SECJ2203: Software Engineering

System Documentation (SD)

Health Consultation System

Version 2.0

31/5/2023

Faculty of Computing

Prepared by: <Group 3>

Name	Matric No
Gan Heng Lai	A21EC0176
Ng Kai Zheng	A21EC0101
Lew Chin Hong	A21EC0044
Yeo Chun Teck	A21EC0148

Revision Page

a. **Overview**

Describe the content of the current version of SD (see below – the note in red).

The current SD has an Introduction section with a brief purpose statement, project scope, definitions, acronyms, and abbreviations, as well as a section for user characteristics and system features. Multiple use cases (UC) with specifications, activity diagrams, and system sequence diagrams are included in the Specific Requirements section. Additionally, the SD includes a section for software system attributes, performance, and other requirements, as well as a section for design constraints.

b. **Target Audience**

Healthcare providers: doctors and pharmacists who will use the health consultation system to consult with patients and handle prescription medications.

Patients: individuals who will use the health consultation system to get medical advice, prescriptions, and delivery of medications.

Health sector managers: persons in charge of managing the health sector and making choices about the establishment and adoption of the health consultation system.

c. **Project Team Members**

List the team members in a table by stating their roles and the status for each assigned task e.g. by sections for this SD version (complete, partially complete, incomplete). If the assigned tasks are not done and have been assigned to other team members, state accordingly.

Member Name	Role	Task	Status
Yeo Chun Teck	Team Leader	Completed 1.1, 1.2, 1.3, and 1.4. Also provided assistance in reviewing the entire project.	Completed
Ng Kai Zheng	System Analyst	Completed 1.5, 2.1, and 2.3.	Completed
Gan Heng Lai	Software Developer	Completed 2.2.1 and 2.2.2.	Completed
Lew Chin Hong	Tester	Completed 2.2.3 and 2.4	Completed

d. **Version Control History**

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Yeo Chun Teck	Completed Chapter 1, Sections 1.1, 1.2, 1.3, and 1.4. Also provided assistance in reviewing the entire project. Chapter 3, 4 and 5 contribute by all member.	02/05/2023
1.0	Ng Kai Zheng	Completed Chapter 1, Section 1.5, and Section 2.1 and 2.3 of Chapter 2. Chapter 3, 4 and 5 contribute by all member.	02/05/2023
1.0	Gan Heng Lai	Completed Section 2.2.1 and 2.2.2 of Chapter 2. Chapter 3, 4 and 5 contribute by all member.	02/05/2023
1.0	Lew Chin Hong	Completed Section 2.2.3 and 2.4 of Chapter 2. Chapter 3, 4 and 5 contribute by all member.	02/05/2023

Note:

This System Documentation (SD) template is adapted from IEEE Recommended Practice for Software Requirements Specification (SRS) (IEEE Std. 830-1998), Software Design Descriptions (SDD) (IEEE Std. 10161998 1), and Software Test Documentation (IEEE Std. 829-2008) that are simplified and customized to meet the need of SECJ2203 course at Faculty of Computing, UTM. Examples of models are from Arlow and Neustadt (2002) and other sources stated accordingly.

Table of Contents

1. Introduction

1.1 Purpose

The purpose of system documentation for a Health Consultation system is to provide a comprehensive and detailed overview of the system's design, development, and testing. This documentation should include the System Requirements Specification (SRS), which outlines the functional and non-functional requirements of the system, as well as the System Design Document (SDD), which describes the system's architecture and design. Additionally, the System Testing Document (STD) should be included, which outlines the testing procedures used to verify that the system meets the specified requirements. The intended audience for this document may include developers, project managers, quality assurance testers, system managers, doctors, patients, the Specialized Medical Team(SMT), and pharmacists involved in the development and implementation of the Health Consultation system. The documentation serves as a reference for understanding the system's capabilities, limitations, and design decisions, and helps ensure that the system is built and tested to meet the needs of its users (AltexSoft, 2019).

1.2 Scope

This System Documentation (SD) pertains to the software product called “Health Consultation System” and is designed to provide a clear understanding of the system development process, including the System Requirements Specification (SRS), System Design Document (SDD), and System Testing Document (STD)

The “Health Consultation System” software product will allow the physicians and pharmacists to provide medical advice to patients and conduct a further discussion with their patients via online. The software product would be developed to handle the task of registration and qualification of doctors in the online system for providing services to patients. This software product would also perform as a fast and precise communication platform between doctors and patients for transferring medical information between both parties including the result of a professional diagnosis, prescribed medication, laboratory test results or other relevant papers. The software product would provide the feature of arranging appointments for medical services. Along with this, the patients would be given freedom in their participation in these services including selecting the sort of consultation they desire, modifying or canceling appointments due to difficulty in the aspect of time or other reasons. The “Health Consultation System” would ease up the task of updating, storing and organizing or managing the data that is related to the patients, doctors, appointments, consultations, and prescriptions. This would allow doctors to track the latest status of conditions. The software product would also be designed to improve the user experience by providing patients with a more user-friendly platform, allowing them to control the system more freely. Besides medical services, the software product has extended

the scope of service to the medicine delivery service. In this service, the software product would guarantee a smooth and fast medication delivery procedure. The product would prioritise the security and protection of user information, ensuring that patient data is kept confidential and secure and it would not provide any additional functionality that is not related to healthcare services.

1.3 Definitions, Acronyms and Abbreviations

Definitions of all terms, acronyms and abbreviations used are to be defined here.

Term	Definition
SD	System Documentation - Exists to explain product functionality, unify project-related information, and allow for discussing all significant questions arising between stakeholders and developers
SRS	System Requirement Specification - An overview of software under development and its intended purpose
SDD	System Design Document - A comprehensive document that outlines the architecture, components, and functionality of a system or software application, providing a detailed description of its design, implementation, and integration aspects.
STD	System Testing Document - A document that describes the planned approach and strategies for testing a system or software application, including test objectives, test cases, test scenarios, and the overall testing process.
HTML	Hypertext Markup Language - The standard markup language used for creating web pages, and defining the structure and content of a webpage using tags and elements
CSS	Cascading Style Sheets - A style sheet language used to describe the presentation and formatting of an HTML document, enabling the separation of content from its visual representation on web pages.
TCP/IP	Transmission Control Protocol/Internet Protocol - A set of communication protocols that define how data is transmitted and exchanged over networks
SSL	Secure Sockets Layer - A cryptographic protocol that secures internet communication by encrypting and authenticating data transmitted between web servers and clients
RAM	Random Access Memory - A type of computer memory that allows data to be stored and accessed randomly, providing temporary storage for running programs

