3 for structuring and styling the web pages.
- JavaScript for client-side interactivity.
- Bootstrap for creating responsive and consistent designs.

2. Application Logic Layer (Backend):

The backend is responsible for processing all business logic, handling user requests, and managing communication with the database.

- **Core Functionalities:**
 - Scheduling appointments with conflict resolution to avoid overlaps.
 - Verifying patient registration and handling secure authentication.
 - Managing role-based access to features for patients, doctors, and administrators.
 - Handling notifications for events like appointment confirmations and updates.
- **APIs:**

PHP scripts are used to handle requests from the frontend and interact with the database for CRUD operations.

3. Data Management Layer :

The data layer ensures the secure storage and management of healthcare data, supporting both structured and unstructured formats.

- **Structured Data (SQL):**

A relational database (e.g., MySQL) is used to store patient details, appointment schedules, medical records metadata.
- **Security Measures:**
 - All data is encrypted during storage and transit.
 - Role-based access controls restrict unauthorized modifications.

4. Cloud Infrastructure Layer:

The cloud infrastructure ensures the scalability, reliability, and availability of the HMS, meeting the needs of both small clinics and larger healthcare organizations.

- **Hosting:**

The application is hosted on a cloud platform and a secure hosting provider, ensuring robust uptime and performance.

- **Load Balancing:**

Traffic is distributed across servers to handle peak loads and ensure smooth operation.

- **Backup and Recovery:**

Automated backups are scheduled to protect against data loss, and disaster recovery mechanisms are in place for quick restoration.

3. Software Architecture and Design

Based on the requirements of the Healthcare Management System (HMS), we decided to build a three-tier architecture system that separates the application into three distinct layers for enhanced scalability, maintainability, and performance.

The architecture includes:

1. Presentation Tier (Frontend):

Also known as the frontend, this layer is responsible for handling the user interface and displaying information to the users. It consists of an HTML-based frontend enhanced with CSS, JavaScript, and Bootstrap, enabling patients, doctors, and administrators to interact with HMS effectively.

2. Application Tier (Backend):

Also known as the backend, this layer encompasses the business logic and application processing. It includes a PHP-based backend that processes requests received from the frontend, interacts with the MySQL database, and returns responses to the users.

3. Data Tier (Database):

Also known as the database tier, this layer is responsible for storing and managing the data required by HMS. A MySQL database is used to store and retrieve data related to patients, doctors, administrators, appointments, and the overall system.

This three-tier design ensures modularity, scalability, and secure data handling, aligning with the requirements of the Healthcare Management System.

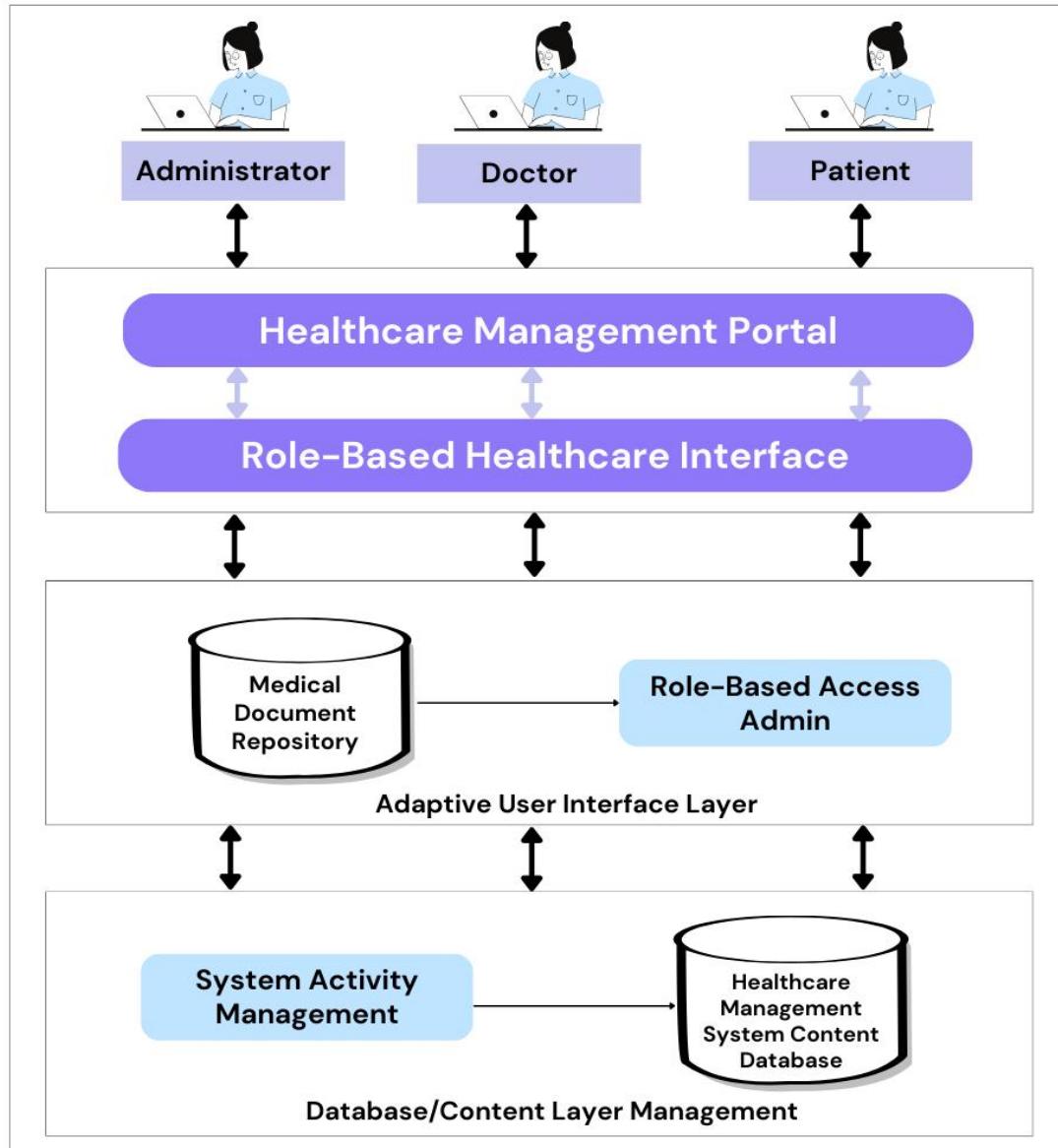


Figure 3- Abstract View of HMS's Architecture

Functional Components of HMS:

- The Healthcare Management System (HMS) provides user registration and login functionality to allow patients, doctors, and administrators to access their respective accounts securely.
- **Patients** can register on the system, upload medical documents such as lab reports and CT scans, view their medical history, and schedule appointments with their preferred doctors. They also receive notifications for appointment confirmations, reminders, and updates.
- **Doctors** have the ability to view and manage patient profiles, track appointments, prescribe treatments, and send notifications to patients. They can also update treatment plans and resolve scheduling conflicts.
- **Administrators** manage user accounts, monitor activities across the system. They ensure that patient data is securely stored and accessed only by authorized users.
- HMS integrates with external services for email notifications and data security protocols to protect sensitive healthcare information.
- HMS supports real-time scheduling functionality, ensuring that any conflicts in doctor or patient availability are automatically resolved.

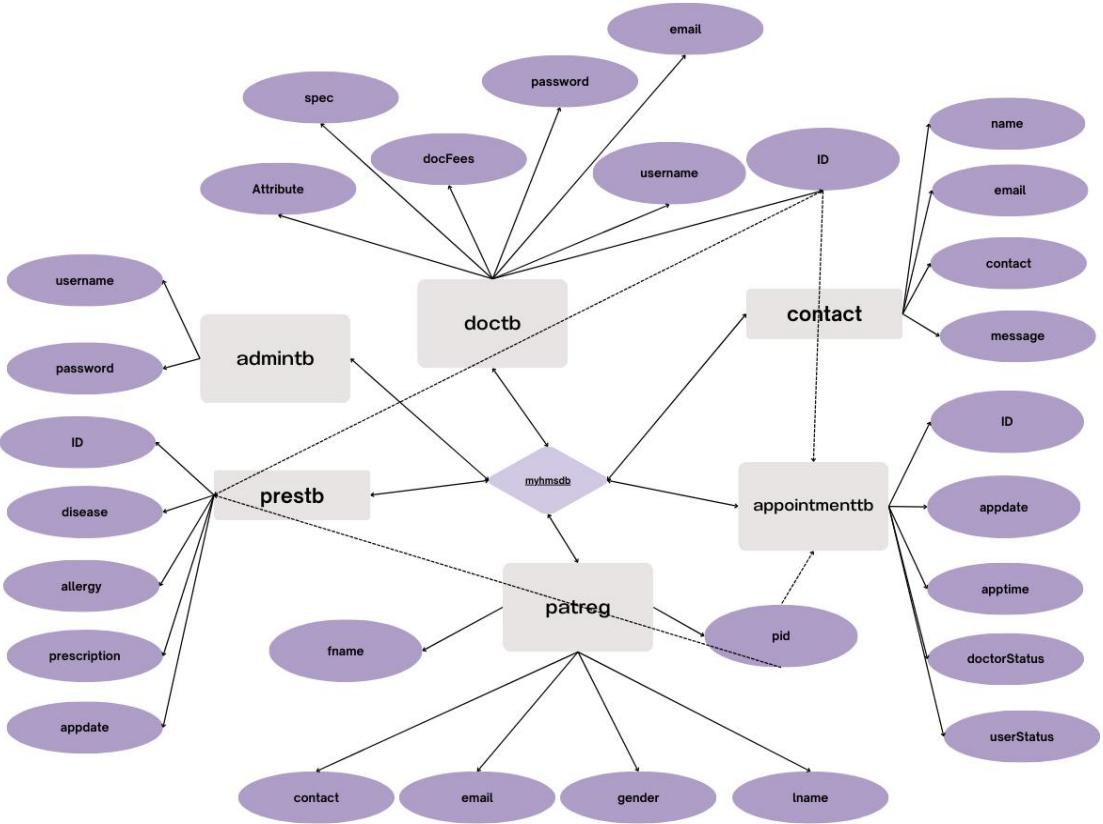
Real-World Constraints of AWS for HMS:

In AWS, deploying resources such as EC2 instances comes with its own set of challenges. By default, EC2 instances are assigned a public IPv4 address, but configuring the public IPv4 DNS address requires manual intervention through the AWS Management Console. This process can be complex and time-consuming, particularly for developers who are less familiar with AWS's configuration settings.

Additionally, managing instance-level settings, such as securing communication channels and configuring custom domains, may involve multiple steps that require a deep understanding of networking and AWS services. For example, while AWS provides Elastic Load Balancing and security groups for managing traffic and access, properly setting them up without disrupting the application's availability requires careful attention.

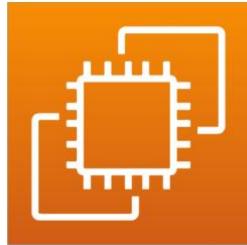
We believe that automating certain configurations, such as enabling DNS by default or offering simplified templates for common use cases, could make AWS more accessible for developers. Such improvements would streamline deployment workflows, enhance usability, and reduce the risk of misconfigurations for healthcare management systems like HMS.

