Online Doctor Reservation System

Project Proposal

Supervisor

Ms. Hafsah Mehmood + Maam Mehwish Javed

Submitted by

Neha Arshad
51009

Hadia Imtiaz
50649

Faculty of Computing, Riphah International University, Islamabad.

29th October,2024

Table of Content

1. Introduction

The ***Online Doctor Reservation System*** is a software application designed to make scheduling appointments easier for patients and healthcare professionals. By allowing patients to book appointments directly online, the system simplifies the process of appointment, reduces the administrative work load, and smoother management of doctor availability. This system is developed using **Java** to build user interfaces and **MySQL** for managing and storing the data, ensuring efficient storage and retrieval of patient, doctor, and appointment information. This project aims to enhance both patient convenience and doctor availability management.

2. Objective

The primary objective of the ***Online Doctor Reservation System*** is to provide a simple and efficient platform where patients can book appointments with their preferred doctors based on availability. This system aims to reduce wait times, eliminate scheduling conflicts, and improve healthcare service delivery rather than administrative tasks. The project demonstrates how object-oriented programming principles, integrating Java with a MySQL back-end for robust data management

3. Problem Description

The **Online Doctor Reservation System** solves common problems with traditional appointment booking methods. Patients often have to wait a long time and deal with scheduling conflicts when booking appointments by calling or visiting clinics. This manual process is not only time-consuming but also lead to errors, such as double bookings or miscommunication between patients and staff. Furthermore, clinics and hospitals also face a lot of extra work trying to manage appointments without a good system in place. The lack of real-time data on doctor availability can lead to patient frustration and decreased satisfaction. By automating the appointment booking process, this system makes it easier for

patients to get healthcare, reduces administrative burdens, and enhances communication between patients and doctors, leading to better healthcare outcomes.

4. Project Scope

This project covers the development of an ***Online Doctor Reservation System*** using Java and MySQL. The system supports the following functionalities:

1. **User Authentication**: Patients and doctors can log in using secure credentials.
2. **Doctor Management**: Doctors can update their availability, view appointments, and manage their schedule.
3. **Patient Management**: Patients can register, view doctor profiles, and book available time slots.
4. **Appointment Booking**: Real-time reservation system where patients can schedule and view their appointments.
5. **Payment Module:** After successfully booking an appointment with their preferred doctor, patients will allow patients to make payments of their appointment, securely record transaction details, and generate receipts upon successful payment
6. **Database Integration**: The system will use **MySQL** to store user information, doctor details, and appointment records, ensuring data consistency and security

5. Requirements

1. User Authentication

1.1 The System should allow users to login using social media platform.

1.2 The Users should create accounts with unique email addresses and passwords.

1.3 Through reset password feature, The Users should reset their password using registered email addresses.

1.4 The system should allow users to login to the system using valid email and password.

2. Doctor Management

2.1 The system should allow doctors to update their profiles with personal and professional information, such as specialization, experience, and contact details.

2.2 The system should enable doctors to set, update, and view their availability schedule.

2.3 The system should allow doctors to view their upcoming appointments.

2.4 The system should allow doctors to view upcoming patient details and their appointment time.

2.5 The system should permit doctors to cancel or reschedule appointments

2.6 The system should allowing doctors to access past appointments for reference.

2.7 The system should enable doctors to create, edit, and delete advices for patients by their unique IDs.

3. Patient Management

3.1 The system should allow patients to add their basic information such as name, contact details, and health history.

3.2 The system should allow patients to update their profiles.

3.3 The system should enable patients to view doctor profiles, including specializations, experience, location and availability.

3.4 The system should enable patients to view their appointment history.

3.5 The system should allow patients to submit feedback regarding their consultation sessions with doctors.

3.6 The system should enable patients to view or search advice by their unique IDs.

4. Appointment Booking

4.1 The system should allow patients to book appointments based on doctor availability.

4.2 The system should allow patients to cancel or reschedule appointments.

4.3 When once an appointment is booked or canceled, The system should automatically update the availability status of doctors.

4.4 The system should send notifications to doctors upon appointment request, cancelation, or rescheduling.

4.5 The system should send notifications to patients upon their appointment acceptance or cancellation.

4.6 The system should manage conflicts by preventing double bookings of the same time slot for a doctor.

5. Payment Module

5.1 To confirm the patient appointment booking with the selected doctor, The system should automatically generate a payment receipt, detailing the appointment time, doctor detail, total charges, discount offer, registration fees and patient detail.

5.2 The system should support various secure payment options, including credit/debit cards, digital wallets, and online bank transfers.

5.3 After successful payment, The system should allow patients to download an automatically generated digital copy of their booking receipt in PDF format.

5.4 The system should maintain a record of appointment receipts for future reference.

5.5 The system should give access patients to view or download past receipts from their profile

6. Database Integration

6.1 The system should use MySQL to store user, doctor, and appointment data.

6.2 The system should support efficient queries to allow quick retrieval of require data.

6.3 The system should maintain data integrity, ensuring no data duplication

6. Solution Application Areas / How DBMS are better for this System?

A ***Database Management System (DBMS)*** is highly beneficial in various areas of an **Online Doctor Reservation System**. One of the main areas is efficient data storage and fast access. A DBMS organizes and stores large amounts of information, like patient

profiles, doctor availability, and appointment schedules, in a way that can be quickly accessed. This means users can view doctor schedules in real time and book appointments without delay, making the experience smooth and satisfying.

A DBMS ensures data integrity and consistency, which means all information is accurate and reliable. The DBMS use primary and foreign keys, to prevent mistakes like double-booking or data duplication.

A DBMS are also provides **security** features as health data is sensitive. A DBMS protects patient and doctor information by allowing only authorized users to access specific data. This helps build trust in the app, as users know their data is safe from unauthorized access, maintaining their privacy.

Finally, a DBMS provides **backup and recovery** support, which keeps data safe in case of system failures. Automated backups mean that if there's an issue, the system can quickly restore important information, reducing downtime and maintaining a positive user experience. Overall, using a DBMS makes the system more efficient, secure, and reliable, making it an essential part of a well-functioning Online Doctor Reservation System.

7. Tools/Technology

The **Online Doctor Reservation System** requires both hardware and software tools to ensure a smooth, efficient experience for users. The primary software technologies include Java, used for building the user interface, and MySQL, which manages and stores the database securely. Additionally, a web server like Apache Net-beans is required to host the application. For hardware, Networking hardware, like routers, ensures stable internet connectivity, allowing real-time booking and communication between patients and doctors.

