Metropolitan University of Tirana

Master in Artificial Intelligence, First Year

Advanced Software Engineering Final Project

Theme: **Patient Management System**

June 13th, 2025

Group Members:
Juela Kadriu
Klevis Kadriu

Abstract

The Patient Management System (PMS) presented in this project is a comprehensive web-based solution designed to optimize healthcare operations within clinical environments. Developed under the project name CarePulse, the system centralizes patient data management, appointment scheduling, medical record keeping, and doctor-patient communication, addressing many of the operational inefficiencies present in Albanian healthcare facilities.

The project follows a complete software engineering lifecycle, beginning with detailed requirement gathering and analysis, followed by system design, implementation, and validation. Functional and non-functional requirements were carefully defined to meet both user needs and regulatory compliance, including GDPR and national healthcare data protection standards. The system supports multiple user roles—patients, doctors, and administrative staff—each with distinct access privileges and responsibilities.

Key system functionalities were modeled using advanced design methodologies, including use case diagrams, ER diagrams, BPMN workflows, activity diagrams, state and sequence diagrams, communication diagrams, and CRC cards. The technical implementation leverages modern web technologies such as Next.js, TypeScript, Tailwind CSS, and Twilio to ensure scalability, maintainability, and secure patient interaction.

The final product not only streamlines healthcare workflows and improves patient experiences but also ensures data integrity, system security, and future extensibility. This project demonstrates the successful application of advanced software engineering principles to the development of a healthcare information system tailored to real-world operational needs.

Table of Contents

1. Summary 6

1.1 Project Overview 6

1.2 Purpose 6

2. Product Description.. 7

2.1 Product Context 7

2.2 User Characteristics...8

2.3 Technologies Used...9

3. Requirements.. 10

3.1 Functional Requirements 10

• Client Functional Requirements..10

• Doctor Functional Requirements..11

• Admin Functional Requirements..11

3.2 Non-Functional Requirements.. 12

- User Interface Requirements... 12

- Usability.. 12

- Efficiency 12

- Dependability 13

- Security.. 13

- Accessibility 13

3.3 Domain Requirements.. 17

4. Software Design.. 18

4.1 Use Cases.. 18

4.2 Use Case Diagram (Figure 1) 20

4.3 ER Diagram and Relational Schema..21

• ER Diagram (Figure 2) 22

• RS Diagrams (Figures 3 & 4) 23

4.4 Activity Diagrams... 25

• Register a New Patient (Figure 5) 26

• Update Patient File (Figure 6) 27

• Schedule Appointment (Figure 7) 28

• Cancel or Reschedule Appointment (Figure 8) 29

4.5 BPMN Diagrams... 30

• Patient Registration (Figure 9) 31

• Cancel or Reschedule Appointment (Figure 10) 32

• Scheduling Appointment (Figure 11) 33

• Updating Patient File (Figure 12) 34

4.6 State Diagram (Figure 13) 35

4.7 Sequence Diagram... 37

• Sequence Diagram (Figure 14) 38

• Sequence Diagram Part 1 (Figure 15) 39

• Sequence Diagram Part 2 (Figure 16) 40

4.8 Communication Diagram (Figure 17) 41

4.9 CRC Cards... 43

4.10 Class Diagram (Figure 18) 46

4.11 Techniques to Find Underlying Problems..47

• 5 Whys 47

• 3 R's (Role, Rule, Route) 48

5. Implementation... 49

5.1 Technology Evaluation... 49

5.2 Prototypes and Sketches... 51

• 1st Sketch (Figure 19) 51

• 2nd Sketch (Figure 20) 52

• 3rd Sketch (Figure 21) 53

5.3 Detailed Designs..54

• Registration Page (Figure 22) 54

• Appointment Dashboard (Figure 23) 55

• Doctor Directory Page (Figure 24) 56

5.4 Software Screenshots.. 57

• Register Page (Figure 25) 57

• Log in Page (Figure 26) 58

• Request Appointment (Figure 27) 59

• Admin View (Figure 28) 60

• Schedule Appointment Page (Figure 29) 61

• Appointment Status Page (Figure 30) 62

• Cancel Appointment (Figure 31) 63

6. Conclusion 64
7. References... .... .65

1. Summary

1.1 Project Overview

The **Patient Management System** is a web-based application designed to streamline and centralize healthcare-related operations within a medical facility. The system will enable administrative staff, doctors, and nurses to efficiently manage patient information, appointments, and medical records, while providing patients with secure access to their health data and upcoming appointments.

Key features include:

- Patient registration and profile management
- Appointment scheduling and rescheduling
- Doctor and nurse dashboards with access to patient records
- Prescription management and report uploads
- Notifications and reminders for appointments and tests
- Secure login and role-based access for different user types (admin, doctor, nurse, patient)

This system will help reduce paperwork, improve communication between departments, and ensure a smoother experience for both staff and patients.

1.2 Purpose

The Patient Management System improves the efficiency and accuracy of healthcare services by replacing manual, paper-based processes with a streamlined digital solution. It reduces administrative workload, minimizes errors in patient data handling, and allows doctors and staff to access critical information quickly. For patients, it offers a convenient way to manage appointments and view medical records—enhancing overall communication, reducing wait times, and improving the quality of care.

2. Product Description

Our proposed Patient Management System (PMS), “CarePulse,” is designed to simplify and streamline the interaction between patients, medical staff, and healthcare administrators. It facilitates the end-to-end process of managing appointments, medical records, prescriptions, and communications within a clinical or hospital setting. The lack of a unified and modern digital interface in many Albanian clinics poses challenges for both patients and professionals. CarePulse aims to address this gap by offering a centralized platform that automates workflows, enhances record accessibility, and ensures a smoother experience for all users.
The platform is built to be intuitive and accessible even for users with limited technical proficiency. Whether a doctor needs to track treatment history, a receptionist needs to verify patient information, or a patient wants to reschedule a consultation, the system ensures these operations are performed securely and efficiently.

2.1 Product Context

The PMS platform is independent and self-contained. While other healthcare systems may focus on narrow internal tasks (such as scheduling or billing), CarePulse offers a comprehensive suite of functionalities that integrate patient data, clinical records, and administrative tasks under one interface.
All information is securely created, stored, and accessed within the platform, in compliance with GDPR and Albanian health data regulations. The system is designed to function without relying on third-party applications, ensuring higher security and greater control for healthcare providers.
A technical overview and system diagrams are available in the internal documentation, showcasing how modules such as patient registration, prescription history, and appointment scheduling interact seamlessly.

2.2 User Characteristics

The system is designed for three primary user groups:

• Patients:

- Register and log in
- Book, cancel, or reschedule appointments
- Access medical records and prescription history
- Receive automated notifications for upcoming visits
- View visit summaries and communicate with doctors
- Submit feedback regarding services

• Doctors:

- Log in to a personal dashboard
- View upcoming appointments and patient history
- Record diagnoses and issue prescriptions
- Update treatment progress notes
- Access analytics on visit frequency and patient load

• Receptionists (Administrators):

- Review and approve new patient registrations
- Verify appointments and manage daily visit queues
- Modify or reschedule bookings upon request
- Send alerts or reminders to patients
- Generate summary reports for each day/week

2.3 Technologies used

To build an efficient and modern Patient Management System, we conducted a thorough evaluation of available technologies. After testing several tools and frameworks, we selected **Next.js**, **TypeScript**, **Twilio**, and **Tailwind CSS**. These technologies collectively support robust backend logic, modern frontend design, secure communication, and scalable development. Next.js enables fast performance through server-side rendering, while TypeScript ensures type safety and maintainable code. Twilio was chosen for its reliable SMS and telemedicine capabilities, crucial for patient notifications and virtual consultations. Tailwind CSS streamlines UI development with its utility-first approach. Each technology was explored through sample applications to confirm its fit for our project's needs, balancing performance, usability, and long-term maintainability.

3. Requirements

3.1 Functional Requirements

Client

Req#	**Requirement**	**Comments**	**Priority**	**Date Rvwd**	**SME Reviewed / Approved**
FR_C_01	The system should allow patients to register an account.	Patients must provide name, age, contact, and medical ID.	1	15/04/25	Project Supervisor
FR_C_02	The system should allow patients to book appointments online.	Patient selects preferred doctor and time from availability.	1	18/04/25	Project Supervisor
FR_C_03	The system should allow patients to view their appointment history.	Past appointments and doctor feedback are shown.	2	23/04/25	Project Supervisor
FR_C_04	The system should enable patients to cancel appointments.	Patients can cancel 24 hours before the scheduled time.	2	28/04/25	Project Supervisor
FR_C_05	The system should allow patients to message their assigned doctors.	Text-based messaging interface with alerts.	2	03/05/25	Project Supervisor

Doctor

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
FR_D_01	The system should allow doctors to view their daily schedule.	Dashboard with appointments and patient details.	1	06/05/25	Project Supervisor
FR_D_02	The system should allow doctors to update appointment status.	Mark as completed, cancelled, or no-show.	1	09/05/25	Project Supervisor
FR_D_03	The system should allow doctors to write and view patient records.	Includes notes, prescriptions, and medical history.	1	13/05/25	Project Supervisor
FR_D_04	The system should enable doctors to chat with assigned patients.	Chat log must be stored for reference.	2	17/05/25	Project Supervisor

Admin

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
FR_A_01	The system should allow the admin to manage user accounts.	Create, delete, and edit patient and doctor accounts.	1	21/05/25	Project Supervisor
FR_A_02	The system should provide admin with system usage statistics.	Daily logins, appointments booked, cancellations.	2	25/05/25	Project Supervisor
FR_A_03	The system should allow admin to send system-wide notifications.	Example: maintenance alerts or emergency announcements.	2	28/05/25	Project Supervisor

3.2 Non-Functional Requirements

3.2.1 Product Requirements

3.2.1.1 User Interface Requirements

General UI Requirements:

- The user interface must be clean, intuitive, and accessible to both medical staff and patients with minimal training.
- The application must support various screen sizes, including desktops, tablets, and mobile devices.
- The main navigation should be aligned vertically on the left side of the screen, while the central area should display relevant user content (appointments, patient files, messaging, etc.).
- Separate landing pages will be provided for doctors, patients, and administrative staff, each showing tailored information and access.
- The login and registration forms will vary depending on the user type (patient or medical staff).

Users UI Requirements:

Patient:

- Dashboard includes upcoming appointments, past visit summaries, and communication options.
- Ability to search available appointment slots by doctor or specialty.
- Ability to cancel or reschedule existing appointments.
- Messages and notifications (e.g., reminders or prescription updates) must be clearly visible on login.
- Secure access to personal medical history and prescription details.

Doctor:

- Dashboard provides daily schedule, upcoming patient visits, and urgent alerts.
- Easy access to patient records, visit notes, and the ability to upload prescriptions.
- Ability to chat with assigned patients through a secure chat interface.
- Notifications for appointment updates or system messages.

Admin:

- Dashboard provides access to user management, platform statistics, and system notifications.
- Tools to create system-wide messages (e.g., downtime notices).
- Ability to review usage patterns and export audit logs.

3.2.1.2 Usability

The platform must be designed for ease of use, especially for patients who may have limited digital literacy. Tooltips, error warnings, and step-by-step flows will be integrated to guide users. A help center will be available with FAQs and video walkthroughs. Minimal clicks will be required to complete core actions (e.g., booking or cancelling an appointment).

3.2.1.3 Efficiency

Performance Requirements:

- As a web-based system, platform performance is dependent on internet speed.
- Designed to support at least 25 simultaneous users initially (scalable to 200+).
- Server should ensure an average response time under 0.2 seconds for database queries and UI loading.

Space Requirements:

- Each patient record, including appointment history and medical notes, is expected to require approximately 1 MB per year.
- System must be scalable to support 10,000+ users with 10+ years of history.
- Database storage requirements estimated at 500 GB for 15 years of full operation.

3.2.1.4 Dependability

Availability:

- The platform will be accessible 24/7 for both patients and staff.
- Downtime for updates or backups will be limited to midnight hours during weekends, and announced ahead of time.
- 99.8% uptime target.

Reliability:

- System should function with minimal interruption during high usage hours (Monday mornings, evenings).
- Errors or crashes should trigger immediate alerts to the maintenance team.
- Automatic session recovery and backup systems must be in place.

Monitoring:

- Every user action will be logged for security and debugging purposes.
- Logs will be reviewed weekly, and anomalies flagged for admin review.

Maintenance:

- Developers will maintain the system remotely, responding to issues raised via the platform's internal messaging feature.
- Maintenance cycles and patches will be deployed monthly or as needed.

Integrity:

- The system will perform regular checks on database consistency and user session handling.
- Alerts will be issued for unauthorized access attempts or corrupted records.

3.2.1.5 Security

- All data must be stored encrypted using AES-256 encryption.

- All communications will be secured via HTTPS with valid SSL/TLS certificates.
- Two-factor authentication (2FA) will be mandatory for doctors and admin users.
- Role-based access control (RBAC) will restrict access to sensitive features.
- Automatic logout after 10 minutes of inactivity.
- Full audit trail will be logged for record updates and prescription entries.
- Passwords will be stored hashed and salted using industry-standard algorithms (e.g., bcrypt).

3.2.1.6 Accessibility

- The platform will follow WCAG 2.1 AA accessibility standards.
- All UI elements will be navigable via keyboard.
- Screen reader support will be integrated for visually impaired users.
- Color contrast ratios will be optimized for clarity and visual disability accommodation.
- All form fields will include associated labels and ARIA roles where needed.

3.2.2 Organizational Requirements

3.2.2.1 Environmental Requirements

The Patient Management System will be used to streamline the operations of healthcare clinics by enabling them to manage patient appointments, medical records, and communication in a unified platform. It will be optimized for everyday clinical workflows and accessible in both private and public healthcare environments.

Power Supply:
Clinics using the system must have a stable power supply to ensure consistent system uptime.

Backup power systems (e.g., UPS) are recommended to avoid interruptions during patient interactions or record updates.

Internet Connection:
The platform requires a stable internet connection for real-time operations. Clinics may use Wi-Fi or wired Ethernet. Patients can access the system through any standard home broadband or mobile data connection.

3.2.2.2 Operational Requirements

The platform is structured to serve three main user groups with distinct operational roles:

- **Doctors**:
  Will use the platform to review schedules, access and update patient medical records, and communicate with patients.
- **Administrative Staff (Admins)**:
  Will manage user accounts (patient/doctor), system statistics, and ensure proper functioning of clinic-side operations.
- **Patients**:
  Will be able to book, cancel, or review appointments, access their medical history, and communicate with their doctor.

Each role has dedicated permissions and access levels within the system to ensure both usability and data security.

3.2.3 External Requirements

3.2.3.1 Regulatory Requirements

This software will comply with all legal standards related to personal data handling and electronic health records in Albania and the EU. It will implement the following frameworks:

- **Law No. 9887, dated 10.03.2008**, on the Protection of Personal Data, as amended;
- GDPR (General Data Protection Regulation), enforceable since May 25, 2018;
- Applicable eHealth and telemedicine provisions.

The system will collect and store:

- Patient full name, contact info, medical ID, appointment history, and optional health insurance details.

- Doctor user accounts will store professional credentials, consultation history, and prescription data.

All user data will be encrypted and stored in compliance with EU privacy regulations. Session tracking will be implemented using anonymized tokens and no personally identifiable information will be used without user consent.

3.2.3.2 Ethical Requirements

All users must consent to the collection and use of personal data upon registration. Patients will be informed how their information is used for appointment scheduling, communication, and record keeping.

Collected information includes:

- For patients: Full name, contact number, email, date of birth, appointment history.
- For doctors: Full name, email, specialization, and schedule availability.

Data will not be shared outside the system except in accordance with legal obligations or with explicit patient consent.

3.3 Domain Requirements

The platform is developed specifically for outpatient and clinic-based care environments. It focuses on digitizing routine patient interactions such as:

- Booking/rescheduling appointments;
- Maintaining visit history and prescriptions;
- Facilitating direct communication between doctor and patient;
- Allowing administrative oversight without compromising data privacy.

The system is modular and can be extended for hospital use, emergency triage management, or integration with pharmacy services in future versions.

4. Software Design

4.1 Use cases

UC_1.1 – Register a New Patient

Summary	**Patient registers a new account in the system.**
Actors	Patient
Description	The patient creates an account by filling out a registration form with required personal details. The system checks if the email has been used before and either completes the registration or shows an error message.
Pre-condition	Patient has not registered before.
Post-condition	New account created and patient is notified of successful registration.

UC_1.2 – Update Patient File

Summary	**Receptionist updates a patient's file with diagnosis and prescription.**
Actors	Receptionist, Doctor
Description	Receptionist accesses the patient's file using their ID. The doctor provides the diagnosis and indicates whether a prescription is needed. The receptionist enters the diagnosis, prescription (if any), and uploads a report.
Pre-condition	Patient must have an existing file in the system.
Post-condition	Patient file updated and saved with new medical information.

UC_1.3 – Schedule Appointment

Summary	**Patient books an appointment with a doctor.**
Actors	Patient, Admin
Description	Patient logs in and selects preferred doctor and time slot. The admin checks availability and authorizes the appointment. If the slot is unavailable, the system asks the patient to choose another.
Pre-condition	Patient must be logged in or must create an account.
Post-condition	Appointment scheduled and confirmation sent to the patient.

UC_1.4 – Cancel or Reschedule Appointment

Summary	**Patient cancels or reschedules an existing appointment.**
Actors	Patient, Receptionist
Description	Patient accesses their scheduled appointments. If they choose to cancel, the receptionist confirms and removes it. If rescheduling, a new time slot is selected, and the receptionist checks and authorizes it.
Pre-condition	Patient must be logged in and have an existing appointment.
Post-condition	Appointment canceled or rescheduled with confirmation sent.