HMS Entity Relationship Diagram



HMS's Entity Relationship Diagram

4. Used Cloud Services and its Interfaces



AWS EC2 Instance



AWS IAM

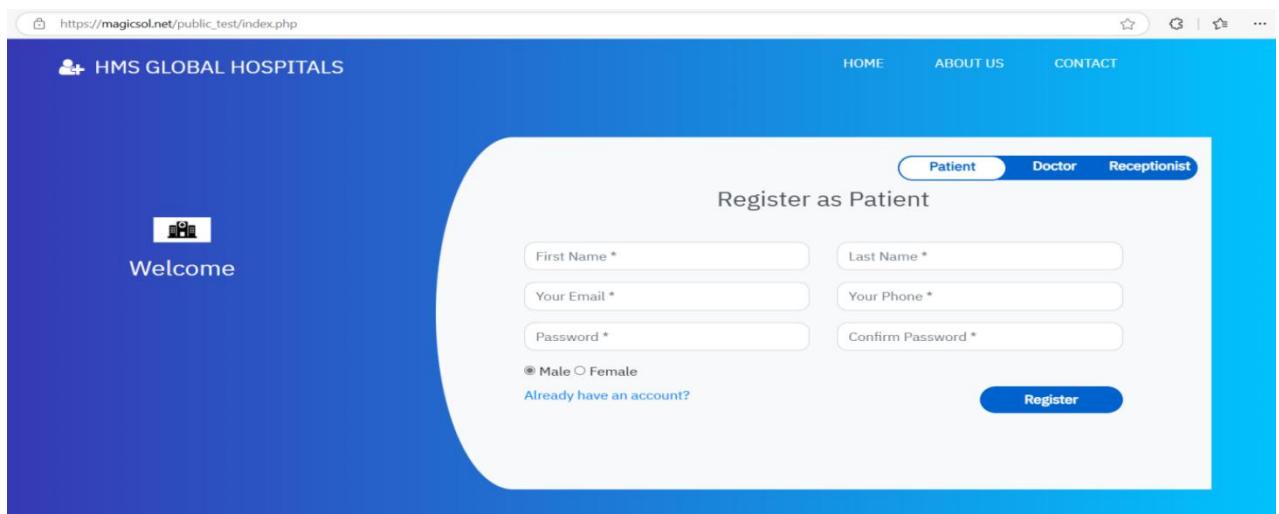
We utilized a range of AWS services to support the implementation and deployment of our system. These services included Amazon Relational Database Service (RDS), Amazon Elastic Compute Cloud (EC2) instances, security groups, and AWS Identity and Access Management (IAM). These services provided a reliable infrastructure for hosting, managing, and securing the application.

We leveraged the RDS service as our managed database solution, enabling us to store and access relational MySQL data efficiently and securely. The EC2 instances were employed to host the application, allowing us to deploy the components in a scalable and flexible environment that can handle varying workloads.

To ensure secure access, security groups were configured to manage inbound and outbound traffic, acting as virtual firewalls to control access to the instances and database. Additionally, AWS IAM was utilized to manage user roles and permissions, ensuring proper access control and authentication for various system users, including administrators and doctors.

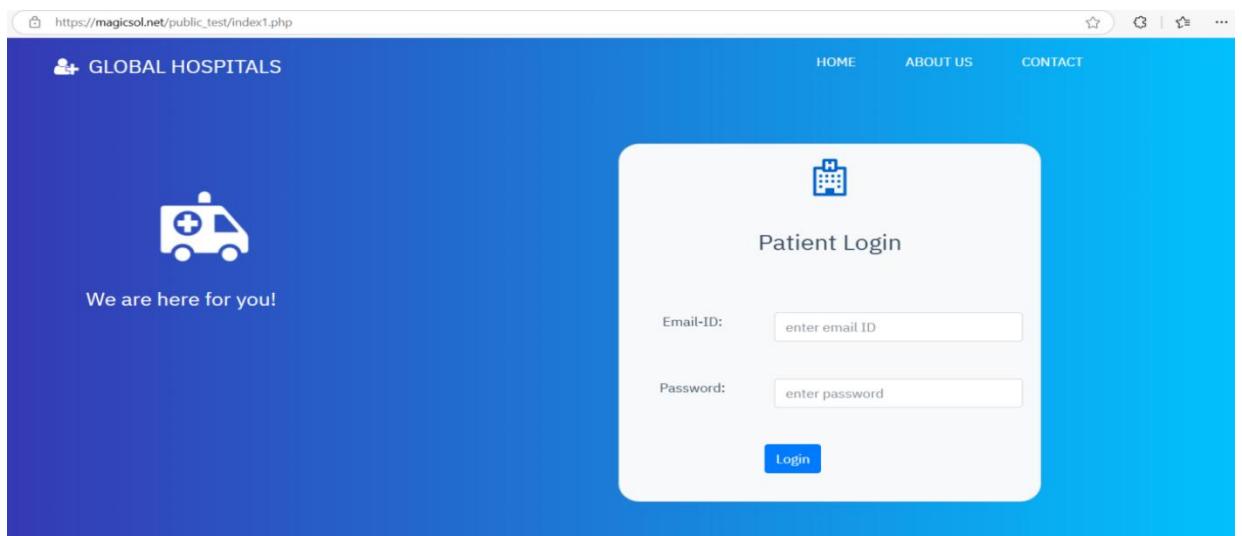
These AWS services collectively provided a scalable, secure, and reliable foundation for deploying and managing our system in the cloud environment. This architecture ensures efficient operation, robust data handling, and seamless communication within the application.

User Interfaces:



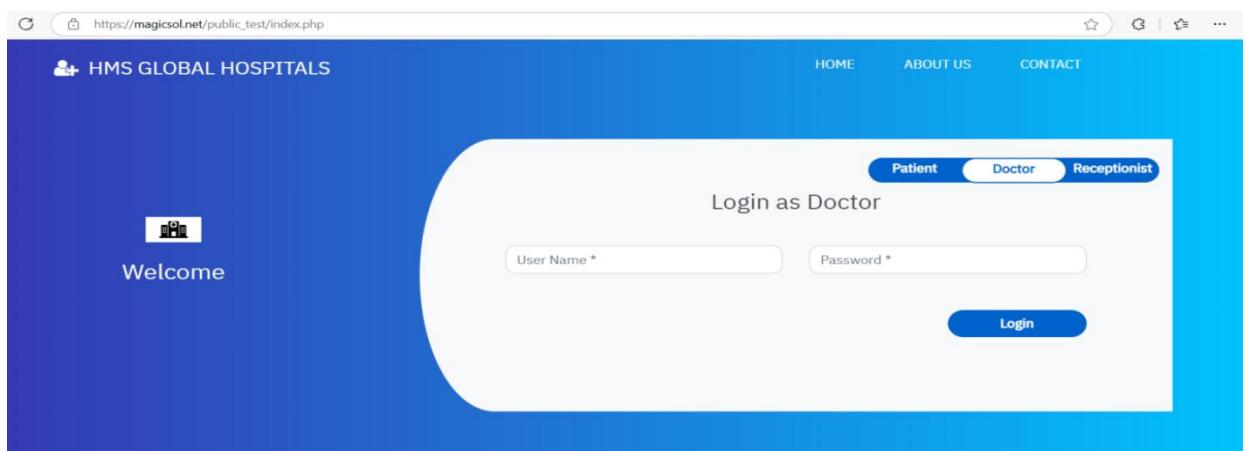
The screenshot shows a web browser window for the URL https://magicsol.net/public_test/index.php. The page title is "HMS GLOBAL HOSPITALS". The main content area features a "Welcome" message and a "Patient Login" form. A large, semi-transparent callout box is overlaid on the right side, containing a "Register as Patient" form. This register form includes fields for First Name, Last Name, Your Email, Your Phone, Password, and Confirm Password. It also includes gender selection (Male/Female) and a link for existing users. At the bottom right of the register form is a blue "Register" button.

The Register Interface for Patient



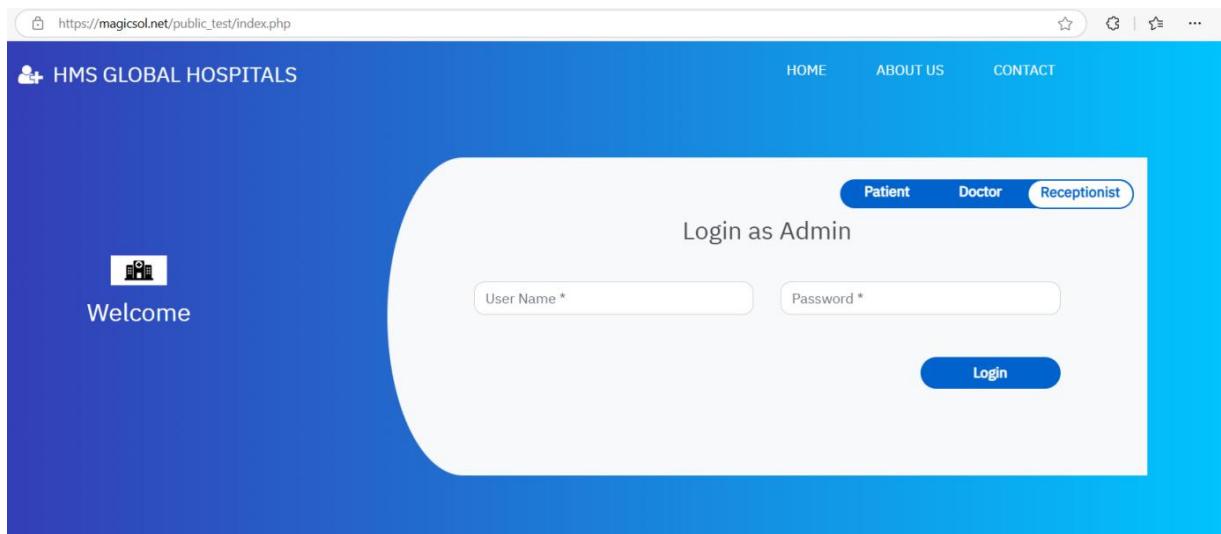
The screenshot shows a web browser window for the URL https://magicsol.net/public_test/index1.php. The page title is "GLOBAL HOSPITALS". The main content area features a "We are here for you!" message and a "Patient Login" form. A large, semi-transparent callout box is overlaid on the right side, containing a "Patient Login" form. This login form includes fields for Email-ID and Password, both with placeholder text "enter email ID" and "enter password" respectively. At the bottom right of the login form is a blue "Login" button.

The Login Interface for Patient



The screenshot shows a web browser window for the URL https://magicsol.net/public_test/index.php. The page title is "HMS GLOBAL HOSPITALS". The main content area features a "Welcome" message and a "Patient Login" form. A large, semi-transparent callout box is overlaid on the right side, containing a "Login as Doctor" form. This login form includes fields for User Name and Password, both with placeholder text "User Name *" and "Password *". At the bottom right of the login form is a blue "Login" button.

The Login Interface for Doctor



The Login Interface for Admin

A screenshot of the Patient Dashboard. The top navigation bar shows "Global Hospital" and "Logout". Below it, a welcome message says "Welcome Salam Hammad". The dashboard features a sidebar with "Dashboard", "Book Appointment", "Appointment History", and "Prescriptions". To the right, there are four main sections: "Book My Appointment" (with "Book Appointment" link), "My Appointments" (with "View Appointment History" link), "Prescriptions" (with "View Prescription List" link), and "Patient Data" (with "Patient Data Entry" link).

Patient Dashboard

A screenshot of the Patient Appointment Booking Page. The top navigation bar shows "Global Hospital" and "Logout". Below it, a welcome message says "Welcome Salam Hammad". The dashboard sidebar is identical to the previous screenshot. To the right, there is a large form titled "Create an appointment" with fields for Specialization (dropdown), Doctors (dropdown), Consultancy Fees (text input), Appointment Date (date input), Appointment Time (time input), and a "Create new entry" button.

Patient Appointment Booking Page

Global Hospital Logout

Welcome Salam Hammad

- Dashboard
- Book Appointment**
- Appointment History
- Prescriptions

Create an appointment

Specialization:	<input type="text" value="Internal Medicine"/>
Doctors:	<input type="text" value="Dinesh"/>
Consultancy Fees	<input type="text" value="700"/>
Appointment Date	<input type="text" value="01/15/2025"/>
Appointment Time	<input type="text" value="8:00 AM"/>

[Create new entry](#)

/admin-panel.php

localhost says

Select a time or date in the future!

OK

Select a time or date in the future!

