F24-143-D-HealthBridge

Project Team

Muhammad Abdullah	21I-0643
Muneel Haider	21I-0640
Abdullah Zahoor	21I-2481

Session 2021-2025

Supervised by

Mr. Muhammad Aadil Ur Rehman

Co-Supervised by

Dr. Zeshan Khan

Department of Computer Science

National University of Computer and Emerging Sciences
Islamabad, Pakistan

June, 2025

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Pakistan's healthcare system consists of public and private sectors, which provide various healthcare services such as hospitals, clinics, and diagnostic centers. The public sector is affiliated with the federal and provincial governments that offer various services which is either free or heavily subsidized, with the main focus on the general population, especially those living in rural areas. On the other hand, the private sector provides a lot more special and highly individual care services relatively, but at a higher cost, through hospitals, clinics, and diagnostic centers.

In both sectors, the process of receiving healthcare begins with booking an appointment. For public hospitals, patients generally visit the outpatient department (OPD), where they ask for consultancy with doctors for a specific problem and register. Furthermore, the appointments are typically on first-come, first-served basis. Alternatively, private hospitals offer booking in advance via phone calls or an online management system which significantly reducing waiting times. After booking an appointment, the doctors may consult patients to get diagnostic tests with specialists before diagnosing a problem or a recommending a treatment.

For instance, if a doctor suspects that further diagnosis is necessary, they refer the patients for additional diagnostic tests such as blood tests, X-ray, MRI, CT scans. The patients then visit the diagnostic centers for taking tests and then receive a report accordingly depending on the scans which sometimes take a few hours or several days. When the reports are generated, the patient returns to the doctor and again wait for their turn to get a follow-up consultation from the doctor who reviews the findings and decides on the appropriate course of action or recommends a treatment process along with the prescription. This structured approach is consistent for both public and private healthcare systems nationwide, in order to ensure complete diagnosis and management to assist patients.

1.1 Existing Solutions

Several solutions have been developed in the recent years to cater such healthcare problems, especially for the patient-doctor communication, diagnostics, and appointment management to reduce further delays which affect many patients' health. These systems consist of advanced technologies such as telehealth, automated workflows, and artificial intelligence in order to address such challenges. Below is the detailed analysis of the most well-known applications that have tried to eliminate such problems.

Table 1.1: Comparison of Existing Solutions

System Name	System Overview	System Limitations
Oladoc	A digital healthcare service in Pakistan that offers appointment booking (via video call or physical).	Mostly not applicable in rural areas. Threat to data privacy.
Marham	Enables patients to book appointments with specific doctors for better communication.	Reliant on digital literacy and only used for appointment booking.
Qure.ai	Focuses on developing AI tools for CT and X-ray scans assistance. Enhances diagnostic accuracy for radiologists.	Focuses on X-ray and CT scans only and not MRI scans. Potential errors in complex scans.
HeartFlow	Provides an artery disease diagnosis using advanced imaging algorithms along with 3D artery models.	Restricted to only artery diseases and high cost.
Zebra Medical Vision	An AI-based analysis for brain bleeds and orthopedic scans which is beneficial for assisting radiologists.	Relies heavily on high-quality images and faces many challenges for a diverse healthcare system.
Aidoc	An AI-based platform for several specialties (cardiology and neurology) which provides accurate diagnosis.	Its only limited to specific specialties along with the high cost.

However, the above solutions are only effective in their own domains as they lack an all-in-one platform for addressing such challenges on a broader aspect, more particularly in regions like Pakistan especially for rural areas. Some of the key gaps include:

- AI-driven liver diagnostics for tumor and fatty liver detection with accurate results.
- Online Appointment scheduling and real-time consultations (chat system and video call) for better accessibility.
- Comprehensive support for patients especially in rural areas for enhanced healthcare on-time delivery.

By integrating such features in a single application, HealthBridge not only addresses such

limitations that already exist in these applications, but also enhances the overall efficiency, accessibility and provides better solutions for modern real-time problems.

1.2 Problem Statement

The healthcare system in Pakistan for both public and private sectors, face various problems which affect the overall quality and accessibility of the services. These challenges create difficulties for both patients and doctors in several ways. Most of the healthcare services in rural areas consist of public hospitals and clinics. These systems provide free medication and treatment for most of the patients due to which many patients prefer such medical services. However, these facilities are mostly inadequate as they provide very limited advanced diagnostics and specialized care generally.

Hospitals mainly in urban areas are mostly overcrowded as many patients are forced to wait in long queues which hinders the severity of the problem for patients. Hence, further endangering the health of the patients. Most of the hospitals especially in rural areas do not offer a pre-booking appointment system especially in underdeveloped areas. Alternatively, private hospitals appointment systems are more efficient but cannot be accessed through digital and mobile services.

Delays in diagnostic reports have caused several critical issues to the healthcare system whereby almost every hospital in the country would take a number of diagnostic reports and test results for each patient, mostly in the case of specialized procedures like MRI, CT scans and related diagnostics reports. Such reports are usually processed by the diagnostic centers within hours or sometimes even days, which leads to further delays to the whole process of diagnosing a problem and recommending some treatment for the patient.

Patients mostly visit doctors multiple times for consultation and follow-ups, mainly after receiving the test results that are previously recommended by the doctors. This increases the wait time and leads to more logistical challenges. Patients follow a navigated healthcare system, moving from doctors to specialists and diagnostic centers without getting a final result or a recommended treatment. Such system leads to inefficiencies as patient is unclear of the problem and the doctor remains confused.

Such problems highlight the need for improving the overall healthcare accessibility and efficiency in Pakistan. Although there are apps available for appointment booking and report generation, they are hosted on different platforms. Patients need to navigate separate apps to utilize these services, making the process more complicated and time-consuming. While AI-assisted diagnostic apps exist, very few focus on liver diseases, an area that requires more attention, especially given the rising prevalence of liver-related health issues.

1.3 Scope

HealthBridge is an all-in-one platform which aims to provide a liver-related medical diagnostics by integrating an AI-Powered healthcare services. The platform allows the users to book an appointment with their relevant doctors, whether for physical or video consultations, ensuring accessibility across all regions, including those with limited facilities. It features diagnostics specifically for liver diseases such Tumor, Cancer, and Fatty Liver (hepatic steatosis). These reports are generated in real-time with basic terminologies that are easy to comprehend. Such AI model also aims to assist healthcare professionals and patients. HealthBridge is meant to be accessible on web and mobile applications to ensure the ease of use for users.

This platform aims to reduce the delays which are often associated with traditional diagnostic methods by providing an instant report and treatment recommendations when necessary. Focusing exclusively on liver diagnostics, HealthBridge aims to deliver a centralized and efficient solution, ensuring the quality of healthcare is accessible and reliable for patients in need.

Table 1.2: Summary of Multimodal AI Healthcare Research

Reference	**Year**	**Objective**	**Model**	**Key Findings / Limitations**
Multimodal for Healthcare	2024	AI diagnostics in medical imaging	RAG (Retrieval Augmented Generation)	Improved diagnostic accuracy; limited dataset diversity and potential overfitting
Enhanced Diagnostics in Radiology	2022	Multimodal data integration in diagnostics	Multimodal Models	Enhanced diagnostic capabilities; high computational cost
Multimodal RAG for Medical Radiology	2021	Large Language Models in clinical support	GPT 3.5	Relevant medical text generation; relies on fine-tuning and risk of inaccuracies
Radiologic Assistant Model	2023	Fact-aware generation in radiology reports	Fact MM-RAG	Improved factual accuracy in reports; dependence on retrieval systems
Medical Vision Language Models	2023	Few-shot learning in clinical apps	Med-Flamingo	Effective with minimal data; struggles with specialized knowledge

1.4 Modules

The modules listed below focus on each specific aspect and features of the project in order to ensure a structured approach for the development. These modules comprise of web application, HealthBridge virtual assistant, diagnostic liver model, and android app interface for a comprehensive healthcare solution.

1.4.1 Web Application

- Signup Page: Design and implementation of Signup Pages, unique for every type of user.
- Login Page: Design and implementation of Login page, same for every type of user.
- Authentication: Implement necessary authentication methods using tools like JWT or OAuth2.
- Patient Interaction Platform: All pages related to searching, booking, and using AI Model.
- Doctor Dashboard and Services: All pages related to booking, communicating and using AI Model.
- Design Database: Designing our projects database, with ERDs, and relational schemas.
- Backend APIs: APIs will be made for our project and be used later to connect our webapp with the server.

1.4.2 HealthBridge Virtual Assistant

- Chatbot Page Design: Design a single page for the Chatbot, which will flow like a conversation until a result can be concluded.
- Connecting Chatbot: Our Chatbot will be fine-tuned and connected to our backend.

1.4.3 Diagnostic Liver Model

- Labelling: We will start labelling our datasets and prepare them for segmentation.
- Model Training: Develop the neural networks required to train our model.
- Segmentation: Create algorithms for automatic segmentation before proceeding with training and data cleaning.

1.4.4 Android App Interface

- Portal: Algorithms developed for communication between users (patient and doctor) will be improved.

- Android Application Development: Use existing APIs from the web app to develop Android application.
- Patient Interaction Platform: All pages related to searching, booking, and using AI Model.
- Doctor Dashboard and Services: All pages related to booking, communicating and using AI Model.
- Design Database: Designing our projects database, with ERDs, and relational schemas.