4.2 Use Case Diagram

The Use Case Diagram identifies the main actors (Patient, Doctor, Receptionist) and the functionalities they interact with. It highlights the core use cases such as Login, Book Appointment, Cancel Appointment, and Request Prescription.

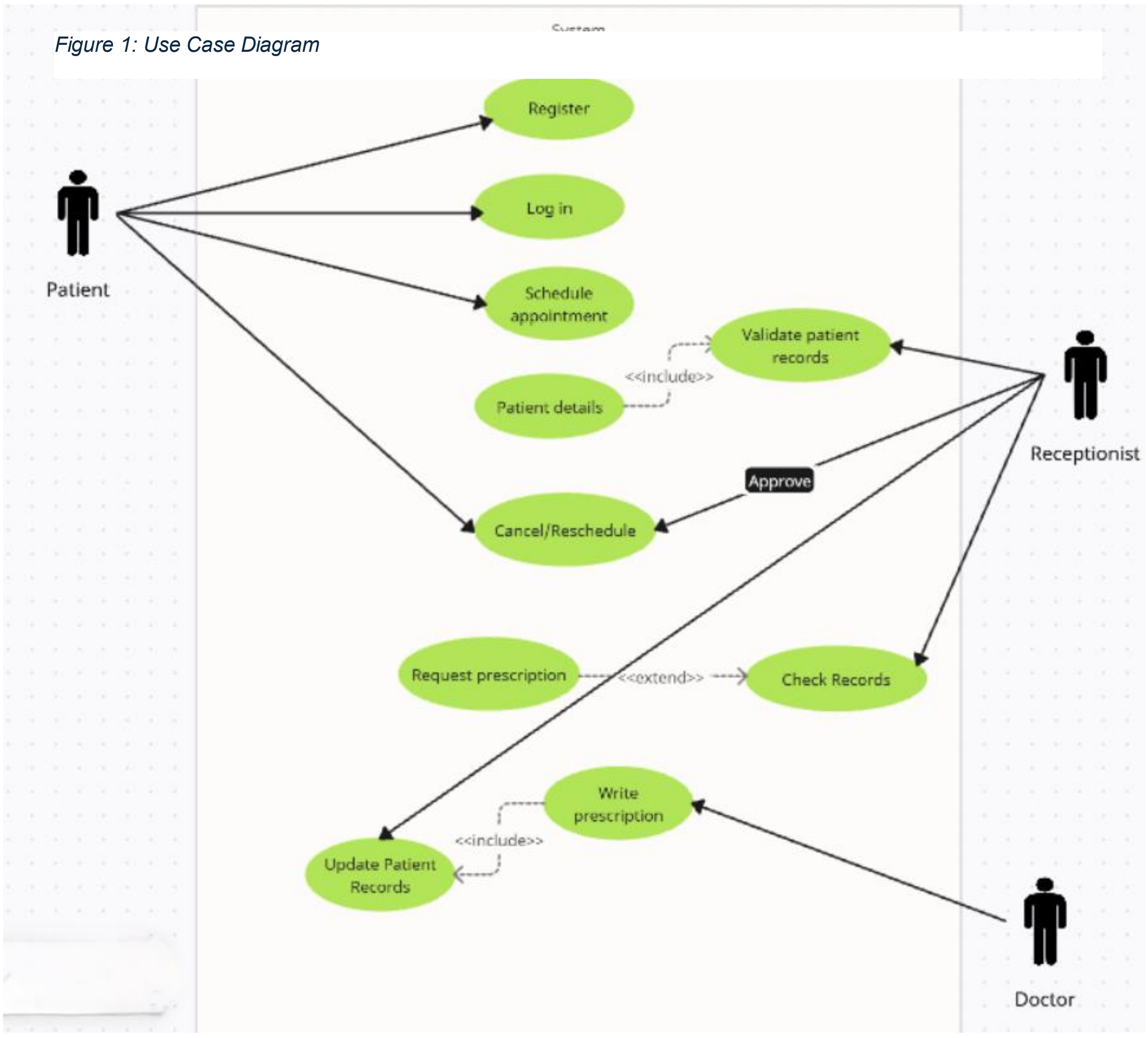
Actors
Receptionist
Doctor
Patient

Use Cases
Receptionist
Register New Patient
Schedule Appointment (include: Check Doctor Availability)
Update Patient Info
Check-in Patient (extend: Assist in Check-in)

Doctor
View Patient History
Create Prescription (include: View Patient History)
Record Diagnosis
View Appointments

Patient
View Appointments
View Medical Records (include: View Diagnosis, Prescriptions)
Send Message

Figure 1: Use Case Diagram



4.3 ER Diagram and Relational Schema (RS)

4.3.1 ERD

The ERD defines the database schema of the system. It includes entities such as Patient, Appointment, Doctor, Receptionist, Prescription, and Login Session along with their attributes and relationships.

Entities and Attributes:

1. patients
   - Attributes: PatientID, Name, DateOfBirth, Gender, Email, Phone, Address, Password
   - Primary Key: PatientID
   - Represents the core user of the system.

2. appointments
   - Attributes: AppointmentID, PatientID, ReceptionistID, Date, Time, Status, DoctorID
   - Primary Key: AppointmentID
   - Links patients with doctors and receptionists.

3. prescriptions
   - Attributes: PrescriptionID, AppointmentID, Medication, Dosage, Notes, DoctorID
   - Primary Key: PrescriptionID
   - Linked to both appointments and doctors.

4. doctors
   - Attributes: DoctorID, Name, Specialization, Email, Phone
   - Primary Key: DoctorID

5. receptionists
   - Attributes: ReceptionistID, Name, Email, Phone
   - Primary Key: ReceptionistID

6. login_sessions
   - Attributes: SessionID, PatientID, LoginTime, LogoutTime, Status
   - Primary Key: SessionID
   - Tracks login activity of patients.

Relationships:

One-to-Many:

- One patient can have many appointments.
- One doctor can handle many appointments and prescriptions.
- One appointment can lead to one prescription.
- One receptionist can manage many appointments.

Entity Integrity is maintained with proper foreign keys (e.g., DoctorID, PatientID, ReceptionistID in appointments).

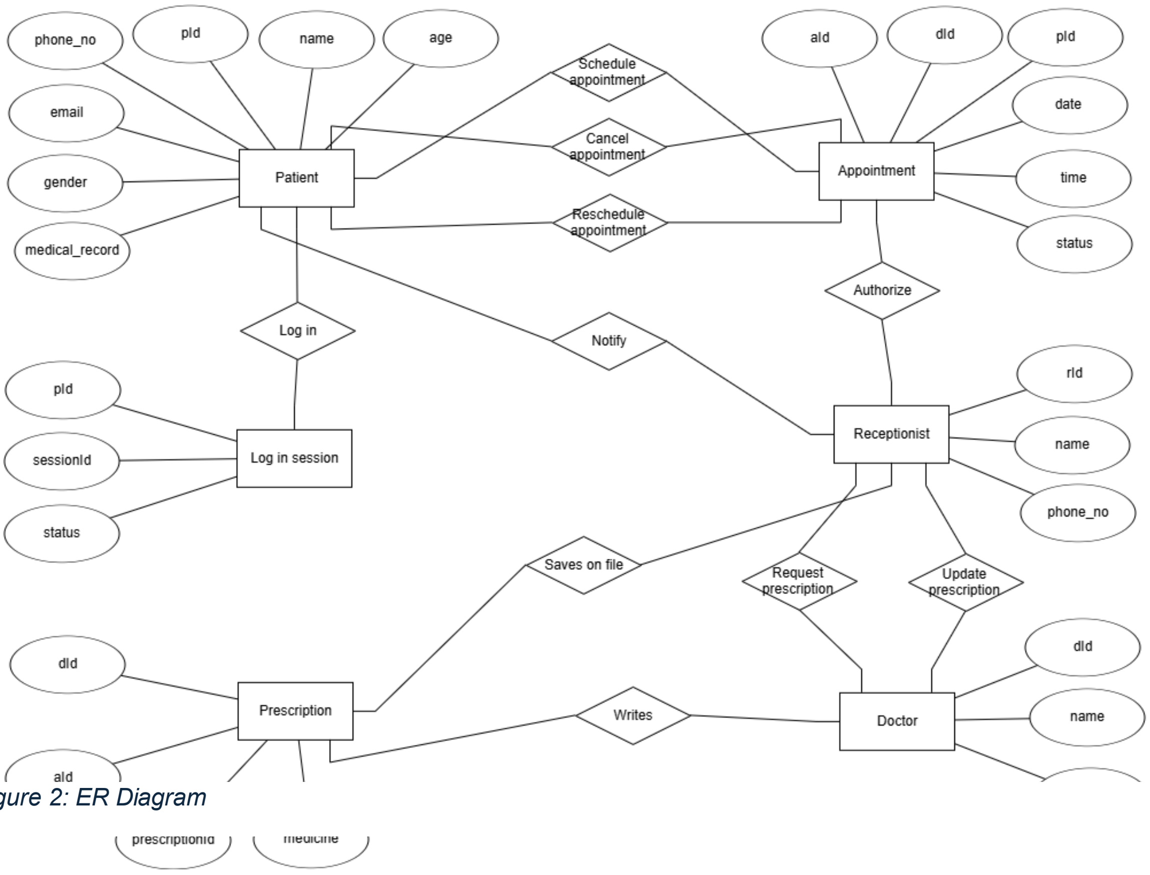


Figure 2: ER Diagram

4.3.2 RS

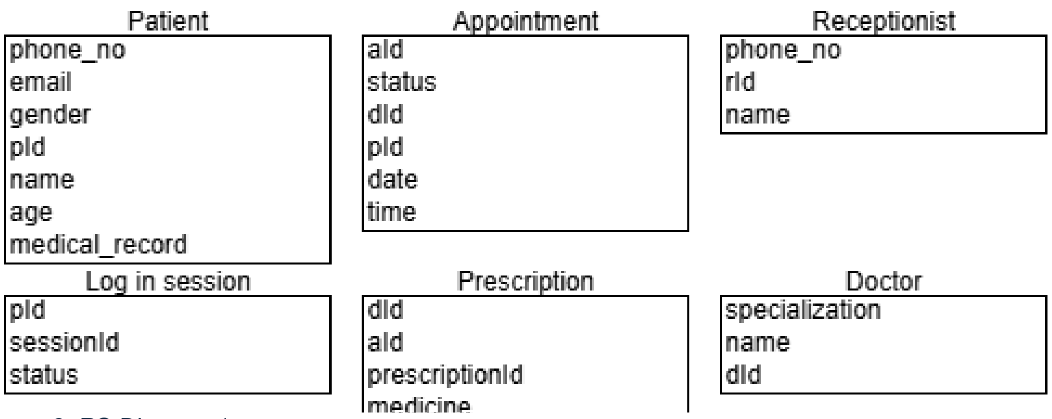


Figure 3: RS Diagram 1

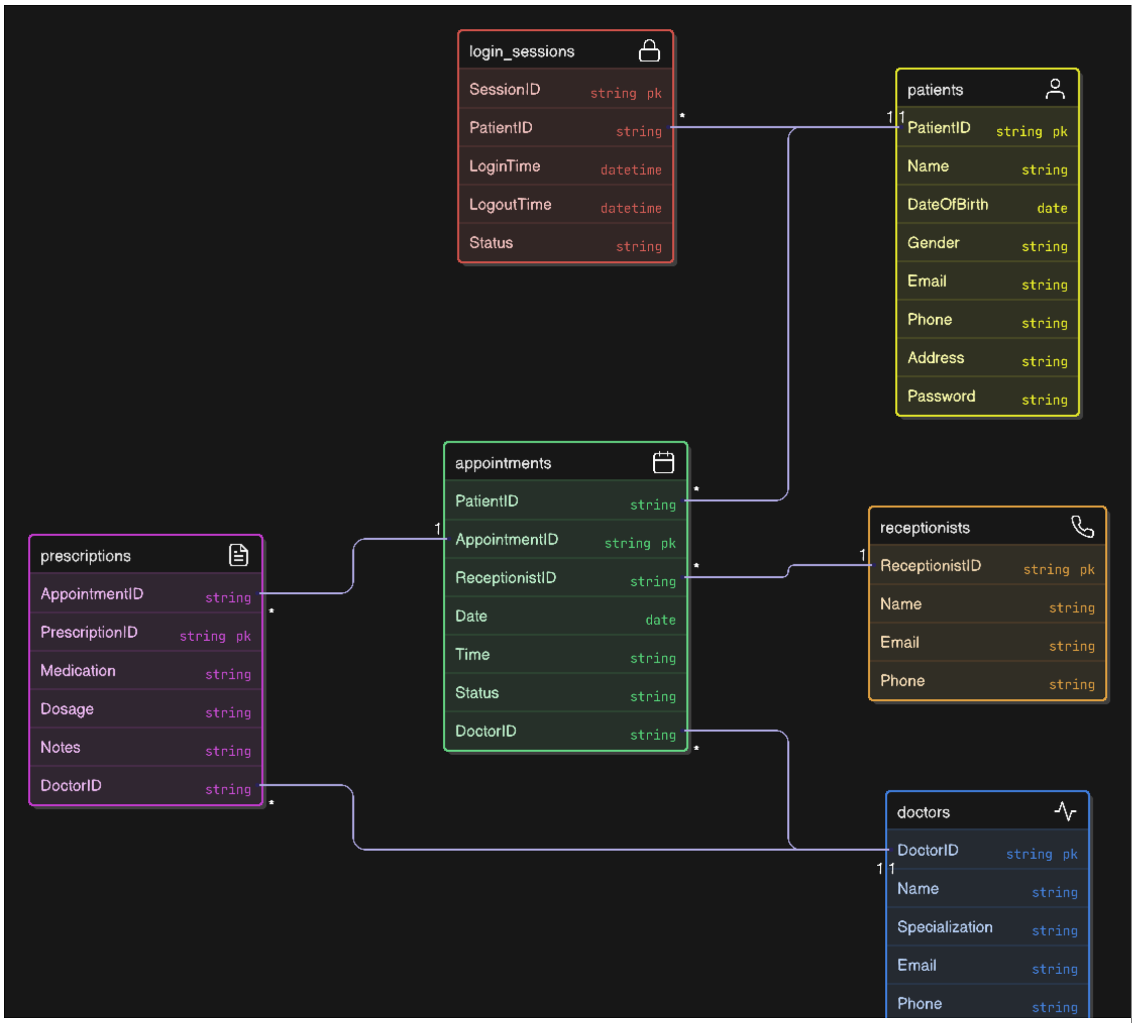


Figure 4: RS Diagram2

4.4 Activity Diagrams

The Activity Diagram represents the dynamic workflow of the system, particularly for login and appointment management. It includes action nodes, decision points, and parallel flows showing how users interact with the system.

4.4.1 Register a New Patient

The patient initiates registration by creating an account and filling out a form. Upon submission, the system validates the email address. If the email is already in use, an error message is displayed and the user is prompted to revise. If the email is unique, the registration is marked successful, and the patient receives a notification confirming successful registration.

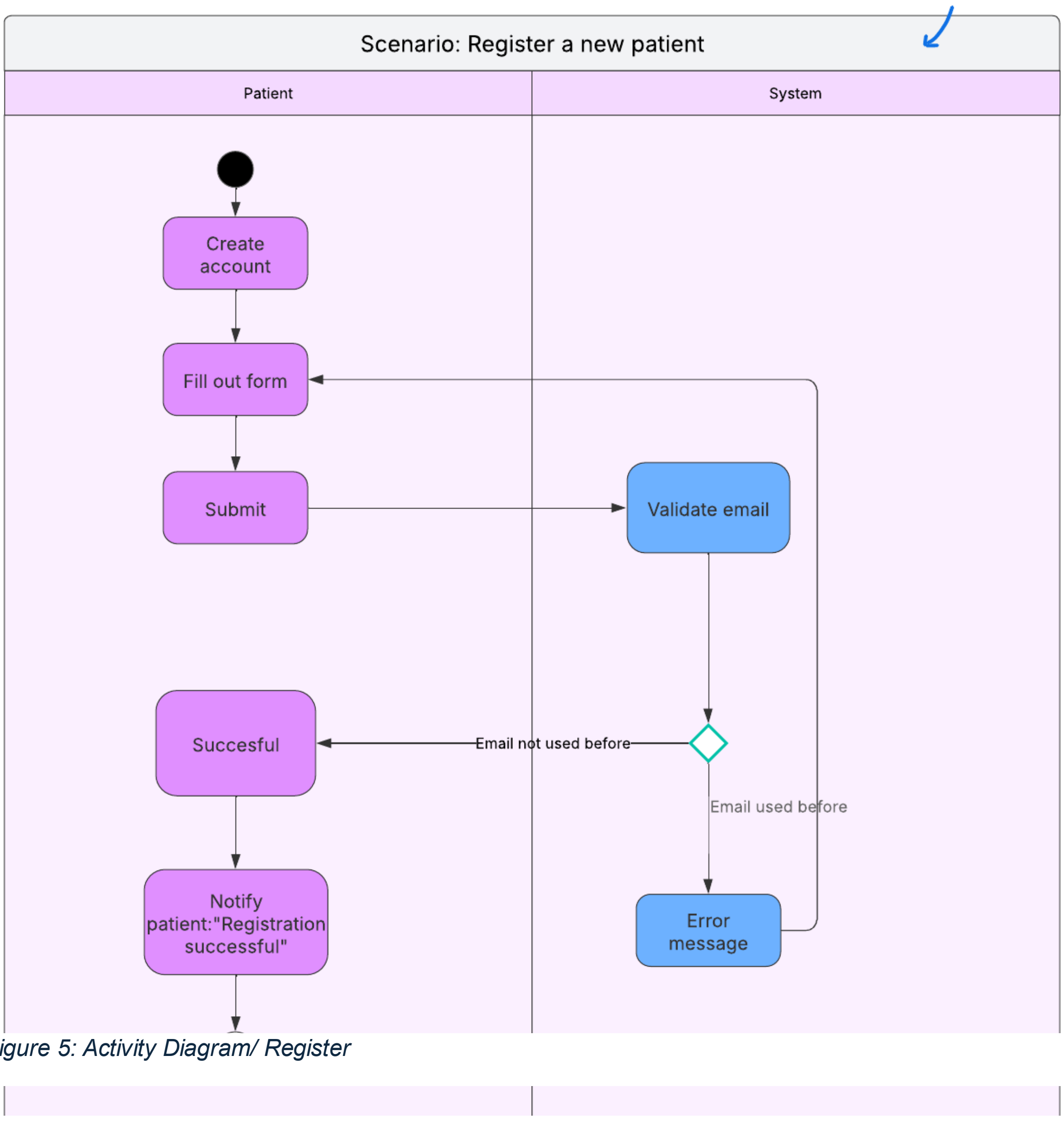


Figure 5: Activity Diagram/ Register

4.4.2 Update Patient File

In this scenario, the receptionist begins by entering the patient's ID and opening their file. They then request the patient's diagnosis from the doctor. After receiving the diagnosis, the receptionist adds it to the file. If a prescription is needed, the doctor provides it and the receptionist records it. A report is then attached to the file, and the updates are saved to complete the process.