1.4 References

1. Ian Sommerville (2022). Software Engineering (10th ed.). Pearson.
2. Bell, D. (2004). The sequence diagram. IBM Developer. https://developer.ibm.com/articles/the-sequence-diagram/
3. Nishadha (2022). Use Case Diagram Relationships Explained with Examples. Creately. https://creately.com/blog/diagrams/use-case-diagram-relationships/
4. Van Galen, W. (2012). Use Case Goals, Scenarios And Flows. batimes. https://www.batimes.com/articles/use-case-goals-scenarios-and-flows/
5. System documentation: Features, purpose and contents | MIS. (2015, December 19). Your Article Library. https://www.yourarticlelibrary.com/management/mis-management/system-documentation-features-purpose-and-contents-mis/70408
6. AltexSoft. (2019). Technical documentation in software development: Types, best practices, and Tools. https://www.altexsoft.com/blog/business/technical-documentation-in-software-development-types-best-practices-and-tools/

1.5 Overview

There are two parts of system documentation in the Health Consultation System: Introduction and Specific Requirements. The introduction includes:

- Purpose: Clearly states the objective and goal of the Health Consultation System, outlining what it aims to achieve
- Scope: Defines the boundaries and extent of the system, specifying the functionalities and capabilities it will encompass
- Definitions, acronyms & abbreviations: Lists any abbreviations or acronyms used in the documentation, providing quick reference or definitions for readers
- References: Includes a list of external sources, standards, or documents that are referenced or consulted during the development of the Health Consultation System
- Overview of the proposed system: Offers a high-level summary of the proposed system, providing a brief description of its major components, functions, and benefits.

For the specific requirements, there are four sub-parts which are:

- User Characteristics: Describes the intended users of the Health Consultation System, including their roles, skills, and level of technical expertise, to ensure the system caters to their specific needs
- System Features: Outlines the desired features and functionalities of the system, specifying the tasks it should be able to perform and the services it should offer to users
- Software System Attributes: Outlines the desired features and functionalities of the system, specifying the tasks it should be able to perform and the services it should offer to users
- Performance and Other Requirements: states any specific performance targets, such as response time or system capacity, as well as any additional non-functional requirements like accessibility, compatibility, or regulatory compliance
- Design Constraints: Identifies any limitations or constraints that may impact the design and development of the Health Consultation System

The software development process for this project follows the waterfall model. The waterfall model is a linear and sequential approach to software development, where each phase is completed before moving on to the next. The project progresses through distinct phases, including requirements gathering, system design, implementation, testing, and deployment. This model ensures a systematic and structured approach to development, with clear documentation and deliverables at each stage. The waterfall model is particularly suitable for projects with well-defined requirements and a stable scope, which is the case for our project.

2. Specific Requirements

2.1 User characteristics

2.1.1 Patients

- Patients using the system should be familiar with basic computer functions and feel at ease using resources and web-based tools.
- Patients are actively involved in the healthcare journey and desire to interact with healthcare experts, look for medical counsel and take part in consultations.
- They may have varying degrees of medical knowledge and comprehension. Someone may need more thorough explanations and guidance while someone may have a basic understanding of healthcare concepts.
- They anticipate that the system would prioritize healthcare features over other features, ensuring a focused and committed user experience.
- They should trust the system may offer them access to physician recommendations, prescription services and laboratory test results.

2.1.2 Doctor

- The system's doctors have substantial medical training and competence in their respective fields.
- Doctors should have a solid understanding of technology and be familiar with the use of digital platforms and online communication tools.
- Doctors should accommodate the needs of patients with different medical backgrounds as patients accessing the system may have a variety of health issues from minor aches and pains to severe symptoms
- They are able to engage in meaningful conversations with patients to comprehend their issues and effectively medical facts to patients.

2.1.3 Specialized Medical Team(SMT)

- SMT should have the expertise and experience necessary to assess and validate a doctor's credentials and registration.
- SMT should be swift and effective in its operations, reviewing and approving doctor registration as soon as possible.
- SMT should act with a high degree of integrity and professionalism, assuring openness and impartiality in the verification process.

2.1.4 Pharmacist

- Pharmacists need to be extremely knowledgeable about pharmaceuticals, including dosage, drug interactions, and pharmaceutical formulations.
- They should make sure that prescription medications are supplied on time and safely while following all applicable laws and safety guidelines.
- They should be adept at effectively monitoring and distributing medications and ensure that patients receive the proper meds based on the prescriptions issued by the doctor after having the consultation.
- They guarantee appropriate labelling of pharmaceuticals, check patient profiles for allergies or contraindications, and confirm the accuracy of prescriptions.

2.2 System Features

The Health Consultation System is a web-based software product that allows patients to receive medical advice and consultation from authorised doctors and pharmacists remotely The system will be available via desktop and mobile devices, and a stable internet connection will be required. The software will provide asynchronous or synchronous consultations, with doctors delivering professional diagnoses, prescribing medication, and requesting laboratory testing as needed.

Modern web technologies such as HTML, CSS, JavaScript, and Python will be used to create the Health Consultation System. For data storage and processing, the system will rely on a cloud-based server infrastructure. The system will be compatible with popular web browsers such as Google Chrome, Firefox, and Safari.

The product's interface will be straightforward and user-friendly, enabling users to register for the application, arrange appointments, and get in touch with healthcare providers including chemists and doctors. The interface will be optimized for both desktop and mobile devices.

There won't be any specialised devices needed for the system to work. The only thing the user will need is a device that can connect to the internet. The system will use standard software protocols such as HTTP, TCP/IP, and SSL for secure data transmission hence avoiding data leakage from the

server. All communication between the system and users will be encrypted using SSL against unauthorized access.

The system will keep user data on a server infrastructure based in the cloud, which will offer scalable and reliable data storage.

The Health Consultation System operates 24/7, providing patients with the flexibility to make appointments at their convenience. However, doctors and pharmacists will only be available during their respective working hours to provide consultations and prescriptions. A user-friendly interface allows users to schedule appointments, choose between synchronous and asynchronous consultations, and access the system. For a seamless user experience, the system also has real-time cloud-based databases and communication interfaces.