Global Hospital Logout

Welcome Salam Hammad

- Dashboard
- Book Appointment
- Appointment History**
- Prescriptions

Doctor Name	Consultancy Fees	Appointment Date	Appointment Time	Current Status	Action
Ganesh	550	2020-02-14	10:00:00	Cancelled by Doctor	Cancelled
Dinesh	700	2020-02-28	10:00:00	Cancelled by You	Cancelled
Amit	1000	2020-02-19	03:00:00	Cancelled by You	Cancelled
ashok	500	2020-02-29	20:00:00	Active	<button style="background-color: red; color: white; border: none; padding: 2px 5px;">Cancel</button>
ashok	500	2025-01-22	12:00:00	Active	<button style="background-color: red; color: white; border: none; padding: 2px 5px;">Cancel</button>

Patient Appointment History

Welcome Brad Pitt

Dashboard	Appointment ID	Appointment Date	Appointment Time	Diseases	Allergies	Prescriptions	Bill Payment
Dinesh	16	2020-03-25	10:00:00	Cold	Nothing	Take some rest	Pay Bill

Prescription for the patient

04 Explaining the Dev X PMA localhost / 127.0.0.1 / myhmsdb / X Enter Patient Data X +

Download video from this page 2 X /indexbac.html

Enter Patient Data

Name:
Salam Hammad

Age:
22

Patient data:
Fever
Sore Throat

Upload medical files:
Choose Files salam.docx

Upload videos:
Choose Files Video.mp4

Send Data

Patient Information Entry Interface

Data Received:

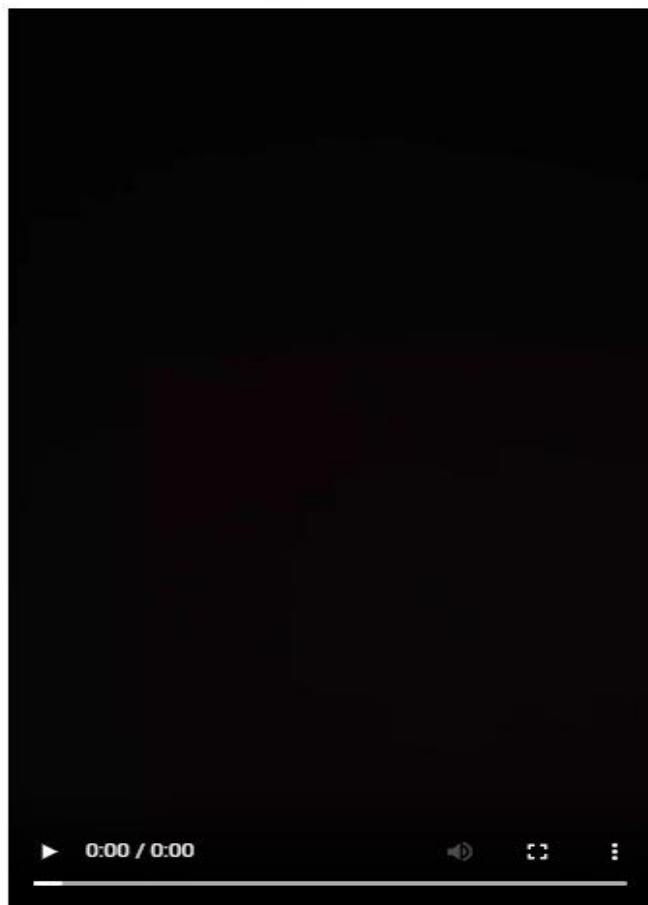
User Name: Salam Hammad
User Number:
Age: 22
Hospital Data: Fever Sore Throat

Uploaded files:

salam.docx

Uploaded Videos:

تحميل الفيديو من هذه الصفحة



[Return to home page](#)

The interface after entering the patient's data

Enter Patient Data +

master/indexbac.html

localhost says

Data has been successfully sent!

OK

Name:

Patient information has been successfully saved

Global Hospital [Logout](#)

Welcome ashok



[Dashboard](#)

[Appointments](#)

[Prescription List](#)



[View Appointments](#)

[Appointment List](#)



[Prescriptions](#)

[Prescription List](#)

Doctor Dashboard

Global Hospital [Logout](#)

Welcome ashok

Dashboard
Appointments
Prescription List

Patient ID	Appointment ID	First Name	Last Name	Gender	Email	Contact	Appointment Date	Appointment Time	Current Status	Action
11	4	Salam	Hammad	Female	salamhammad2003@gmail.com	9768946252	2020-02-29	20:00:00	Active	<button style="background-color: red; color: white; border: none; padding: 2px 5px;">Cancel</button>
1	15	Malak	Abu Sharar	Male	malak@gmail.com	9876543210	2024-12-28	10:00:00	Active	<button style="background-color: red; color: white; border: none; padding: 2px 5px;">Cancel</button>
1111	16	salam	Hammad	female	salamhammad2003@gmail.com	4754437347	2025-01-22	12:00:00	Active	<button style="background-color: red; color: white; border: none; padding: 2px 5px;">Cancel</button>

Doctor's Appointment Management Interface

Global Hospital [Logout](#)

localhost says

Are you sure you want to cancel this appointment ?

Welcome ashok

Dashboard
Appointments
Prescription List

Patient ID	Appointment ID	First Name	Last Name	Gender	Email	Contact	Appointment Date	Appointment Time	Current Status	Action
11	4	Salam	Hammad	Female	salamhammad2003@gmail.com	9768946252	2020-02-29	20:00:00	Active	<button style="background-color: red; color: white; border: none; padding: 2px 5px;">Cancel</button>
1	15	Malak	Abu Sharar	Male	malak@gmail.com	9876543210	2024-12-28	10:00:00	Active	<button style="background-color: red; color: white; border: none; padding: 2px 5px;">Cancel</button>
1111	16	salam	Hammad	female	salamhammad2003@gmail.com	4754437347	2025-01-22	12:00:00	Cancelled by You	Cancelled

Appointment Cancellation Confirmation

-panel.php?ID=15&cancel=update

localhost says

Your appointment successfully cancelled

This message confirms the successful cancellation of the selected appointment, providing reassurance to the user that the action has been completed


Global Hospital
Logout
Enter contact number
Search

Welcome ashok

Dashboard	Patient ID	First Name	Last Name	Appointment ID	Appointment Date	Appointment Time	Disease	Allergy	Prescribe	Send Email
Appointments	11	Shraddha	Kapoor	4	2020-02-29	20:00:00	vfrggggg	rerrrrrr	rrrrrrr	<button>Send</button>
Prescription List	1	Ram	Kumar	15	2024-12-28	10:00:00	fbhbhjbehbjhf	jrrjhrhrhrhr	hrhjijrjkjk	<button>Send</button>
	1111	salam	ab	16	2025-01-22	12:00:00	???? ?????	???? ?????	???? ?????	<button>Send</button>

Doctor's Prescription Management Interface


Global Hospital
Logout
Enter contact number
Search

Send Email

To: salamhammad2003@gmail.com

Subject: Follow-Up Appointment Confirmation

Message:

Dear Salam Hammad,

I hope this message finds you well. I wanted to remind you about your upcoming follow-up appointment scheduled for January 15, 2025, at 10:00 AM. This session will focus on reviewing your progress and addressing any concerns or symptoms you might be experiencing.

Please let me know if you need to reschedule or if there's anything specific you'd like to discuss during the appointment.

Looking forward to seeing you then.

Best regards,

Dr. Ashok

Internal Medicine

Close **Send Email**

Activate Windows
Go to Settings to activate Windows.

Send Email Feature


Global Hospital
Logout

WELCOME RECEPTIONIST

Dashboard

- [Doctor List](#)
- [Patient List](#)
- [Appointment Details](#)
- [Prescription List](#)
- [Add Doctor](#)
- [Delete Doctor](#)
- [Queries](#)



Doctor List

[View Doctors](#)



Patient List

[View Patients](#)



Appointment Details

[View Appointments](#)



Prescription List

[View Prescriptions](#)



Manage Doctors

[Add Doctors | Delete Doctors](#)

Admin Dashboard

WELCOME RECEPTIONIST

Dashboard
Doctor List
Patient List
Appointment Details
Prescription List
Add Doctor
Delete Doctor
Queries

Doctor Name	Specialization	Email	Password	Fees
ashok	Internal Medicine	ashok@gmail.com	ashok123	500
arun	Dermatology	arun@gmail.com	arun123	600
Dinesh	Internal Medicine	dinesh@gmail.com	dinesh123	700
Ganesh	Pediatrics	ganesh@gmail.com	ganesh123	550
Kumar	Pediatrics	kumar@gmail.com	kumar123	800
Abbis	Dermatology	abbis@gmail.com	abbis123	1500
Tiwary	Pediatrics	tiwary@gmail.com	tiwary123	450
amit	Dermatology	amit@gmail.com	amit123	1000

Activate Windows
Go to Settings to activate Windows.

This interface provides the receptionist with a comprehensive list of all doctors and their details

WELCOME RECEPTIONIST

Dashboard
Doctor List
Patient List
Appointment Details
Prescription List
Add Doctor
Delete Doctor
Queries

Patient ID	First Name	Last Name	Gender	Email	Contact	Password
1	Ram	Kumar	Male	ram@gmail.com	9876543210	ram123
2	Alia	Bhatt	Female	alia@gmail.com	8976897689	alia123
3	Shahrukh	khan	Male	shahrukh@gmail.com	8976898463	shahrukh123
4	Kishan	Lal	Male	kishansmart0@gmail.com	8838489464	kishan123
5	Gautam	Shankaram	Male	gautam@gmail.com	9070897653	gautam123
6	Sushant	Singh	Male	sushant@gmail.com	9059986865	sushant123
7	Aya	Alkhloot	Female	aya@gmail.com	9128972454	aya1234
8	Kenny	Sebastian	Male	kenny@gmail.com	9809879868	kenny123
9	William	Blake	Male	william@gmail.com	8683619153	william123

Activate Windows
Go to Settings to activate Windows.

This interface provides the receptionist with a comprehensive list of all patients and their details

Global Hospital [Logout](#)

WELCOME RECEPTIONIST

Patient Details										Appointment Status		
Appointment ID	Patient ID	First Name	Last Name	Gender	Email	Contact	Doctor Name	Consultancy Fees	Appointment Date	Appointment Time	Appointment Status	
1	4	Salam	Hammad	Male	salamhammad2003@gmail.com	8838489464	Ganesh	550	2020-02-14	10:00:00	Cancelled by Doctor	
2	4	Salam	Hammad	Male	salamhammad2003@gmail.com	8838489464	Dinesh	700	2020-02-28	10:00:00	Cancelled by Patient	
3	4	Salam	Hammad	Male	salamhammad2003@gmail.com	8838489464	Amit	1000	2020-02-19	03:00:00	Cancelled by Patient	
4	11	Salam	Hammad	Female	salamhammad2003@gmail.com	9768946252	ashok	500	2020-02-29	20:00:00	Active	
5	4	Hanan	Abu Shaban	Male	hanan@gmail.com	8838489464	Dinesh	700	2020-02-28	12:00:00	Active	
6	4	Hanan	Abu Shaban	Male	hanan@gmail.com	8838489464	Ganesh	550	2020-02-26	15:00:00	Cancelled by Patient	
8	2	Hanan	Abu Shaban	Female	hanan@gmail.com	8976897689	Ganesh	550	2020-03-21	10:00:00	Active	

This interface provides the receptionist with a detailed overview of all patient appointments

Global Hospital [Logout](#)

WELCOME RECEPTIONIST

Prescription List									
Doctor	Patient ID	Appointment ID	First Name	Last Name	Appointment Date	Appointment Time	Disease	Allergy	Prescription
Dinesh	4	11	Kishan	Lal	2020-03-27	15:00:00	Cough	Nothing	Just take a teaspoon of Benadryl every night
Ganesh	2	8	Alia	Bhatt	2020-03-21	10:00:00	Severe Fever	Nothing	Take bed rest
Kumar	9	12	William	Blake	2020-03-26	12:00:00	Sever fever	nothing	Paracetamol -> 1 every morning and night
Tiwary	9	13	William	Blake	2020-03-26	14:00:00	Cough	Skin dryness	Intake fruits with more water content
ashok	11	4	Shraddha	Kapoor	2020-02-29	20:00:00	vfrggggg	rerrrrrr	Activate Windows Go to Settings to activate Windows.

This interface provides the receptionist with a comprehensive list of patient prescriptions

Global Hospital [Logout](#)

WELCOME RECEPTIONIST

<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Dashboard </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Doctor List </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Patient List </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Appointment Details </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Prescription List </div> <div style="background-color: #0070C0; color: white; border: 1px solid #0070C0; padding: 5px; text-decoration: none; font-weight: bold;"> Add Doctor </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Delete Doctor </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Queries </div>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Doctor Name:</td> <td><input type="text" value="Salam"/></td> </tr> <tr> <td>Specialization:</td> <td><input type="text" value="Cardiologist"/></td> </tr> <tr> <td>Email ID:</td> <td><input type="text" value="salamhammad@gmail.com"/></td> </tr> <tr> <td>Password:</td> <td><input type="password" value="*****"/></td> </tr> <tr> <td>Confirm Password:</td> <td><input type="password" value="*****"/> Matched</td> </tr> <tr> <td>Consultancy Fees:</td> <td><input type="text" value="1000"/></td> </tr> </table> <p style="text-align: center;">Add Doctor</p>	Doctor Name:	<input type="text" value="Salam"/>	Specialization:	<input type="text" value="Cardiologist"/>	Email ID:	<input type="text" value="salamhammad@gmail.com"/>	Password:	<input type="password" value="*****"/>	Confirm Password:	<input type="password" value="*****"/> Matched	Consultancy Fees:	<input type="text" value="1000"/>
Doctor Name:	<input type="text" value="Salam"/>												
Specialization:	<input type="text" value="Cardiologist"/>												
Email ID:	<input type="text" value="salamhammad@gmail.com"/>												
Password:	<input type="password" value="*****"/>												
Confirm Password:	<input type="password" value="*****"/> Matched												
Consultancy Fees:	<input type="text" value="1000"/>												

This interface allows the receptionist to add a new doctor by entering their details

/admin-panel1.php

localhost says

Doctor added successfully!