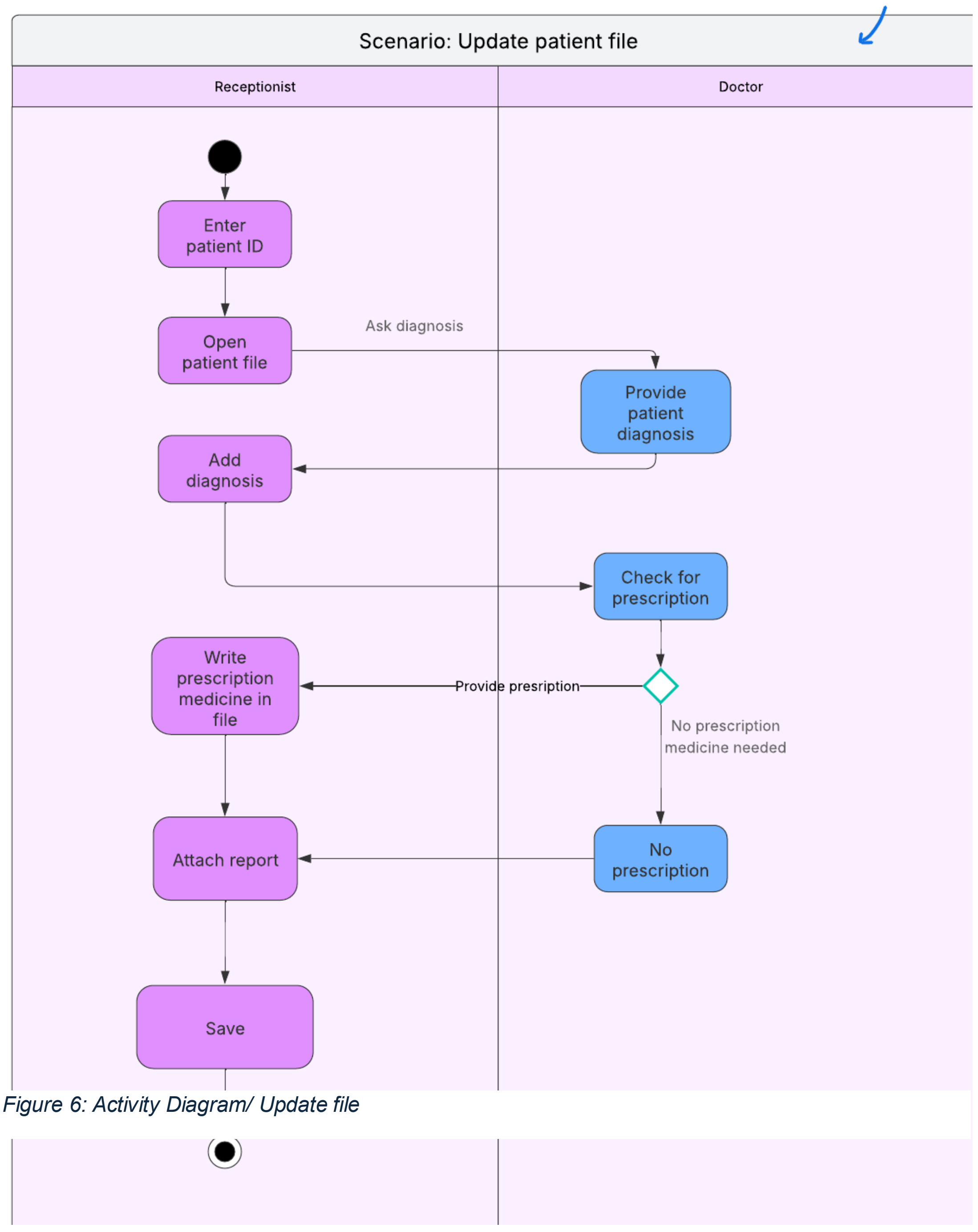


Figure 6: Activity Diagram/ Update file

4.4.3 Schedule Appointment

The patient logs into the system and enters credentials. If login fails, the patient creates a new account. After logging in, the patient proceeds to schedule an appointment by choosing a doctor and a time slot. The admin then authorizes the request and checks for slot availability. If the chosen slot is unavailable, the patient is prompted to choose another. Once a valid slot is found, the appointment is authorized and a success notification is sent.

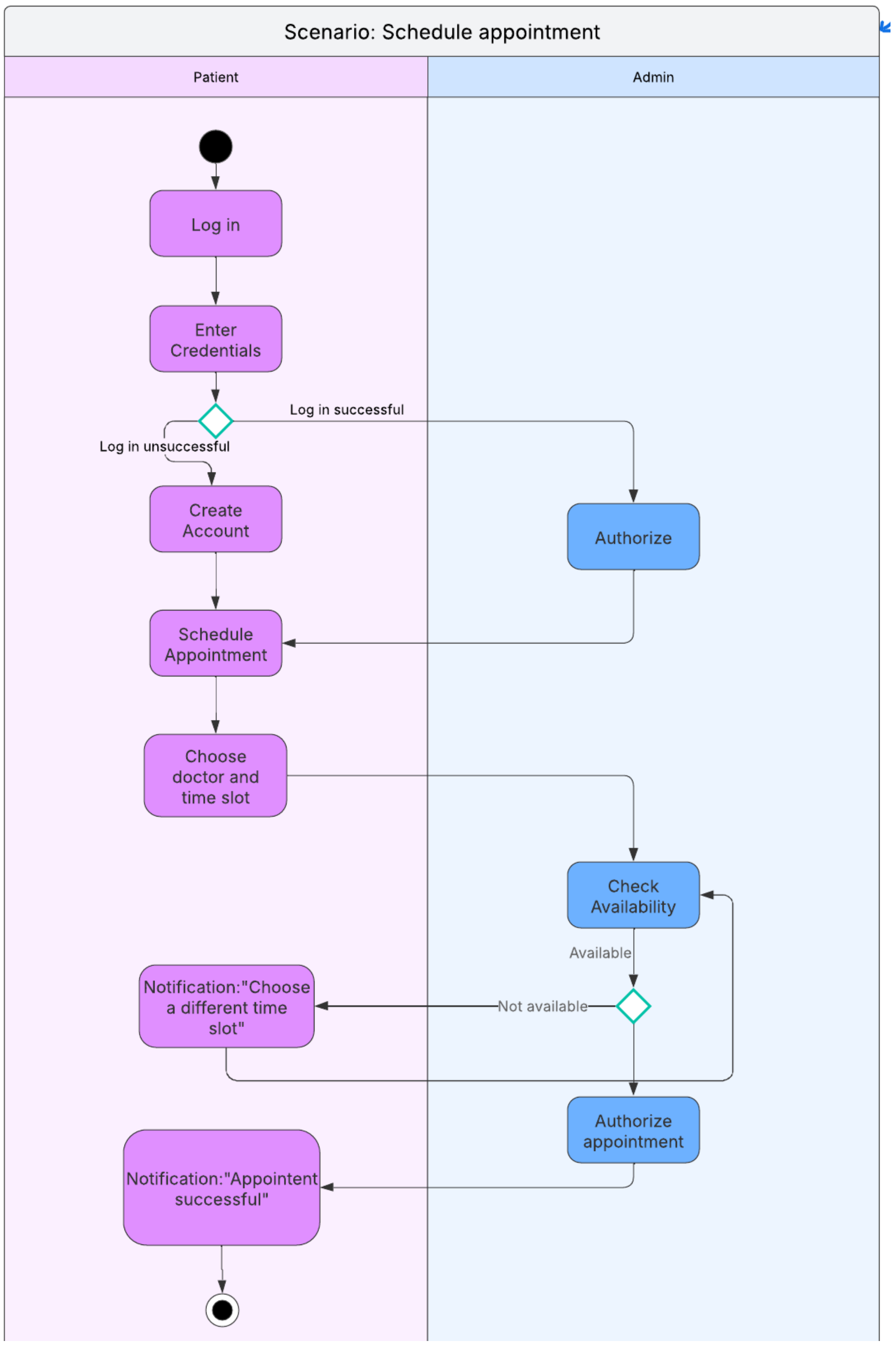


Figure 7:Activity Diagram/ Schedule Appointment

4.4.4 Cancel or Reschedule

Appointment In this scenario, the patient logs into the system and navigates to the "My Appointments" section. After selecting an appointment, the patient can either cancel or reschedule it. If cancelled, the receptionist confirms the cancellation and a confirmation message is sent to the patient. If rescheduling, the patient selects a new time slot, and the receptionist checks availability. If the slot is unavailable, the patient is prompted to choose another. Once an available slot is selected, the receptionist authorizes it, and the patient receives a success notification.

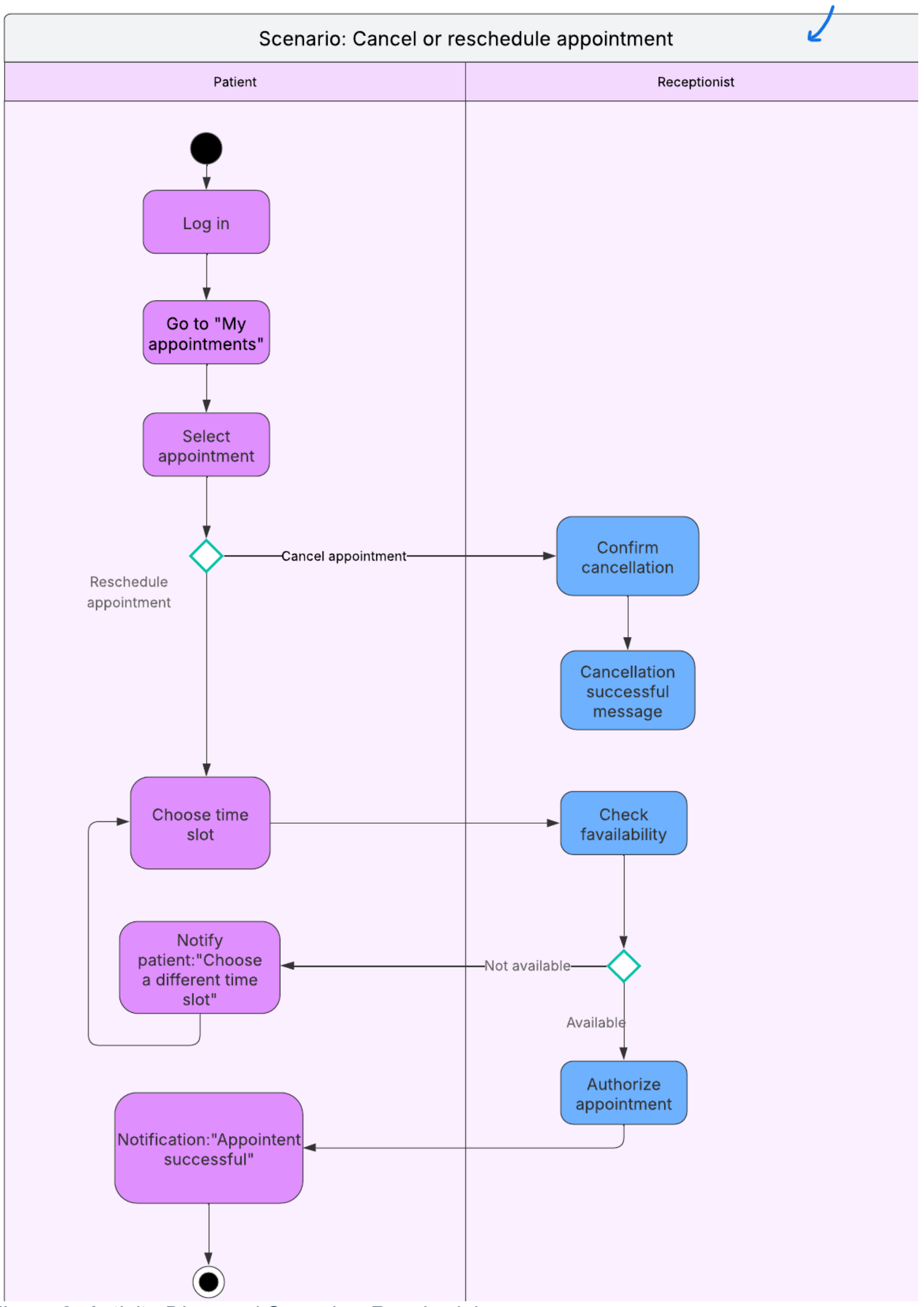


Figure 8: Activity Diagram/ Cancel or Reschedule

4.5 BPMN Diagrams

The BPMN (Business Process Model and Notation) diagram outlines the flow of business processes in the system, from patient login to appointment scheduling and prescription issuance. It emphasizes tasks, gateways, and message flows.

Scenario 1: Patient Registration Process
Actors Involved:

- Patient
- Receptionist

Step-by-Step Process:

Patient Actions:

1. Create Account:
The patient initiates the registration process.
2. Fill Form:
The patient inputs required information such as name, email, password, etc.
3. Submit:
The form is submitted to the system for processing.

Receptionist/System Actions:

4. Validate Email:
The system (represented in the receptionist lane) checks whether the email entered is already registered.
5. Decision Point – Email Check:
   o If the email was used before:
      - An error message is sent to the patient.
      - The process returns to the form-filling step, prompting the user to try again with a different email.

   o If the email is new:
      - The registration is marked as successful.

Final Actions:

6. Notify Patient:
A confirmation message ("Successful") is sent to the patient.
7. End Event:
The registration process is completed.

Purpose:
To ensure that each patient has a unique account by validating email addresses during registration. This prevents duplicate accounts and enhances system integrity.

Key Highlights:
- o Strong email validation logic to avoid redundancy.
- o User feedback loop in case of errors.
- o Clean and automated handoff between patient and receptionist roles.

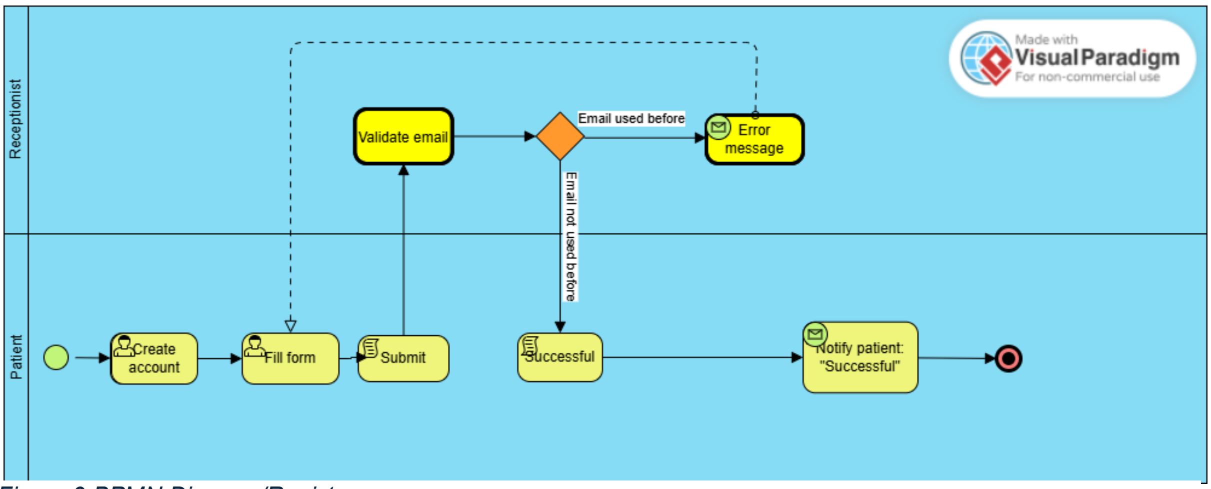


Figure 9 BPMN Diagram/Register

Scenario 2: Cancel or Reschedule Appointment

Actors:

- Patient
- Receptionist

Flow Summary:

1. Patient logs in to the system.
2. They navigate to appointments and select one.
3. A decision point follows: Cancel or Reschedule.

o If Cancel:

- The system asks the Receptionist to confirm cancellation.
- Upon confirmation, the patient is notified.

o If Reschedule:

- The patient is asked to choose a new time slot.
- The system checks availability:
    - If available: Receptionist authorizes the appointment, and the patient is notified of success.
    - If not: the patient is notified to choose another slot and repeats the process.

Purpose: Allows flexibility for patients to manage their appointments while keeping receptionists in the loop for confirmation and approval.