8. Expertise of the Team Members

The project team members familiar with **Java programming** and **MySQL database management**, having studied these in relevant coursework. This foundational knowledge prepares them to handle essential tasks like designing interfaces, managing data, and troubleshooting technical issues. Each team member is equally committed to the project, sharing a strong interest in developing a healthcare solution that can improve

patient access to doctors and streamline scheduling. This shared enthusiasm, equips the team to achieve the project objectives effectively.

9. Database Design

The database design for the **Online Doctor Reservation System** is structured to support efficient, secure, and user-friendly management of patient and doctor interactions. It uses a relational database approach, specifically MySQL. Here's a detailed overview of each entity, its attributes, and the relationships among them, aligning with the project requirements:

1. **Users**

   Attributes: user_id (Primary Key), name, password, email, contact_info

   Description: stores user information to ensure about authenticated

   users can use the app.

2. **Doctor**

   Attributes: doctor_id (Primary Key), name,, email, contact_info, rating.

   Description: stores doctor information, contact details, and ratings.

3. **Specialization**

   Attributes: specialization_id (Primary Key), doctor_id (Foreign Key).

   name, years_of_completion, institute

   Description: use to keep track of specific doctor specializations,

   completion years, and institutions through doctor_id

4. **Schedule**

Attributes: schedule_id (Primary Key), doctor_id (Foreign Key), date, Time

Description: use to view specific doctors availability by date and time

5. **Experience**

Attributes: experience_id (Primary Key), doctor_id (Foreign Key), position_title, organization_name,City ,start_date, end_date

Description: stores doctors' past job titles, organizations, and locations, Duration where they enhance and utilize their skills.

6. **Appointments**

Attributes: appointment_id (Primary Key), patient_id (Foreign Key), doctor_id (Foreign Key), status, consultation time, consultation date,

Description: Allows patients to view their scheduled appointments and provides doctors with details on upcoming appointments, including patient information.

7. **Location**

Attributes: location_id (Primary Key), doctor_id (Foreign Key), city, street, phase, hospital, From_Time,To_Time

Description: Details doctors' practice locations, hours, and hospital affiliations.

8. **Feedback**

Attributes: feedback_id (Primary Key), doctor_id (Foreign Key), patient_id (Foreign Key), rating, review

Description: Records patient reviews and ratings for a specific doctors

9. **Patient**

Attributes: patient_id (Primary Key), name, contact_info, address .

Description: stores patient details and contact details.

10. **Medical Record**

Attributes: medical_record_id (Primary Key),patient_id (Foreign Key), description, reports, disease

Description: Holds patients' health history,and reports,

11. **Past Consultation**

Attributes: consultation_id (Primary Key), patient_id (Foreign Key), appointment_id (Foreign Key), advice, medication_id (Foreign Key)

Description: Tracks details of previous consultations, including advice and medications.

12. **Medication**

Attributes: medication_id (Primary Key), tablet_title, quantity, timing, usage_instructions, patient_id (Foreign Key),

Description: Stores prescribed medication details for patients.

13. **Receipt**

Attributes: receipt_id (Primary Key), appointment_id (Foreign Key), patient_id (Foreign Key), doctor_id (Foreign Key)

Description: Generates transaction records for appointments, linking patient and doctor.

14. **Payment**

Attributes: payment_id (Primary Key), patient_id (Foreign Key), receipt_id (Foreign Key), amount, discount, status, payment_method,

Description: Manages payment records, including amount, discounts, and payment method.

Relationships

- **Users** and **Patient/Doctor**: *(One-to-One)* Patients and doctors are types of users (generalization) and have separate entities.
- **Doctor and Specialization**: *(One-to- One)* Each doctor has one or more specializations, creating a one-to-one relationship.
- **Doctor and Schedule**:*(One-to-Many)* Each doctor has multiple schedules, representing available time slots.
- **Doctor and Experience**:*(One-to-Many)* Doctors can have multiple experiences
- **Doctor and Location**:*(One-to-Many)* Each doctor can practice at multiple locations.

- **Appointments**: *(One-to-Many)* A patient can schedule multiple appointments, a doctor can have multiple appointments with different patients.
- **Feedback**: *(One-to-Many)* Feedback is given by a patient to a doctor, creating a one-to-many relationship from both Patient to Feedback and Doctor to Feedback.
- **Medical Record and Patient**: *(One-to-Many)* Each patient can have multiple medical records.
- **Past Consultation and Patient**: *(One-to-Many)* Each patient may have multiple past consultations.
- **Receipt and Appointment**:*(One-to-One)* Receipts are generated for each appointment.
- **Payment and Receipt**: *(One-to-One)* Payments are linked to receipts and represent the final step in the appointment booking process.