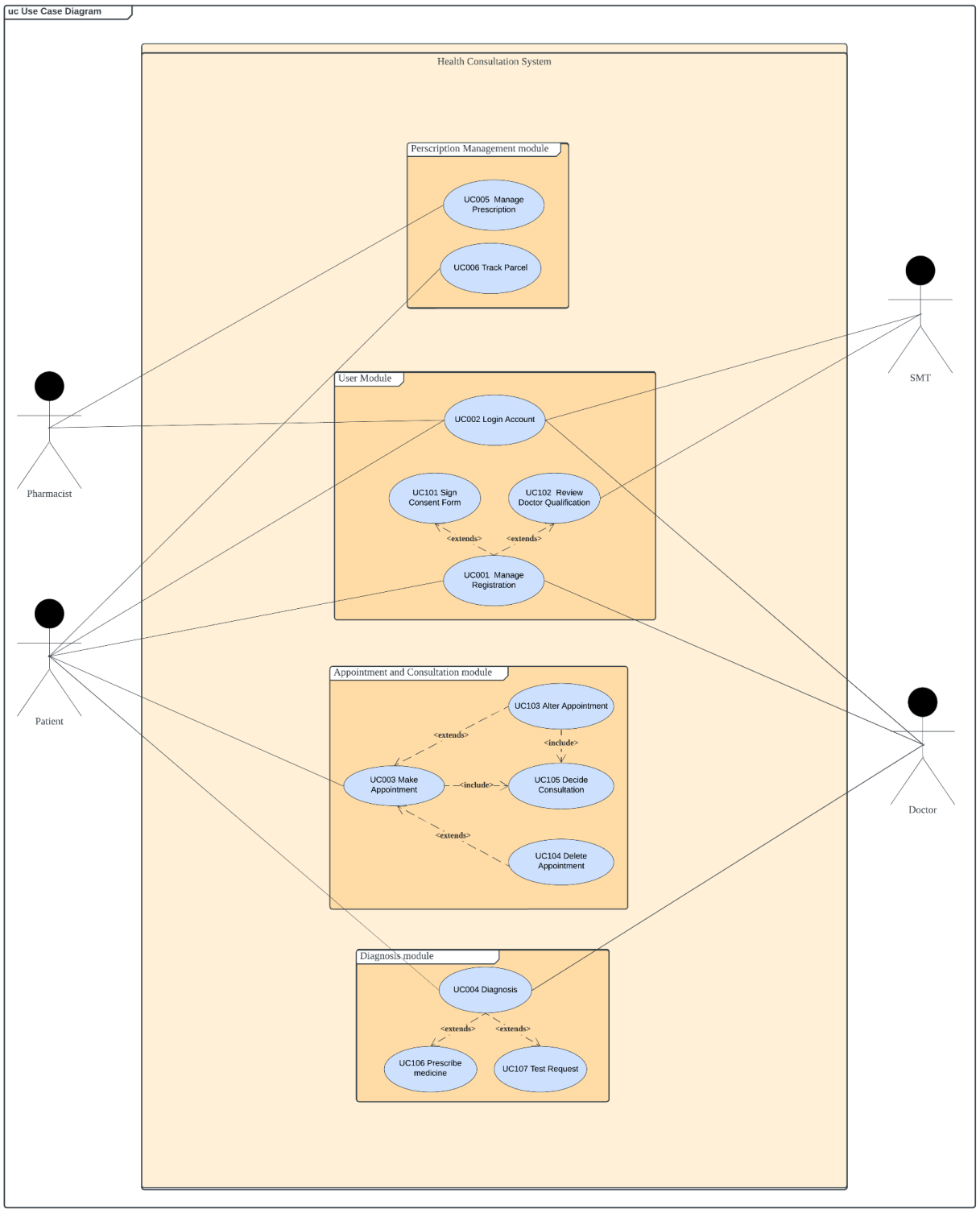


Figure 1.0: Use Case Diagram for <Health Consultation System>

Module	Function	Description
User Module	UC001 – Manage Registration	This use case describes the process of registering for an account
	UC002 - Login Account	This use case describes the process of login for the Patient, Doctor, Pharmacist and SMT.
	UC101 - Consent Form	This use case describes the patient should sign the consent form in order to register an account in the system
	UC102 - Review Doctor Qualification	This use case describes the process of reviewing doctor qualification registration.
Appointment and Consultation module	UC003 - Make Appointment	This use case describes the process of a patient making an appointment.
	UC103 - Alter Appointment	This use case describes the process of a patient altering a scheduled appointment
	UC104 - Delete Appointment	This use case describes the process of a patient deleting an appointment
	UC105 - Decide Consultation	The use case describes the process of a patient deciding his consultation type for his appointment. The use case is commonly used by other use cases.
Diagnosis module	UC004 - Diagnosis	This use case describes the process of the diagnosis for the doctor and patient
	UC106 - Prescribe medicine	This use case describes the process of a doctor prescribing medicine
	UC107 - Test Request	This use case describes the process of requesting lab test results from the doctor and submitting lab test results for the patient
Prescription Management module	UC005 - Manage Prescription	This use case includes the process of a pharmacist verifying and delivering medication following the prescription given by the doctor
	UC006 - Track Parcel	This use case is used by the patient to track the status of his/her parcel (medication)