1.5 Work Division

Table 1.3: Work Division

Name	**Registration**	**Responsibility / Module / Feature**
Muneel Haider	21i-0640	Module 1 - Feat 3, 6; Module 1 - Feat 5; Module 3 - Feat 2; Module 4 - Feat 2; Module 4 - Feat 1
Muhammad Abdullah	21i-0643	Module 1 - Feat 1, 4, 5; Module 1 – Feat 4; Module 4 – Feat 3; Module 4 – Feat 2, 3
Abdullah Zahoor	21i-2481	Module 1 - Feat 2, 8; Module 1 – Feat 5; Module 3 – Feat 1; Module 4 – Feat 1; Module 4 – Feat 2, 3

Chapter 2

Project Requirements

The project requirements define the main functionalities for the HealthBridge system. These requirements are functional and non-functional requirements, which describe the system's core components and operations, performance and usability standards.

2.1 Use-case/Event Response Table/Storyboarding

This section describes how users will interact with the HealthBridge system through various use cases that define certain scenarios and the relevant system response. Each use case provides a detailed interaction for each user (patient, doctor, system admin) and highlights the pre-conditions, main success scenario, and post conditions for the overall flow of events.

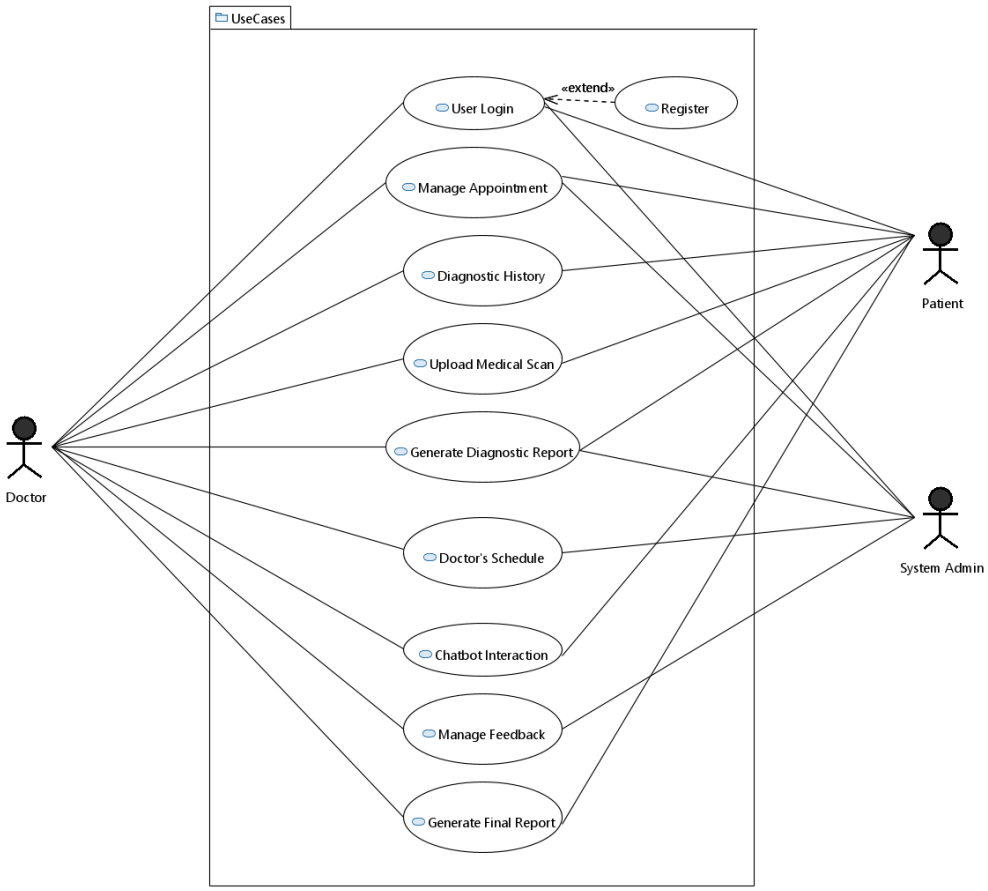

Figure 2.1: Use Case Diagram of HealthBridge System

Use-case 1: User Registration and Login

- **Scope:** HealthBridge System
- **Level:** User goal
- **Primary Actor:** Patient, Doctor, Institute Admin
- **Stakeholders and Interests:**
  - Patient: Secure access to book an appointment and upload scans.
  - Doctors: Access to schedule appointments and download assistive reports for patients safely.
  - System Admin: To ensure that the platform is available for authorized users.
- **Pre-condition:**
  - Users do not have an account on the platform.
  - Invalid credentials.
- **Post-condition:** Users are successfully registered and logged in and are redirected to their dashboards.
- **Main success scenario:**
  1. User navigates to the login/registration page.
  2. User enters credentials (email, CNIC, password).
  3. The system validates the entered credentials.
  4. If valid, the user is redirected to their dashboard.
- **Extensions:**
  - CNIC/email already exists – error message.
  - Invalid credentials – error message.
  - Forgotten password – option to reset.
- **Special Requirements:** Encrypted password storage.
- **Technology and Data Variations:** Input – CNIC, password; Output – success/failure message.

Use-case 2: Schedule Appointments

- **Scope:** HealthBridge System (Appointment Scheduling)
- **Level:** User goal
- **Primary Actor:** Patient, Doctor, System Admin
- **Stakeholders and Interests:**
  - Patient: Can efficiently book, cancel, or reschedule appointments.
  - Doctors: Manage schedules for consultations.
  - System Admin: Ensure seamless appointment scheduling.
- **Pre-condition:**
  - Users must be logged in.
  - Patients view available doctor slots.
- **Post-condition:** Appointment booked and both parties notified.
- **Main success scenario:**
  1. Patient navigates to "Book Appointment".
  2. System displays doctors and their schedules.
  3. Patient selects slot, enters payment info.
  4. System validates payment, confirms booking.
  5. Updates doctor's schedule and notifies both users.
- **Extensions:**
  - Only available slots shown.
  - Cancelled slots are freed up.
- **Special Requirements:** Real-time updates, confirmation notifications.
- **Technology and Data Variations:** Doctor/time selection and appointment confirmation.

Use-case 3: Review Diagnostic Records

- **Scope:** HealthBridge System
- **Level:** User goal
- **Primary Actor:** Patient, Doctor, Institute Admin
- **Stakeholders and Interests:**
  - Patient: Upload and review diagnostic reports.
  - Doctors: View patient history.
  - Institute Admin: Access historical records.
- **Pre-condition:** Logged-in users; patient uploads reports.
- **Post-condition:** All parties access uploaded reports.
- **Main success scenario:**
  1. Patient uploads reports.
  2. System validates and stores them.
  3. Doctors/Admins view reports.
- **Extensions:**
  - No file = error notification.
  - Invalid file path = error message.
- **Special Requirements:** Secure file storage.
- **Technology and Data Variations:** Uploaded medical history files.

Use-case 4: Provide Medical Scans

- **Scope:** HealthBridge System
- **Level:** User goal
- **Primary Actor:** Patient, Doctor
- **Stakeholders and Interests:**
  - Patient: Upload scans for diagnosis.
  - Doctor: Review scan-based diagnostic reports.

  - System Admin: Use scans for future referencing.

- **Pre-condition:** Logged-in users; scans ready.
- **Post-condition:** AI generates diagnostic report.
- **Main success scenario:**
  1. Patient uploads scan.
  2. System checks file format.
  3. AI processes scan.
  4. Diagnostic report generated and shared.
- **Extensions:** Unsupported format = error.
- **Special Requirements:** Secure scan upload.
- **Technology and Data Variations:** DICOM, NII, image formats.

Use-case 5: Generate Diagnostic Report

- **Scope:** HealthBridge System
- **Level:** User goal
- **Primary Actor:** Patient, Doctor
- **Stakeholders and Interests:**
  - Patient: Receives AI-based diagnostics.
  - Doctor: Uses report for consultation.
  - Admin: Monitors accuracy.
- **Pre-condition:** Scan successfully uploaded.
- **Post-condition:** Report generated and distributed.
- **Main success scenario:**
  1. AI processes scan.
  2. Report generated using pre-trained model.
  3. Sent to doctor and patient.
- **Extensions:** AI failure = error message.
- **Special Requirements:** High AI accuracy.
- **Technology and Data Variations:** Scanned data, generated report.

Use-case 6: Doctor's Schedule

- **Scope:** HealthBridge System
- **Level:** User goal
- **Primary Actor:** Doctor, Patient
- **Stakeholders and Interests:**
  - Doctor: Define availability.
  - Patient: View available slots.
  - Admin: Ensure real-time sync.
- **Pre-condition:** Doctor logged in and accessing schedule.
- **Post-condition:** Slots updated.
- **Main success scenario:**
  1. Doctor selects 'Manage Schedule'.
  2. Adds/edits time slots.
  3. System confirms changes.
- **Extensions:** Conflicts = error.
- **Special Requirements:** Real-time updates.
- **Technology and Data Variations:** Date/time slots.

Use-case 7: Chat Assistant

- **Scope:** HealthBridge System
- **Level:** User goal
- **Primary Actor:** Patient, Doctor
- **Stakeholders and Interests:**
  - Patient: Ask queries, receive suggestions.
  - Admin: Maintain uptime.
- **Pre-condition:** User logged in.
- **Post-condition:** Chat completed successfully.