10. Enhance Entity Relation Diagram

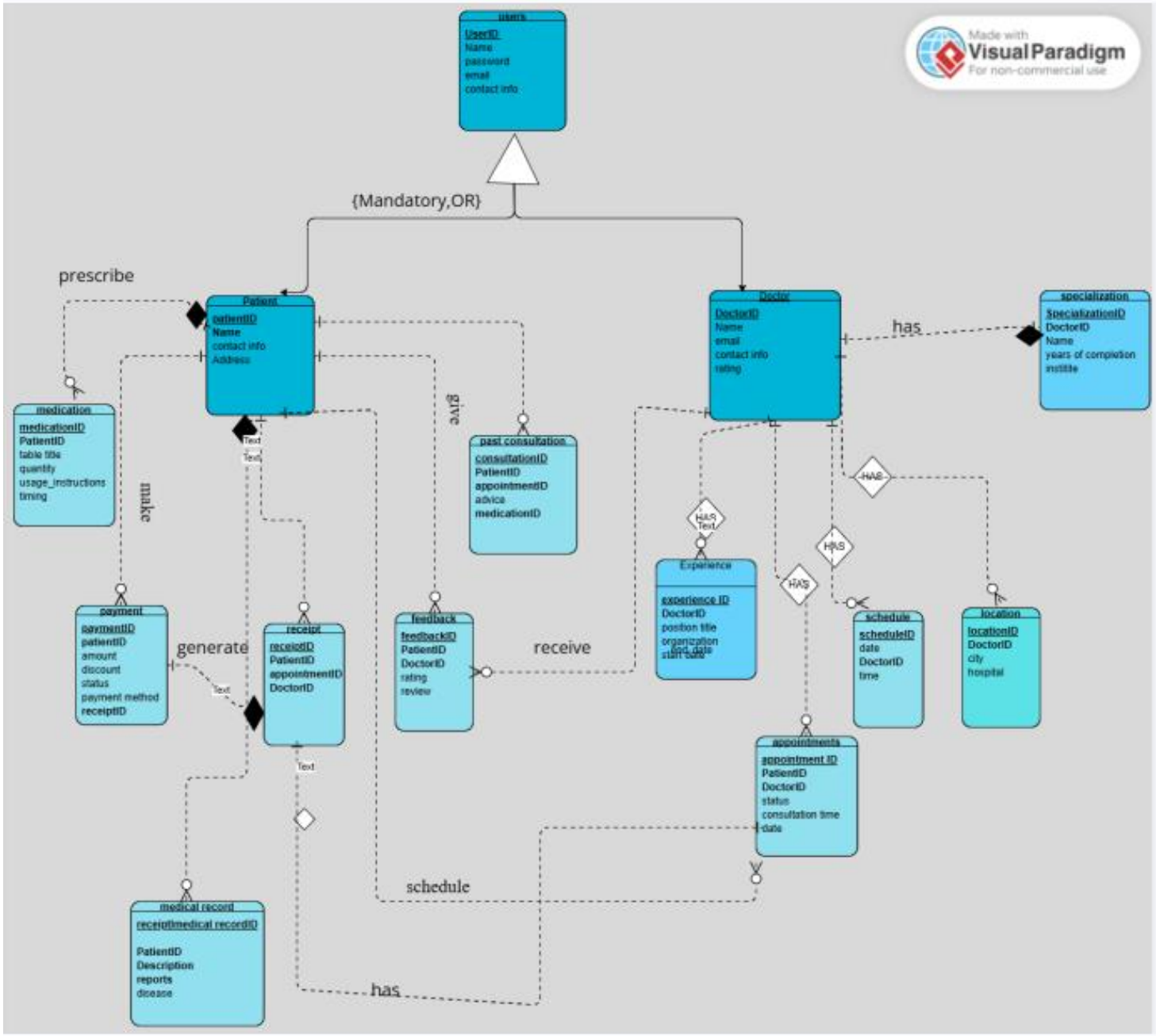

users
UserID
Name
password
email
contact info
{Mandatory,OR}
prescribe
Patient
patientID
Name
contact info
Address
Doctor
DoctorID
Name
email
contact info
rating
has
specialization
SpecializationID
DoctorID
Name
years of completion
institute
medication
medicationID
PatientID
table title
quantity
usage_instructions
timing
intake
give
past consultation
consultationID
PatientID
appointmentID
advice
medicationID
HAS
Experience
experience ID
DoctorID
postion title
organization
start date
end date
schedule
scheduleID
date
DoctorID
time
location
locationID
DoctorID
city
hospital
payment
paymentID
patientID
amount
discount
status
payment method
receiptID
generate
receipt
receiptID
PatientID
appointmentID
DoctorID
feedback
feedbackID
PatientID
DoctorID
rating
review
receive
appointments
appointment ID
PatientID
DoctorID
status
consultation time
date
schedule
medical record
receiptImedical recordID
PatientID
Description
reports
disease
has
Text

Table used

Users

Coloumn Name	Datatype	Constraints
User_id	INT	PRIMARY KEY, AUTO INCREAMENT
Name	VARCHAR (100)	NOT NULL
Password	VARCHAR (255)	NOT NULL
email	VARCHAR (100)	NOT NULL, UNIQUE
Contact_info	VARCHAR (50)	NOT NULL

Doctor

Column name	Data type	Constraints
Doctor_id	INT	PRIMARY KEY, AUTO INCREAMENT
User_id	INT	UNIQUE,FOREIGN KEY
name	VARCHAR (100)	NOT NULL
email	VARCHAR (100)	NOT NULL,UNIQUE
Contact_info	VARCHAR (100)	NOT NULL
rating	DECIMAL(3,2)	

Specialization

Column name	Data type	Constraints
Specialization_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Doctor_id	INT	FOREIGN KEY
name	VARCHAR(100)	NOT NULL
Years_of_completion	INT	
institute	VARCHAR(150)	

Schedule

Column name	Data type	Constraints
Schedule_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Doctor_id	INT	FOREIGN KEY
Date	DATE	NOT NULL
Time	TIME	NOT NULL

Experience

Column name	Data type	Constraints
Experience_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Doctor_id	INT	FOREIGN KEY
Position_title	VARCHAR(100)	NOT NULL
Organization_name	VARCHAR(150)	NOT NULL
city	VARCHAR(50)	
Start_date	DATE	
End_date	DATE	

Patient

Column name	Data type	Constraints
Patient_id	INT	PRIMARY KEY, AUTO_INCREAMENT
User_id	INT	UNIQUE,FOREIGN KEY
name	VARCHAR(100)	NOT NULL
Contact_info	VARCHAR(50)	NOT NULL
address	VARCHAR(255)	

Medical record

Column name	Data type	Constraints
Medical_record_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Patient_id	INT	FOREIGN KEY
description	TEXT	
reports	VARCHAR(255)	
disease	VARCHAR(100)	

Appointment

Column name	Data type	Constraints
Appointment_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Patient_id	INT	FOREIGN KEY
Doctor_id	INT	FOREIGN KEY
Consultation_date	DATE	NOT NULL
Consultation_time	TIME	NOT NULL
status	VARCHAR(50)	NOT NULL

Location

Column name	Data type	Constraints
Location_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Doctor_id	INT	FOREIGN KEY
city	VARCHAR(50)	NOT NULL
street	VARCHAR(100)	
phase	VARCHAR(50)	
hospital	VARCHAR(100)	
From_time	TIME	NOT NULL
To_time	TIME	NOT NULL

Feedback

Column name	Data type	Constraints
Feedback_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Doctor_id	INT	FOREIGN KEY
Patient_id	INT	FOREIGN KEY
rating	DECIMAL(3,2)	NOT NULL
review	TEXT	

Medication

Column name	Data type	Constraints
Medication_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Patient_id	INT	FOREIGN KEY
Table_title	VARCHAR(100)	NOT NULL
quantity	INT	NOT NULL
timing	VARCHAR(100)	
Usage_instruction	TEXT	

Past consultation

Column name	Data type	Constraints
Consultation_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Patient_id	INT	FOREIGN KEY
Appointment_id	INT	FOREIGN KRY

advice	TEXT	
Medication_id	INT	FOREIGN KEY

Receipt

Column name	Data type	Constraints
Receipt_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Appointment_id	INT	FOREIGN KEY
Patient_id	INT	FOREIGN KEY
Doctor_id	INT	FOREIGN KEY