OK

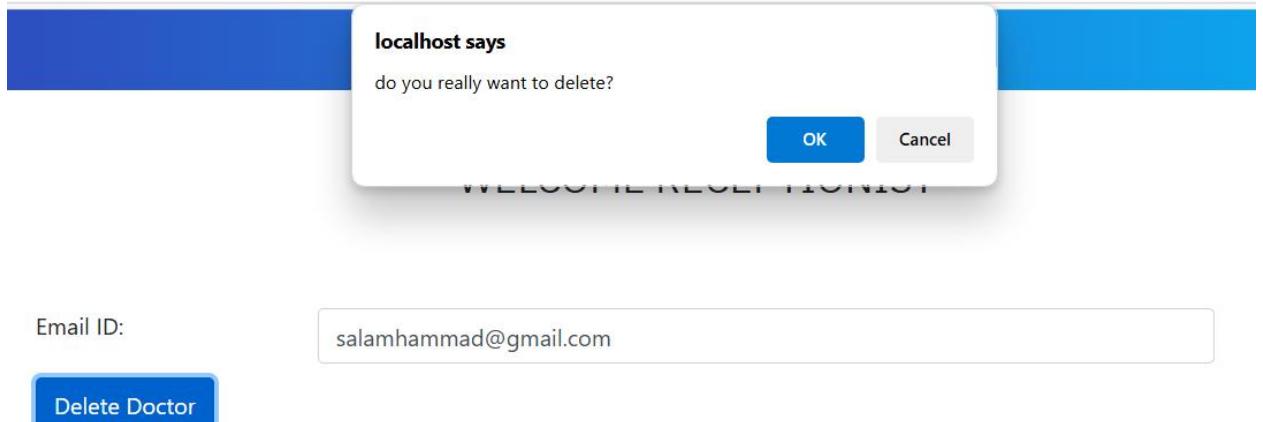
Doctor successfully added by the administrator!

Global Hospital [Logout](#)

WELCOME RECEPTIONIST

<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Dashboard </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Doctor List </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Patient List </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Appointment Details </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Prescription List </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Add Doctor </div> <div style="background-color: #0070C0; color: white; border: 1px solid #0070C0; padding: 5px; text-decoration: none; font-weight: bold;"> Delete Doctor </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Queries </div>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Email ID:</td> <td><input type="text" value="salamhammad@gmail.com"/></td> </tr> </table> <p style="text-align: center;">Delete Doctor</p>	Email ID:	<input type="text" value="salamhammad@gmail.com"/>
Email ID:	<input type="text" value="salamhammad@gmail.com"/>		

This interface allows the receptionist to delete a doctor's record by entering their email ID



This alert prompts the receptionist to confirm the deletion of a doctor's record before proceeding

admin-panel1.php



Doctor successfully removed by the administrator!

A screenshot of a web application interface. At the top, there is a blue header bar with the text "Global Hospital" and "Logout". Below it, the main content area has a white background. On the left side, there is a sidebar menu with the following items: Dashboard, Doctor List, Patient List, Appointment Details, Prescription List, Add Doctor, Delete Doctor, and Queries. The "Queries" item is highlighted with a blue background. In the center, there is a table with the following columns: "User Name", "Email", "Contact", and "Message". The table contains the following data:

User Name	Email	Contact	Message
Anu	anu@gmail.com	7896677554	Hey Admin
Viki	viki@gmail.com	9899778865	Good Job, Pal
Ananya	ananya@gmail.com	9997888879	How can I reach you?
Aakash	aakash@gmail.com	8788979967	Love your site
Mani	mani@gmail.com	8977768978	Want some coffee?
Karthick	karthi@gmail.com	9898989898	Good service
Abbis	abbis@gmail.com	8979776868	Love your service
Asiq	asiq@gmail.com	9087897564	Love your service. Thank you!
Jane	jane@gmail.com	7869869757	I love your service!

This interface displays a list of queries submitted by users, including their names, contact details, and messages, allowing the receptionist to review and respond effectively

Global Hospital [Logout](#)

WELCOME RECEPTIONIST

Dashboard	<input type="text" value="0000000000"/>	Search	
User Name	Email	Contact	Message
Anu	anu@gmail.com	7896677554	Hey Admin
Viki	viki@gmail.com	9899778865	Good Job, Pal
Ananya	ananya@gmail.com	9997888879	How can I reach you?
Aakash	aakash@gmail.com	8788979967	Love your site
Mani	mani@gmail.com	8977768978	Want some coffee?
Karthick	karti@gmail.com	9898989898	Good service
Abbis	abbis@gmail.com	8979776868	Love your service

User Name	Email	Contact	Message
Salam	salamhammad2003@gmail.com	0000000000	Coooo

[Back to your Dashboard](#)

This interface displays the search result related to the user whose details were entered

GLOBAL HOSPITALS [HOME](#) [ABOUT US](#) [CONTACT](#)

Drop Us a Message

[Send Message](#)

Activate Windows
Go to Settings to activate Windows.

Contact Page

/logout1.php

You have logged out.

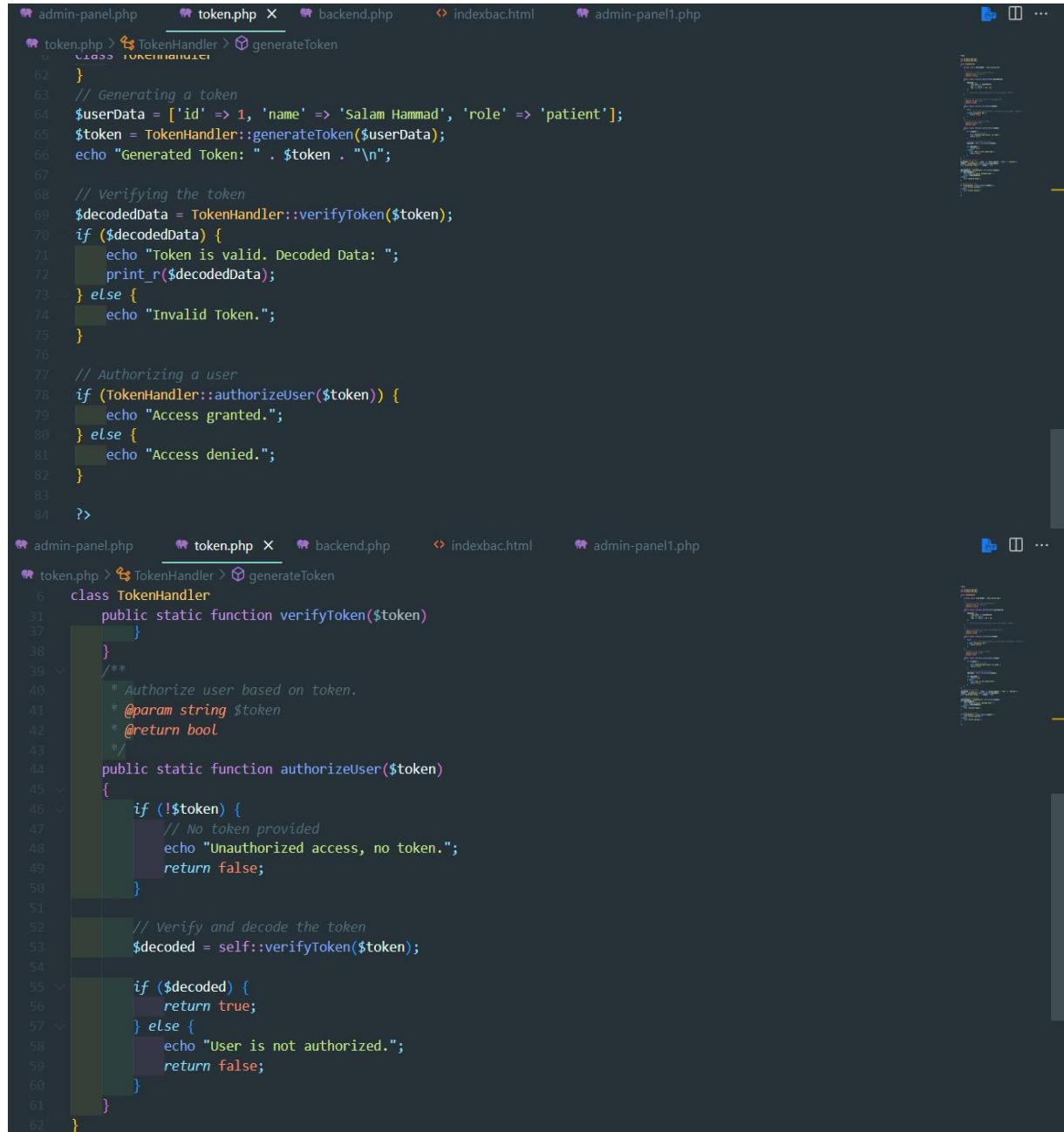
[Back to Home Page](#)

Logout Page

5. Implementation

In this section, we will preview some of the most important code and the logic behind.

Please feel free to check out all the source code of TMS through the provided GitHub link in the references.



```
admin-panel.php      token.php X      backend.php      indexbac.html      admin-panel1.php
token.php > TokenHandler > generateToken
  FUNCTION [F11]
62 }
63 // Generating a token
64 $userData = ['id' => 1, 'name' => 'Salam Hammad', 'role' => 'patient'];
65 $token = TokenHandler::generateToken($userData);
66 echo "Generated Token: " . $token . "\n";
67
68 // Verifying the token
69 $decodedData = TokenHandler::verifyToken($token);
70 if ($decodedData) {
71     echo "Token is valid. Decoded Data: ";
72     print_r($decodedData);
73 } else {
74     echo "Invalid Token.";
75 }
76
77 // Authorizing a user
78 if (TokenHandler::authorizeUser($token)) {
79     echo "Access granted.";
80 } else {
81     echo "Access denied.";
82 }
83
84 ?>
admin-panel.php      token.php X      backend.php      indexbac.html      admin-panel1.php
token.php > TokenHandler > generateToken
  CLASS [F11]
6 class TokenHandler
31     public static function verifyToken($token)
32     {
33     }
34
35 /**
36 * Authorize user based on token.
37 * @param string $token
38 * @return bool
39 */
40 public static function authorizeUser($token)
41 {
42     if (!$token) {
43         // No token provided
44         echo "Unauthorized access, no token.";
45         return false;
46     }
47
48     // Verify and decode the token
49     $decoded = self::verifyToken($token);
50
51     if ($decoded) {
52         return true;
53     } else {
54         echo "User is not authorized.";
55         return false;
56     }
57 }
58
59 }
60
61 }
62 }
```