Table 1.0: Description of Module and Functions for <Health Consultation System>

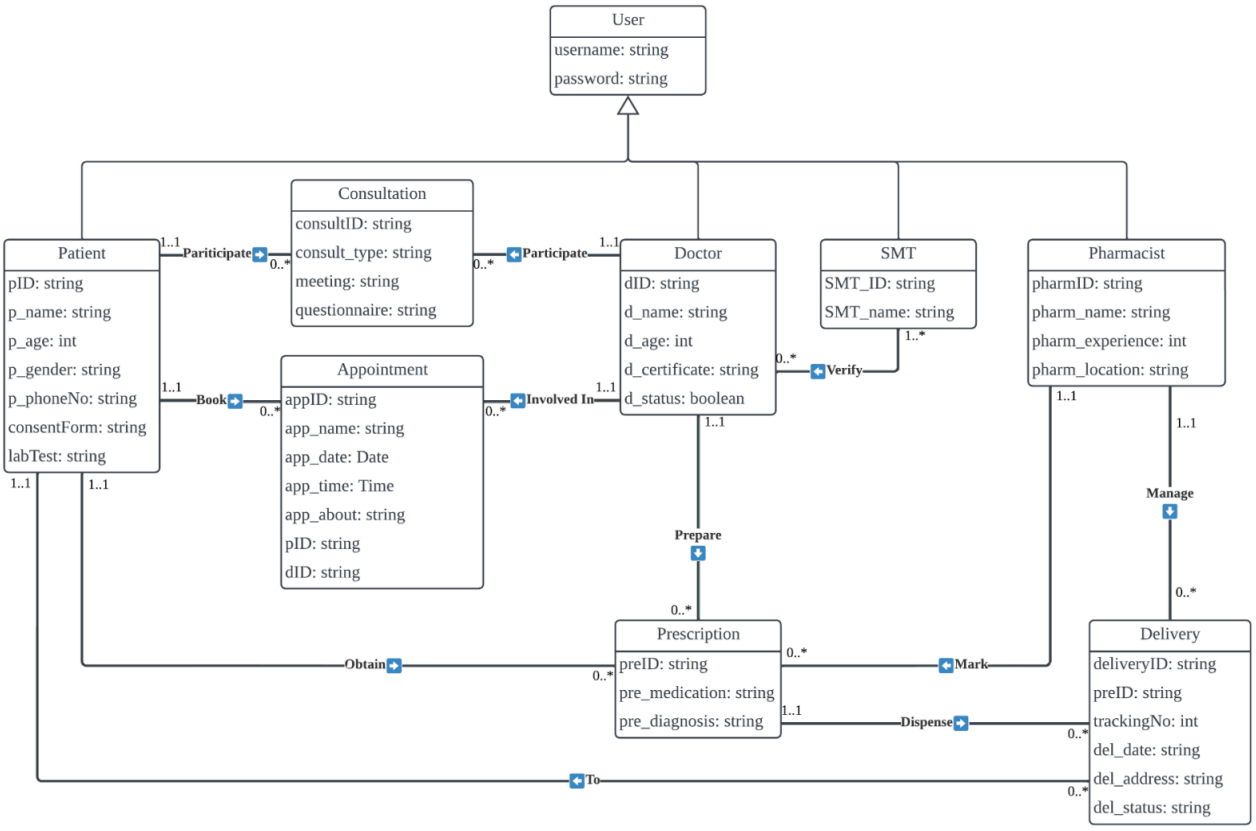


Figure 2.0: Class Diagram for <Health Consultation System>

2.2.1 UC001: Use Case <Manage Registration>

Use Case ID:	UC001
Use Case Name	Manage Registration
Actor	1. Patient 2. Doctor
Extension point	<Sign Consent Form>, <Review Doctor Qualification>
Brief Description	This use case describes the process of registering for an account
Pre-condition	● Have active internet connection to the system ● Haven't owned an account in the system before ● Have intentions to be an active user in the system
Normal Flow	1. The use case starts when the user taps "Create an account" on the login page 2. The system displays a registration page containing registration form 3. The user enters personal information 4. If the actor is "Doctor" 4.1. The user is required to upload medical registration certificate 4.2. <Review Doctor Qualification> 5. If the actor is "Patient" 5.1. <Sign Consent Form> 6. The system checks data accuracy and adds the user account to the user database. 7. System redirects the user to the login page. 8. The use case ends
Postcondition	● The user has created an account record in the system
Exception Flow	1. The user does give not accurate information into system 1.1. The system displays an error message to the user to request to resubmit again 2. The doctor qualification has been
Alternative Flow	1. At any point, the User may leave the system.