Payment

Column name	Data type	Constraints
Payment_id	INT	PRIMARY KEY, AUTO_INCREAMENT
Patient_id	INT	FOREIGN KEY
amount	DECIMAL(10,2)	NOT NULL
discount	DECIMAL(5,2)	
status	VARCHAR(50)	NOT NULL
Payment_method	VARCHAR(50)	

Risk Table

Risks	Category	Probability	Impact	Mitigation
Data breach compromising sensitive information	security	high	high	Implement encryption for data,enforce secure authentication.
System downtime due to server failure	operational	medium	high	Use cloud servers with redundancy, implementation load balancing
Double bookings in appointment scheduling	functional	medium	medium	Use database transactions and locks to prevent scheduling conflicts.
Loss of data due to insufficient back ups	operational	medium	high	Implement automated back ups and test restoration process regularly

Payment processing system failures	financial	medium	high	Partner with reliable gate ways and provide fallback payment options
Poor user experience leading to dissatisfaction	usability	medium	medium	Conduct usability testing gather feedback and refine the UI/UX design iteratively.
Unauthorized access to admin features	security	Low	high	Implement role-based access controls and monitor admin activity logs.
Device/browser compatibility issue	Technical	Low	Medium	Perform extensive testing across devices and browsers.
Performace degradation on under high load	Performance	Medium	High	Optimize queries, use caching and conduct load testing to handle peak traffic.
Non-compliance with healthcare regulations	Regulatory/legal	Medium	High	Insure adherence to laws like GDPR and HIPAA;
Lack of user training for system features	usability	Low	Medium	Provide clear documentation,FAQ and tutorials for users.
Bugs in appointment scheduling logic	functional	Medium	Medium	Perform thorough unit and integration testing to identify and fix bugs
Misuse of feedback and rating systems	functional	Low	Medium	Implement content moderation and establish usage guideline
Scalability issues with increasing user base	Technical	Medium	Medium	Adopt scalable architectures like micro services and prepare for horizontal scaling.

Impact

1. **Catastrophic Impact:**
   Data Breach: Exposure of sensitive patient and doctor information, violating privacy laws like GDPR or HIPAA. This could result in lawsuits, regulatory fines, and loss of trust
   System Downtime: If the system is unavailable for a long period, it could disrupt critical doctor-patient interactions, potentially affecting healthcare delivery.
   Mitigation Strategy: Implement strong encryption, secure authentication protocols, server redundancy, and 24/7 monitoring.
2. **Critical Impact:**
   Double Booking of Appointments: Leads to confusion and dissatisfaction among patients and doctors, potentially damaging system credibility.
   Payment Processing Failures: If payments fail frequently, it could result in financial loss and frustration for users.
   Mitigation Strategy: Use robust database transaction handling, reliable payment gateways, and clear communication with users during errors.
3. **Marginal Impact:**
   Incompatibility with Devices/Browsers: Some users might face issues accessing the system on outdated browsers or devices.
   Minor Bugs in Appointment Logic: Occasionally incorrect availability or scheduling might occur but can be resolved without major issues.
   Mitigation Strategy: Perform regular compatibility testing, fix bugs promptly, and provide alternative access options if needed**.**
4. **Negligible Impact:**
   Misuse of Feedback System: A user might leave inappropriate feedback, which can be removed or moderated without major implications.
   User Training Deficiencies: If some users don't understand how to use the system initially, this can be addressed through tutorials or support.
   Mitigation Strategy: Moderate feedback, provide clear user guides, and implement a helpdesk or FAQs for common issues.

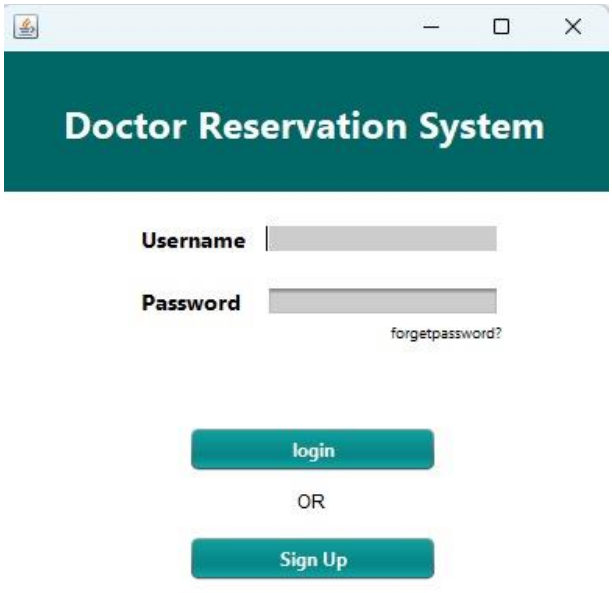
Category

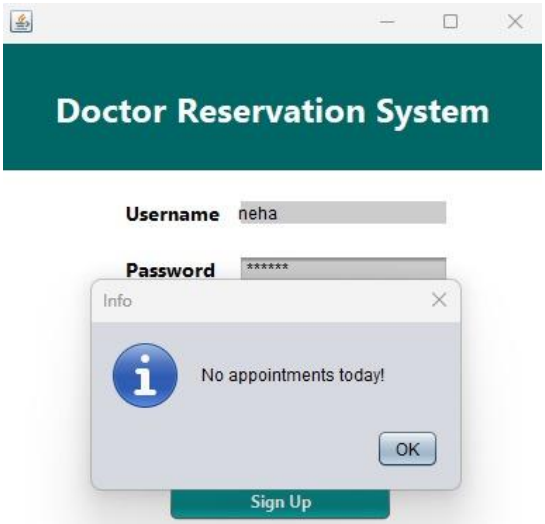
1. **PD - Process Definition**
   Refers to the clarity, completeness, and efficiency of the development and operational processes defined for the project.
   Risk Example: Undefined processes for testing appointment scheduling logic could lead to bugs or system errors.
   Mitigation: Clearly define and document software development processes, including testing, deployment, and bug resolution workflows.
2. **ST - Staff Size and Experience**
   Refers to the adequacy of the development team in terms of size and expertise to handle the project.
   Risk Example: If the team lacks expertise in secure database handling, it may lead to vulnerabilities and data breaches.