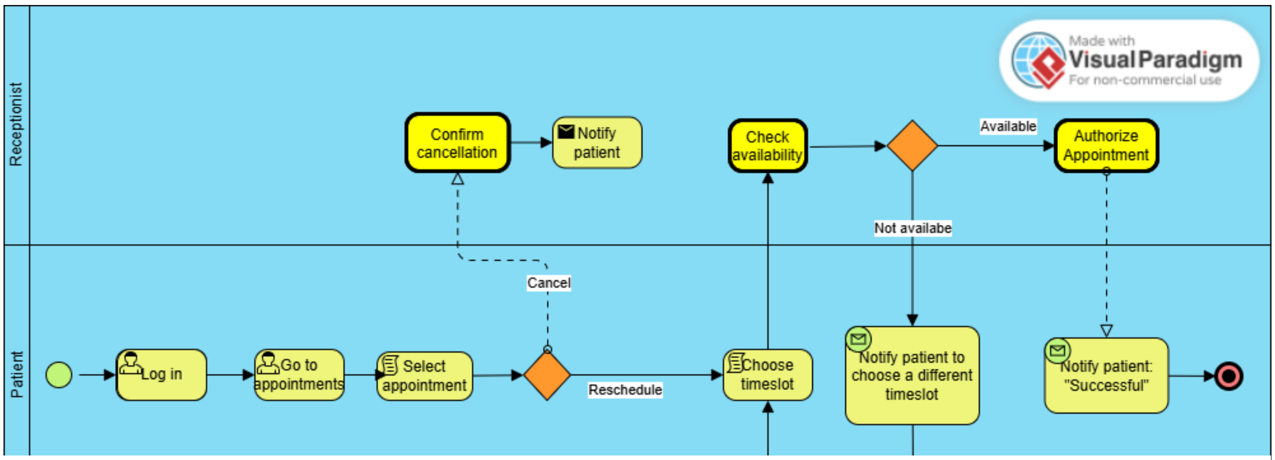


Figure 10: BPMN Diagram/ Cancel or Reschedule

Scenario 3: Scheduling an Appointment

Actors:

- Patient
- Receptionist

Flow Summary:

1. Patient logs in and enters credentials.
2. A decision point determines if the patient has an account:

   o If not, the patient creates a new account.
3. The patient then schedules an appointment.
4. The patient chooses a timeslot.
5. The receptionist checks availability:

   o If available: Appointment is authorized, and the patient is notified.

   o If not: The patient is asked to choose another timeslot, looping back.

Purpose:

Ensures secure access and provides a streamlined process for booking new appointments,

including account creation and system-assisted time slot checking.

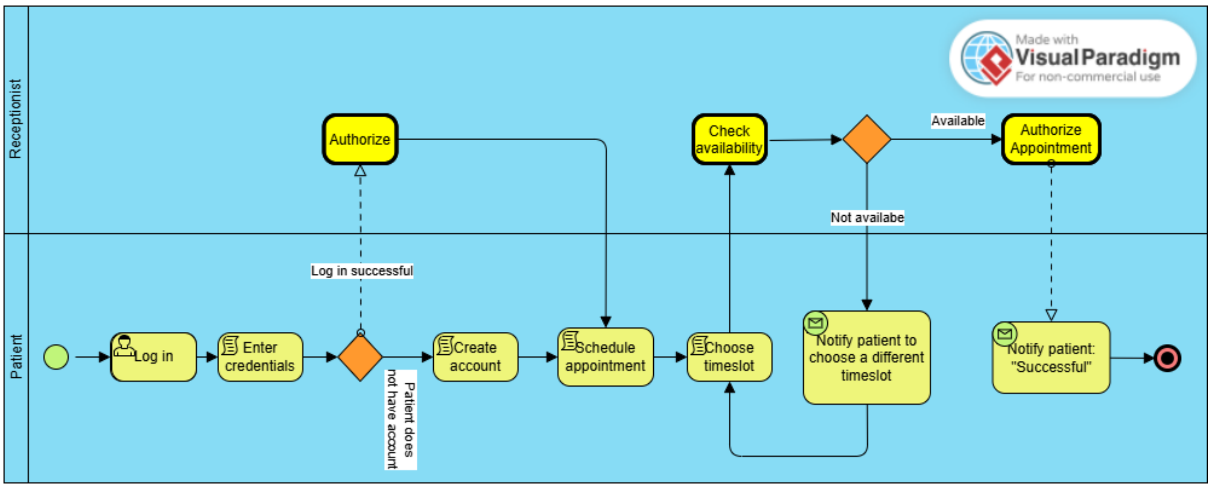


Figure 11: BPMN Diagram/ Schedule appointment

Scenario 4: Updating the Patient File Post-Appointment

Actors:

- Receptionist
- Doctor

Flow Summary:

1. Receptionist enters the patient ID and opens the patient file.

2. The doctor provides a diagnosis, which the receptionist records.

3. The doctor then checks if prescription is needed:

o If no medication is needed: a "No Prescription" message is recorded.

o If medication is needed:

- The doctor writes the prescription in the file.
- The receptionist attaches a report and saves the updated file.

Purpose:

Maintains accurate and up-to-date medical records with collaboration between doctor and receptionist, ensuring prescribed medications and diagnoses are documented correctly.

4.6 State Diagram

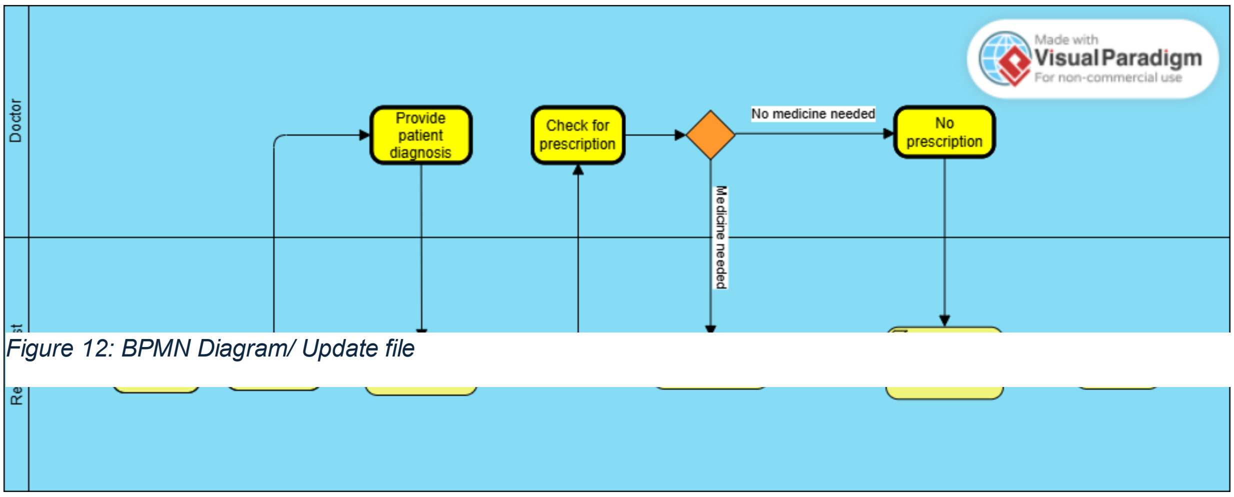
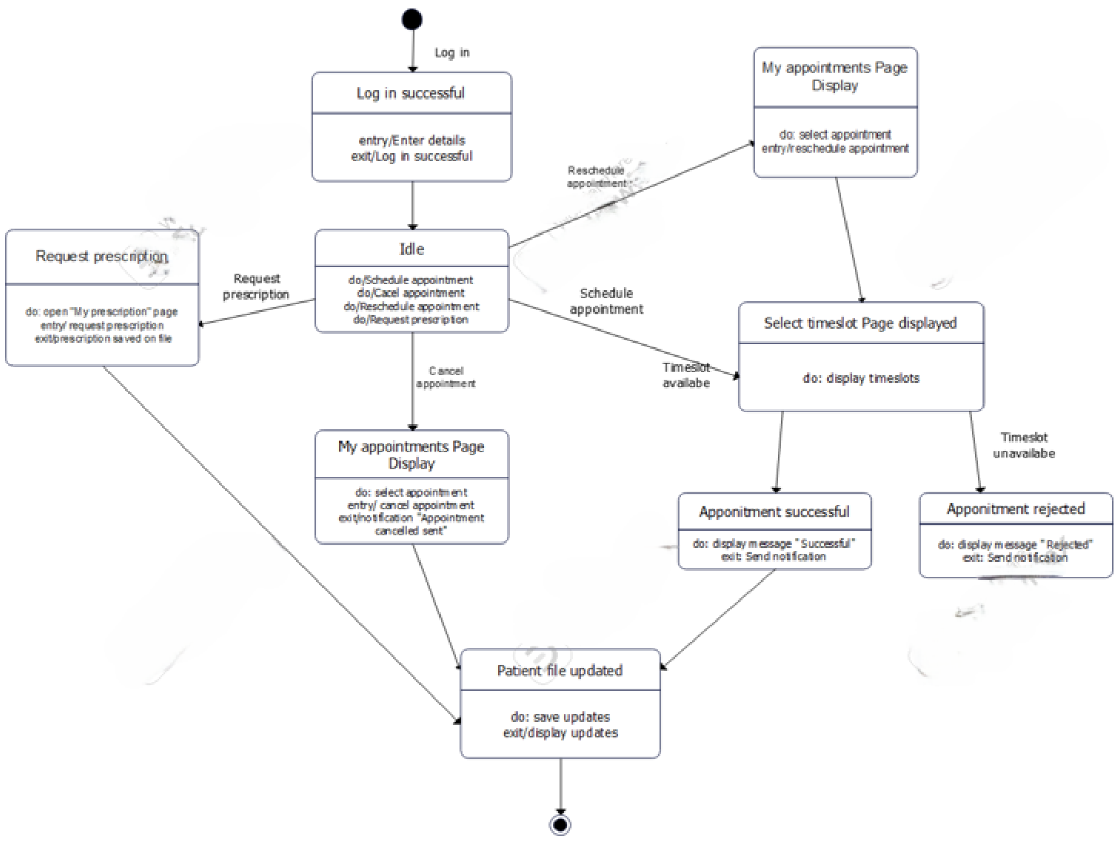


Figure 12: BPMN Diagram/ Update file

The state diagram illustrates the various states and transitions a user (typically a patient or
receptionist) goes through while interacting with a healthcare appointment and file

management system. It starts from the initial login and proceeds through states such as Idle,
where the user can choose to schedule, reschedule, or cancel an appointment, as well as
request a prescription. Depending on the action chosen, the system transitions to corresponding states like Select Timeslot, Appointment Successful/Rejected, or Patient File
Updated. Each state includes specific actions (do, entry, exit) to show what happens when
entering, exiting, or during that state.

Figure 13 State Diagram



How This Helps:

- It provides a clear, sequential understanding of how the system behaves based onuser actions.
- Helps designers and developers visualize the flow of control and how the application transitions between states.
- Useful for debugging and refining system logic and identifying unnecessary or missing states.
- Enhances communication between stakeholders, like developers, testers, and clients, by showing expected system behavior.

How It Differs from BPMN:

- Focus: A state diagram focuses on the system's states and transitions rather than the flow of activities or responsibilities among different actors.
- Actors: BPMN diagrams usually involve multiple participants (like patient, receptionist, admin), whereas state diagrams typically show the behavior of a single object or system.
- Detail Level: BPMN emphasizes process steps and task ownership, while state diagrams highlight status and response to events.
- Use Case: BPMN is best for workflow modeling, whereas state diagrams are more suited for system behavior modeling.

4.7 Sequence Diagram

The sequence diagram shown provides a detailed view of the interactions between various system components and users in a healthcare management system. It captures the chronological flow of messages exchanged when a patient logs in, schedules, cancels, or reschedules an appointment, or requests a prescription. The diagram clearly shows how different actors—Patient, Main Page, Login Page, Database, Appointments Page, Receptionist, and Doctor—collaborate to fulfill user requests.

Key Benefits of the Sequence Diagram:

- Chronological Clarity: It presents the step-by-step interaction of system components over time, making it easier to trace how an event unfolds from initiation to completion.
- Responsibility Tracking: Each vertical lifeline helps identify which component is responsible for which action—useful for debugging or updating system behavior.
- Error and Exception Handling: It accounts for scenarios like unavailable time slots, cancellation denial, or prescription rejection, showing how the system gracefully handles such cases.
- Notification Logic: It clearly illustrates when SMS notifications are sent and under what conditions, which is crucial for user communication.

Comparison with the State Diagram:

- The state diagram focuses on the system's internal state changes based on user events.
- The sequence diagram, on the other hand, emphasizes external interactions and the order of operations between objects or participants in the system.
- While the state diagram helps in modelling system behavior, the sequence diagram is better for designing system collaboration and communication.