Token Generation, Verification & Authorization

admin-panel.php index.php ✘ backend.php indexbac.html admin-panel1.php

index.php > html > body > div.container.register > div.row > div.col-md-9.register-right > div#myTabContent.tab-content > div#home.tab-pane.fade

```
1
44
70     container register" style="font-family: 'IBM Plex Sans', sans-serif;">
71         <div class="row">
72             <div class="col-md-9 register-right" style="margin-top: 40px; left: 80px;">
73                 <div class="tab-pane fade show active" id="home" role="tabpanel" aria-labelledby="home-tab">
74                     <h3 class="register-heading">Register as Patient</h3>
75                     <form method="post" action="func2.php">
76                         <div class="row register-form">
77
78                             <div class="col-md-6">
79                                 <div class="form-group">
80                                     <input type="text" class="form-control" placeholder="First Name *" name="fname" onkeydown="return alphaOnly(event);;" required/>
81                                 </div>
82                                 <div class="form-group">
83                                     <input type="email" class="form-control" placeholder="Your Email *" name="email"/>
84                                 </div>
85                                 <div class="form-group">
86                                     <input type="password" class="form-control" placeholder="Password *" id="password" name="password" onkeyup='check();' required/>
87                                 </div>
88
89                                 <div class="form-group">
90                                     <div class="maxl">
91                                         <label class="radio inline">
92                                             <input type="radio" name="gender" value="Male" checked>
93                                             <span> Male </span>
94                                         </label>
95                                         <label class="radio inline">
96                                             <input type="radio" name="gender" value="Female">
97                                             <span>Female </span>
98                                         </label>
99                                     </div>
100                                     <a href="index1.php">Already have an account?</a>
101                                 </div>
102                             </div>
103                         </div>
104                     </form>
105                 </div>
106             </div>
107         </div>
108     </div>
109
110     <div class="col-md-6">
111         <div class="form-group">
112             <input type="text" class="form-control" placeholder="Last Name *" name="lname" onkeydown="return alphaOnly(event);;" required/>
113         </div>
114         <div class="form-group">
115             <input type="tel" minlength="10" maxlength="10" name="contact" class="form-control" placeholder="Your Phone *" />
116         </div>
117         <div class="form-group">
118             <input type="password" class="form-control" id="cpassword" placeholder="Confirm Password *" name="cpassword" onkeyup='check();' required/><span id='message'></span>
119         </div>
120         <input type="submit" class="btnRegister" name="patsub1" onclick="return checklen();;" value="Register"/>
121     </div>
122 </div>
123 </form>
```

0 🔍 BLACKBOX Chat Add Logs 🔍 CyberCoder Improve Code Share Code Link Ln 93, Col 37 Spaces: 4 UTF-8

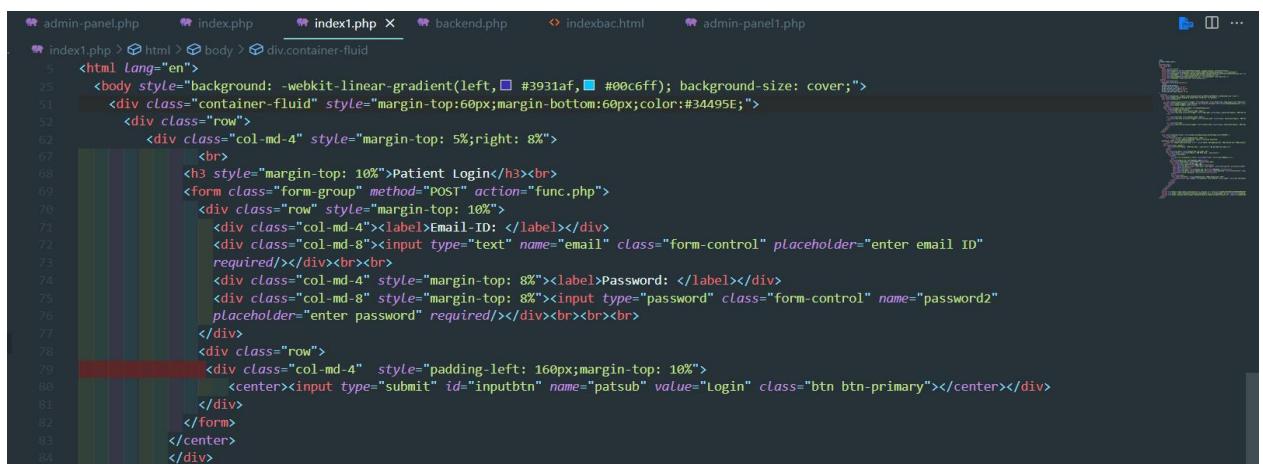
```
<div class="col-md-6">
    <div class="form-group">
        <input type="text" class="form-control" placeholder="Last Name *" name="lname" onkeydown="return alphaOnly(event);;" required/>
    </div>

    <div class="form-group">
        <input type="tel" minlength="10" maxlength="10" name="contact" class="form-control" placeholder="Your Phone *" />
    </div>
    <div class="form-group">
        <input type="password" class="form-control" id="cpassword" placeholder="Confirm Password *" name="cpassword" onkeyup='check();' required/><span id='message'></span>
    </div>
    <input type="submit" class="btnRegister" name="patsub1" onclick="return checklen();;" value="Register"/>
</div>
</div>
```

```
141
142     <div class="tab-pane fade show" id="profile" role="tabpanel" aria-labelledby="profile-tab">
143         <h3 class="register-heading">Login as Doctor</h3>
144         <form method="post" action="func1.php">
145             <div class="row register-form">
146                 <div class="col-md-6">
147                     <div class="form-group">
148                         <input type="text" class="form-control" placeholder="User Name *" name="username3" onkeydown="return alphaOnly(event);;" required/>
149                     </div>
150                 </div>
151                 <div class="col-md-6">
152                     <div class="form-group">
153                         <input type="password" class="form-control" placeholder="Password *" name="password3" required/>
154                     </div>
155
156                     <input type="submit" class="btnRegister" name="docsub1" value="Login"/>
157                 </div>
158             </div>
159         </form>
160     </div>
```



```
3 <div class="tab-pane fade show" id="admin" role="tabpanel" aria-labelledby="profile-tab">
4   <h3 class="register-heading">Login as Admin</h3>
5   <form method="post" action="func3.php">
6     <div class="row register-form">
7       <div class="col-md-6">
8         <div class="form-group">
9           <input type="text" class="form-control" placeholder="User Name *" name="username1"
10          |onkeydown="return alphaOnly(event);" required/>
11         </div>
12       </div>
13       <div class="col-md-6">
14         <div class="form-group">
15           <input type="password" class="form-control" placeholder="Password *" name="password2" required/>
16         </div>
17         <input type="submit" class="btnRegister" name="adsub" value="Login"/>
18       </div>
19     </div>
20   </form>
21 </div>
22 </div>
23 </div>
```



```
4 index1.php > ⚡ HTML > ⚡ body > ⚡ div.container-fluid
5   <html lang="en">
6     <body style="background: -webkit-linear-gradient(left, #3931af, #00c6ff); background-size: cover;">
7       <div class="container-fluid" style="margin-top:60px; margin-bottom:60px; color:#34495e;">
8         <div class="row">
9           <div class="col-md-4" style="margin-top: 5%; right: 8%">
10             <br>
11             <h3 style="margin-top: 10%">Patient Login</h3><br>
12             <form class="form-group" method="POST" action="func.php">
13               <div class="row" style="margin-top: 10%">
14                 <div class="col-md-4"><label>Email-ID: </label></div>
15                 <div class="col-md-8"><input type="text" name="email" class="form-control" placeholder="enter email ID" required/></div><br><br>
16                 <div class="col-md-4" style="margin-top: 8%"><label>Password: </label></div>
17                 <div class="col-md-8" style="margin-top: 8%"><input type="password" class="form-control" name="password2" placeholder="enter password" required/></div><br><br><br>
18               </div>
19             <div class="row">
20               <div class="col-md-4" style="padding-left: 160px; margin-top: 10%">
21                 <center><input type="submit" id="inputbtn" name="patsub" value="Login" class="btn btn-primary"></center></div>
22             </div>
23           </form>
24         </center>
25       </div>
26     </div>
27   </div>
```

The login implementation for the Doctor, Administrator, and Patient in the Healthcare Management System (HMS)

```

15     if (isset($_POST['app-submit'])) {
16         $pid = $_SESSION['pid'];
17         $username = $_SESSION['username'];
18         $email = $_SESSION['email'];
19         $fname = $_SESSION['fname'];
20         $lname = $_SESSION['lname'];
21         $gender = $_SESSION['gender'];
22         $contact = $_SESSION['contact'];
23         $doctor = $_POST['doctor'];
24         $email = $_SESSION['email'];
25         $docFees = $_POST['docFees'];
26
27         $appdate = $_POST['appdate'];
28         $apptime = $_POST['apptime'];
29         $cur_date = date("Y-m-d");
30         date_default_timezone_set('Asia/Kolkata');
31         $cur_time = date("H:i:s");
32         $apptime1 = strtotime($apptime);
33         $appdate1 = strtotime($appdate);
34

```

```

35     if (date("Y-m-d", $appdate1) >= $cur_date) {
36         if ((date("Y-m-d", $appdate1) == $cur_date and date("H:i:s", $apptime1) > $cur_time) or date("Y-m-d", $appdate1) > $cur_date) {
37             $check_query = mysqli_query($con, "select apptime from appointmenttb where doctor='$doctor' and appdate='$appdate' and apptime='$apptime'");
38
39             if (mysqli_num_rows($check_query) == 0) {
40                 $query = mysqli_query($con, "insert into appointmenttb(pid,fname,lname,gender,email,contact,doctor,
41 docFees,appdate,apptime,userstatus,doctorstatus)
42 values($pid,$fname','$lname','$gender','$email','$contact','$doctor','$docFees','$appdate','$apptime','1','1')");
43
44             if ($query) {
45                 echo "<script>alert('Your appointment successfully booked');</script>";
46             } else {
47                 echo "<script>alert('Unable to process your request. Please try again!');</script>";
48             }
49         } else {
50             echo "<script>alert('We are sorry to inform that the doctor is not available in this time or date.
51 Please choose different time or date!');</script>";
52         }
53     } else {
54         echo "<script>alert('Select a time or date in the future!');</script>";
55     }
56 } else {
57     echo "<script>alert('Select a time or date in the future!');</script>";
58 }
59 }
60 }

```

Booking Appointments allows patients to efficiently schedule doctor consultations.

```

61
62     if (isset($_GET['cancel'])) {
63         $query = mysqli_query($con, "update appointmenttb set userStatus='0' where ID = '" . $_GET['ID'] . "'");
64         if ($query) {
65             echo "<script>alert('Your appointment successfully cancelled');</script>";
66         }
67     }
68

```

Cancel Appointments .

```

admin-panel.php X
admin-panel.php > ...
68
69  function generate_bill()
70  {
71      $con = mysqli_connect("localhost", "root", "", "myhmsdb");
72      $pid = $_SESSION['pid'];
73      $output = '';
74      $query = mysqli_query($con, "select p.pid,p.ID,p.fname,p.lname,p.doctor,p.apptdate,p.apptime,p.disease,p.allergy,
75      p.prescription,a.docFees from prestb p inner join appointmenttb a on p.ID=a.ID and p.pid = '$pid' and p.ID = '" . $_GET['ID'] . "'");
76      while ($row = mysqli_fetch_array($query)) {
77          $output .= '
78          <label> Patient ID : </label>' . $row["pid"] . '<br/><br/>
79          <label> Appointment ID : </label>' . $row["ID"] . '<br/><br/>
80          <label> Patient Name : </label>' . $row["fname"] . ' ' . $row["lname"] . '<br/><br/>
81          <label> Doctor Name : </label>' . $row["doctor"] . '<br/><br/>
82          <label> Appointment Date : </label>' . $row["apptdate"] . '<br/><br/>
83          <label> Appointment Time : </label>' . $row["apptime"] . '<br/><br/>
84          <label> Disease : </label>' . $row["disease"] . '<br/><br/>
85          <label> Allergies : </label>' . $row["allergy"] . '<br/><br/>
86          <label> Prescription : </label>' . $row["prescription"] . '<br/><br/>
87          <label> Fees Paid : </label>' . $row["docFees"] . '<br/>
88      ';
89  }
90
91  return $output;
92 }

```

```

95 if (isset($_GET["generate_bill"])) {
96     require_once("TCPDF/tcpdf.php");
97     $obj_pdf = new TCPDF('P', PDF_UNIT, PDF_PAGE_FORMAT, true, 'UTF-8', false);
98     $obj_pdf->setCreator(PDF_CREATOR);
99     $obj_pdf->setTitle("Generate Bill");
100    $obj_pdf->setHeaderData('', '', PDF_HEADER_TITLE, PDF_HEADER_STRING);
101    $obj_pdf->setHeaderFont(array(PDF_FONT_NAME_MAIN, '', PDF_FONT_SIZE_MAIN));
102    $obj_pdf->setFooterFont(array(PDF_FONT_NAME_MAIN, '', PDF_FONT_SIZE_MAIN));
103    $obj_pdf->setDefaultMonospacedFont('helvetica');
104    $obj_pdf->setFooterMargin(PDF_MARGIN_FOOTER);
105    $obj_pdf->setMargins(PDF_MARGIN_LEFT, '5', PDF_MARGIN_RIGHT);
106    $obj_pdf->setPrintHeader(false);
107    $obj_pdf->setPrintFooter(false);
108    $obj_pdf->SetAutoPageBreak(TRUE, 10);
109    $obj_pdf->setFont('helvetica', '', 12);
110    $obj_pdf->AddPage();
111
112    $content = '';
113    $content .= '<br/>
114        <h2 align="center"> Global Hospitals</h2></br>
115        <h3 align="center"> Bill</h3>';
116    $content .= generate_bill();
117    $obj_pdf->writeHTML($content);
118    ob_end_clean();
119    $obj_pdf->Output("bill.pdf", 'I');
120 }