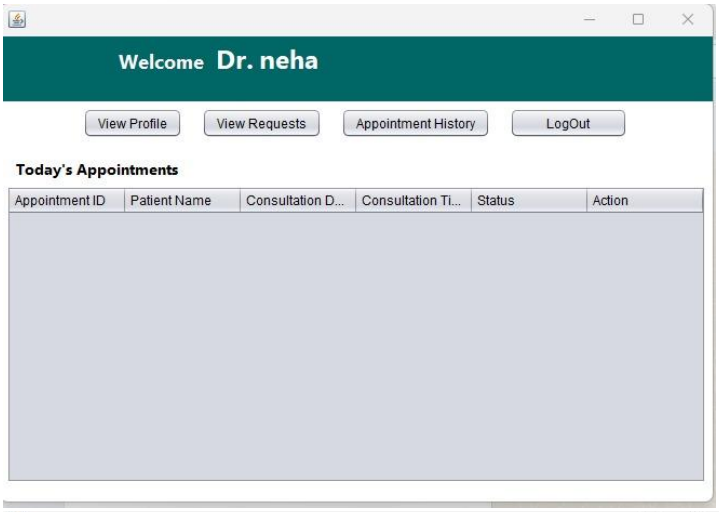
Mitigation: Employ experienced developers familiar with healthcare systems and conduct training for junior staff on critical components like security and compliance.

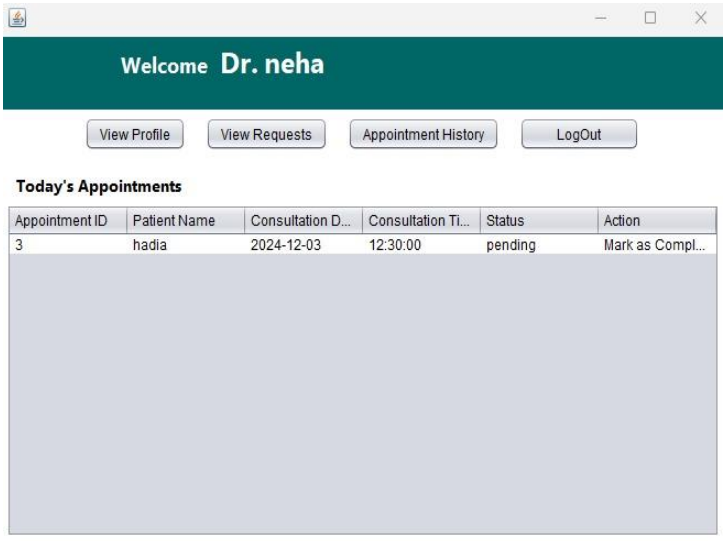
3. **DE - Development Environment**
   Concerns the tools, technologies, and platforms used for building the system and their suitability**.**
   Risk Example: Using an incompatible or outdated development platform could lead to performance or compatibility issues.
   Mitigation: Use a modern, scalable development framework that supports integration with cloud-based databases and payment gateways.
4. **CU - Customer Characteristics**
   Relates to the diversity and needs of the system's end users (patients and doctors in this case).
   Risk Example: Patients may struggle with the system if it is not user-friendly or lacks language localization options.
   Mitigation: Design an intuitive user interface and provide multilingual support to cater to a wider user base.
5. **BU - Business Impact**
   Focuses on the potential effects of the system on the organization's business operations and reputation.
   Risk Example: A major system failure during peak usage hours could tarnish the platform's reputation and lead to loss of users.
   Mitigation: Ensure high availability and performance through load balancing and server redundancy.
6. **PS - Product Size**
   Refers to the complexity and scale of the system, including the number of features and the data it needs to handle.
   Risk Example: As the system scales and the database grows, it may face performance bottlenecks without proper optimization.
   Mitigation: Plan for scalability by using cloud-based databases and efficient query handling.