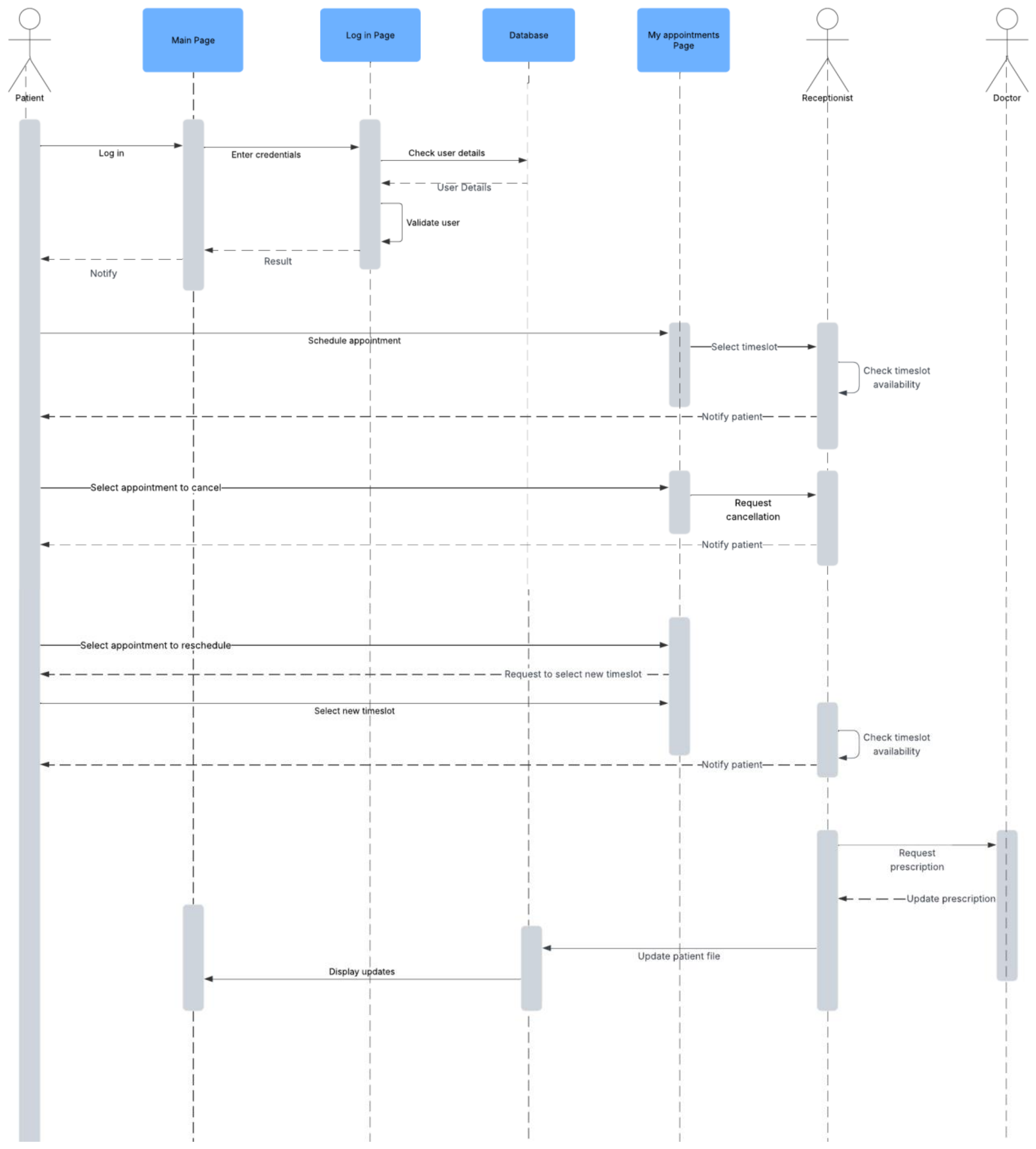


Figure 14: Sequence Diagram

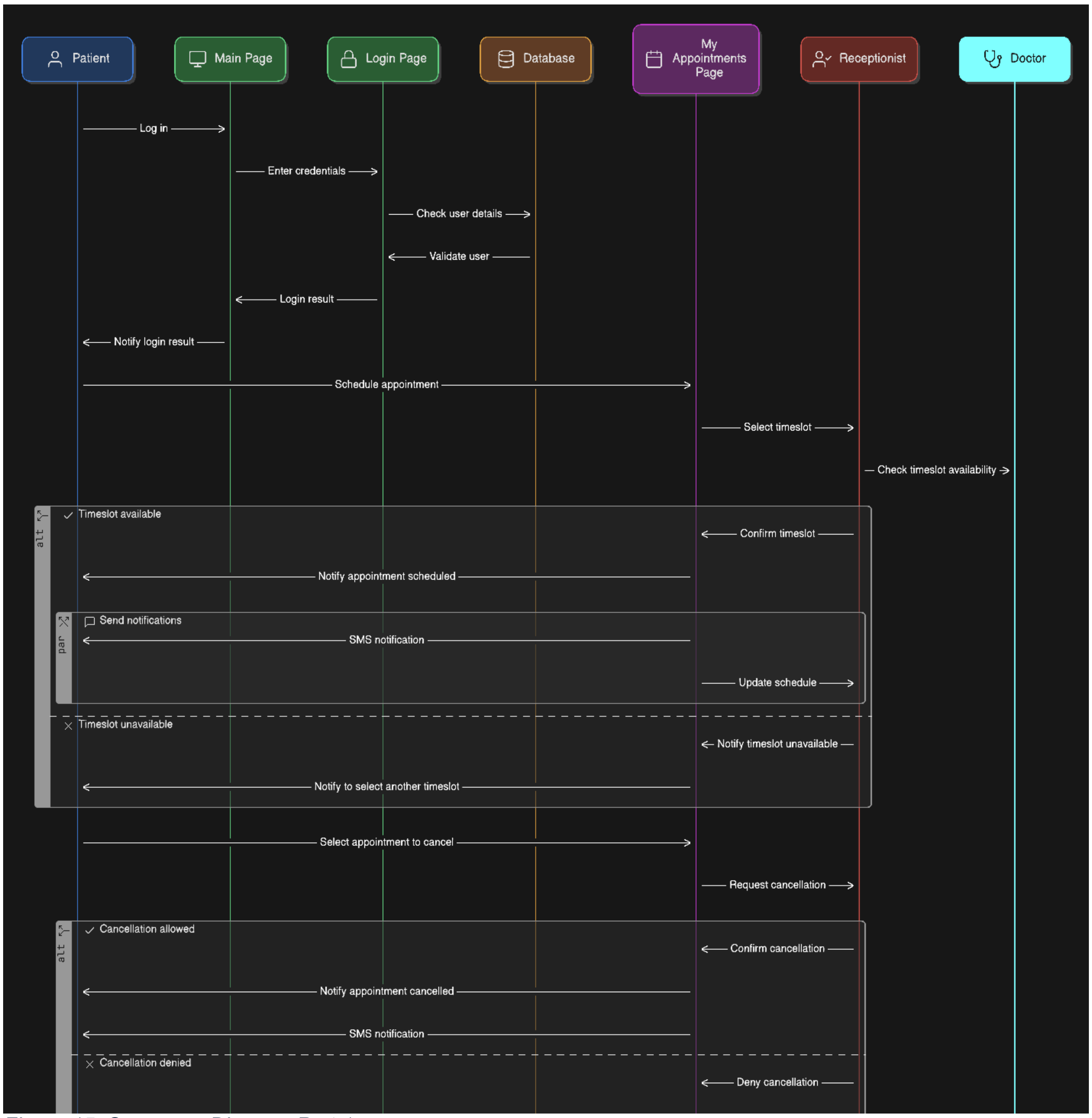


Figure 15: Sequence Diagram Part 1

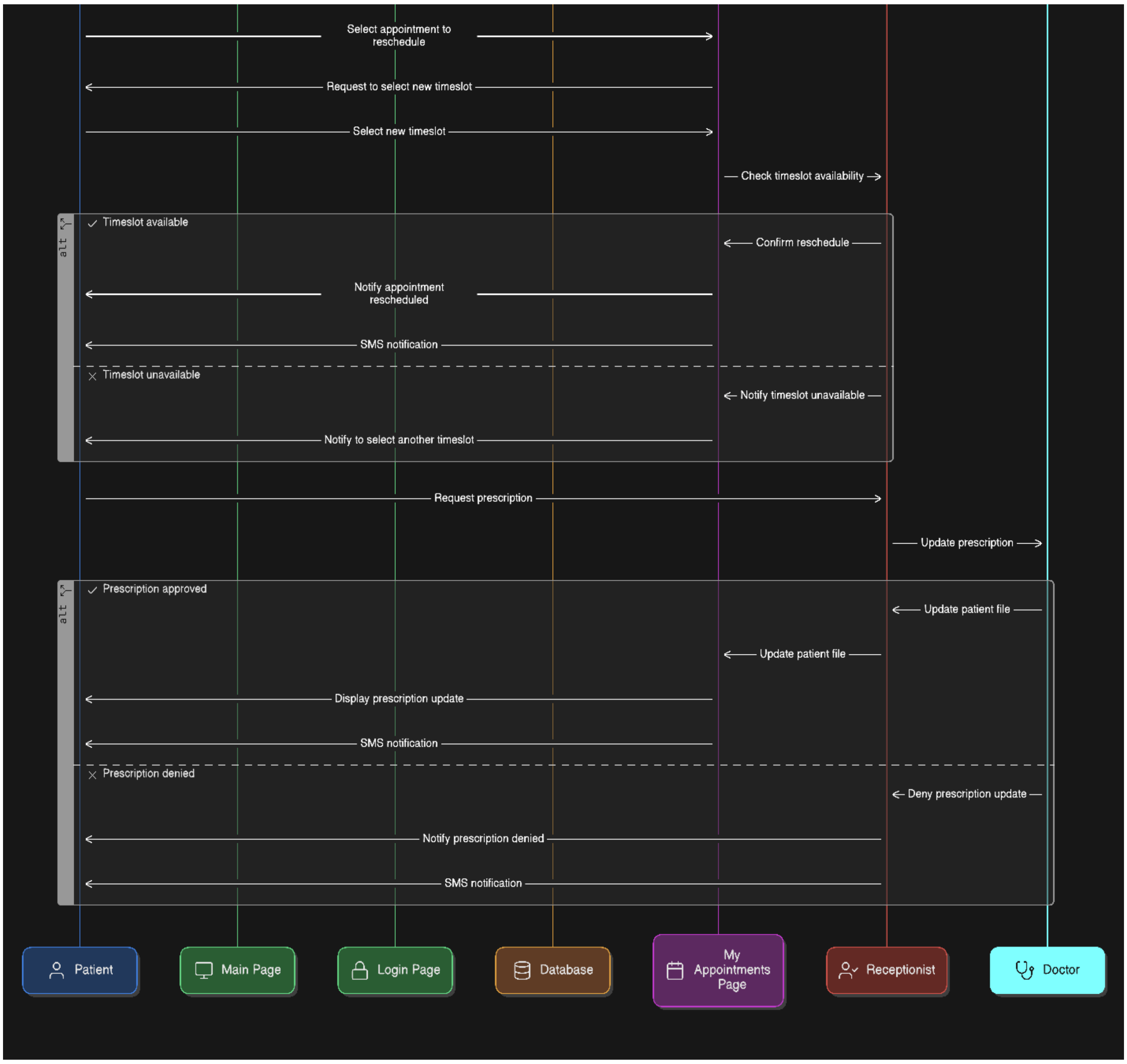


Figure 16: Sequence Diagram Part 2

4.8 Communication Diagram

The communication diagram depicted below offers a structured view of the interactions between different components and actors within the Patient Management System. It focuses on the messages exchanged to complete typical user operations like login, appointment scheduling, cancellation, rescheduling, and prescription requests.

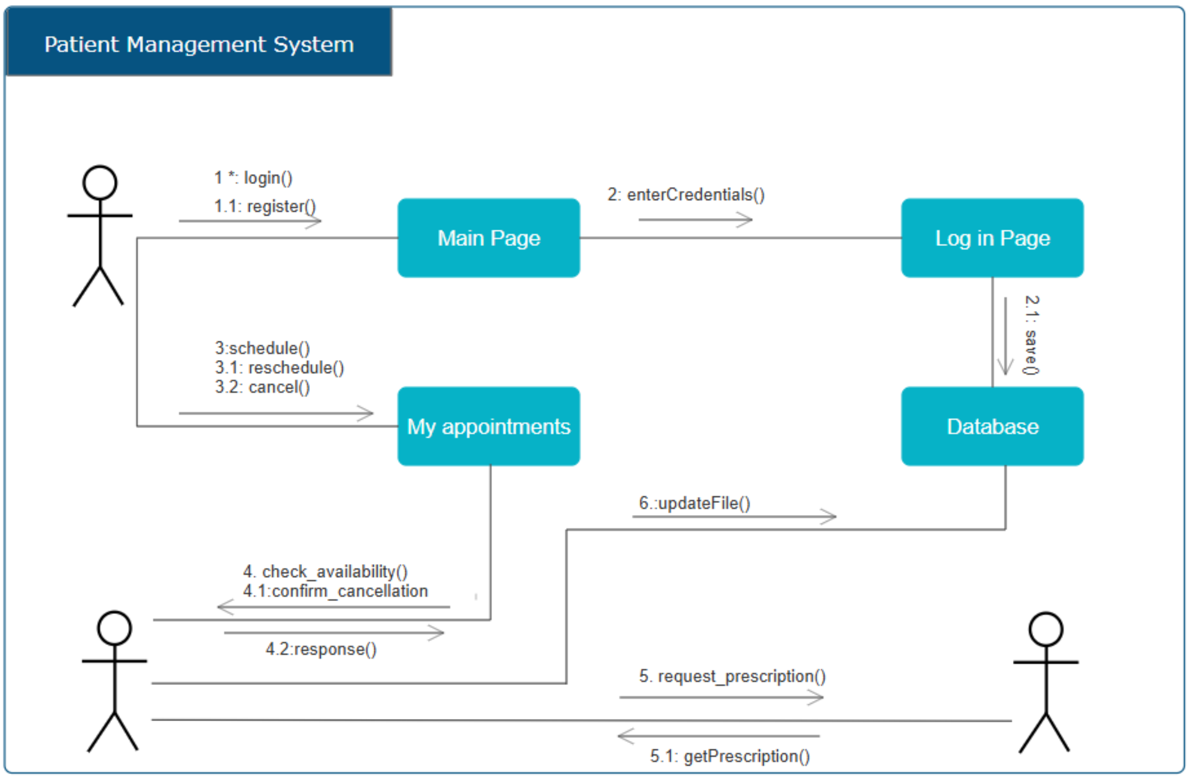


Figure 17: 7. Communication Diagram

Key Features of the Communication Diagram:

- Interaction Focused: Unlike sequence diagrams, which emphasize the order of interactions over time, this diagram emphasizes which objects interact and the messages passed between them.
- Numbered Messages: Each interaction is labeled in sequence (e.g., 1, 1.1, 2...) showing message flow in a logical order without being bound to time as rigidly as in sequence diagrams.

- Component Relationships: It helps illustrate how key system components (e.g., Main Page, My Appointments, Database, etc.) collaborate to complete tasks.
- Actor Roles: The diagram highlights the roles of Patient and Doctor, helping to clarify who initiates what action and how the system responds.

Benefits of the Communication Diagram:

- Simplified View: It provides a compact overview of the system's message exchange patterns, ideal for understanding system behavior at a glance.
- Modular Understanding: Shows modular responsibility — e.g., login validation via the login page and database, appointment management via the My Appointments page.
- Traceability: Easily trace how a user's action (e.g., request_prescription) flows through the system components.

Comparison with Sequence Diagram:

- Communication diagrams highlight message paths between components, while sequence diagrams emphasize temporal order and lifelines.
- Communication diagrams are more static and structural, ideal for showing how objects are connected, whereas sequence diagrams are dynamic, better for understanding the timing and order of interactions.

4.9 CRC Cards

CRC (Class-Responsibility-Collaborator) cards help identify the responsibilities of each class and how they collaborate. They were used to define the core classes of the system and their interactions before designing the class diagram.

Patient Class

- CRC: Responsibilities like login, view appointments, request/cancel/reschedule appointments, and request prescriptions.
- Class Diagram: Implements methods for all these responsibilities: login(), logout(), viewAppointment(), etc.
- Collaborators:
  - CRC shows Appointment, LoginSession, Prescription, and Receptionist.
  - UML backs this with foreign key links and method calls to related classes.

Patient	
Responsibilities	**Collaborators**
Log in to system	Appointment
View appointments	Receptionist
Request appointments	LoginSession
Cancel/reschedule appointments	Prescription
Request prescriptions	

Appointment Class

- CRC: Focused on scheduling and status tracking.
- Class Diagram: Supports with methods like scheduleAppointment(), cancelAppointment(), checkStatus(), etc.
- Collaborators:
    - CRC lists Patient, Doctor, and Receptionist.
    - UML shows foreign keys and references to those classes.

Appointment	
Responsibilities	**Collaborators**
Schedule appointments	Patient
Track appointment status (scheduled, cancelled, rescheduled)	Doctor
	Receptionist

Doctor Class

- CRC: Responsibilities include confirming availability and providing prescriptions.
- Class Diagram: Implements checkAvailability() and writePrescription() accordingly.
- Collaborators: Patient, Prescription, and Receptionist, clearly shown in both models.

Doctor	
Responsibilities	**Collaborators**
Confirm availability	Prescription
Provide prescriptions	Receptionist

Receptionist Class

- CRC: Administrative hub—handles appointment management and prescription forwarding.
- Class Diagram: Implements these with scheduleAppointment(), cancelAppointment(), and forwardPrescriptionRequest().
- Collaborates with all three actors (Patient, Doctor, Appointment).

Receptionist	
Responsibilities	**Collaborators**
Schedule/cancel/reschedule appointments	Appointment
Forward prescription requests	Patient
Notify patients	Doctor

Prescription Class

- CRC: Stores and links medication info to appointments and doctors.
- Class Diagram: Includes necessary attributes (medication, dosage, notes) and methods like create() and view().

Prescription	
Responsibilities	**Collaborators**
Store medication and dosage info	Receptionist
Link to appointments	Doctor

Login_Sessions Class

- CRC: Responsible for tracking logins, validating users.
- Class Diagram: Contains appropriate fields (loginTime, logoutTime) and methods like startSession() and endSession().

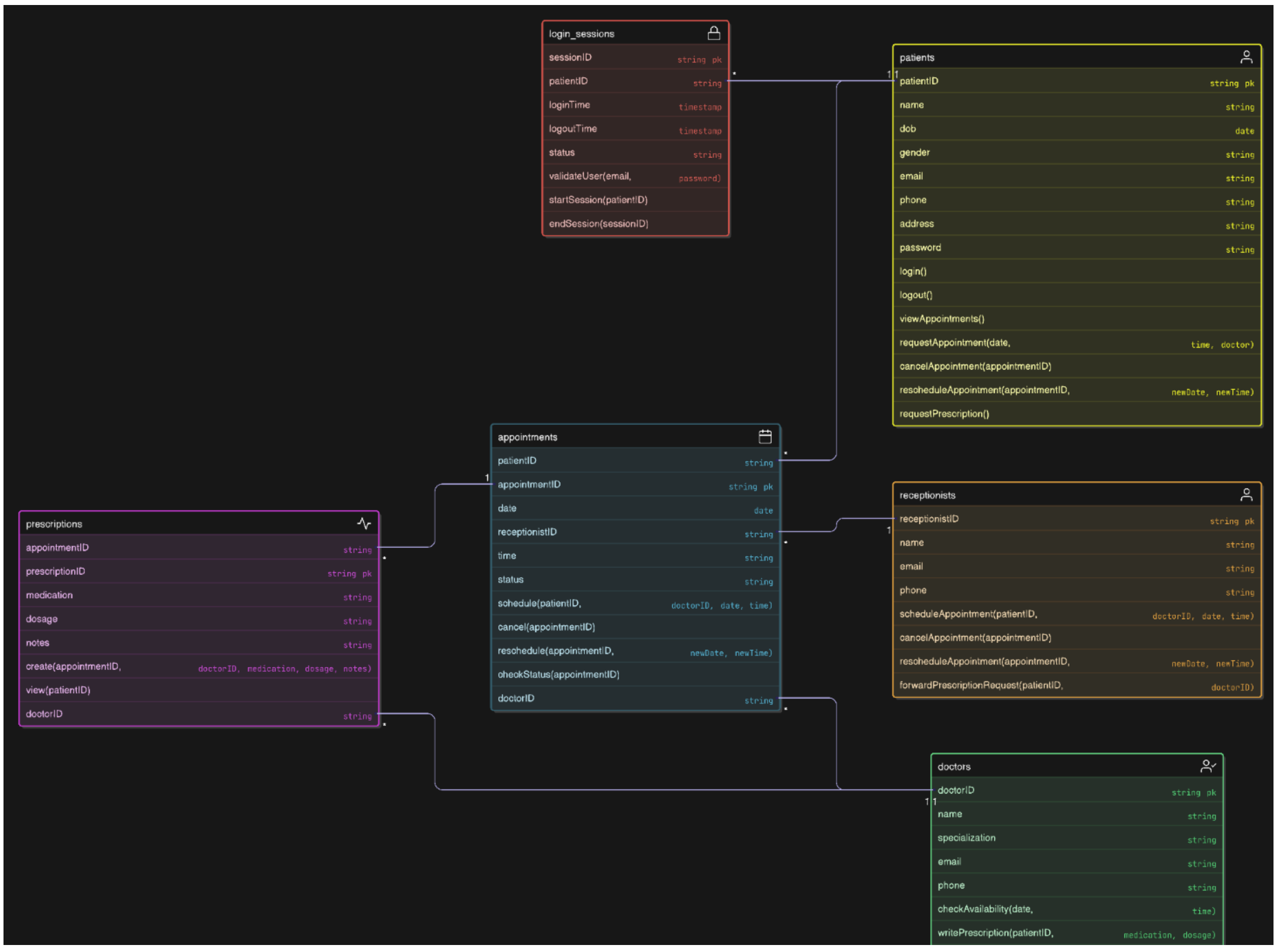
LoginSession	
Responsibilities	**Collaborators**
Track login/logout times	Patient
Validate login credentials	Database

Why should we use CRC?

CRC (Class–Responsibility–Collaborator) cards are used primarily during the early stages of object-oriented design to clarify the role of each class in a system before any coding begins. By identifying what each class is responsible for and which other classes it collaborates with, CRC cards help promote high cohesion and low coupling—ensuring each class has a focused purpose while relying on others for supporting tasks. This method encourages clean, maintainable design and serves as a lightweight, collaborative way for teams to brainstorm and visualize system behaviour. CRC also acts as a bridge between functional requirements and technical design by helping translate user stories or use cases into class responsibilities. It allows designers to quickly spot redundancy, missing functionality, or overloaded classes. Overall, CRC cards provide a simple yet powerful way to model the behaviour and interactions of classes in a system, focusing on how responsibilities are logically distributed.