Table 2.0 Use case Description for UC001: Use Case <Manage Registration>

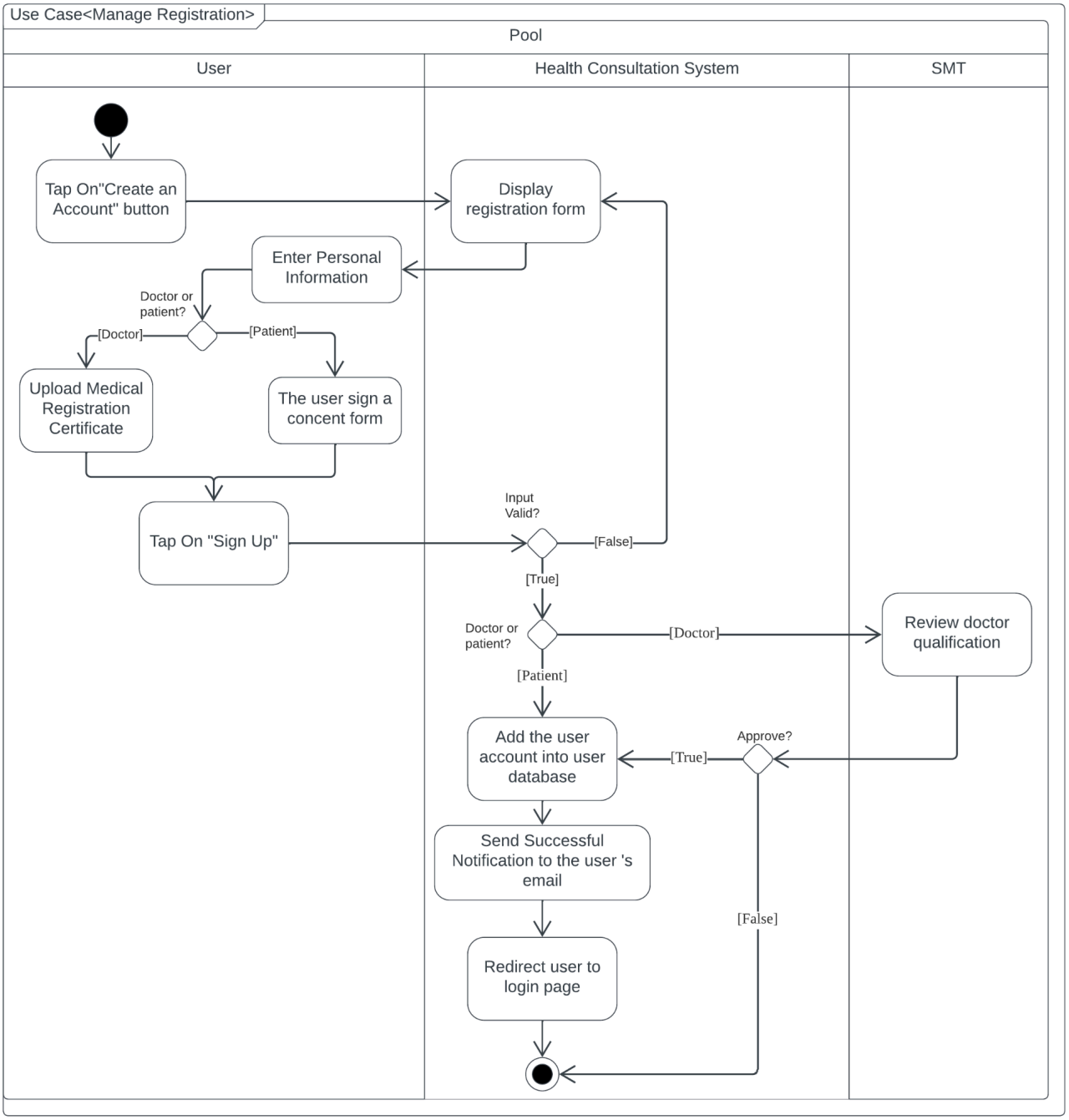


Figure 4.0 Activity Diagram for UC001: Use Case<Manage Registration>

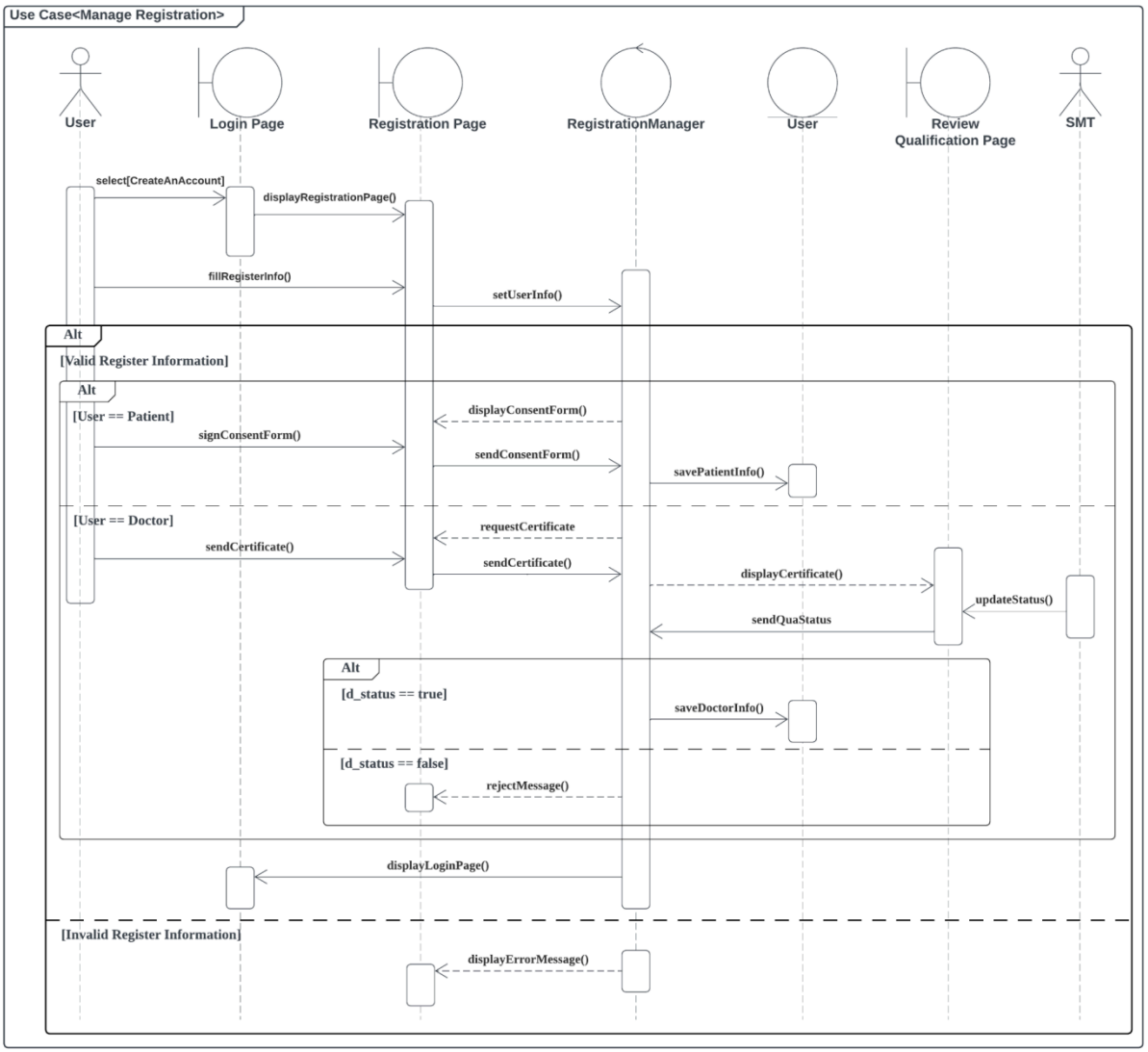


Figure 3.0: Sequence Diagram for UC001: Use Case <Manage Registration>

2.2.2 UC101: Extension Use Case <Sign Consent Form>

Use Case ID:	UC101
Extends	UC001 Manage Registration at <Sign Consent Form>
Use Case Name	Sign Consent Form
Actor	Patient
Brief Description	This use case describes the patient should sign the consent form in order register an account in the system
Pre-condition	● Patient has entered their personal information ● Haven't signed the consent form in the system before
Normal Flow	1. The system displays the consent form to the patient interface 2. The patient signs the consent form 3. The patient submits back consent form into the system
Postcondition	● The consent form has been submitted into the system database

Table 3.0 Extension Use case Description for UC101: Extension Use Case <Sign Consent Form>

2.2.3 UC102: Extension Use Case <Review Doctor Qualification>

Use Case ID:	UC102
Extends	UC001 Manage Registration at <Review Doctor Qualification>
Use Case Name	Review Doctor Qualification
Actor	1. Specialized Medical Team (SMT) 2. Doctor
Brief Description	This use case describes the process of reviewing doctor qualification registration.
Pre-condition	● Have active internet connection to the system ● Doctor has submitted their doctor qualification registration
Normal Flow	1. The use case starts when the doctor's certificate and information are displayed at the SMT interface. 2. SMT reviews the submitted information. 3. If the status is "true" then 3.1 The doctor's information is updated in the system. 4. else 4.1 Notify the doctor regarding the rejection. 5. Use case ends.
Postcondition	● The doctor's qualification status is updated in the system.

Alternative Flow	1. At any point, the User may leave the review page.

Table 4.0 Use case description for UC102: Extension Use Case <Review Doctor Qualification>

2.2.4 UC002: Use Case <Login Account>

Use Case ID:	UC002
Use Case Name	Login Account
Actor	1. Patient 2. Doctor 3. Pharmacist 4. SMT
Brief Description	This use case describes the process of login for the Patient, Doctor, Pharmacist and SMT.
Pre-condition	● Have active internet connection to the system ● User has registered their own account (For patient) ● User's doctor qualification has been approved (For doctor) ● User has the account provided by the company (For Pharmacist and SMT)
Normal Flow	1. The use case starts when the user enters their login information on the login page. 2. The user taps the "submit" button after filling in the information. 3. The system redirects the user to main page 4. Use case ends.
Postcondition	● Users successfully log in to the system
Exception flow	1. The user failed to submit the correct login information 1.1. The system displays an error message and asks the user to resubmit the login information.