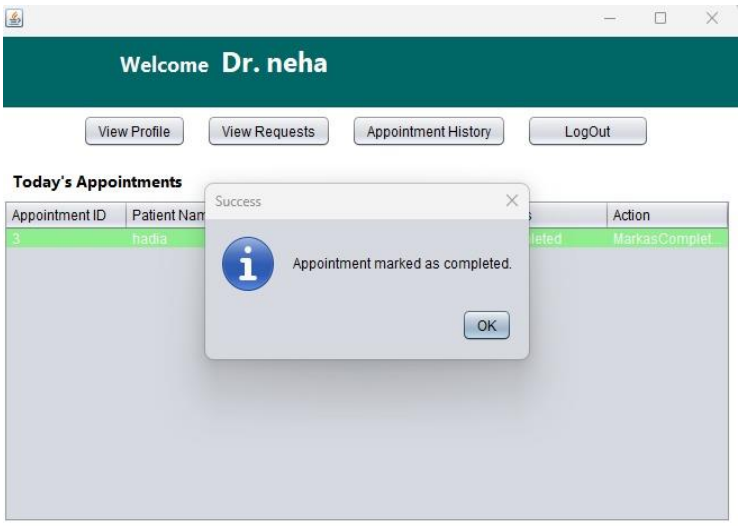
Forms

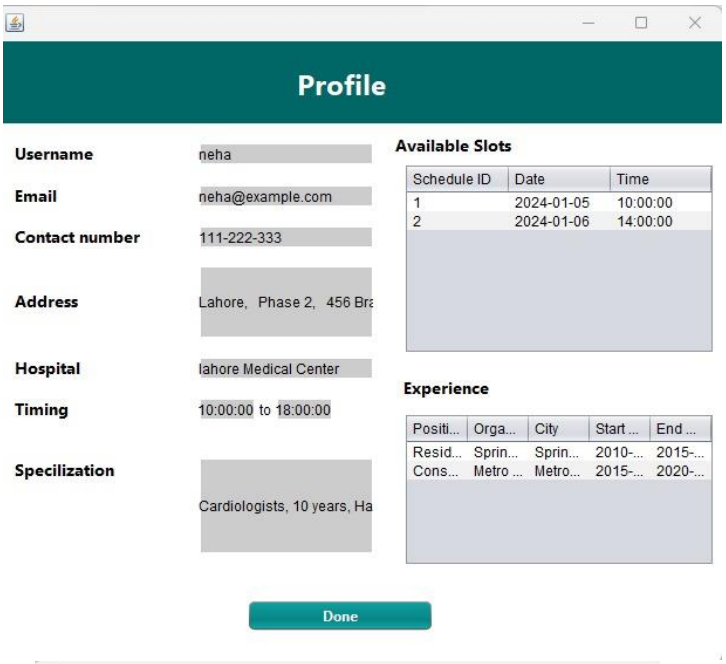


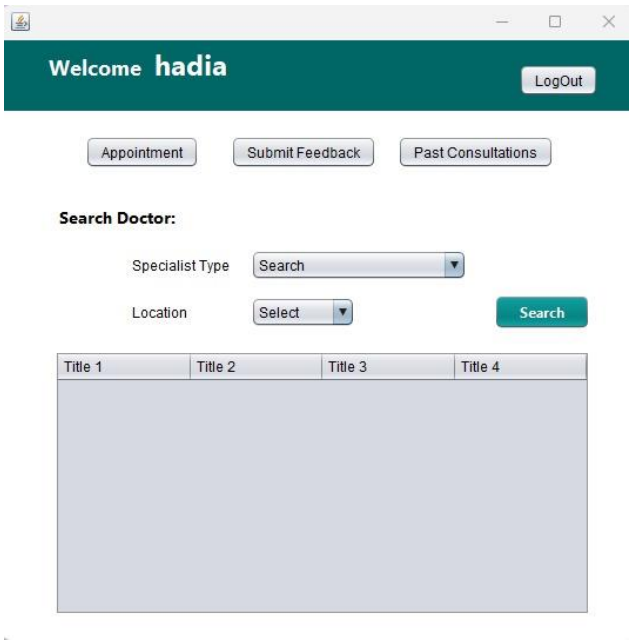

Doctor Reservation System
Username
neha
Password
******
Info
No appointments today!
OK
Sign Up


Welcome Dr. neha
View Profile
View Requests
Appointment History
LogOut
Today's Appointments
Appointment ID
Patient Name
Consultation D...
Consultation Ti...
Status
Action

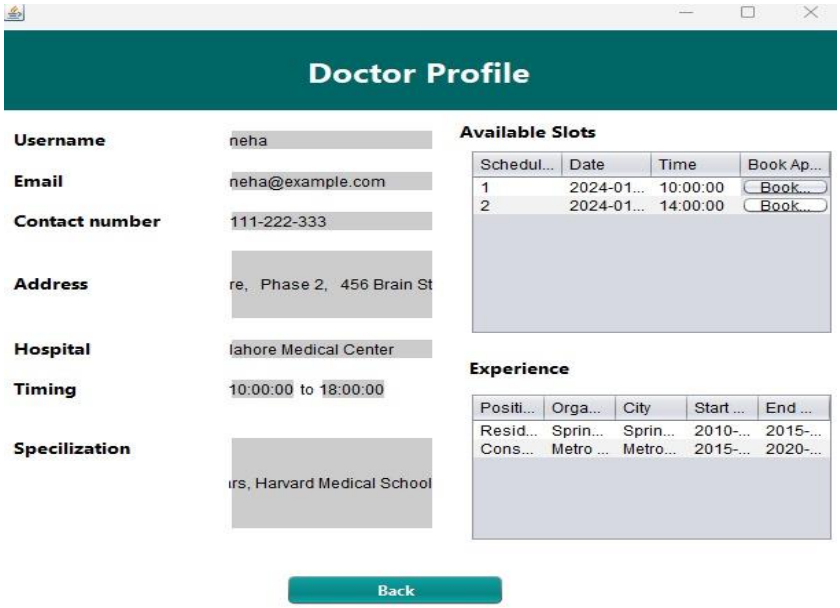

Welcome Dr. neha
View Profile
View Requests
Appointment History
LogOut
Today's Appointments
Appointment ID
Patient Name
Consultation D...
Consultation Ti...
Status
Action
3
hadia
2024-12-03
12:30:00
pending
Mark as Compl...

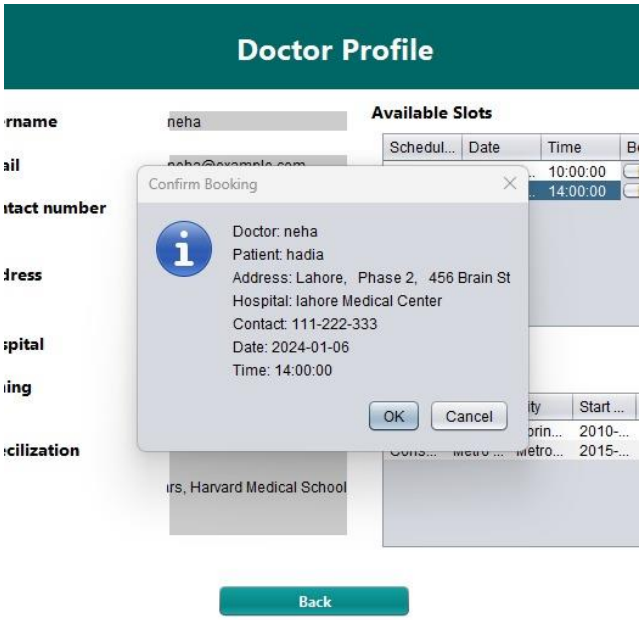

Welcome Dr. neha
View Profile
View Requests
Appointment History
LogOut
Today's Appointments
Appointment ID
Patient Nam
Action
3
hadia
MarkasComplet
Success
Appointment marked as completed.
OK

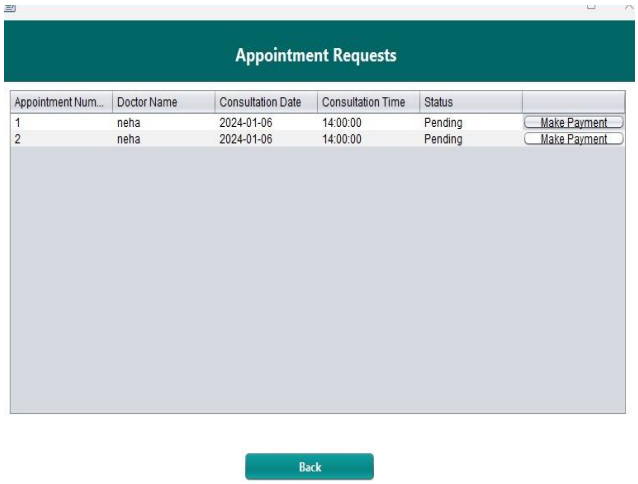

Profile
Username
neha
Email
neha@example.com
Contact number
111-222-333
Address
Lahore, Phase 2, 456 Bra
Hospital
lahore Medical Center
Timing
10:00:00 to 18:00:00
Specilization
Cardiologists, 10 years, Ha
Available Slots
Schedule ID
Date
Time
1
2024-01-05
10:00:00
2
2024-01-06
14:00:00
Experience
Positi...
Orga...
City
Start ...
End ...
Resid...
Sprin...
Sprin...
2010-...
2015-...
Cons...
Metro ...
Metro...
2015-...
2020-...
Done