- **Main success scenario:**
  1. User initiates chat.
  2. Sends query.
  3. Assistant replies in real time.
- **Extensions:** System delay = error.
- **Special Requirements:** Relevant, fast replies.
- **Technology and Data Variations:** Prompt + assistant reply.

Use-case 8: Provide Feedback

- **Scope:** HealthBridge System
- **Level:** User goal
- **Primary Actor:** Patient, Doctor, Admin
- **Stakeholders and Interests:**
  - Patient/Doctor: Submit feedback.
  - Admin: Analyze for improvement.
- **Pre-condition:** Logged in and completed session.
- **Post-condition:** Feedback saved.
- **Main success scenario:**
  1. User selects feedback option.
  2. Submits comments and ratings.
  3. System stores it.
- **Extensions:** Storage failure = retry prompt.
- **Special Requirements:** Must store per-user data.
- **Technology and Data Variations:** Ratings/comments.

Use-case 9: Generate Final Report

- **Scope:** HealthBridge System
- **Level:** User goal
- **Primary Actor:** Patient, Doctor, AI Model
- **Stakeholders and Interests:**
  - Patient: Receives verified report.
  - Doctor: Finalizes diagnosis.
  - AI Model: Provides supporting data.
- **Pre-condition:** Scan uploaded and AI report generated.
- **Post-condition:** Final report available.
- **Main success scenario:**
  1. AI generates report.
  2. Doctor verifies and adds notes.
  3. System delivers final version to patient.
- **Extensions:** Incomplete or ambiguous reports trigger reprocessing.
- **Special Requirements:** Must include full diagnosis + treatment steps.
- **Technology and Data Variations:** AI report + doctor's input.

2.2 Functional Requirements

Module 1: Web Application

- The user is successfully able to register and login to the platform.
- The user is properly authenticated before being able to access the platform.
- The user's respective pages have all use cases implemented.

Module 2: User Management System

- The database communicates with the web application.
- APIs cannot be accessed outside of our web application.

Module 3: HealthBridge Virtual Assistant

- The chatbot responds according to the user's prompt.
- Successfully communicates with backend.
- Accurately suggests recommended doctors.
- Can process various JSON body formats for future AI integration.

Module 4: Diagnostic Liver Model

- Datasets are correctly labeled.
- Returns result in 2 formats: highlighted image + JSON body.

Module 5: Android App Interface

- Communicates with backend.
- Remains synchronized with the web application.

2.3 Non-Functional Requirements

Reliability

- AI model accuracy above 80%.
- Accurate doctor recommendations based on problem input.

Usability

- By implementing fail-safe mechanisms and try-catch error handling, the application will remain operational and will not crash when encountering errors.
- Auto-logout after 15 minutes of inactivity unless user selects permanent login.

Performance

- The system employs structured API schemas for the authentication context rather than simple endpoints.

- Supports 25 users within 5 seconds.
- The Virtual Assistant shall present a brief questionnaire and, based on user responses, recommend the most appropriate doctor within seconds.

Security

- All sensitive data encrypted.
- Authorization ensures only permitted feature access.
- Secure authentication with valid token sessions.

Chapter 3

System Overview

The System Overview provides a more comprehensive description of the HealthBridge System. It showcases how several components of the system work using architecture design, data design, domain model, activity diagram, data-flow diagram, system-level sequence diagrams, and state transition diagram. These mentioned diagrams highlight the integration of modules like web application, liver diagnostic model, and virtual assistant in order to follow a systematic approach.

3.1 Architectural Design

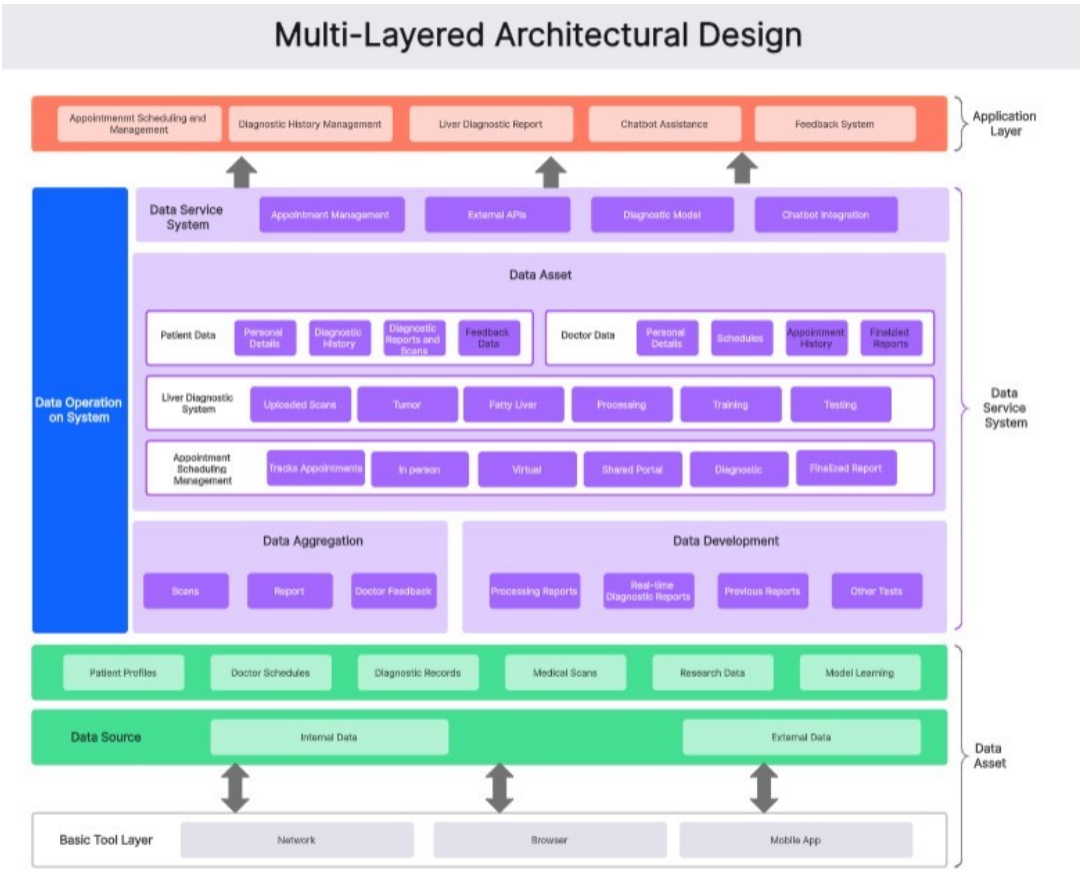

Figure 3.1: Multi-Layered Architectural Design of HealthBridge System

3.2 Data Design

The HealthBridge system utilizes a MongoDB database to manage the various entities involved in the online appointment booking process, video consultations, AI-based disease detection, and automated report generation. MongoDB's flexibility and scalability are well-suited for managing the complex and dynamic nature of healthcare data, such as patient records, doctor schedules, and AI-generated reports.

3.2.1 Appointment Data

- **appointmentid**: Unique identifier for the appointment.
- **doctorid**: Refers to the doctor assigned.
- **patientid**: Refers to the booking patient.
- **astatus, aaccepted**: Manage statuses.
- **adate, atime**: Date and time.
- **appointmenttype**: "Regular Checkup" or "Video Consultation"
- **areminder**: Automated reminders.
- **paymentid**: Links to payment.

3.2.2 Doctor Data

- **doctorid**: Unique identifier.
- **name, email, phone, specialization**
- **dConsultFee, videoCallFacility**
- **dOperatingDay, workingslots**

3.2.3 Patient Data

- **patientid**: Unique identifier.
- **name, email, phone, address, city**
- **medicalHistory, allergies, medicationList**
- **appointmenthistory**

3.2.4 Prescription and Reports

- **prescriptionid**, **doctorid**, **patientid**, **appointmentid**

- **prescriptionStatus**, **Prescription**, **prescriptionCategory**

- **reportid**, **reportDocument**, **reportSummary**, **reportReviewer**, **diseaseDetectionResult**

3.2.5 Diagnostic Model

HealthBridge's AI model for liver disease detection accepts MRI/CT scans and generates automated diagnostic results stored in **AI disease detection result** field in the report collection.

3.2.6 Conclusion

The HealthBridge system leverages MongoDB collections such as appointments, doctors, patients, and payments with embedded AI and third-party integrations for complete healthcare delivery.