Table 5.0 Use case Description for UC002: Use Case <Login Account>

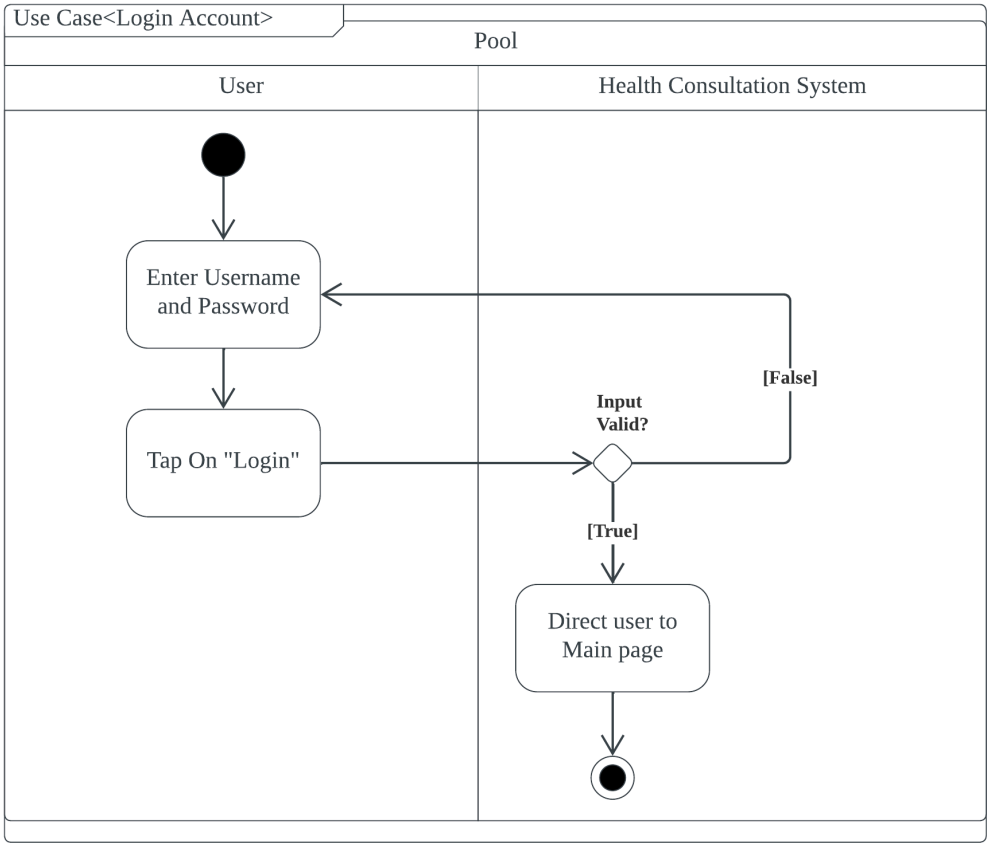


Figure 5.0 Activity diagram for UC002: Use Case <Login Account>

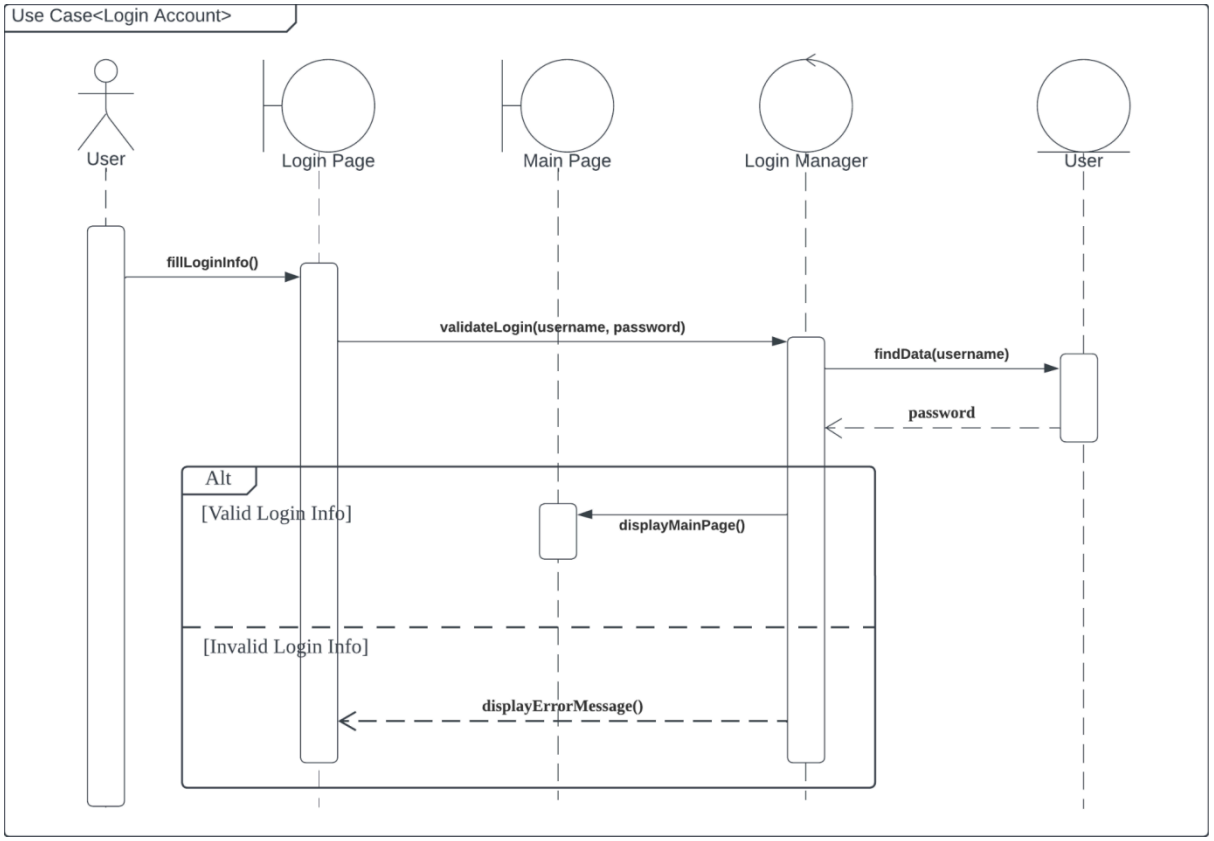


Figure 6.0 Sequence diagram for UC002: Use Case <Login Account>

2.2.5 UC003: Use Case <Make Appointment>

Use Case ID:	UC003
Use Case Name	Make Appointment
Extension Point	<Alter Appointment> <Delete Appointment>
Actor	1. Patient
Brief Description	This use case describes the process of a patient making an appointment
Pre-condition	• The patient has logged into the system. • The patient intends to make a new appointment
Normal Flow	1. The use case starts when the user clicks “Make New Appointment” option in the AppointmentList Page 2. The system displays an appointment Page for the user to fill in 3. The user submits the appointment page to the system 4. The system validates appointment form 5. If the appointment form is invalid 5.1. Exception 1 is followed 6. If the appointment form is valid 6.1. The system saves the appointment data into the database 6.2. The system displays a successful message 6.3. Include <Decide Consultation> 6.4. The system saves the consultation data into the database 6.5. Display a successful message 7. The use case ends
Alternative Flow 1	1. On AppointmentList Page, the user may alter the scheduled appointment 1.1. <Alter Appointment>
Alternative Flow 2	1. On AppointmentList Page, the user may delete the scheduled appointment 1.1. <Delete Appointment>
Exception Flow	1. The system displays an error message
Postcondition	• The appointment is successfully made. • The consultation is successfully made for the corresponding appointment