```

Generate Bill.

```

newfunc.php X
newfunc.php > ...
1  <?php
2  $con=mysqli_connect("localhost","root","","myhmsdb");
3  if(isset($_POST['update_data']))
4  {
5      $contact=$_POST['contact'];
6      $status=$_POST['status'];
7      $query="update appointmenttb set payment='$status' where contact='$contact';";
8      $result=mysqli_query($con,$query);
9      if($result)
10         header("Location:updated.php");
11  }

```

Update appointment payment status by contact.

```

313
314     <script>
315         document.getElementById('spec').onchange = function foo() {
316             let spec = this.value;
317             console.log(spec)
318             let docs = [...document.getElementById('doctor').options];
319
320             docs.forEach((el, ind, arr) => {
321                 arr[ind].setAttribute("style", "");
322                 if (el.getAttribute("data-spec") != spec) {
323                     arr[ind].setAttribute("style", "display: none");
324                 }
325             });
326         }
327     </script>

```

Display Doctors by Specialization.

```

336
337     <script>
338         document.getElementById('doctor').onchange = function updateFees(e) {
339             var selection = document.querySelector(`[value=${this.value}]`).getAttribute('data-value');
340             document.getElementById('docFees').value = selection;
341         }
342     </script>

```

Update Fees on Doctor Selection.

```

378     <div class="tab-pane fade" id="app-hist" role="tabpanel" aria-labelledby="list-pat-list">
379         <table class="table table-hover">
380             <thead>
381                 <tr>
382                     <th scope="col">Doctor Name</th>
383                     <th scope="col">Consultancy Fees</th>
384                     <th scope="col">Appointment Date</th>
385                     <th scope="col">Appointment Time</th>
386                     <th scope="col">Current Status</th>
387                     <th scope="col">Action</th>
388                 </tr>
389             </thead>
390             <tbody>
391                 <?php
392
393                     $con = mysqli_connect("localhost", "root", "", "myhmsdb");
394                     global $con;
395
396                     $query = "select ID,doctor,docFees,apptime,apptime,userstatus,doctorstatus from appointmenttb
397                     where fname ='$fname' and lname ='$lname'";
398                     $result = mysqli_query($con, $query);
399                     while ($row = mysqli_fetch_array($result)) {
400
401                         <tr>
402                             <td><?php echo $row['doctor']; ?></td>
403                             <td><?php echo $row['docFees']; ?></td>
404                             <td><?php echo $row['apptime']; ?></td>
405                             <td><?php echo $row['apptime']; ?></td>
406                             <td>
407                                 <?php if (($row['userStatus'] == 1) && ($row['doctorStatus'] == 1)) {
408                                     echo "Active";
409                                 }
410                             </td>

```

```

409 }
410     if (($row['userStatus'] == 0) && ($row['doctorStatus'] == 1)) {
411         echo "Cancelled by You";
412     }
413     if (($row['userStatus'] == 1) && ($row['doctorStatus'] == 0)) {
414         echo "Cancelled by Doctor";
415     }
416 ?></td>
417 <td>
418     <?php if (($row['userStatus'] == 1) && ($row['doctorstatus'] == 1)) { ?>
419         <a href="admin-panel.php?ID=<?php echo $row['ID'] ?>&cancel=update"
420             onClick="return confirm('Are you sure you want to cancel this appointment ?')"
421             title="Cancel Appointment" tooltip-placement="top" tooltip="Remove"><button
422                 class="btn btn-danger">Cancel</button></a>
423     <?php } else {
424         echo "Cancelled";
425     } ?>
426     </td>
427 </tr>
428 <?php } ?>
429 </tbody>
430 </table>
431 <br>
432 </div>

```

View Appointment History.

```

434 <div class="tab-pane fade" id="list-pres" role="tabpanel" aria-labelledby="list-pres-list">
435     <table class="table table-hover">
436         <thead>
437             <tr>
438                 <th scope="col">Doctor Name</th>
439                 <th scope="col">Appointment ID</th>
440                 <th scope="col">Appointment Date</th>
441                 <th scope="col">Appointment Time</th>
442                 <th scope="col">Diseases</th>
443                 <th scope="col">Allergies</th>
444                 <th scope="col">Prescriptions</th>
445                 <th scope="col">Bill Payment</th>
446             </tr>
447         </thead>
448         <tbody>
449             <?php
450                 $con = mysqli_connect("localhost", "root", "", "myhmsdb");
451                 global $con;
452                 $query = "select doctor.ID, appdate, apptime, disease, allergy, prescription from prestb where pid='$pid'";
453                 $result = mysqli_query($con, $query);
454                 if (!$result) {
455                     echo mysqli_error($con);
456                 }

```

```

457             while ($row = mysqli_fetch_array($result)) {
458                 ?
459                     <tr>
460                         <td><?php echo $row['doctor']; ?></td>
461                         <td><?php echo $row['ID']; ?></td>
462                         <td><?php echo $row['appdate']; ?></td>
463                         <td><?php echo $row['apptime']; ?></td>
464                         <td><?php echo $row['disease']; ?></td>
465                         <td><?php echo $row['allergy']; ?></td>
466                         <td><?php echo $row['prescription']; ?></td>
467                     <td>
468                         <form method="get">
469                             <a href="admin-panel.php?ID=<?php echo $row['ID'] ?>">
470                                 <input type="hidden" name="ID" value="<?php echo $row['ID'] ?>" />
471                                 <input type="submit" onclick="alert('Bill Paid Successfully');" name="generate_bill"
472                                     class="btn btn-success" value="Pay Bill" />
473                             </a>
474                         </form>
475                     </td>
476                 </tr>
477             <?php } ?
478         </tbody>
479     </table>
480     <br>
481 </div>

```

View Prescriptions.

```

484 <div class="tab-pane fade" id="list-messages" role="tabpanel" aria-labelledby="list-messages-list">...</div>
485 <div class="tab-pane fade" id="list-settings" role="tabpanel" aria-labelledby="list-settings-list">
486   <form class="form-group" method="post" action="func.php">
487     <label>Doctors name: </label>
488     <input type="text" name="name" placeholder="Enter doctors name" class="form-control">
489     <br>
490     <input type="submit" name="doc_sub" value="Add Doctor" class="btn btn-primary">
491   </form>
492 </div>

```

admin-panel.php indexbac.html admin-panel1.php

```

1 admin-panel1.php > ...
2
3 $con = mysqli_connect("localhost", "root", "", "myhmsdb");
4 include('newfunc.php');
5 if (isset($_POST['docsub'])) {
6   $doctor = $_POST['doctor'];
7   $dpassword = $_POST['dpassword'];
8   $demail = $_POST['demail'];
9   $spec = $_POST['special'];
10  $docFees = $_POST['docFees'];
11  $query = "insert into doctb(username,password,email,spec,docFees)values('$doctor', '$dpassword', '$demail', '$spec', '$docFees')";
12  $result = mysqli_query($con, $query);
13  if ($result) {
14    echo "<script>alert('Doctor added successfully!');</script>";
15  }
16 }

```

Add Doctor.

```

18 if (isset($_POST['docsub1'])) {
19   $demail = $_POST['demail'];
20   $query = "delete from doctb where email='$demail';";
21   $result = mysqli_query($con, $query);
22   if ($result) {
23     echo "<script>alert('Doctor removed successfully!');</script>";
24   } else {
25     echo "<script>alert('Unable to delete!');</script>";
26   }
27 }
?>

```

Delete Doctor.

```

247 <?php
248 $con = mysqli_connect("localhost", "root", "", "myhmsdb");
249 global $con;
250 $query = "select * from doctb";
251 $result = mysqli_query($con, $query);
252 while ($row = mysqli_fetch_array($result)) {
253   $username = $row['username'];
254   $spec = $row['spec'];
255   $email = $row['email'];
256   $password = $row['password'];
257   $docFees = $row['docFees'];
258
259   echo "<tr>
260     <td>$username</td>
261     <td>$spec</td>
262     <td>$email</td>
263     <td>$password</td>
264     <td>$docFees</td>
265   </tr>";
266 }
?>
</tbody>

```

View Doctor List.

```

296 $con = mysqli_connect("localhost", "root", "", "myhmsdb");
297 global $con;
298 $query = "select * from patreg";
299 $result = mysqli_query($con, $query);
300 while ($row = mysqli_fetch_array($result)) {
301     $pid = $row['pid'];
302     $fname = $row['fname'];
303     $lname = $row['lname'];
304     $gender = $row['gender'];
305     $email = $row['email'];
306     $contact = $row['contact'];
307     $password = $row['password'];
308
309     echo "<tr>
310         <td>$pid</td>
311         <td>$fname</td>
312         <td>$lname</td>
313         <td>$gender</td>
314         <td>$email</td>
315         <td>$contact</td>
316         <td>$password</td>
317     </tr>";
318 }
319 ?>

```

View Patient List.

```

415 $query = "select * from appointmenttb;";
416 $result = mysqli_query($con, $query);
417 while ($row = mysqli_fetch_array($result)) {
418     ?
419     <tr>
420         <td><?php echo $row['ID']; ?></td>
421         <td><?php echo $row['pid']; ?></td>
422         <td><?php echo $row['fname']; ?></td>
423         <td><?php echo $row['lname']; ?></td>
424         <td><?php echo $row['gender']; ?></td>
425         <td><?php echo $row['email']; ?></td>
426         <td><?php echo $row['contact']; ?></td>
427         <td><?php echo $row['doctor']; ?></td>
428         <td><?php echo $row['docFees']; ?></td>
429         <td><?php echo $row['appdate']; ?></td>
430         <td><?php echo $row['apptime']; ?></td>
431         <td>
432             <?php if (($row['userstatus'] == 1) && ($row['doctorstatus'] == 1)) {
433                 echo "Active";
434             }
435             if (($row['userstatus'] == 0) && ($row['doctorstatus'] == 1)) {
436                 echo "Cancelled by Patient";
437             }
438             if (($row['userstatus'] == 1) && ($row['doctorstatus'] == 0)) {
439                 echo "Cancelled by Doctor";
440             }
441             ?></td>
442     </tr>
443     <?php } ?>

```

View Appointment Details.

```

<?php
$con = mysqli_connect("localhost", "root", "", "myhmsdb");
global $con;
$query = "select * from prestb";
$result = mysqli_query($con, $query);
while ($row = mysqli_fetch_array($result)) {
    $doctor = $row['doctor'];
    $pid = $row['pid'];
    $ID = $row['ID'];
    $fname = $row['fname'];
    $lname = $row['lname'];
    $appdate = $row['appdate'];
    $apptime = $row['apptime'];
    $disease = $row['disease'];
    $allergy = $row['allergy'];
    $pres = $row['prescription'];

    echo "<tr>
        <td>$doctor</td>
        <td>$pid</td>
        <td>$ID</td>
        <td>$fname</td>
        <td>$lname</td>
        <td>$appdate</td>
        <td>$apptime</td>
        <td>$disease</td>
        <td>$allergy</td>
        <td>$pres</td>
    </tr>";
}
?>

```

View Prescription List.

```

admin-panel.php      indexbac.html      appsearch.php X
appsearch.php > html > body
<html>
<body>
<?php
include("newfunc.php");
if(isset($_POST['app_search_submit'])){
{
    $contact=$_POST['app_contact'];
    $query = "select * from appointmenttb where contact= '$contact'";
    $result = mysqli_query($con,$query);
    $row=mysqli_fetch_array($result);
    if($row['fname']==" " & $row['lname']==" " & $row['email']==" " & $row['contact']==" " & $row['doctor']==" "
    & $row['docFees']==" " & $row['appdate']==" " & $row['apptime']==" "){
        echo "<script> alert('No entries found! Please enter valid details');
        window.location.href = 'admin-panel1.php#list-doc';</script>";
    }
} else {