3.3 Domain Model

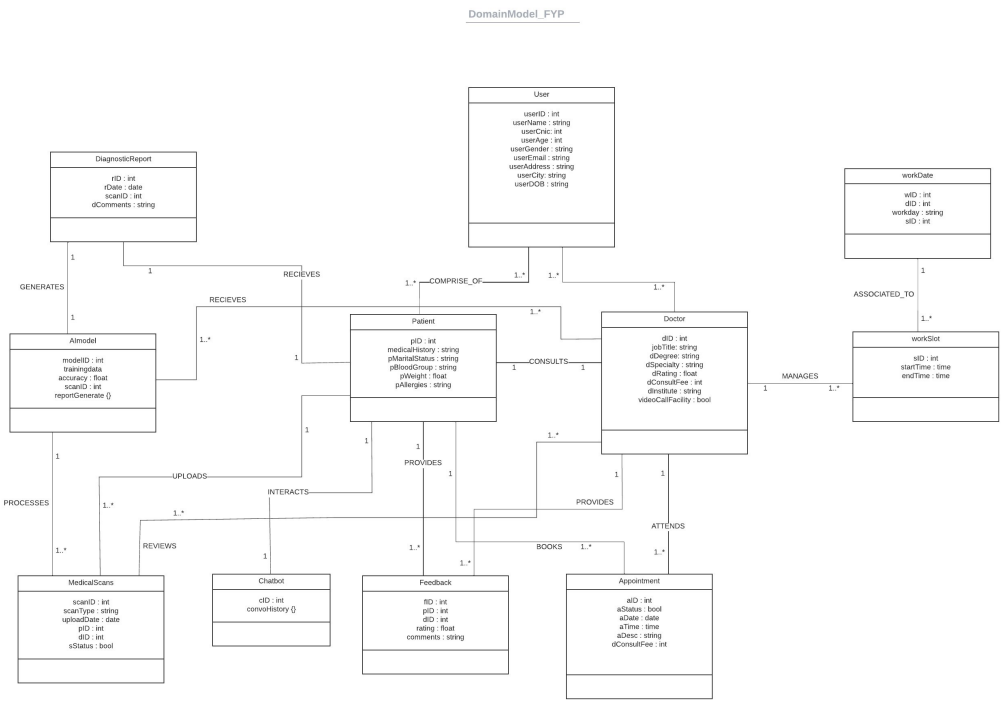

Figure 3.2: Domain Model of HealthBridge System

3.4 Design Models [Up to the Current Iteration]

3.4.1 Activity Diagram

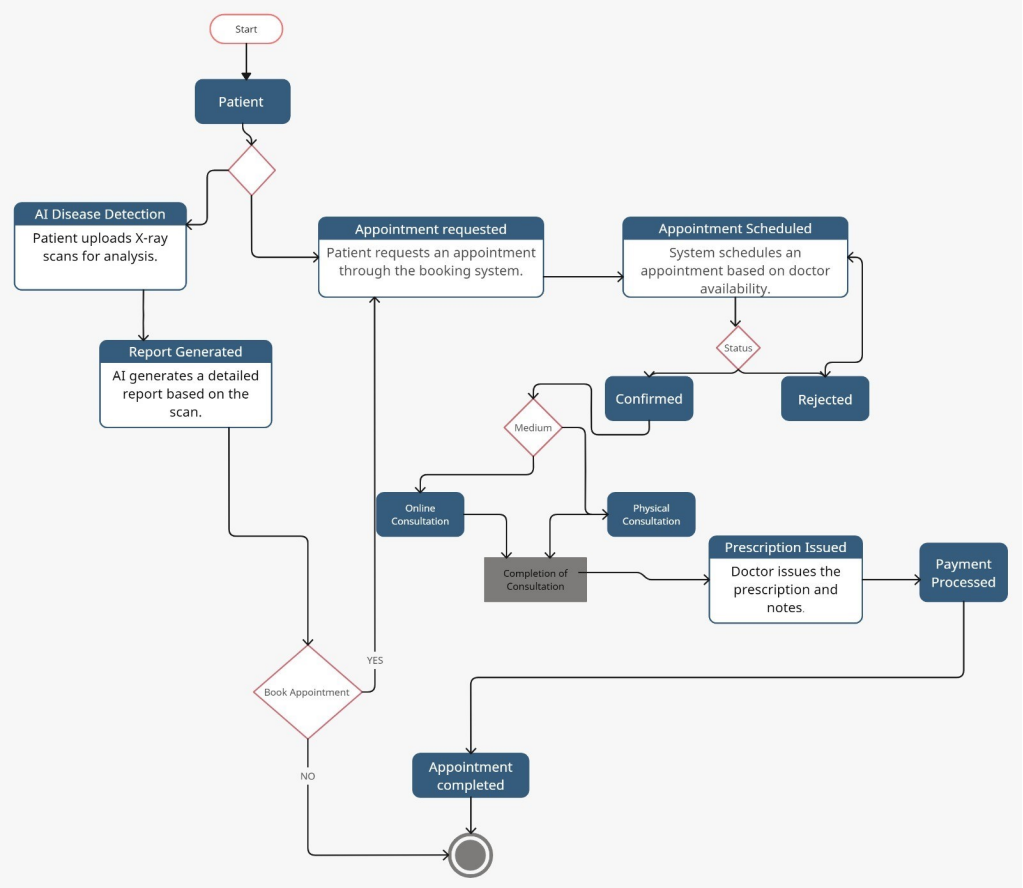

Figure 3.3: Activity Diagram of HealthBridge Workflow

3.4.2 Data Flow Diagram

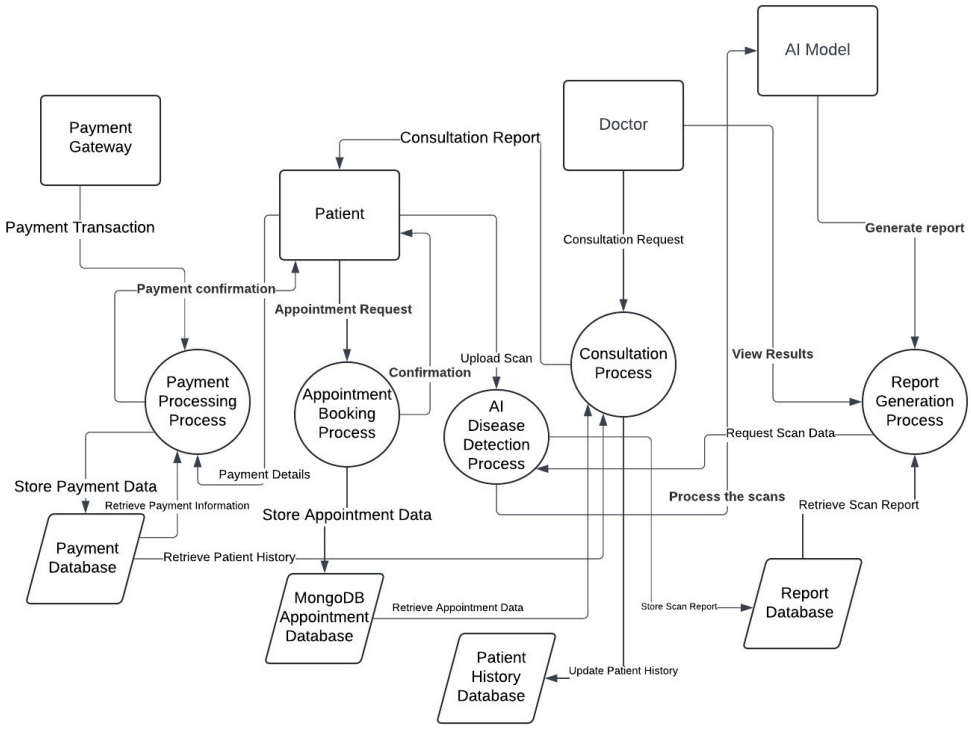

Figure 3.4: Data Flow Diagram of HealthBridge System

3.4.3 System-Level Sequence Diagrams

Login

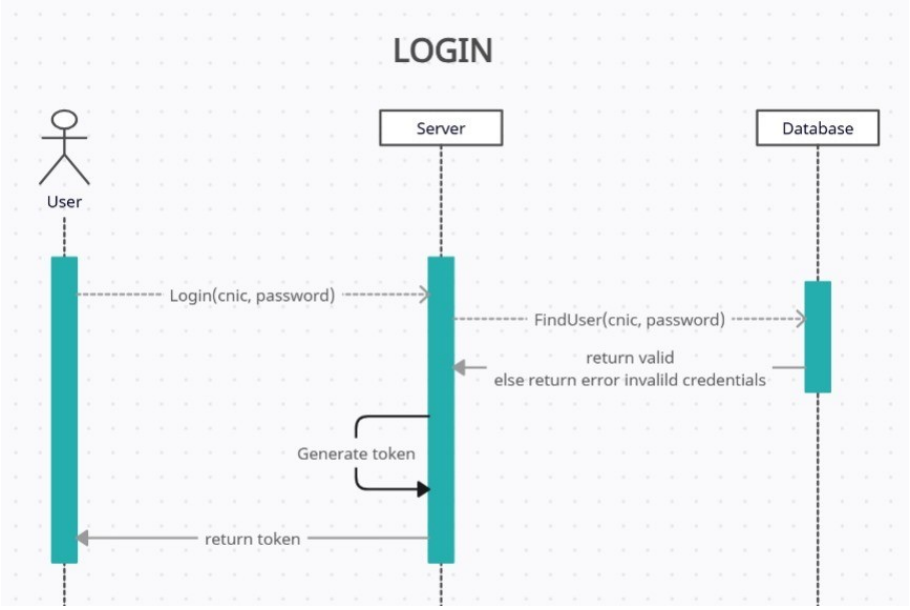

Figure 3.5: Login Sequence

Registration

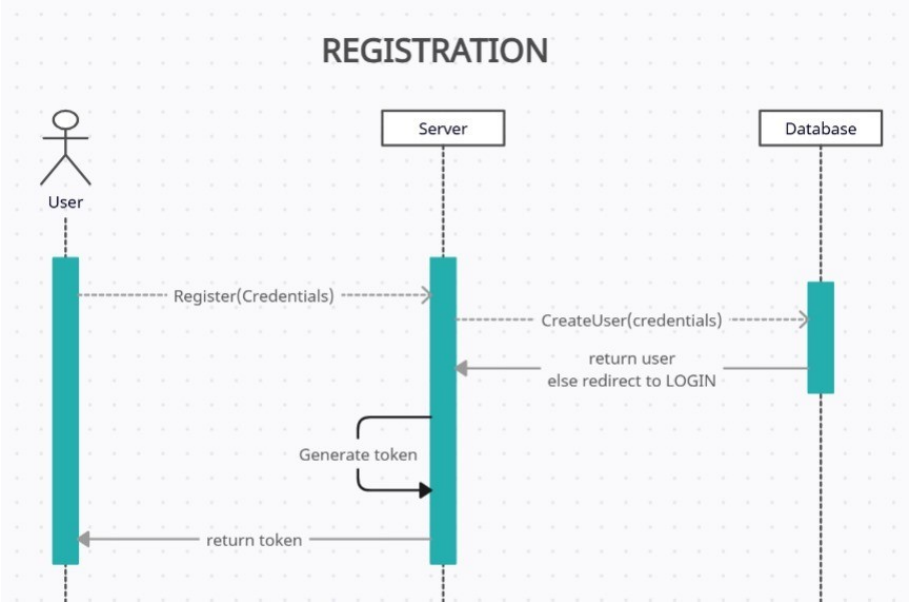

Figure 3.6: Registration Sequence

Give Feedback

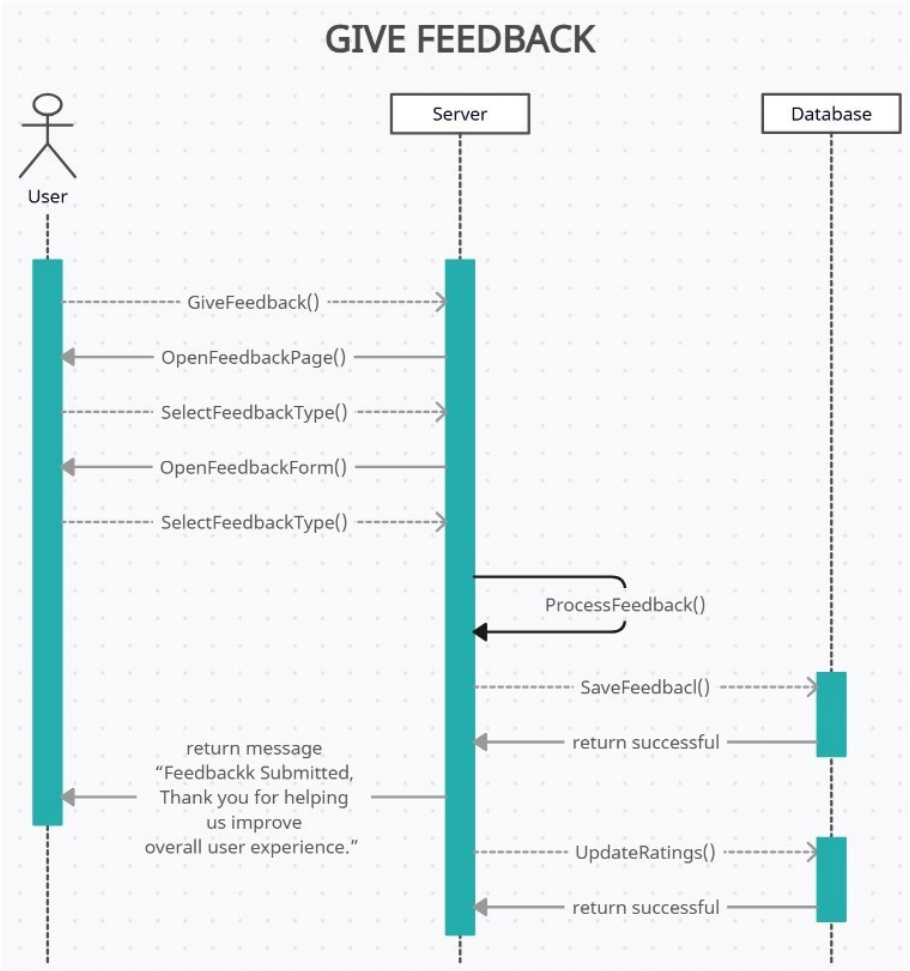

Figure 3.7: Give Feedback Sequence

Book Appointment

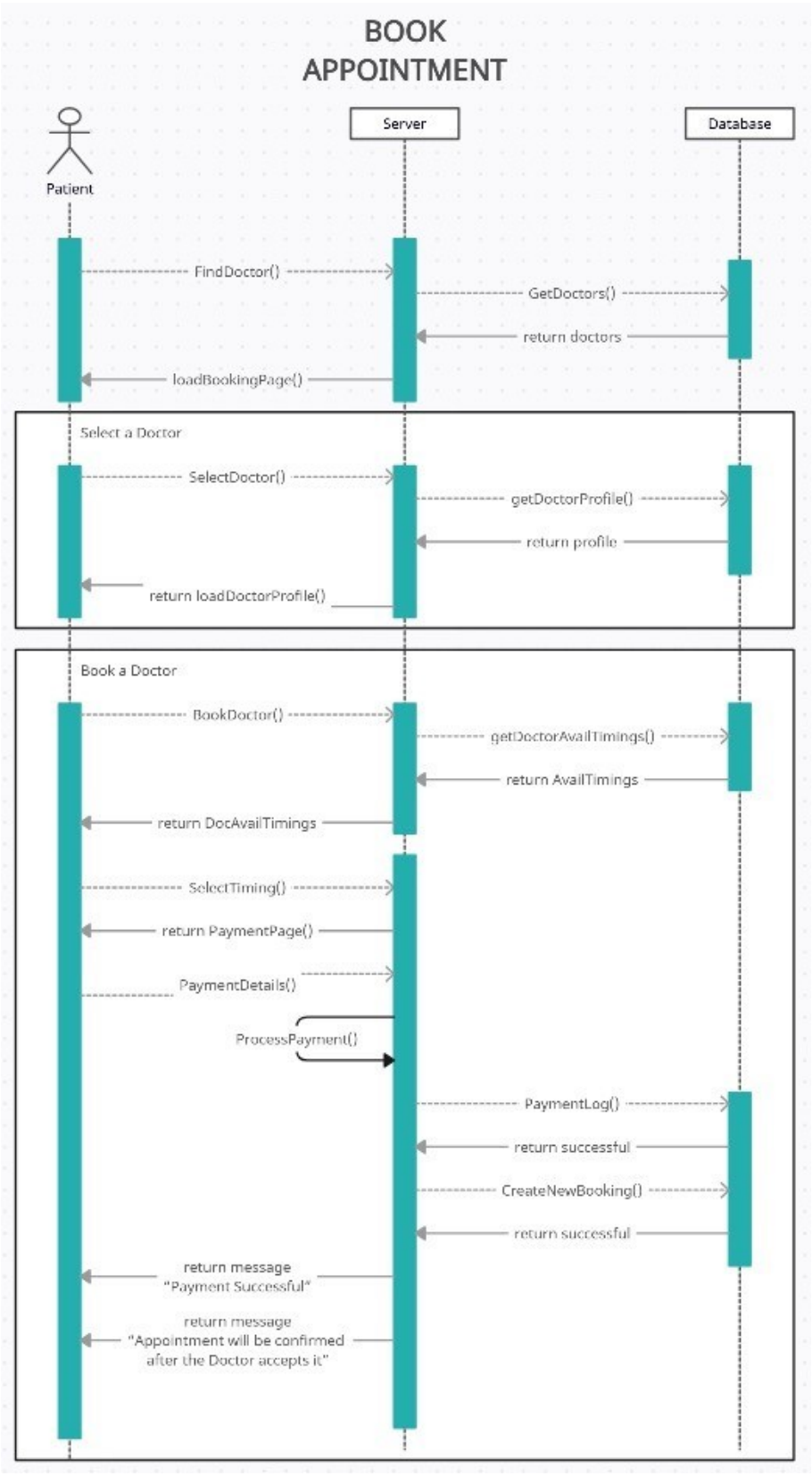

Figure 3.8: Book Appointment Sequence

Chatbot Interaction

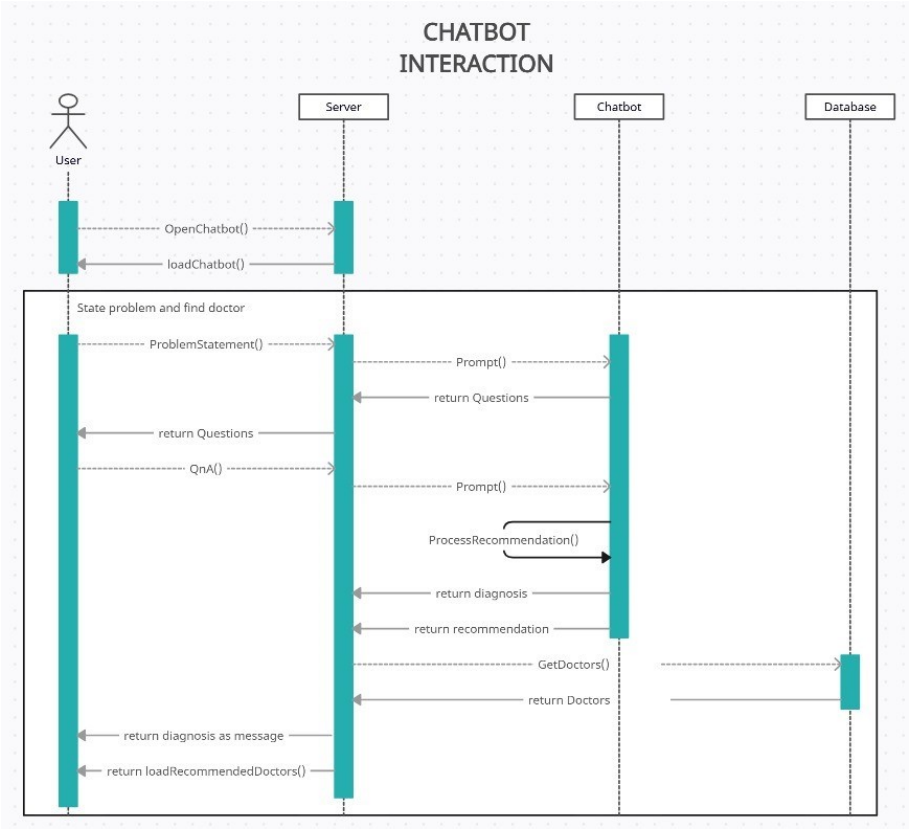

Figure 3.9: Chatbot Interaction Sequence

Diagnostic History

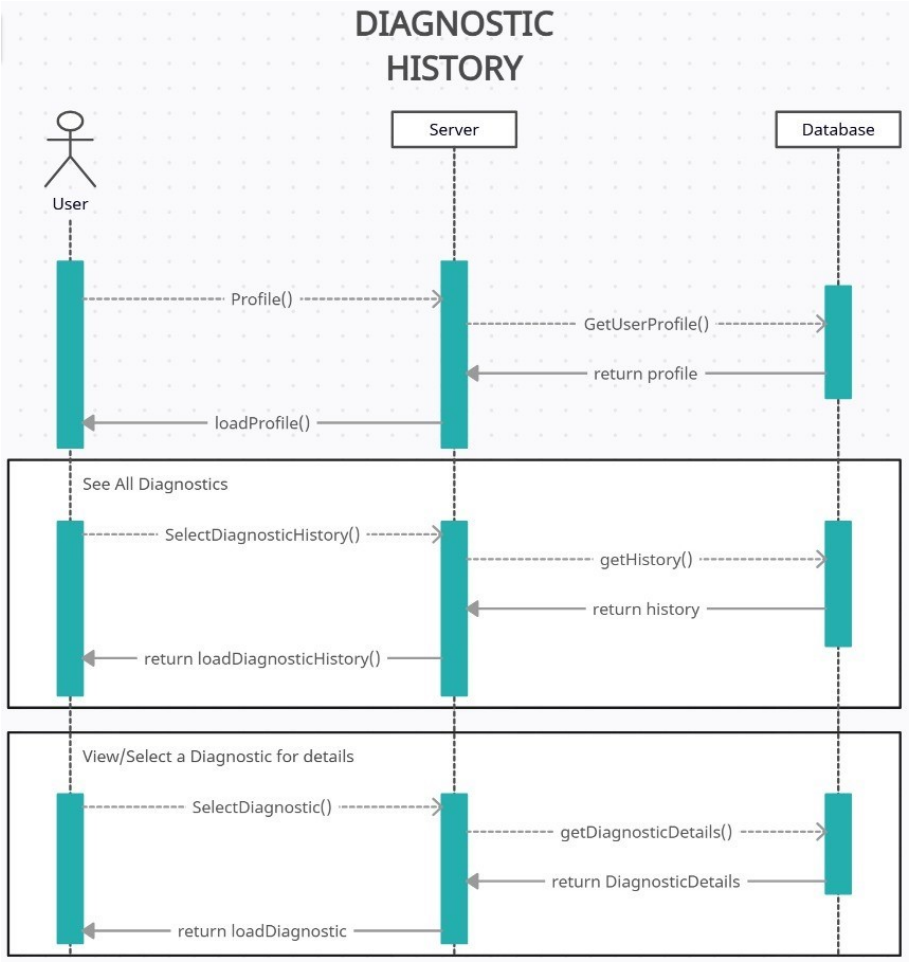

Figure 3.10: Diagnostic History Sequence

3.4.3.1 Manage Appointments

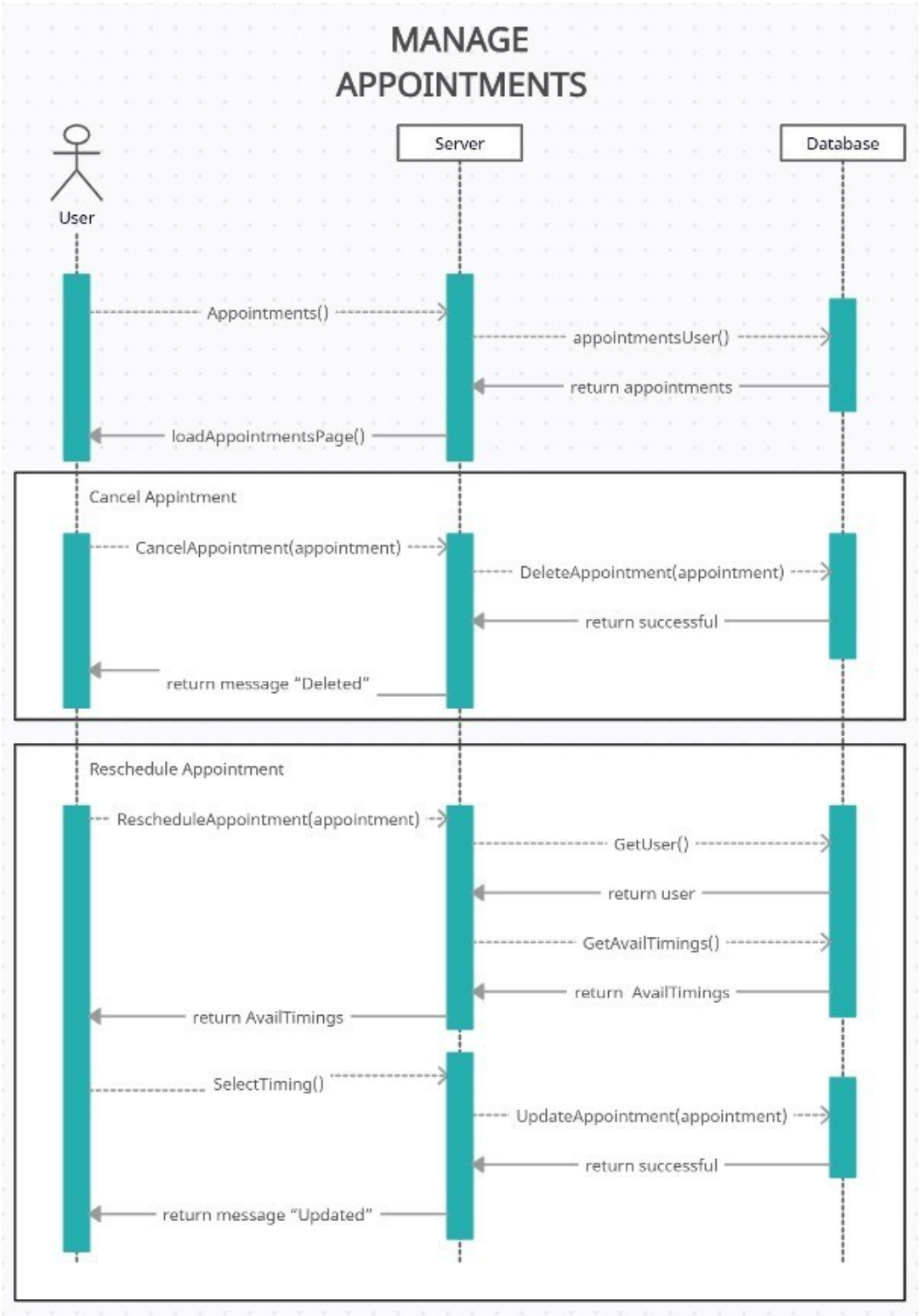

Figure 3.11: System-Level Sequence Diagram: Manage Appointments