Table 6.0 Use case description for UC003: Use Case <Make Appointment>

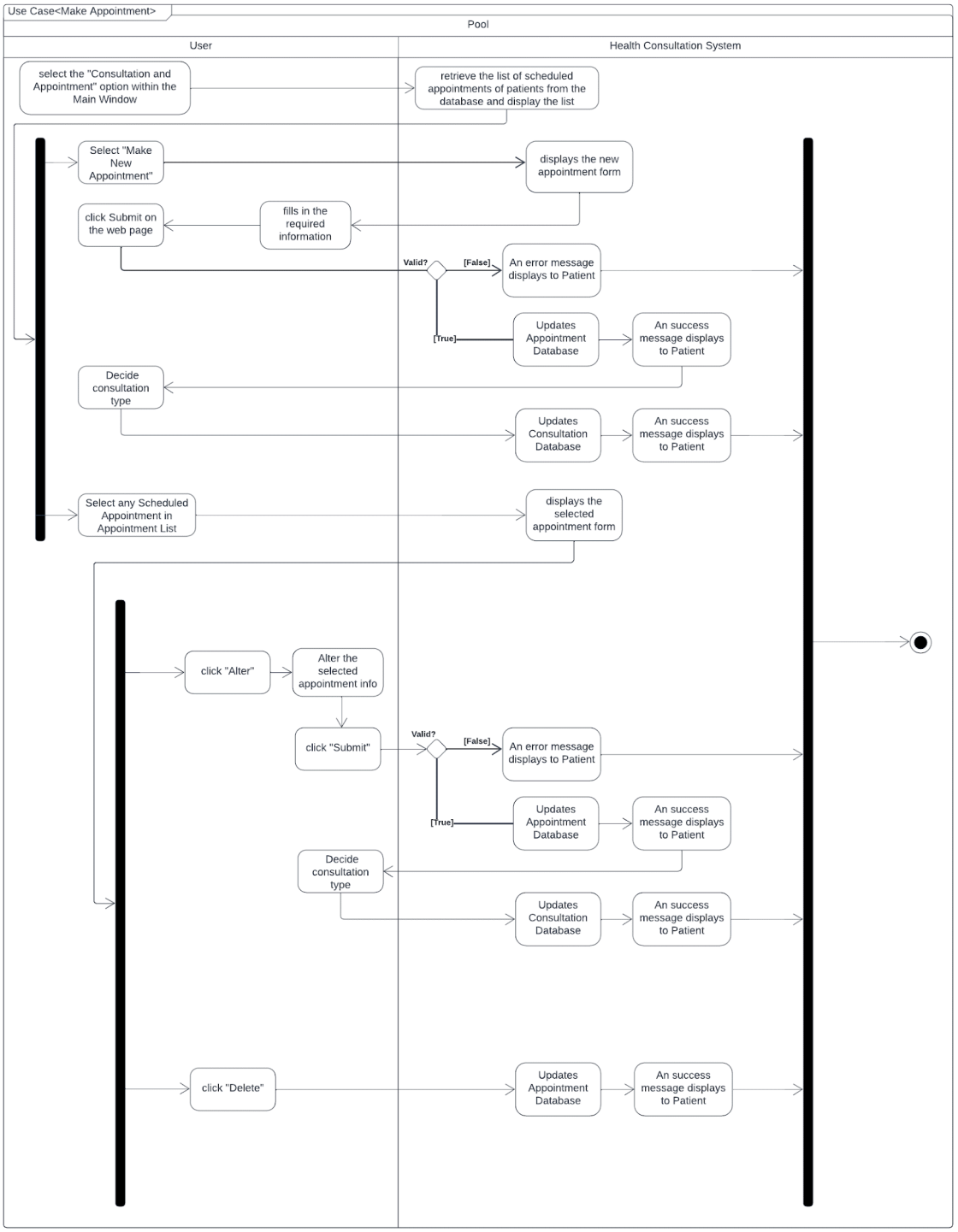
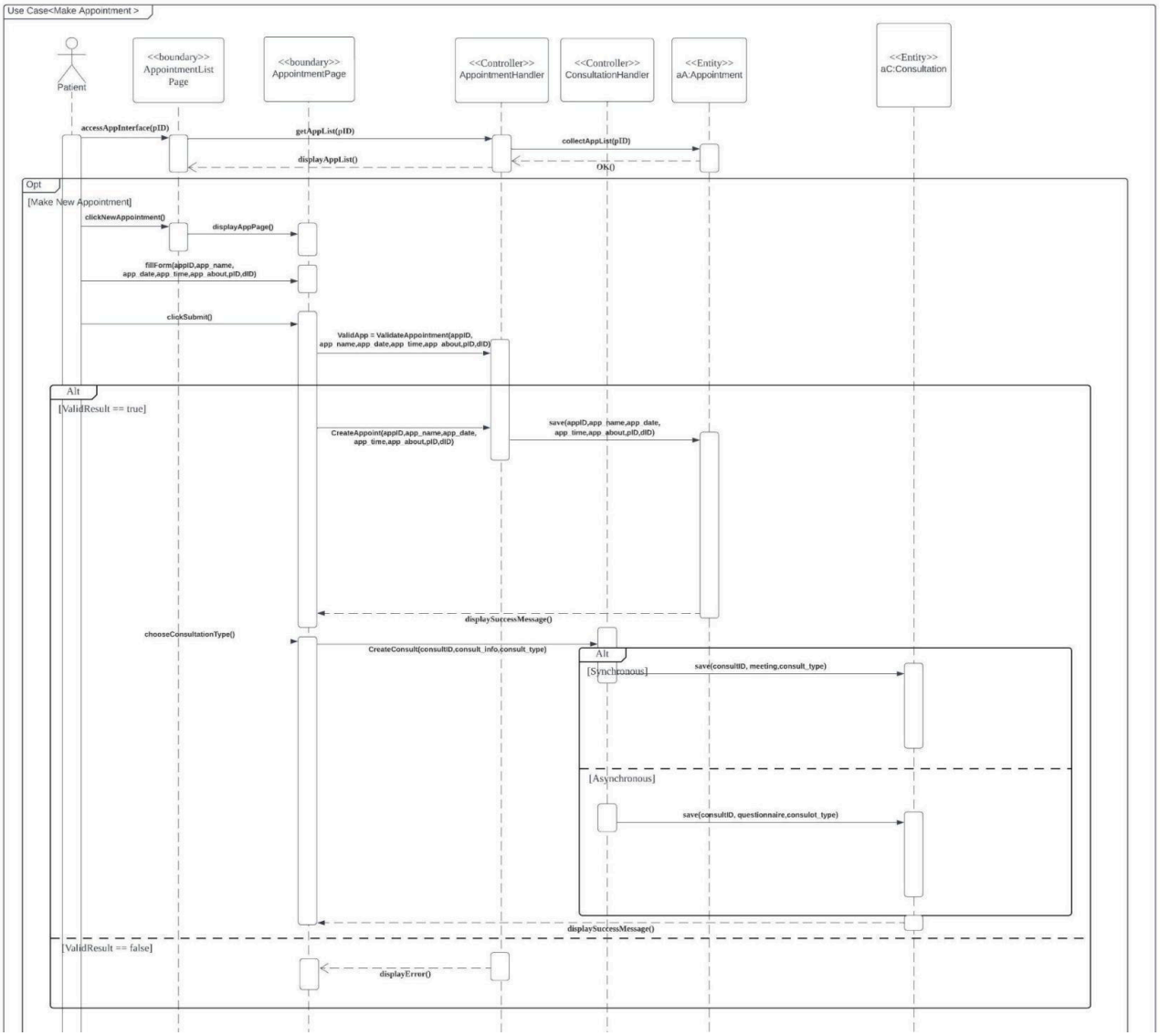