```

Search Appointments by Contact.

```

admin-panel.php
indexbac.html
    <html lang="en">
        <body>
            <div class="container">
                <form id="hospitalForm" action="backend.php" method="POST" enctype="multipart/form-data">
                    <input type="text" id="username" name="username" placeholder="Enter your name" required>
                    <label for="age">Age:</label>
                    <input type="number" id="age" name="age" placeholder="Enter your age" required>
                    <label for="hospitalData">Patient data:</label>
                    <textarea id="hospitalData" name="hospitalData" rows="4" placeholder="Enter case details"></textarea>
                    <label for="files">Upload medical files:</label>
                    <input type="file" id="files" name="files[]" multiple>
                    <label for="videos">Upload videos:</label>
                    <input type="file" id="videos" name="videos[]" accept="video/*" multiple>
                    <button type="submit">Send Data</button>
                </form>
            </div>

```

```

79 <script>
80     const form = document.getElementById('hospitalForm');
81     form.addEventListener('submit', async (event) => {
82         event.preventDefault();
83         const formData = new FormData(form);
84         try {
85             const response = await fetch('backend.php', {
86                 method: 'POST',
87                 body: formData,
88             });
89             if (response.ok) {
90                 alert('Data has been successfully sent!');
91             } else {
92                 alert('There was an error sending the data. Please try again.');
93             }
94         } catch (error) {
95             alert(`Error: ${error.message}`);
96         }
97     });
98 </script>
99 </body>
100 </html>

```

```

admin-panel.php
indexbac.html
appsearch.php
backend.php
    <?php
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $username = $_POST['username'] ?? null;
        $usernumber = $_POST['usernumber'] ?? null;
        $age = $_POST['age'] ?? null;
        $hospitalData = $_POST['hospitalData'] ?? null;

        $files_dir = "uploads/files/";
        $videos_dir = "uploads/videos/";

        if (!is_dir($files_dir)) mkdir($files_dir, 0777, true);
        if (!is_dir($videos_dir)) mkdir($videos_dir, 0777, true);

        echo "<h3>Data Received:</h3>";
        echo "User Name: $username<br>";
        echo "User Number: $usernumber<br>";
        echo "Age: $age<br>";
        echo "Hospital Data: $hospitalData<br><hr>";

        if (isset($_FILES['files'])) {
            echo "<h3>Uploaded files:</h3>";
            foreach ($_FILES['files']['tmp_name'] as $key => $tmp_name) {
                $file_name = basename($_FILES['files']['name'][$key]);
                $file_path = $files_dir . time() . "_" . $file_name;

                if (move_uploaded_file($tmp_name, $file_path)) {
                    echo "<a href='$file_path' target='_blank'>$file_name</a><br>";
                } else {
                    echo "Failed to upload file: $file_name<br>";
                }
            }
        }
    }

```

Patient Data Entry and Receiving user data.

```

admin-panel.php      indexbac.html      appsearch.php      doctor-panel.php      doctorsearch.php      send_email.php X
send_email.php > ...
1. <?php
2. use PHPMailer\PHPMailer\PHPMailer;
3. use PHPMailer\PHPMailer\Exception;
4. require 'phpmailer/PHPMailer.php';
5. require 'phpmailer/Exception.php';
6. require 'phpmailer/SMTP.php';

```

```

18  <?php
19  if ($_SERVER["REQUEST_METHOD"] == "POST") {
20      $to = $_POST['email'];
21      $subject = $_POST['subject'];
22      $message = $_POST['message'];
23      $mail = new PHPMailer(true);
24      try {
25          $mail->isSMTP();
26          $mail->Host = 'smtp.gmail.com';
27          $mail->SMTPAuth = true;
28          $mail->Username = '235380@ppu.edu.ps';
29          $mail->Password = '408365070';
30          $mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;
31          $mail->Port = 587;
32          $mail->setFrom('235380@ppu.edu.ps', 'Hospital');
33          $mail->addAddress($to);
34          $mail->isHTML(true);
35          $mail->Subject = $subject;
36          $mail->Body = $message;
37          $mail->send();
38      }
39  }

```

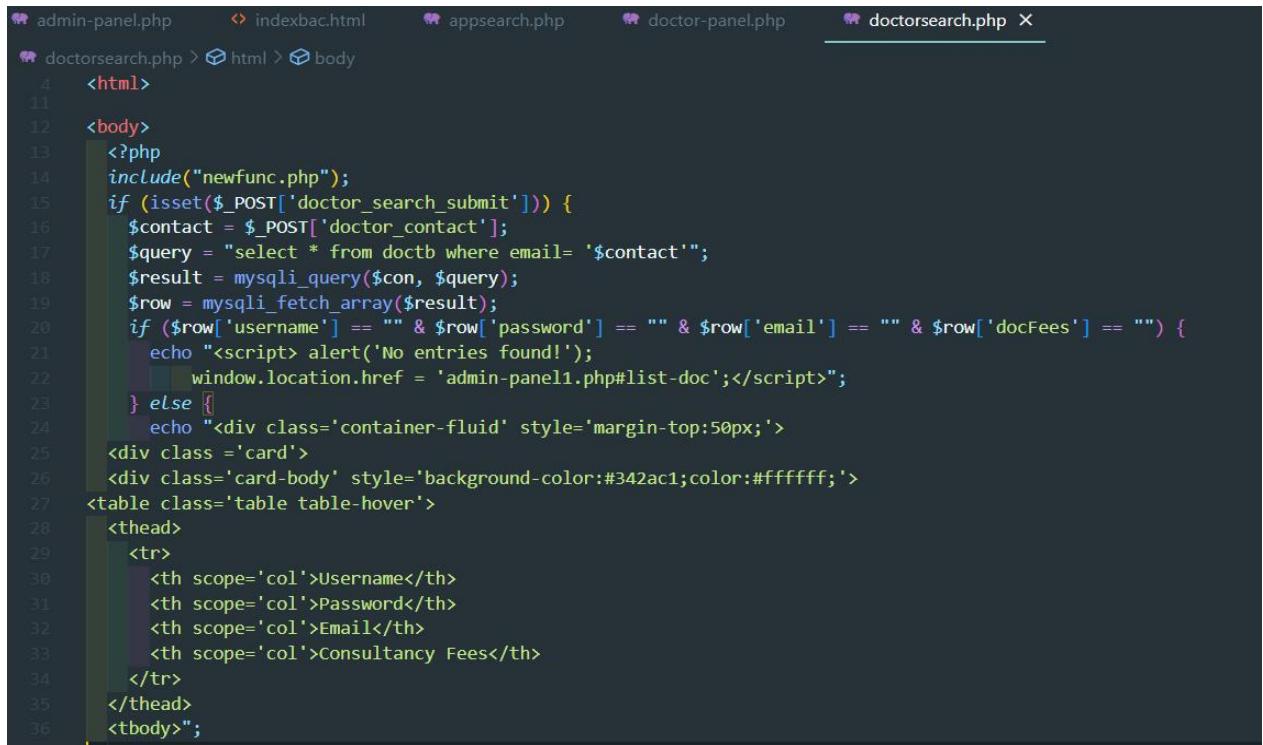
Sending an email using PHPMailer with SweetAlert2 notifications for success or failure.

```

admin-panel.php      indexbac.html      appsearch.php      doctor-panel.php X
doctor-panel.php > html > body > div#emailModal.modal.fade > div.modal-dialog > div.modal-content > form
31  <html lang="en">
32      <body style="padding-top:50px;">
33          <div class="container-fluid" style="margin-top:50px;">
34          </div>
35          <!-- Modal -->
36          <div class="modal fade" id="emailModal" tabindex="-1" role="dialog" aria-labelledby="emailModalLabel" aria-hidden="true">
37              <div class="modal-dialog" role="document">
38                  <div class="modal-content">
39                      <form method="POST" action="send_email.php">
40                          <div class="modal-header">
41                              <h5 class="modal-title" id="emailModalLabel">Send Email</h5>
42                              <button type="button" class="close" data-dismiss="modal" aria-label="Close">
43                                  <span aria-hidden="true">&times;

```

Emailing enables doctors to share details with patients.



The screenshot shows a code editor with multiple tabs at the top: admin-panel.php, indexbac.html, appsearch.php, doctor-panel.php, and doctorsearch.php (which is the active tab). The code in the editor is a PHP script for searching doctors by email. It includes logic to check if a search was submitted, query the database for matching records, and display the results in a table. If no entries are found, it displays an alert message.

```
4      <html>
5
6      <body>
7          <?php
8              include("newfunc.php");
9
10             if (isset($_POST['doctor_search_submit'])) {
11                 $contact = $_POST['doctor_contact'];
12                 $query = "select * from doctb where email= '$contact'";
13                 $result = mysqli_query($con, $query);
14                 $row = mysqli_fetch_array($result);
15                 if ($row['username'] == "" & $row['password'] == "" & $row['email'] == "" & $row['docFees'] == "") {
16                     echo "<script> alert('No entries found!');</script>";
17                     window.location.href = 'admin-panel1.php#list-doc';
18                 } else {
19                     echo "<div class='container-fluid' style='margin-top:50px;'>
20                         <div class='card'>
21                             <div class='card-body' style='background-color:#342ac1;color:#fffff;'>
22                                 <table class='table table-hover'>
23                                     <thead>
24                                         <tr>
25                                             <th scope='col'>Username</th>
26                                             <th scope='col'>Password</th>
27                                             <th scope='col'>Email</th>
28                                             <th scope='col'>Consultancy Fees</th>
29                                         </tr>
30                                     </thead>
31                                     <tbody>";
```

Search for doctors by email to view details or check availability.

The same logic used in the doctor management component is also applied to the patient management and admin management features, ensuring a consistent and streamlined approach throughout the system. This includes handling user data, secure login, and role-specific functionalities.

For further insights on the code, you can freely check the GitHub repository provided in the references. It includes the complete file structure, detailed explanations, and line-by-line comments to clarify the implementation.

These steps outline the initial setup for deploying the application on AWS and integrating it with GitHub for version control and code management.

```
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
New release '24.04.1 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.
```

```
Last login: Thu Jan  9 06:34:10 2025 from 18.202.216.51  
ubuntu@ip-172-31-21-63:~$ sudo yum update -y  
sudo yum install -y httpd git
```

```
ubuntu@ip-172-31-21-63:~$ sudo systemctl start httpd  
sudo systemctl enable httpd
```

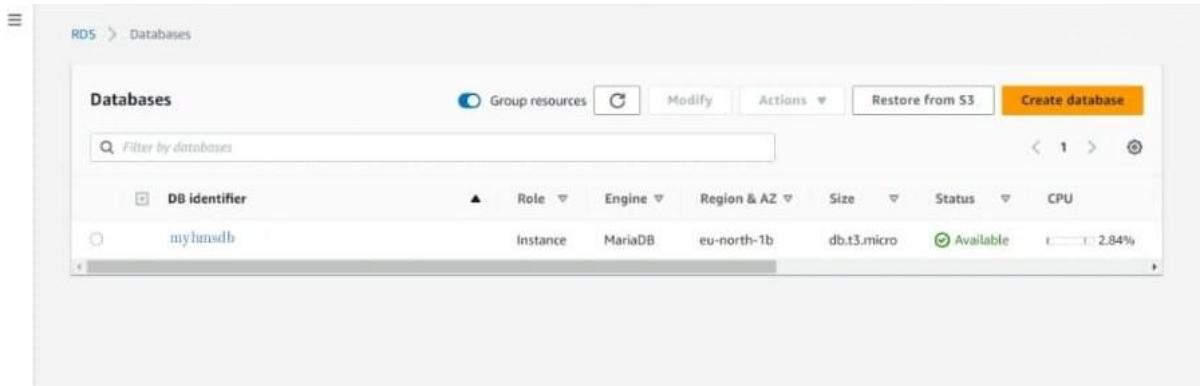
```
ubuntu@ip-172-31-21-63:~$  
ubuntu@ip-172-31-21-63:~$ sudo amazon-linux-extras enable php8.0  
sudo yum install php php-cli php-mysqlnd -y
```

```
ubuntu@ip-172-31-21-63:~$  
ubuntu@ip-172-31-21-63:~$ git config --global user.name "Aya kh"  
git config --global user.email "ayakh2002@gmail.com"
```

```
ubuntu@ip-172-31-21-63:~$ cd var/www/  
ubuntu@ip-172-31-21-63:/var/www$ sudo git clone https://github.com/ayakh2002/hospitalManagement
```

6. Data Model

The data model of our system is implemented using MySQL, which is hosted on Amazon RDS (Relational Database Service).