3.4.3.2 Upload Scans and Generate Report

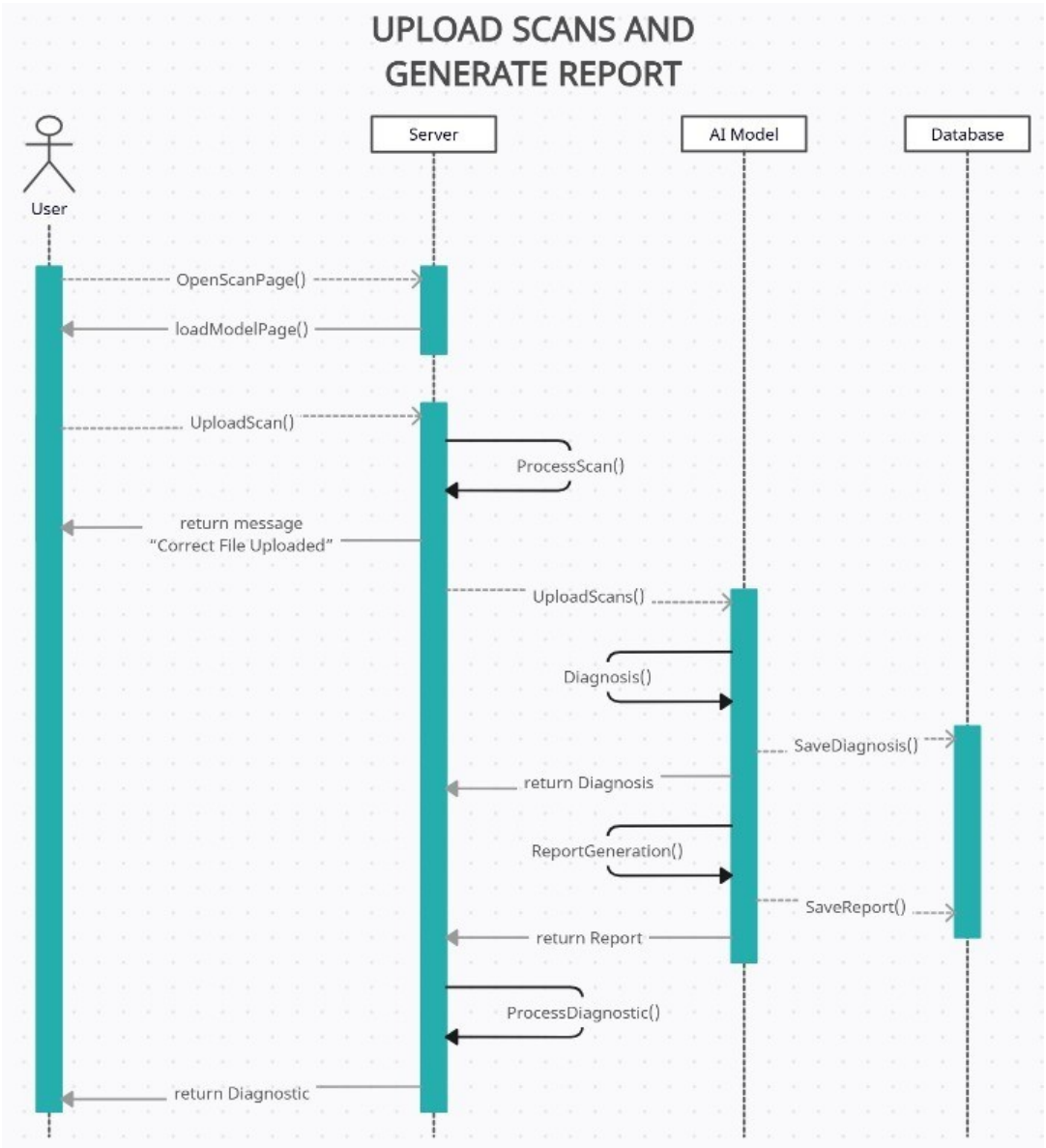

Figure 3.12: System-Level Sequence Diagram: Upload Scans and Generate Report

3.4.4 State Transition Diagram

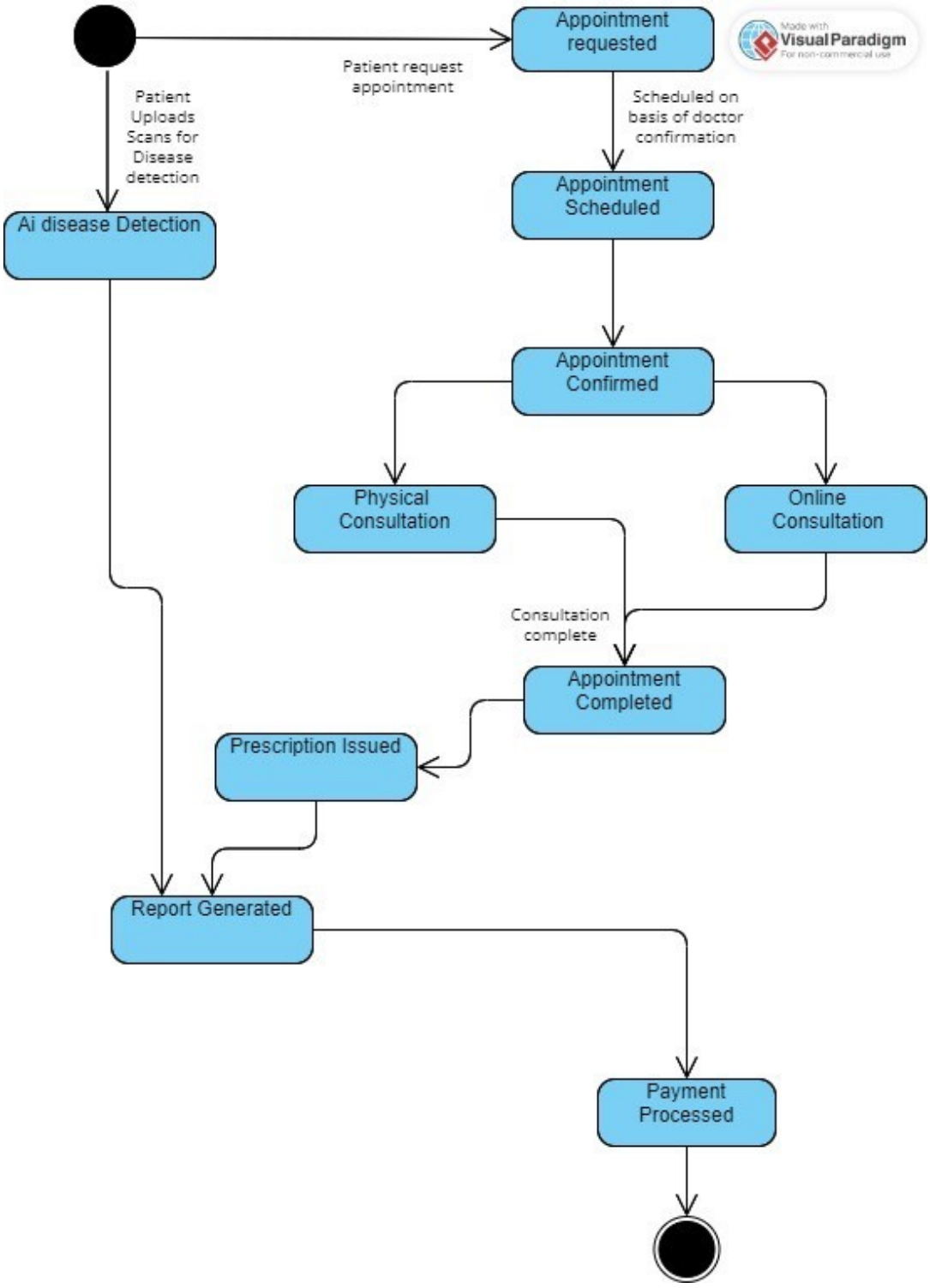

Figure 3.13: State Transition of Appointment and Diagnosis Flow

Chapter 4

Implementation and Testing

4.1 Algorithm Design

4.1.1 Secure User Registration with Cryptographic Hashing

To ensure the secure storage of user credentials, this algorithm validates inputs, hashes passwords, and issues JWTs.
Input: name, CNIC, password
Output: Securely hashed user credentials, JWT token

```
BEGIN
    IF NOT validateCnicFormat(cnic) THEN
        THROW Exception("Cnic Validation Failed: Invalid Format")
    END IF
    IF calculateEntropy(password) < MINIMUM_SECURITY_THRESHOLD THEN
        THROW Exception("Password Too Weak: Minimum entropy not met")
    END IF
    salt ← generateSalt(SALT_ROUNDS)
    hashedPassword ← bcrypt.hash(password, salt)
    userObject ← constructObject({name, cnic, hashedPassword})
    databaseResponse ← commitToDatabase(userObject)
    IF NOT databaseResponse.success THEN
        THROW Exception("Database Write Failed")