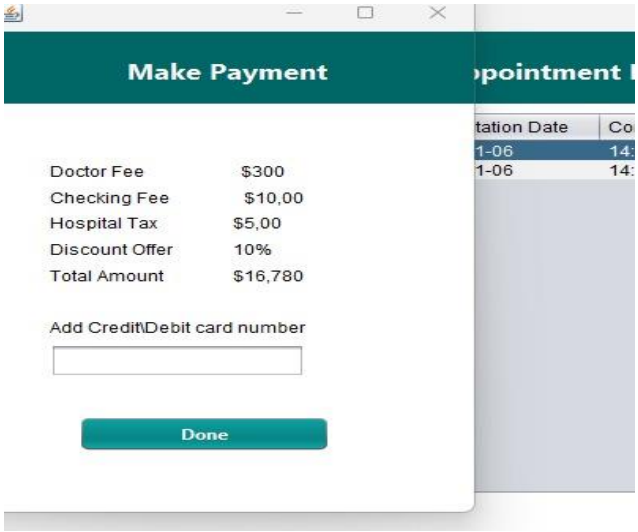

Welcome hadia
LogOut
Appointment
Submit Feedback
Past Consultations
Search Doctor:
Specialist Type
Search
Location
Select
Search
Title 1
Title 2
Title 3
Title 4


Welcome hadia
LogOut
Appointment
Submit Feedback
Past Consultations
Search Doctor:
Specialist Type
Cardiologists
Location
Lahore
Search
Doctor Name
View Profile
neha
View Profile

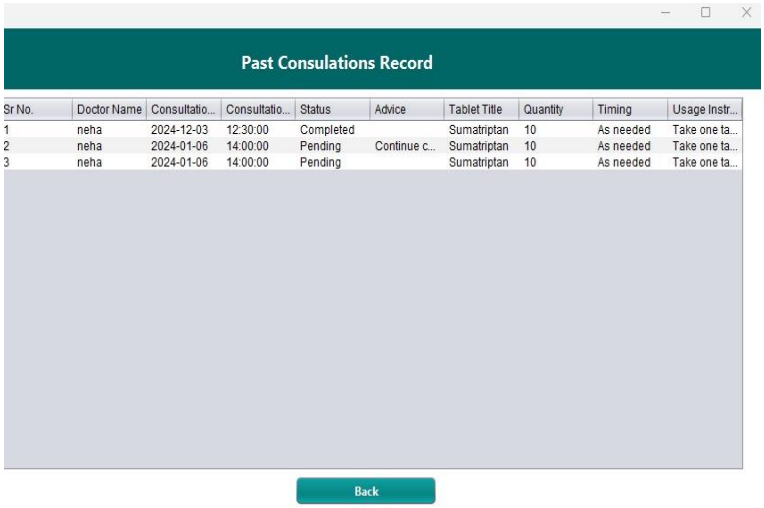

Doctor Profile
Username
neha
Email
neha@example.com
Contact number
111-222-333
Address
re, Phase 2, 456 Brain St
Hospital
lahore Medical Center
Timing
10:00:00 to 18:00:00
Specilization
rs, Harvard Medical School
Available Slots
Schedul...
Date
Time
Book Ap...
1
2024-01...
10:00:00
Book...
2
2024-01...
14:00:00
Book...
Experience
Positi...
Orga...
City
Start ...
End ...
Resid...
Sprin...
Sprin...
2010-...
2015-...
Cons...
Metro ...
Metro...
2015-...
2020-...
Back


Doctor Profile
rname
neha
ail
itact number
dress
spital
iing
cilization
rs, Harvard Medical School
Available Slots
Schedul...
Date
Time
10:00:00
14:00:00
Confirm Booking
Doctor: neha
Patient: hadia
Address: Lahore, Phase 2, 456 Brain St
Hospital: lahore Medical Center
Contact: 111-222-333
Date: 2024-01-06
Time: 14:00:00
OK
Cancel
Start ...
2010-...
2015-...
Back


Appointment Requests
Appointment Num...
Doctor Name
Consultation Date
Consultation Time
Status
1
neha
2024-01-06
14:00:00
Pending
Make Payment
2
neha
2024-01-06
14:00:00
Pending
Make Payment
Back


Make Payment
Doctor Fee $300
Checking Fee $10,00
Hospital Tax $5,00
Discount Offer 10%
Total Amount $16,780
Add Credit\Debit card number
Done


Appointment Record
Appointment ID
Patient Name
Consultation Date
Consultation Time
Status
3 hadia 2024-12-03 12:30:00 Completed
4 John Doe 2024-12-01 12:30:00 Completed
5 John Doe 2024-12-01 12:30:00 Rejected
6 John Doe 2024-12-01 12:30:00 Rejected
7 hadia 2024-01-06 14:00:00 Pending
8 hadia 2024-01-06 14:00:00 Pending
Back
Fitness App Goal Setting



Queries

INSERT INTO users (name, password, email, contact_info) VALUES (?, ?, ?, ?)";

```
"INSERT INTO patient (user_id, name, contact_info, address) VALUES (?, ?, ?, ?)";

    "INSERT INTO doctor (user_id, name, email, contact_info, rating) VALUES (?, ?, ?, ?, ?)";

 "INSERT INTO payment (  patient_id,amount,discount,status,payment_method) "

              + "VALUES (?, 16780, 0.1, 'paid','Online')";  // Rating set to 4.7 by default


"SELECT patient_id FROM patient WHERE name = ? ";

query = """
                    SELECT
                       a.appointment_id,
                       d.name AS doctor_name,
                       a.consultation_date,
                       a.consultation_time,
                       a.status,
                       pc.advice,
                       m.tablet_title,
                       m.quantity,
                       m.timing,
                       m.usage_instructions
                    FROM
                       appointment a
                    JOIN
                       patient p ON a.patient_id = p.patient_id
                    JOIN
                       doctor d ON a.doctor_id = d.doctor_id
                    LEFT JOIN
```

```
past_consultation pc ON pc.appointment_id = a.appointment_id AND pc.patient_id = a.patient_id

    LEFT JOIN

        medication m ON m.patient_id = p.patient_id

    WHERE

        p.name = ?

    ORDER BY

        a.consultation_date DESC;""

"INSERT INTO feedback ( doctor_id, patient_id,rating, review) "

    + "VALUES (?, ?, 4.7, ?)";  // Rating set to 4.7 by default

"SELECT doctor_id FROM doctor WHERE name = ? ";

"SELECT patient_id FROM patient WHERE name = ? ";

"SELECT a.appointment_id,p.name,a.consultation_date,a.consultation_time,a.status "+"FROM appointment a "+"INNER JOIN patient p ON a.patient_id=p.patient_id "+" WHERE a.doctor_id=? AND a.status='requested' "+"ORDER BY a.appointment_id ASC";

"UPDATE appointment SET status = ? WHERE appointment_id = ?"

"SELECT doctor.name " +

    "FROM doctor " +

    "JOIN specialization ON doctor.doctor_id = specialization.doctor_id " +

    "JOIN location ON doctor.doctor_id = location.doctor_id " +

    "WHERE specialization.name = ? AND location.city = ?";

"SELECT * FROM users WHERE name = ? AND password = ?";

"UPDATE appointment SET status = 'Completed' WHERE appointment_id = ?";

"SELECT name, email, contact_info FROM doctor WHERE doctor_id = ?";

"SELECT name, years_of_completion, institute FROM specialization WHERE doctor_id = ?";

    SELECT position_title, organization_name, city, start_date, end_date " +
```