4.10 Class Diagram

The class diagram effectively models a clinic management system, clearly separating responsibilities across different roles such as patients, doctors, and receptionists. It defines appropriate attributes and methods for each class, ensuring that the system's core functions—like scheduling appointments, managing prescriptions, and handling logins—are well represented. The relationships between entities are logically structured, and the inclusion of dedicated classes for appointments, prescriptions, and session management shows thoughtful system organization. Overall, the diagram presents a coherent and functional design suitable for a healthcare context.



login_sessions	
sessionID	string pk
patientID	string
loginTime	timestamp
logoutTime	timestamp
status	string
validateUser(email,	password)
startSession(patientID)	
endSession(sessionID)	

patients	
patientID	string pk
name	string
dob	date
gender	string
email	string
phone	string
address	string
password	string
login()	
logout()	
viewAppointments()	
requestAppointment(date,	time, doctor)
cancelAppointment(appointmentID)	
rescheduleAppointment(appointmentID,	newDate, newTime)
requestPrescription()	

appointments	
patientID	string
appointmentID	string pk
date	date
receptionistID	string
time	string
status	string
schedule(patientID,	doctorID, date, time)
cancel(appointmentID)	
reschedule(appointmentID,	newDate, newTime)
checkStatus(appointmentID)	
doctorID	string

prescriptions	
appointmentID	string
prescriptionID	string pk
medication	string
dosage	string
notes	string
create(appointmentID,	doctorID, medication, dosage, notes)
view(patientID)	
doctorID	string

receptionists	
receptionistID	string pk
name	string
email	string
phone	string
scheduleAppointment(patientID,	doctorID, date, time)
cancelAppointment(appointmentID)	
rescheduleAppointment(appointmentID,	newDate, newTime)
forwardPrescriptionRequest(patientID,	doctorID)

doctors	
doctorID	string pk
name	string
specialization	string
email	string
phone	string
checkAvailability(date,	time)
writePrescription(patientID,	medication, dosage)

Figure 18: Class Diagram

4.11 Techniques to find underlying problems

4.11.1 5 Why’s

The **5 Whys** is a simple but powerful **root cause analysis technique** used to explore the cause-and-effect relationships underlying a problem. The method involves asking "Why?" five times to drill down to the **root cause** of an issue, rather than just addressing surface symptoms.

Problem: A patient was unable to book an appointment online.

---

1. Why was the patient unable to book an appointment online?

→ Because the system showed no available time slots.

2. Why were there no available time slots?

→ Because the receptionist had not updated the doctor's availability in the system.

3. Why did the receptionist not update the availability?

→ Because there was no notification or reminder for the receptionist to do so.

4. Why was there no notification/reminder mechanism in place?

→ Because the system design did not include automated task reminders for staff.

5. Why did the system design omit automated reminders?

→ Because the project requirements did not account for workflow automation features.

Root Cause: The initial system requirements lacked workflow automation (e.g., reminders for staff), leading to manual dependency and missed updates.

4.11.2 The 3 R's

The 3 R's — Roll, Rule, and Route — are often used in business process management or workflow design, especially in automation systems or enterprise applications. Here's what each one means:

---

1. Roll (or Role)

- Refers to who performs the task.
- It's about assigning the right person or role to a specific step in a process.
- Example: A Doctor writes prescriptions, a Receptionist schedules appointments.

---

2. Rule

- Refers to the business logic or conditions that govern the process.
- These are if-then conditions, validations, or decision-making logic.
- Example: "Only a doctor can approve a prescription," or "A patient must have a valid appointment before receiving a prescription."

---

3. Route

- Refers to the flow or path of the process.
- It defines where the task goes next, depending on the rules or outcomes.
- Example: After a prescription is approved, it routes to the pharmacy system or notifies the patient.

5. Implementation

5.1 Technology Evaluation for Patient Management System

Next.js

Pros

Next.js is a powerful React framework that offers server-side rendering (SSR) and static site generation, which significantly enhance the performance and search engine optimization (SEO) of your application—crucial for public-facing components like doctor profiles or patient dashboards. It also includes built-in routing, image optimization, and API routes, making it ideal for building both the frontend and backend logic of your system within one framework. This can simplify tasks like appointment booking or user authentication, while also boosting overall development speed and scalability.

Cons

Next.js can introduce complexity for beginners who are not familiar with concepts like server-side rendering (SSR), static generation, or hybrid routing. Managing SSR and client-side rendering properly may lead to increased cognitive load and debugging difficulties. Furthermore, deploying and scaling Next.js apps with advanced features might require a deeper understanding of server environments or cloud platforms. The framework also updates frequently, which can result in breaking changes or the need for continuous learning to stay current.

TypeScript

Pros

TypeScript brings type safety and enhanced development tools to your project. By adding static types to your JavaScript code, it helps catch bugs early during development rather than at runtime. This leads to more robust and maintainable code, especially as your application grows in size and complexity. TypeScript also improves the developer experience by enabling features like autocompletion, better documentation, and easier refactoring, which collectively lead to higher productivity and fewer logical errors.

Cons

TypeScript, although beneficial for large-scale applications, comes with a steep learning curve for developers who are new to typed languages. It requires more verbose code compared to JavaScript, which can slow down development initially. Setting up proper type definitions, especially for third-party libraries or custom data structures, can sometimes be time-consuming and frustrating. Also, overusing types can lead to rigid code that's harder to refactor if the app's requirements change frequently.

Twilio

Pros

Twilio is an industry-leading communications platform that allows you to integrate SMS, voice, and messaging services into your system. For a Patient Management System, this means you can send appointment reminders, confirmations, or critical alerts directly to patients' phones via SMS or WhatsApp. Twilio also offers options for integrating voice or video calls—ideal for telemedicine features. Importantly, Twilio ensures high standards of security and compliance with data protection regulations like HIPAA or GDPR, which is essential when dealing with medical information.

Cons

Twilio, despite being a robust and feature-rich communication platform, can become expensive as your application's messaging or calling volume increases. It uses a pay-as-you-go model, and costs can escalate if you're sending frequent SMS reminders or using voice services heavily. Integration with Twilio APIs also demands careful setup, particularly when dealing with user authentication, secure message handling, or voice call routing. In regions with limited telecom infrastructure, message delivery may not always be consistent or timely.

Tailwind CSS

Pros

Tailwind CSS is a utility-first CSS framework that lets you design user interfaces rapidly and consistently. Instead of writing custom styles, you apply pre-built utility classes directly in your HTML or JSX, which drastically speeds up the process of building and maintaining UI components. Tailwind ensures that your app maintains a clean, modern, and responsive design with minimal effort. It also supports customization through themes and extensions, making it easy to tailor the design to match your project's branding or specific user experience goals.

Cons

Tailwind CSS, while great for rapid UI development, often leads to bloated HTML due to the heavy use of utility classes inline. This can make the markup harder to read and maintain, especially for developers not familiar with the framework. Unlike traditional CSS, Tailwind moves styling out of separate style sheets, which can be jarring for teams used to a more component-based or semantic styling approach. Additionally, customizing Tailwind beyond its default configurations may require time and effort to fully understand its theming system and configuration files.

5.2 Prototypes and sketches

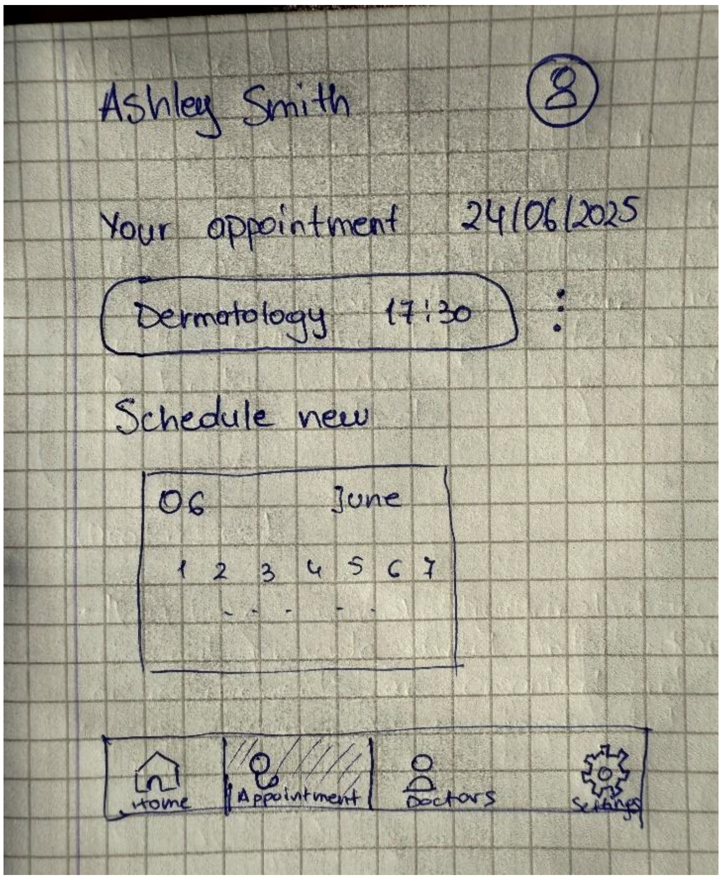


Figure 19: 1st Sketch

This screen is a patient-facing dashboard that displays upcoming appointment details and allows the user to schedule a new one.

Key Elements:

- User Info: Displays the patient's name (Ashley Smith).
- Appointment Details: Shows a scheduled dermatology appointment for June 24, 2025, at 17:30.
- Schedule New Appointment: Includes a simple calendar UI for the first week of June to pick a new appointment date.
- Navigation Bar: Bottom menu includes:
  - Home
  - Appointment (highlighted)
  - Doctors
  - Settings

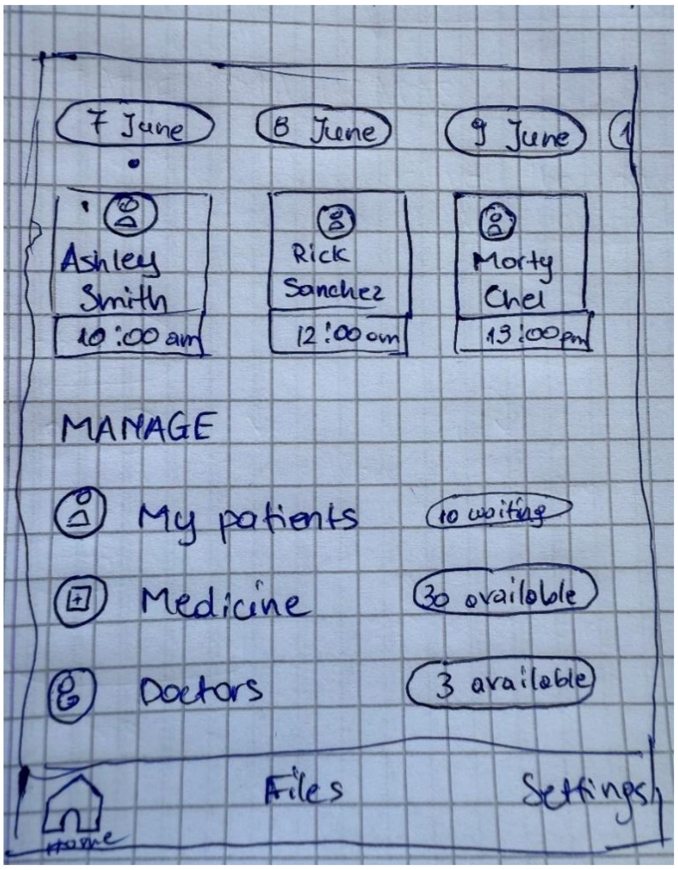


Figure 20: 2nd Sketch

This view is intended for doctors or administrative staff to manage daily patient appointments and access management features.

Key Elements:

- Daily Schedule View: Displays scheduled patients per day (June 7–9), including:
  - Ashley Smith (10:00 AM)
  - Rick Sanchez (12:00 PM)
  - Morty Chel (1:00 PM)
- Management Section:
  - My Patients: 10 waiting
  - Medicine: 30 available
  - Doctors: 3 available
- Navigation Bar: Includes Home, Files, and Settings

This is a high-fidelity digital mock-up showing a refined dashboard interface for the app.

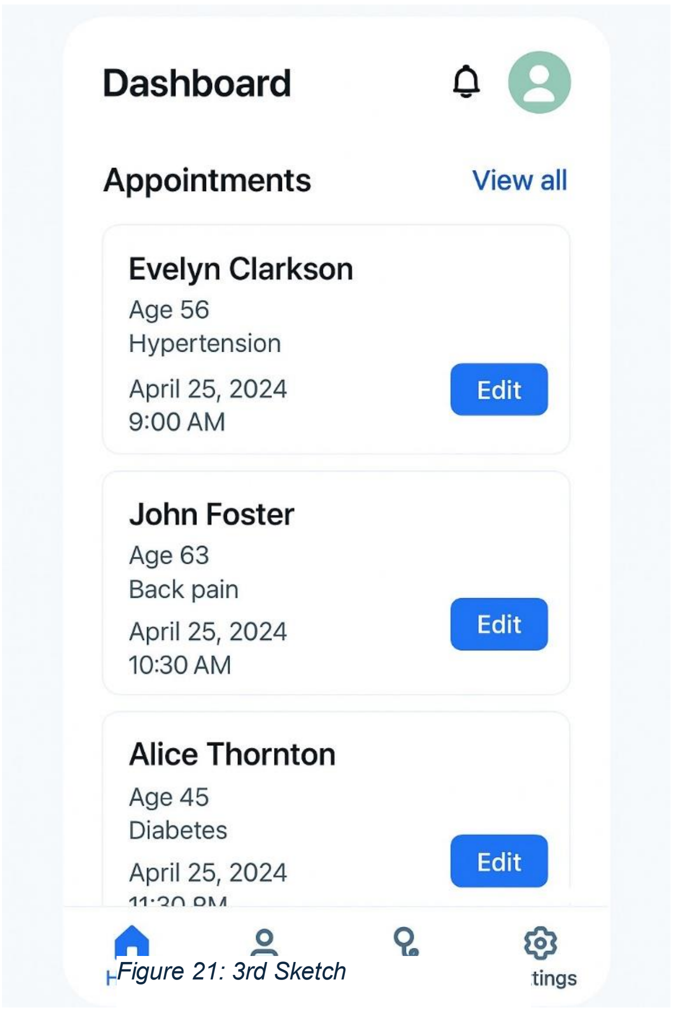
Key Elements:

- Dashboard Title: Clear heading for user context.
- Appointments List:

  Patient cards with:

  - Name
  - Age
  - Health condition
  - Appointment date & time
  - Edit button
- Navigation Bar: Modern icons for:
  - Home (active)
  - Patients
  - Doctors
  - Settings

Purpose:
This represents the polished user interface using Tailwind CSS, giving a glimpse into the final design of the application. It reflects a clean and responsive layout suitable for both patients and medical staff.



Evelyn Clarkson
Age 56
Hypertension
April 25, 2024
9:00 AM

John Foster
Age 63
Back pain
April 25, 2024
10:30 AM

Alice Thornton
Age 45
Diabetes
April 25, 2024
11:30 PM

Figure 21: 3rd Sketch

5.3 Detailed Designs

Page 1: User Registration Page (Welcome to CarePulse)

Purpose:
To allow new users to register for the CarePulse platform.

Key Functionalities:

- Full Name, Email, Phone Number Inputs: Fields where the user can enter their basic personal details.
- Icons next to input fields visually guide the user on what information to enter.
- "Get Started" Button: Submits the registration information and likely navigates to the next step of onboarding.
- "Already have an account?" Link + Log In Button: Redirects users to the login page if they already have an existing account.
- Bottom Navigation Bar: Provides quick access to Home, Appointments, Doctors, and Settings.

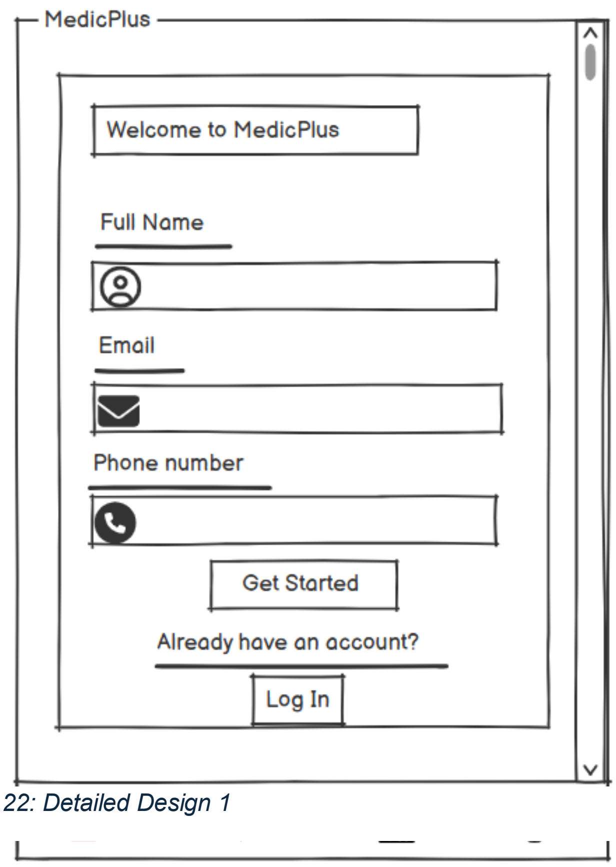


Figure 22: Detailed Design 1

Page 2: Appointment Dashboard Page

Purpose:
To manage current and future appointments, and help the user find the clinic's location.