    END IF
    tokenPayload ← encode({userId: userObject.id}, JWT_SECRET)
    RETURN generateJWT(tokenPayload, EXPIRATION_TIME)
END
```

Figure 4.1: Secure User Registration with Cryptographic Hashing

4.1.2 Appointment Scheduling with Slot Validation

Dynamically handles appointment booking with real-time slot validation and conflict resolution.
Input: userId, docId, slotDate, slotTime
Output: Successful booking confirmation or conflict resolution failure

```
BEGIN
    doctor ← queryDoctorDatabase(docId)
    IF NOT doctor.available THEN
        RETURN "Error: Doctor currently unavailable"
    END IF
    slotMatrix ← doctor.slots_booked
    IF slotMatrix[slotDate][slotTime] EXISTS THEN
        RETURN "Conflict Detected: Slot already reserved"
    END IF
    mutex ← lockSlotUpdate(doctor.id, slotDate, slotTime)
    IF NOT mutex.acquired THEN
        RETURN "Concurrency Error: Slot modification locked"
    END IF
    appointmentEntry ← constructObject({userId, docId, slotDate, slotTime})
    databaseResponse ← commitToDatabase(appointmentEntry)
    Update slotMatrix[slotDate][slotTime] ← RESERVED
    Release mutex
    IF NOT databaseResponse.success THEN
        THROW Exception("Database Write Failed for Appointment")
    END IF
    RETURN "Appointment Successfully Scheduled"
END
```

Figure 4.2: Appointment Scheduling with Slot Validation

4.1.3 Dynamic Doctor Availability State Toggle

Flips a doctor's availability status with transactional safeguards for atomic updates.
Input: docId
Output: Doctor availability status flipped in the database

```
BEGIN
    doctorData ← fetchDoctorFromDatabase(docId)
    IF doctorData IS NULL THEN
        THROW Exception("Doctor Record Not Found")
    END IF
    transactionBegin ← initiateDatabaseTransaction()
    IF transactionBegin.failed THEN
        THROW Exception("Database Transaction Initiation Failed")
    END IF
    newAvailability ← !doctorData.available
    databaseUpdateResponse ← updateDatabase({id: docId, available: newAvailability})
    IF NOT databaseUpdateResponse.success THEN
        rollbackTransaction(transactionBegin)
        THROW Exception("Database Update Failed: Rolled Back")
    END IF
    commitTransaction(transactionBegin)
    RETURN "Availability Status Updated Successfully"
END
```