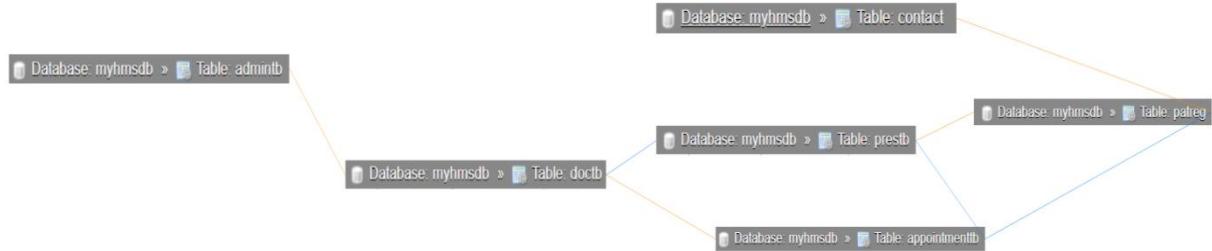
The Database Instance in RDS.

In RDS, the MySQL database is stored and its various entities and their related information are managed. This includes data related to doctors, patients, appointments, and other system-specific entities. The data model consists of tables, as illustrated in the previous figures, that represent different entities, and the relationships between these entities define the structure and organization of the data.

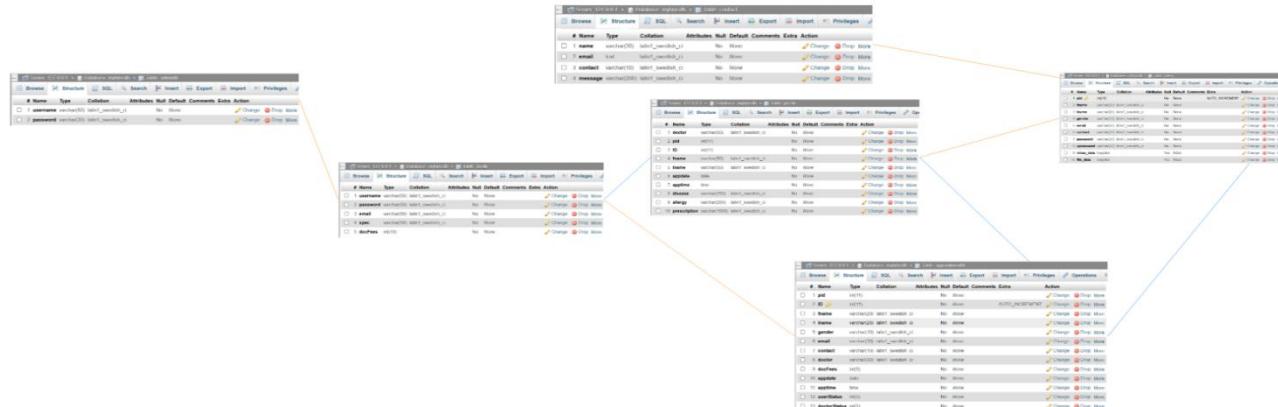
As shown in the previous figures, the data itself is stored within the MySQL database instance provided by Amazon RDS, a fully managed database service that simplifies the deployment, management, and scaling of databases in the cloud. It offers features such as automated backups, high availability, and performance monitoring.

By utilizing MySQL with Amazon RDS, the Hospital Management System (HMS) benefits from a reliable and scalable database solution that ensures data integrity, availability, and optimal performance. The data model is designed to meet the requirements of HMS, enabling efficient storage, retrieval, and manipulation of data to support the functionality and operations of the hospital management system.

The Database Schema:



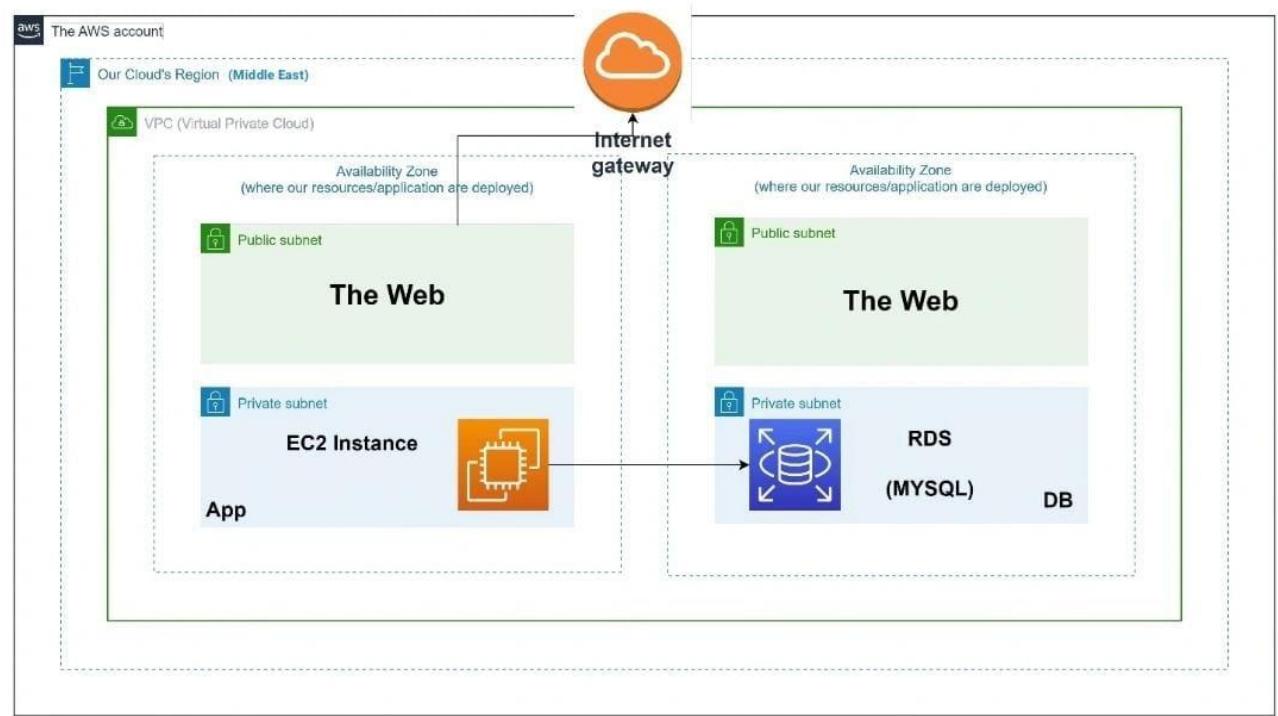
Database Schema Pt.1



Database Schema Pt.2 (Detailed)

7. Architecture of the AWS.

Amazon Web Services (AWS) is a leading and widely adopted cloud computing platform, offering a vast range of services designed to meet the needs of diverse industries. With its global infrastructure and advanced tools, AWS provides scalable, reliable, and secure solutions for businesses of all sizes, including hospitals and healthcare organizations. It is trusted by millions of users, ranging from innovative startups to large enterprises, to optimize costs, enhance operational efficiency, and accelerate innovation.



AWS is designed to provide one of the most secure and flexible cloud computing environments available today. Its robust infrastructure is built to meet the stringent security requirements of industries such as healthcare, finance, and government agencies. AWS ensures data protection through a comprehensive suite of security tools, offering over 300 security, compliance, and governance features. With support for 98 security standards and compliance certifications, AWS guarantees that data stored across its services, including those used in the HMS project, can be encrypted and safeguarded effectively. This ensures that patient and healthcare data remain confidential and secure while enabling scalability and reliability for the system.

Having said that, the AWS architecture for HMS is designed as follows:

1. EC2 Instances:

HMS requires Amazon EC2 instances to host all its essential components. These instances provide scalable and reliable compute resources, enabling the system to handle varying workloads efficiently. The deployment ensures high availability and fault tolerance.

2. RDS (Relational Database Service):

HMS uses Amazon RDS to store and manage its MySQL database. The RDS instance is configured with appropriate storage capacity, instance type, and automated backups to ensure data persistence, reliability, and seamless database management.

3. IAM (Identity and Access Management):

AWS IAM is utilized to manage user access and permissions to AWS resources securely. It allows administrators to define fine-grained access controls, ensuring that only authorized users, such as administrators, doctors, and patients, can interact with the system's components.

These are the key components and services that are a part of the AWS architecture for HMS. They collectively ensure that the system operates in a secure, scalable, and reliable cloud environment, meeting the operational requirements of the healthcare management system.

8. Deployment on the Platform

Here are the steps in which we deployed HMS on AWS:

- **Launched the EC2 Instance:** Using the AWS Management Console, we navigated to the EC2 service and selected "Launch Instance" to create a virtual server for hosting the HMS application.
- **Chose Instance Type:** We selected an instance type that met the compute, memory, and storage requirements of HMS for optimal performance.
- **Configured the Instance:** Configured options such as the number of instances, network settings, and security groups to meet the application requirements.
- **Added Storage:** Defined the storage size and type to support the HMS application files efficiently.
- **Set Up RDS (Relational Database Service):** Configured Amazon RDS for managing the MySQL database. This included selecting the instance type, defining storage capacity, and setting up automated backups for secure database management.
- **Configured Security Groups:** Defined inbound and outbound rules to control network traffic, ensuring secure access to the EC2 instance and RDS.
- **Review and Launch:** Verified all configurations and launched the EC2 instance. The status and details of the instance were monitored using the AWS Management Console.
- **Key Pair Selection:** Generated and downloaded a key pair to securely connect to the instance from a local system for deployment and monitoring.

9. User Support in HMS

User service in HMS is designed to ensure a seamless experience for administrators, doctors, and patients. The application provides multiple touchpoints and interactions to deliver a high-quality user experience. Each of these interactions plays a vital role in shaping the user's perception of the system's efficiency and effectiveness.

1. Patient's Convenience and Accessibility:

Patients can book appointments with doctors through the application, providing them with the flexibility to select a doctor based on specialization, availability, and preferences. The system allows patients to view their appointment history, receive reminders for upcoming appointments, and access their medical records securely.

2. Doctor's Streamlined Management Tools:

Doctors can manage their schedules efficiently, including viewing upcoming appointments and their patients' basic medical history. They can also prescribe medications, add notes, and update patient records in the system, ensuring accurate and timely information for future consultations.

3. Administrator's Centralized Control:

The administrator has full control over managing the system. This includes registering new doctors, managing patient records, monitoring appointment bookings, and handling any disputes or conflicts that arise. Administrators ensure that the system operates securely by managing user roles and permissions.

4. Enhanced Data Security in the Cloud:

HMS leverages AWS services to securely store patient and doctor data in the cloud. Only authorized personnel, such as administrators and doctors, can access and modify the data. This ensures that sensitive medical records are protected from unauthorized access and tampering.

5. Automated Notification System:

HMS integrates notifications to keep users informed. Patients receive reminders for appointments and updates about their consultations, while doctors are notified of any changes in their schedules. This helps maintain effective communication between all stakeholders.

6. Doctor's Flexibility in Decision-Making:

Doctors have the autonomy to manage their interactions with patients, such as approving appointment requests, updating patient records, and prescribing medications. This ensures that each doctor has complete control over their patient management.

7. Patient's Peace of Mind with Data Management:

HMS eliminates the need for patients to worry about managing their health records manually. The cloud-based system securely stores patient records, schedules, and prescriptions, providing a comprehensive and reliable solution for their healthcare needs.

8. Simple and Intuitive Interface:

The application is designed with a user-friendly interface, making it easy for patients, doctors, and administrators to navigate and perform their tasks without technical expertise.

These features collectively ensure that HMS delivers a secure, efficient, and high-quality user experience for all its users, fostering trust and reliability in the healthcare management process.

10. Conclusion

HMS is a robust and comprehensive healthcare management system designed to streamline interactions between administrators, doctors, and patients. With features such as appointment booking, patient record management, and secure data handling, the system ensures a seamless and efficient user experience. By leveraging AWS services like EC2 instances, RDS, and IAM, HMS guarantees scalability, security, and reliable data storage. Its cloud-based architecture provides an efficient framework for managing medical records, scheduling appointments, and maintaining secure communication. Overall, HMS enhances collaboration among users, simplifies healthcare processes, and offers a user-friendly platform for delivering effective and reliable healthcare services.

References

The GitHub Report for HMS:

<https://github.com/aya-alkahlot/Hospital-Management-System>

https://magicsol.net/public_test/index.php