Key Functionalities:

- Upcoming Appointments Section: Displays existing appointment details (date, time, department).
- "Schedule New Appointment" Calendar: Lets users select a date to book a new appointment.
- "Our Location" Section: Integrates a map view to show the physical location of the clinic.
- User Profile Icon: Provides access to account settings or user information.
- Bottom Navigation Bar: Same as Page 1.

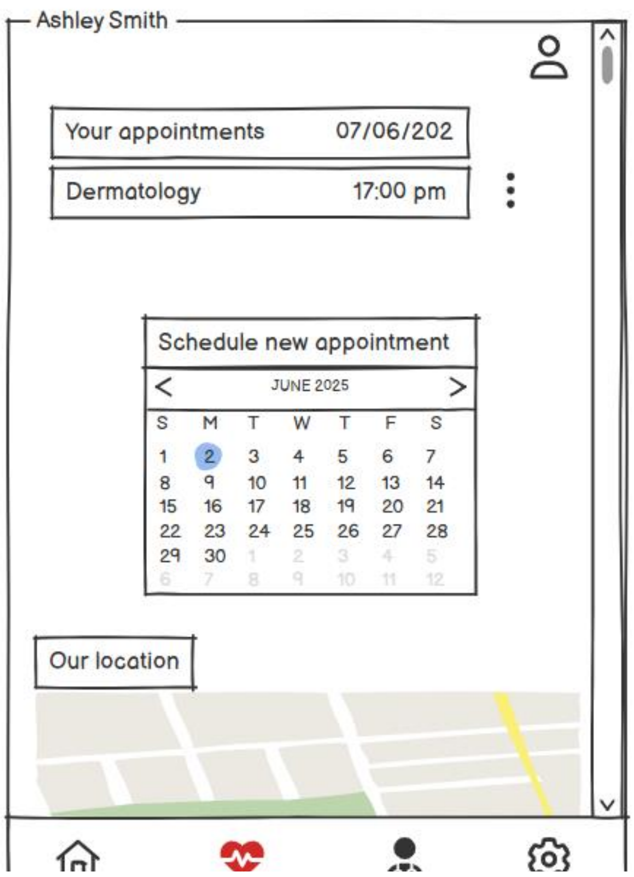


Figure 23: Detailed Design 2

Page 3: Doctor Directory Page

Purpose:
To help users browse and view information about available doctors.

Key Functionalities:

- Doctor Cards: Each doctor is represented with an image, name, and specialization (e.g., Cardiologist, Ophthalmologist).
- "More Info" Button: Likely provides additional details such as availability, experience, or the ability to book an appointment with that doctor.
- Top Banner with User Name: Indicates the logged-in user.
- Bottom Navigation Bar: Remains consistent for easy navigation.

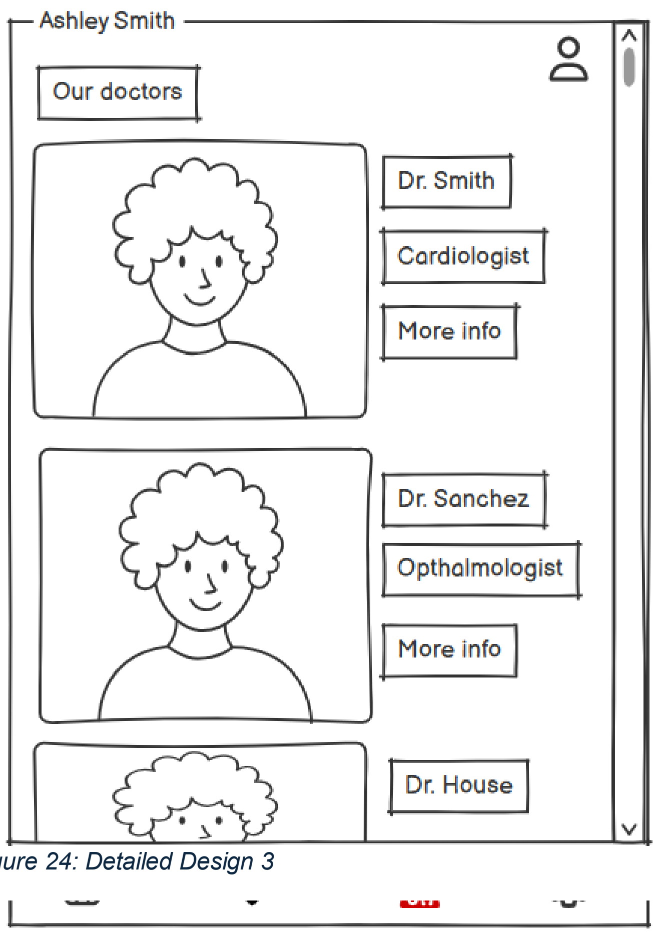
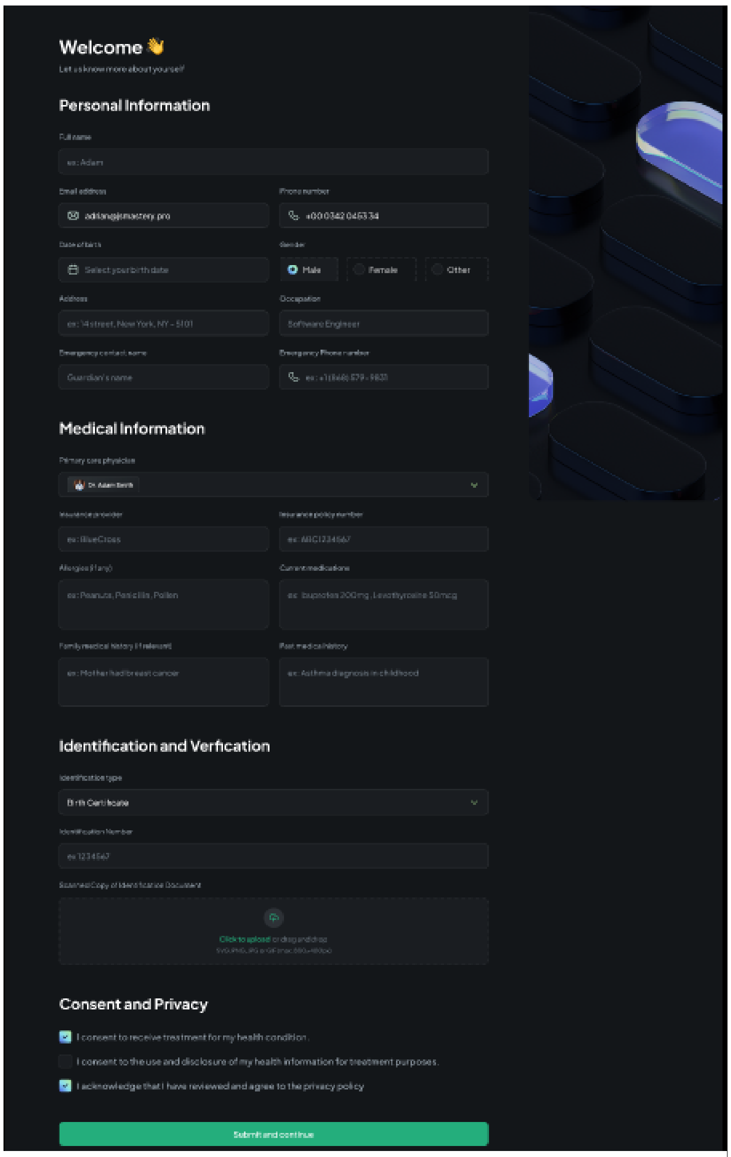


Figure 24: Detailed Design 3

5.4 Software Screenshots

Register page



Welcome 👋

Let us know more about yourself

Personal Information

Full name: ex: Adam

Email address: adrian@jsmastery.pro

Phone number: +00 0342 0453 34

Date of birth: Select your birth date

Gender: [✓ Male] Female Other

Address: ex: 14 street, New York, NY - 5101

Occupation: Software Engineer

Emergency contact name: Guardian's name

Emergency Phone number: ex: +1(868) 579-9831

Medical Information

Primary care physician: Dr. Adam Smith

Insurance provider: ex: BlueCross

Insurance policy number: ex: ABC1234567

Allergies (if any): ex: Peanuts, Penicillin, Pollen

Current medications: ex: Ibuprofen 200mg, Levothyroxine 50mcg

Family medical history (if relevant): ex: Mother had breast cancer

Past medical history: ex: Asthma diagnosis in childhood

Identification and Verfication

Identification type: Birth Certificate

Identification Number: ex 1234567

Scanned Copy of Identification Document

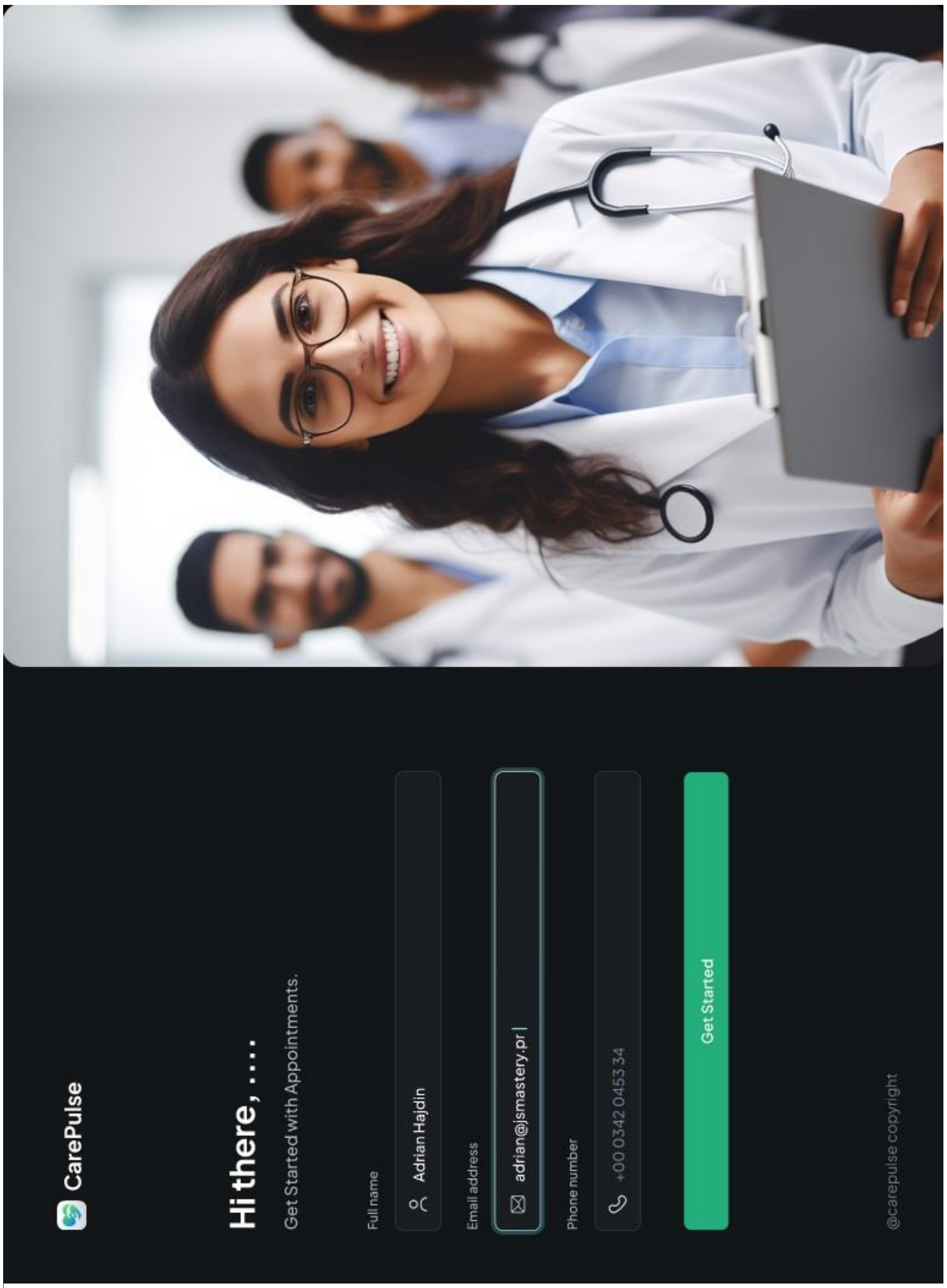
Click to upload or drag and drop
SVG, PNG, JPG or GIF (max. 800x400px)

Consent and Privacy

- [✓] I consent to receive treatment for my health condition.
- [ ] I consent to the use and disclosure of my health information for treatment purposes.
- [✓] I acknowledge that I have reviewed and agree to the privacy policy

Figure 25: Register Page

Log in Page



Hi there, ...

Get Started with Appointments.