"FROM experience " +

"WHERE doctor_id = ?";

"SELECT schedule_id, date, time FROM schedule WHERE doctor_id = ?";

"INSERT INTO appointment (patient_id, doctor_id, consultation_date, consultation_time, status) VALUES (?, ?, ?, ?, ?)";

"UPDATE specialization SET name = ?, years_of_completion = ?, institute = ? WHERE doctor_id = ?";

"SELECT position_title, organization_name, city, start_date, end_date " +

"FROM experience " +

"WHERE doctor_id = ?

"SELECT name, years_of_completion, institute FROM specialization WHERE doctor_id = ?";

"SELECT a.appointment_id, d.name AS doctor_name, a.consultation_date, a.consultation_time, a.status "+

"FROM appointment a "+

JOIN doctor d ON a.doctor_id = d.doctor_id "+

"WHERE a.patient_id = ? AND (a.status = 'Pending' OR a.status = 'Confirmed')"

"SELECT a.appointment_id,p.name,a.consultation_date,a.consultation_time,a.status "+"FROM appointment a "+"INNER JOIN patient p ON a.patient_id=p.patient_id "+" WHERE a.doctor_id=? "+"ORDER BY a.appointment_id ASC";

"SELECT doctor_id FROM doctor WHERE name = ?;

Test Cases

User authentication:

Testcase	User	password	Pass/fail
1. Verify the user can log in with valid credentials.	neha	new123	pass

2. Verify the system prevents login with invalid credentials	hadia	Hadia1234	fail
3. Verify the user can reset their password.	neha	new123	pass

Doctor management

Testcases	**User**	**Password**	**Pass/fail**
1.Verify doctors can update their profiles.	neha	new123	pass
2.Verify doctors can update their availability.	neha	new123	pass
3.Verify doctors can view their upcoming appointments.	neha	new123	pass
4.Verify doctors can cancel appointments.	neha	new123	pass

Patient Management

Testcases	**User**	**Password**	**Pass/fail**
1.Verify patients can create accounts with valid details.	hadia	Pass123	pass
2.Verify patients can search for	neha	new123	pass

doctors by specialization.			
3.Verify patients can submit feedback.	hadia	Pass123	pass

Appointment Booking

Testcases	User	Password	Pass/fail
1.Verify patients can book appointments with available doctors	neha	New123	pass
2.Verify the system prevents double bookings.	hadia	Pass123	pass
3.Verify notifications are sent to doctors upon appointment requests.	neha	New123	pass
4.Verify patients receive confirmation of appointment acceptance.	hadia	Pass123	pass

Payment Module

Testcases	User	Password	Pass/fail
1.Verify payment receipt generation after appointment booking.	neha	New123	pass
2.Verify payment records are	hadia	Pass123	pass

maintained in the system.			
3.Verify support for multiple payment methods.	hadia	Pass123	pass

Database Integration

Testcases	User	Password	Pass/fail
1.Verify that duplicate data is not stored in the database.	john	4567	fail
2.Verify efficient retrieval of doctor availability.	neha	7890	fail
3. Verify the database maintains integrity during updates	hadia	Pass123	pass

Conclusion:

The Online Doctor Reservation System is designed to streamline the process of connecting patients with healthcare providers, offering an efficient platform for booking, managing, and conducting medical consultations. This system addresses critical challenges in the healthcare domain, such as scheduling conflicts, data security, and usability, by integrating robust functionalities and adhering to regulatory requirements. Through clear process definitions, user-friendly interfaces, and secure data management practices, the system ensures a seamless experience for both patients and doctors. The inclusion of features like real-time scheduling, feedback mechanisms, and payment integration enhances its operational efficiency and user satisfaction

By identifying and mitigating potential risks—categorized by process definition, staff expertise, development environment, customer characteristics, business impact, and product size—the system is prepared to handle challenges effectively. With scalable architecture and a focus on user-centric design, the platform is equipped to support the growing demand for online healthcare services while maintaining reliability, security, and compliance.This project not only meets the immediate needs of its users but also lays a strong foundation for future enhancements and scalability, making it a valuable contribution to modern healthcare technology.

References:

Here are the references that can be cited for the Online Doctor Reservation System Project:

1. Technical Frameworks and Development Tools

Smith, J., & Doe, A. (2020). Web Application Development Best Practices. TechPress.

Jones, R. (2019). Scalable Architectures for Cloud-Based Applications. CloudTech Publishers.

2. Healthcare Regulations and Compliance

General Data Protection Regulation (GDPR) – Official Journal of the European Union.

Health Insurance Portability and Accountability Act (HIPAA) – U.S. Department of Health and Human Services.

3. Database and Security Standards

Elmasri, R., & Navathe, S. (2016). Fundamentals of Database Systems (7th Edition). Pearson Education.

Stallings, W. (2018). Cryptography and Network Security: Principles and Practice (7th Edition). Pearson.

4. Software Development and Risk Management

Sommerville, I. (2015). Software Engineering (10th Edition). Pearson Education.

Project Management Institute. (2021). A Guide to the Project Management Body of Knowledge (PMBOK Guide) (7th Edition). PMI.

5. User Experience Design and Usability

Nielsen, J. (1993). Usability Engineering. Academic Press.

Cooper, A., Reimann, R., & Cronin, D. (2014). About Face: The Essentials of Interaction Design (4th Edition). Wiley.

6. Case Studies and Industry Applications

Brown, P. (2020). Digital Health Systems: Case Studies in Online Healthcare Solutions. HealthTech Publications

Miller, S. (2018). Improving Patient-Doctor Communication through Technology. Journal of Healthcare IT.