Figure 4.3: Dynamic Doctor Availability State Toggle

4.1.4 User Profile Update with Image Handling

Facilitates secure user profile updates with image uploads to Cloudinary.
Input: userId, newProfileData, imageFile
Output: Success or failure response

```
BEGIN
    user ← fetchUserById(userId)
    IF user IS NULL THEN
        THROW Exception("User Not Found")
    END IF
    IF imageFile EXISTS THEN
        cloudinaryResponse ← uploadToCloudinary(imageFile)
        newProfileData.image ← cloudinaryResponse.url
    END IF
    updateResponse ← updateDatabase({id: userId, data: newProfileData})
    IF NOT updateResponse.success THEN
        THROW Exception("Database Update Failed")
    END IF
    RETURN "Profile Updated Successfully"
END
```

Figure 4.4: User Profile Update with Image Handling

4.1.5 Text Messaging between Patient and Doctor

Facilitates real-time messaging between users by establishing a secure connection and ensuring message delivery through server acknowledgment.
Input: senderId, receiverId, messageContent
Output: Message delivery status (success or failure)

```
BEGIN
    socketConnection ← Establish connection to the messaging server using Socket.IO
    Authenticate sender by verifying senderId with the authentication service
    IF authenticationFails THEN
        RETURN "Authentication Failed"
    END IF

    receiverExists ← Check if receiverId is active in the system
    IF receiverExists IS FALSE THEN
        RETURN "Receiver Not Found"
    END IF

    messagePacket ← Create a structured packet containing:
        senderId
        receiverId
        messageContent
        timestamp ← Get current server time

    Emit messagePacket through the socket connection to the server
    deliveryStatus ← Wait for server acknowledgment of message delivery

    IF deliveryStatus IS "ACK_RECEIVED" THEN
        RETURN "Message Delivered Successfully"
    ELSE
        RETURN "Message Delivery Failed"
    END IF
END
```

Figure 4.5: Text Messaging between Patient and Doctor

4.1.6 Fatty Liver/ Tumor Detection using MRI and Ultrasound Scans

Trains and deploys an InceptionResNetV2-based CNN to classify liver scans and grade fat percentage.
Input: MRI/CT scan dataset
Output: Liver classification and steatosis level report

```
BEGIN
load .mat file → extract: ids, fat%, labels, images
FOR each patient DO
    preprocess all images → resize, normalize, stack RGB
    compute average image → store in X
END FOR

y_class ← binary labels
y_grade ← fat%
split X, y_class, y_grade → train/test sets

baseModel ← InceptionResNetV2(no-top)
x ← Flatten(baseModel output)
x ← CustomScaleLayer(0.17)(x)
classification ← Dense(sigmoid)(x)
grade ← Dense(linear)(x)

model ← Model(inputs, [classification, grade])
freeze baseModel layers
compile(model, optimizer=Adam, losses=[bce, mse], metrics=[acc, mae])
fit(model, X_train, y_train, epochs=10)

save(model)
evaluate on test → print AUC, MAE

FOR each test sample DO
    predict class & grade
    IF class > 0.5 THEN
        map grade → steatosis level
        print "Fatty Liver, Grade", level
    ELSE
        print "Healthy Liver"
    END IF
END FOR
show sample image
END
```

Figure 4.6: Fatty Liver Detection Model

4.2 External APIs/SDKs

API and Version	Description	Purpose of Usage	API Endpoint/Function/Class Used
Socket.IO	Real-time communication library	Enables real-time text, calls	socket.on, socket.emit
Cloudinary	Cloud-based media management	Managing profile images and medical reports	cloudinary.uploader.upload
MongoDB (Mongoose)	Data storage	Database to store platform data	mongoose.connect
JWT	Authentication library	Issues and verifies tokens for authentication	jwt.sign, jwt.verify

Table 4.1: External APIs/SDKs Overview

4.3 Testing Details

4.3.1 Unit Testing

Each unit test is designed to test a specific function or method independently from other components, helping to identify issues directly related to the functionality being tested.

Test Case ID	Test Objective	Precondition	Steps	Test Data	Expected Result	Post-condition	Pass/Fail
TC001	Verify user registration	User provides valid details	1. Enter name, email, password. 2. Click 'Register'.	Name: John Email: john@example.com Password: Pass@123	User account created and JWT token generated	User details stored in DB	Pass
TC002	Verify user login	User is registered	1. Enter valid email and password. 2. Click 'Login'.	Email: john@example.com Password: Pass@123	Login successful, JWT token issued	User session starts	Pass
TC003	Verify appointment booking	Doctor is available	1. Select doctor and date/time slot. 2. Confirm booking.	Doctor ID: 123 Date: 2024-12-10 Time: 10:00 AM	Appointment booked successfully	Slot reserved for user	Pass
TC004	Verify image upload to Cloudinary	User uploads valid image	1. Select image file. 2. Click 'Upload'.	File: profile.jpg	Image uploaded and URL returned	URL stored in database	Pass
TC005	Analyze uploaded image	User uploads random image	1. Click Upload 2. Upload random image	File: Name.png	AI could not recognize image	Incorrect image rejected	Pass
TC006	Fatty liver image analysis	Valid fatty liver image uploaded	1. Click Upload 2. Upload fatty liver image	File: fatty.jpg	AI model generates report	Report downloadable	Pass
TC007	Tumor image analysis	Valid tumor image uploaded	1. Click Upload 2. Upload tumor image	File: tumor.jpg	AI model generates report	Report downloadable	Pass

Table 4.2: Unit Testing Results

Chapter 5

Conclusions and Future Work

5.1 Conclusion

HealthBridge has been developed as a comprehensive, end-to-end healthcare platform that fully implements every specified use case according to the agreed timeline and scope. The web application (Module 1) and mobile interface (Module 4) provide secure user registration and authentication, appointment scheduling with real-time slot validation, payment processing, profile management, and two-way messaging between patients and doctors.

For Module 2, the Virtual Assistant facilitates patients through the complete workflow of finding their desired available doctor by answering several questions in basic layman terms, while the AI-powered diagnostic liver model (Module 3) processes uploaded MRI or CT scans accurately to detect fatty liver and tumors, generating a detailed report.

Each sequence diagram, data flow diagram, and algorithm has been implemented precisely in Chapters 3 and 4. HealthBridge has strictly adhered to the scheduled milestones throughout each iteration — requirement gathering (FYP-1), design implementation, and testing — ensuring all functional and non-functional requirements are met.

Furthermore, unit testing demonstrated consistent success across critical operations such as user authorization and authentication, appointment booking, image upload, and diagnostic report generation.

To sum up, HealthBridge succeeds in delivering an integrated, user-centric, AI-powered healthcare solution that effectively addresses the diagnostic and communication gaps in Pakistan's medical landscape.

5.2 Future Work

Building on the multi-layered, service-oriented architecture and extensible AI pipeline developed in this project, the following enhancements are envisioned:

- **Broader Liver Disease Coverage:** The current model detects fatty liver and tumors. In future development, the same processing layers and convolutional backbone can be retrained or fine-tuned to recognize and classify additional liver conditions — including all hepatitis variants (A, B, C, D, E), various stages of cirrhosis, and focal lesions — by integrating new, expertly annotated MRI/CT datasets into the training workflow.

- **Multi-Organ Diagnostics:** One of the most prevalent diseases in Pakistan after liver disease is lung disease. Since the AI-powered liver diagnostics of HealthBridge have separate preprocessing and reporting layers, the AI engine can be adapted to other organs. Therefore, with organ-specific datasets, segmentation, and classification, HealthBridge can evolve into a multi-organ diagnostic healthcare service.
- **Performance Optimization:** For better performance, each service layer (API gateway, authentication, appointments, virtual assistant) can be containerized using Kubernetes. Furthermore, by deploying GPU-accelerated endpoints and load-balancing the message queues, HealthBridge will ensure low-latency responses even under heavy load, which can prove to be beneficial in the near future.
- **Accessibility Improvements:** As our main target audience is the population living in urban areas of Pakistan, we can add support for regional languages like Urdu for both web and mobile interfaces. For better user experience, features like high-contrast themes, offline mode, and customizable font options can also be introduced.

In conclusion, by focusing on such enhancements, HealthBridge's multi-layered design can be scaled seamlessly by extending diagnostic coverage to additional liver diseases, integrating organ-specific models, and facilitating real-world healthcare deployments.

Bibliography

1. Biomed Central, 2024.
2. D-RAX: Domain-specific radiologic assistant leveraging multi-modal data and expert model predictions. *arXiv*, 2024.
3. The Express Tribune, 2024.
4. Fact-aware multimodal retrieval augmentation for accurate medical radiology report generation. *arXiv*, 2024.
5. IHME: Institute for Health Metrics and Evaluation, 2024.
6. Multimodal healthcare AI: Identifying and designing clinically relevant vision-language applications for radiology. *ACM*, 2024.
7. PBS Government Website, 2024.
8. Radio RAG: Factual large language models for enhanced diagnostics in radiology using dynamic retrieval augmented generation. *arXiv*, 2024.
9. RULE: Reliable multimodal RAG for factuality in medical vision language models. *arXiv*, 2024.