Full name

Adrian Hajdin

Email address

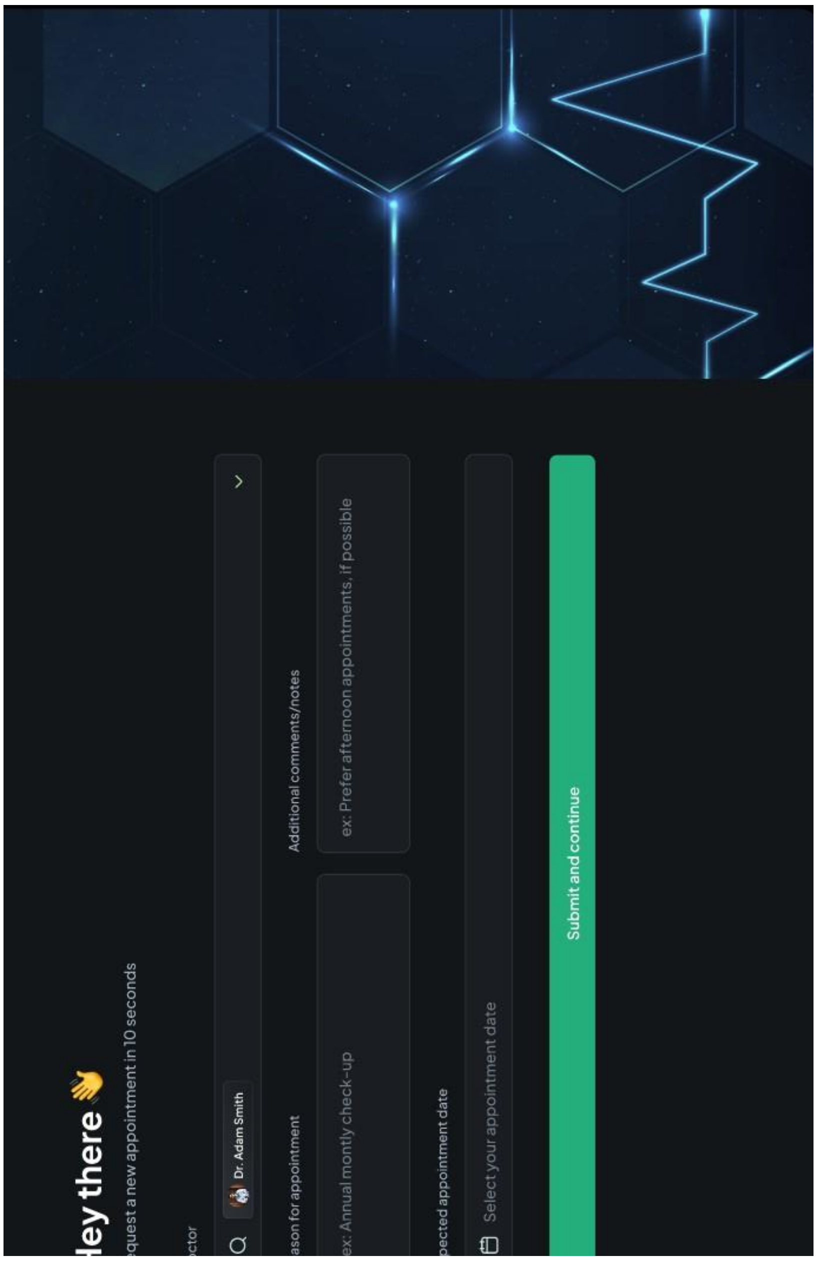
adrian@jsmastery.pr

Phone number

+00 0342 0453 34

Figure 26: Log in Page

Request appointment



ey there 👋

quest a new appointment in 10 seconds

ctor

Dr. Adam Smith

ason for appointment

ex: Annual montly check-up

Additional comments/notes

ex: Prefer afternoon appointments, if possible

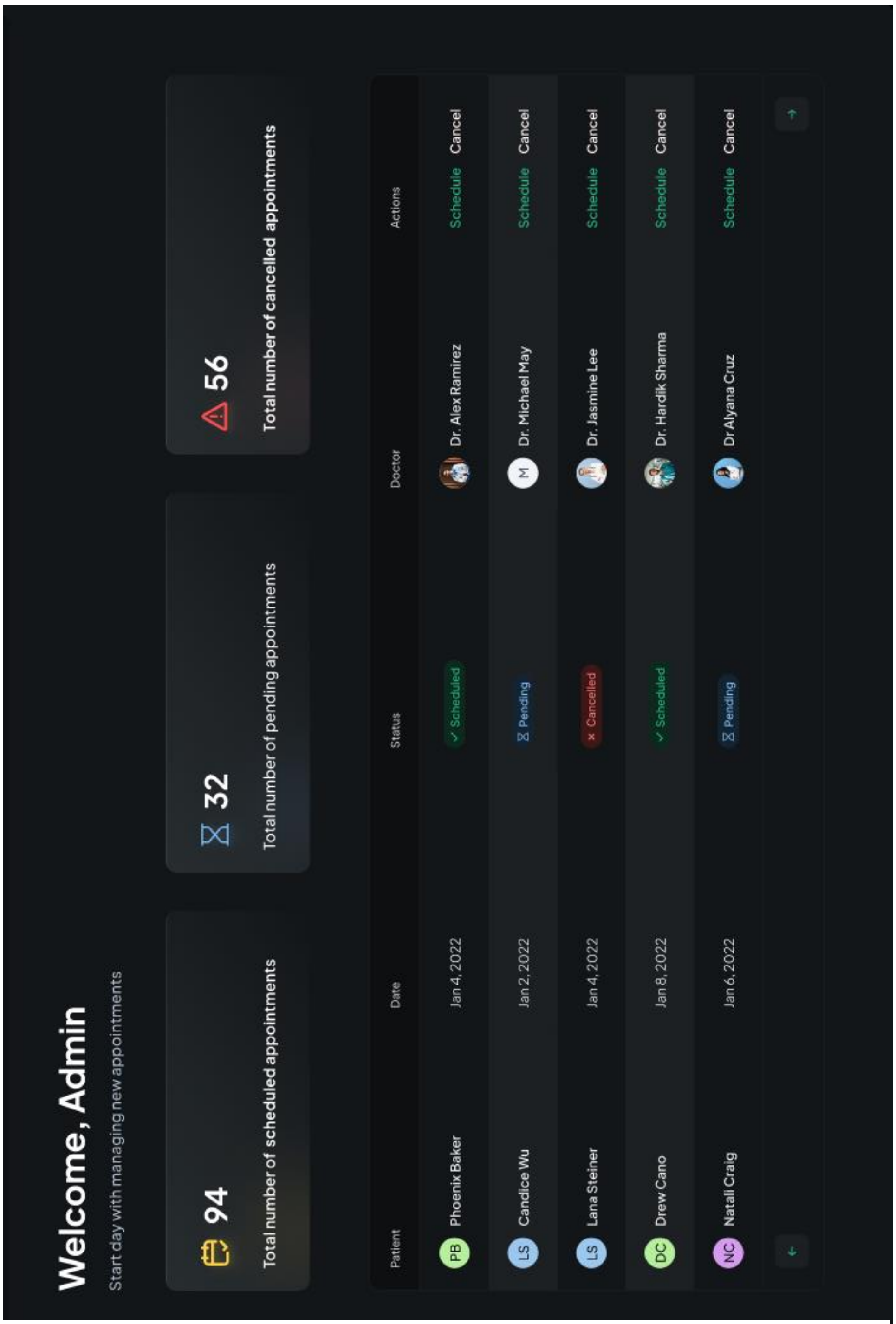
pected appointment date

Select your appointment date

Submit and continue

Figure 27: Request appointment

Admin view



Welcome, Admin

Start day with managing new appointments

94 Total number of scheduled appointments

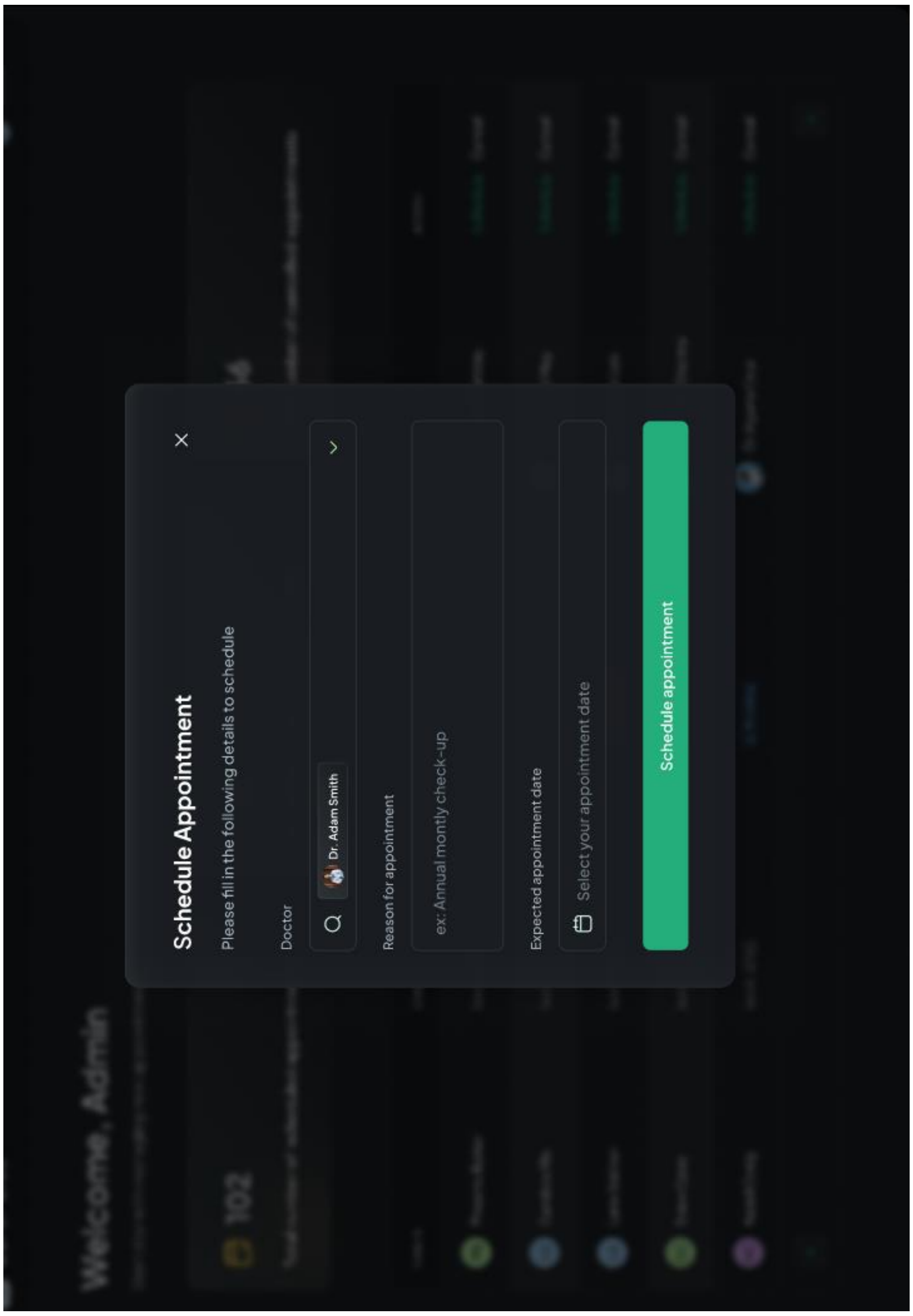
32 Total number of pending appointments

56 Total number of cancelled appointments

Patient	Date	Status	Doctor	Actions
PB Phoenix Baker	Jan 4, 2022	✓ Scheduled	Dr. Alex Ramirez	Schedule Cancel
LS Candice Wu	Jan 2, 2022	Pending	M Dr. Michael May	Schedule Cancel
LS Lana Steiner	Jan 4, 2022	× Cancelled	Dr. Jasmine Lee	Schedule Cancel
DC Drew Cano	Jan 8, 2022	✓ Scheduled	Dr. Hardik Sharma	Schedule Cancel
NC Natali Craig	Jan 6, 2022	Pending	Dr Alyana Cruz	Schedule Cancel

Figure 28: Admin view

Schedule appointment Page



Schedule Appointment

Please fill in the following details to schedule

Doctor

Dr. Adam Smith

Reason for appointment

ex: Annual montly check-up

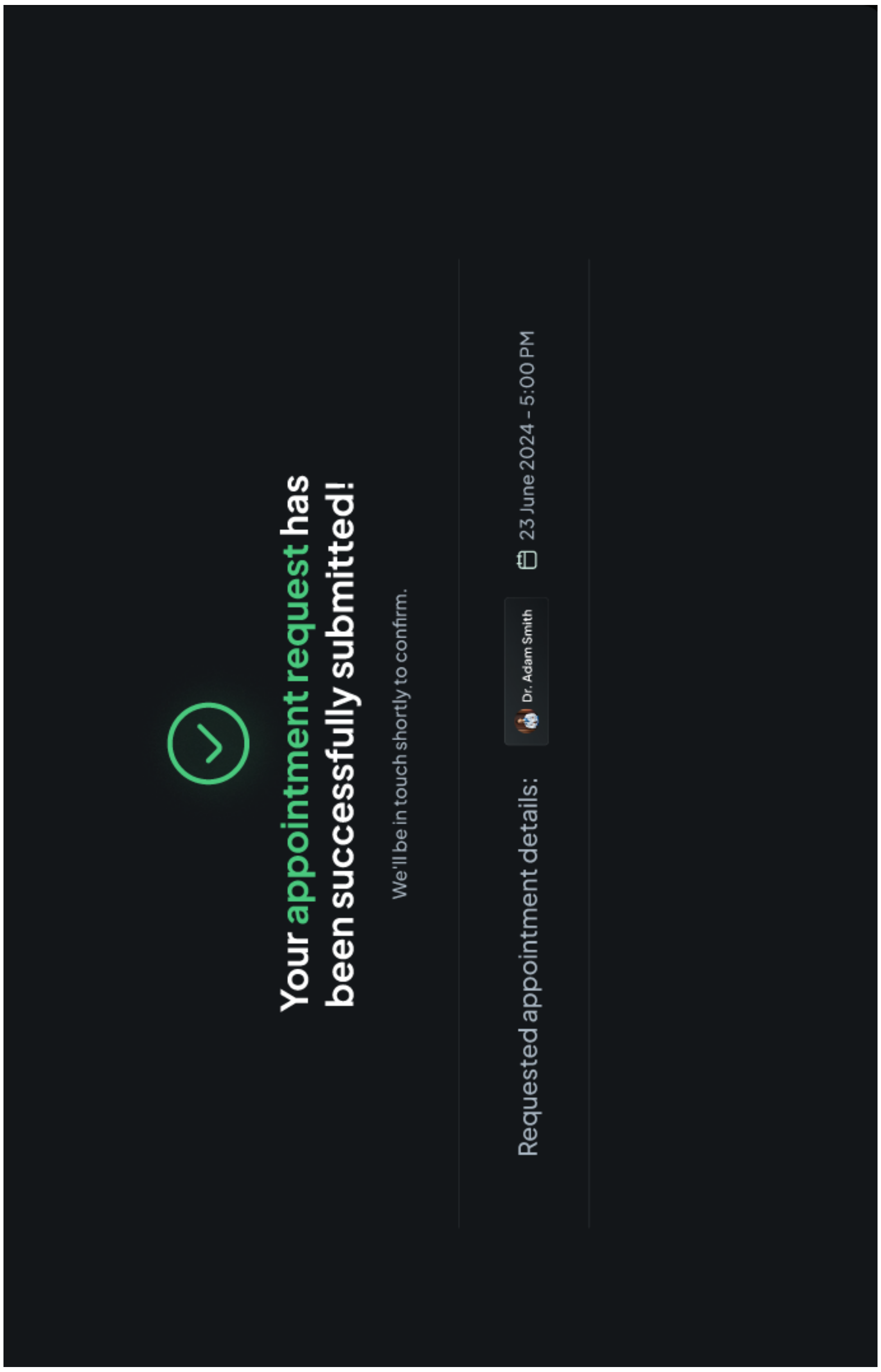
Expected appointment date

Select your appointment date

Schedule appointment

Figure 29: Schedule appointment

Appointment status Page


Your appointment request has been successfully submitted!

We'll be in touch shortly to confirm.

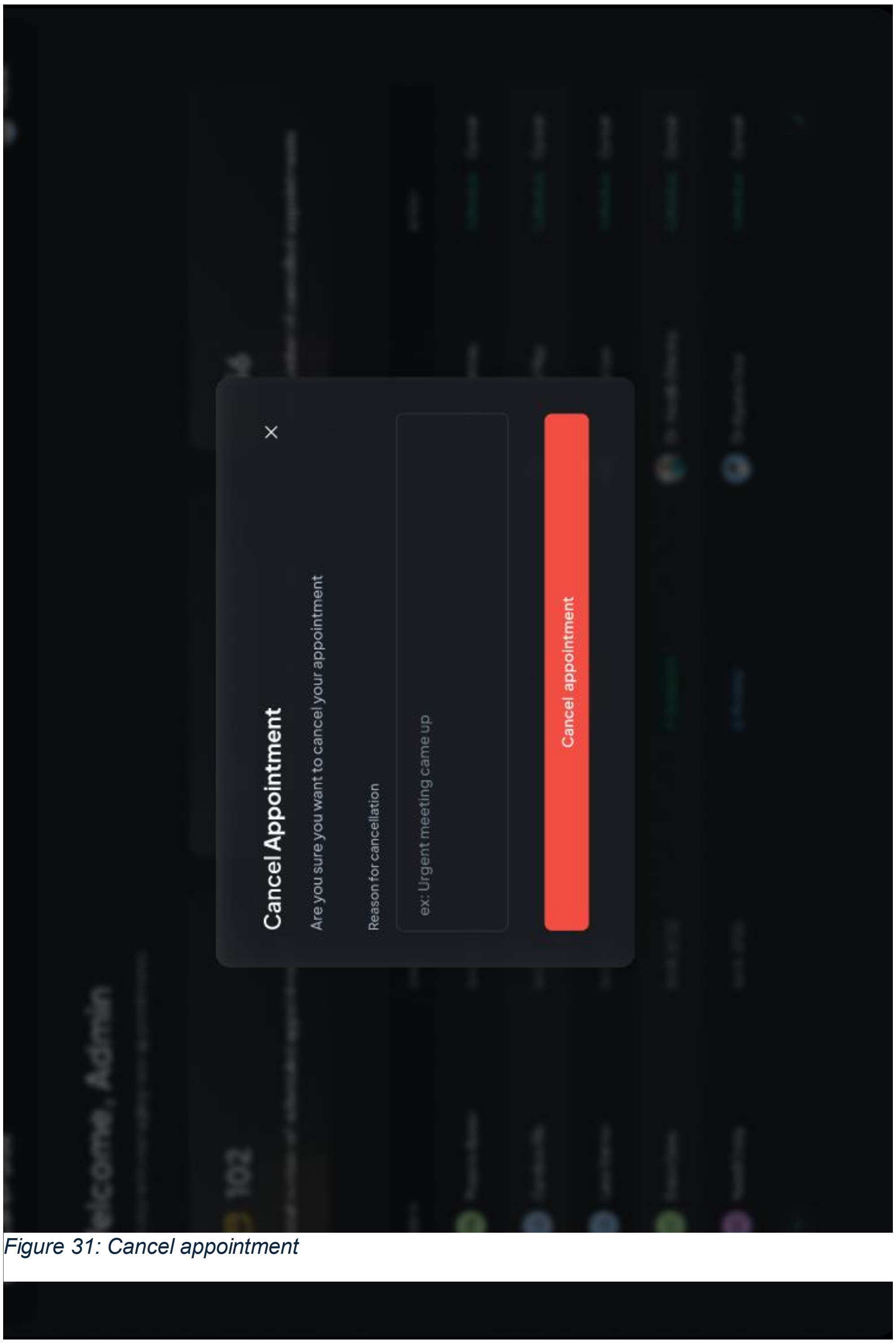
Requested appointment details:

Dr. Adam Smith

23 June 2024 - 5:00 PM

Figure 30: Appointment status page

Cancel appointment


Cancel Appointment

Are you sure you want to cancel your appointment

Reason for cancellation

ex: Urgent meeting came up

Figure 31: Cancel appointment

6. Conclusions

The Patient Management System (CarePulse) designed and developed through this project offers a comprehensive, secure, and user-centric platform aimed at digitizing and optimizing healthcare operations for clinics and outpatient environments in Albania. By addressing inefficiencies in existing manual systems, CarePulse ensures improved coordination between patients, medical staff, and administrative personnel.

The platform successfully integrates core functionalities such as patient registration, appointment scheduling, medical record management, prescription handling, communication modules, and system monitoring. Extensive attention was given to user experience (UX), ensuring accessibility for patients with varying levels of digital literacy while providing advanced control and visibility for healthcare providers.

Key technologies—Next.js, TypeScript, Twilio, and Tailwind CSS—were carefully selected for their scalability, flexibility, and compatibility with healthcare data privacy regulations. Compliance with GDPR, Albanian Data Protection Law, and healthcare regulatory standards ensures secure handling of sensitive personal and medical information.

The project applied advanced software engineering practices, including:

- Use case modeling,
- ER and relational schema design.
- Activity, sequence, BPMN, state, and communication diagrams,
- CRC and class diagrams to ensure maintainability and system scalability.

Challenges such as workflow coordination, system dependability, and real-time communication were analyzed using problem-solving frameworks like the 5 Whys and the 3 R's (Role, Rule, Route), helping to refine system processes and eliminate root causes of potential failures.

Through careful design, modular architecture, and robust security measures, CarePulse provides a strong foundation not only for initial deployment but also for future expansion — including telemedicine integration, pharmacy system collaboration, and broader hospital adoption.

Ultimately, this project demonstrates how modern software engineering, combined with domain-specific requirements, can deliver practical, scalable, and impactful healthcare solutions in Albania and beyond.

7. References

https://www.uml-diagrams.org/

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-practical-guide/

https://creately.com/blog/diagrams/uml-diagram-types-examples/

European Union (2018). General Data Protection Regulation (GDPR), EU Regulation 2016/679.

Republic of Albania. Law No. 9887, dated 10.03.2008, On Protection of Personal Data (amended).

Pressman, R. S., & Maxim, B. R. (2014). Software Engineering: A Practitioner's Approach (8th ed.). McGraw-Hill.

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

Next.js Documentation — https://nextjs.org/docs

TypeScript Documentation — https://www.typescriptlang.org/docs

Twilio API Documentation — https://www.twilio.com/docs

Tailwind CSS Documentation — https://tailwindcss.com/docs

WCAG 2.1 Accessibility Guidelines — https://www.w3.org/TR/WCAG21/