Figure 7.0 Activity diagram for UC003: Use Case <Make Appointment>


Use Case<Make Appointment >
Patient
<<boundary>> AppointmentList Page
<<boundary>> AppointmentPage
<<Controller>> AppointmentHandler
<<Controller>> ConsultationHandler
<<Entity>> aA:Appointment
<<Entity>> aC:Consultation
accessAppInterface(pID)
getAppList(pID)
collectAppList(pID)
displayAppList()
OK()
Opt
[Make New Appointment]
clickNewAppointment()
displayAppPage()
fillForm(appID,app_name, app_date,app_time,app_about,pID,dID)
clickSubmit()
ValidApp = ValidateAppointment(appID, app_name,app_date,app_time,app_about,pID,dID)
Alt
[ValidResult == true]
CreateAppoint(appID,app_name,app_date, app_time,app_about,pID,dID)
save(appID,app_name,app_date, app_time,app_about,pID,dID)
displaySuccessMessage()
chooseConsultationType()
CreateConsult(consultID,consult_info,consult_type)
Alt
[Synchronous]
save(consultID, meeting,consult_type)
[Asynchronous]
save(consultID, questionnaire,consulot_type)
displaySuccessMessage()
[ValidResult == false]
displayError()

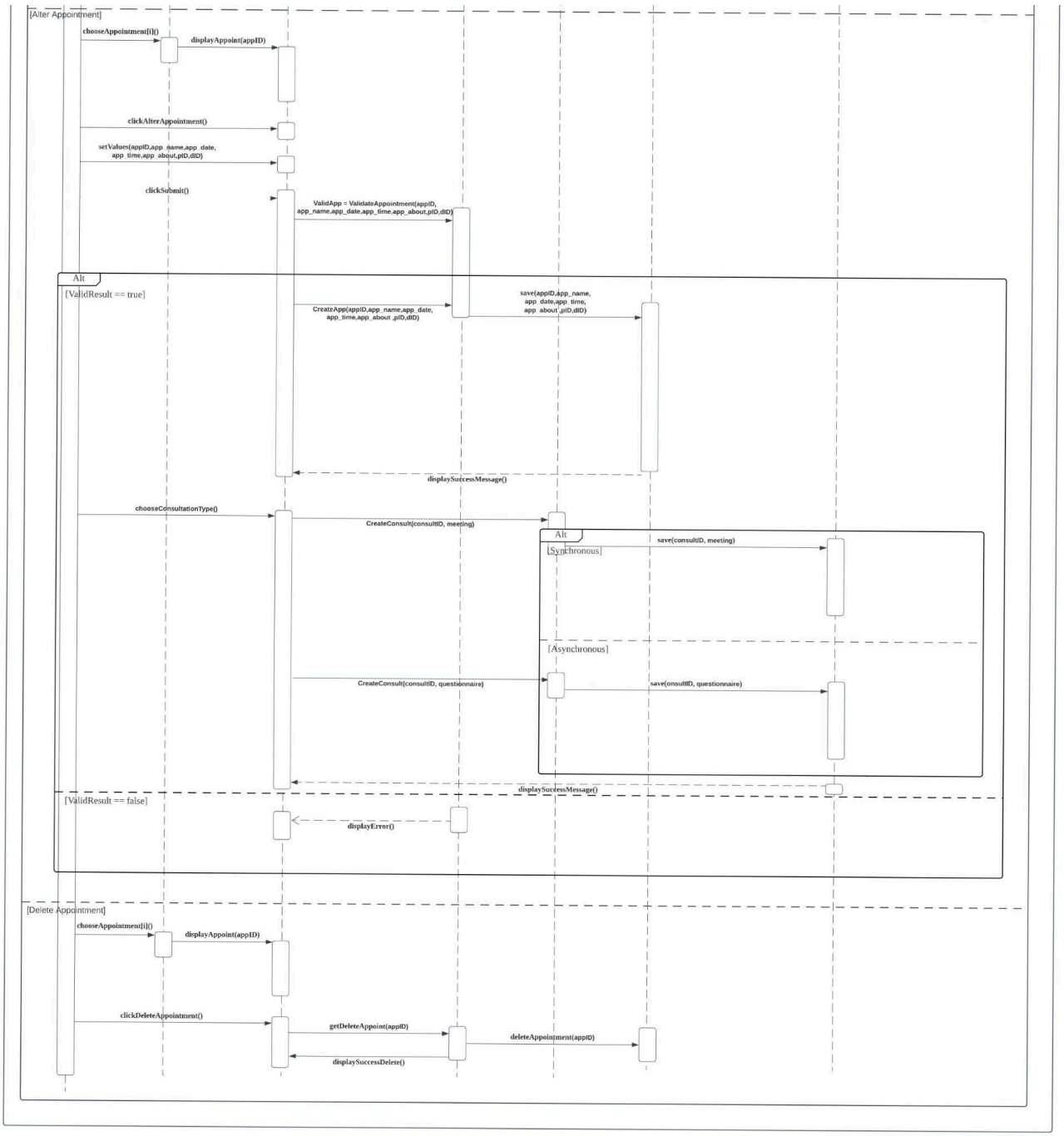


Figure 8.0 Sequence diagram for UC003: Use Case <Make Appointment>

2.2.6 UC103: Extension Use Case <Alter Appointment>

Use Case ID:	UC103
Extends	UC003 Make Appointment at <Alter Appointment>
Actor	1. Patient
Brief Description	This use case describes the process of a patient altering a scheduled appointment
Pre-condition	● The patient has made an appointment ● The patient intends to alter the scheduled appointment
Normal Flow	1. The use case starts when the user selected one of the scheduled appointments in the AppointmentList Page 2. The user taps “Alter appointment” 3. The user alters the appointment details 4. The user submits the altered appointment form 5. The system validates the altered appointment form 6. If the appointment form is invalid 6.1. Exception 1 is followed 7. If the appointment form is valid 7.1. The system saves the altered appointment data into the database 7.2. The system displays a successful message 7.3. Include <Decide Consultation> 7.4. The system saves the consultation data into the database 7.5. Display successful message 8. The use case ends
Exception Flow	1. The system displays an error message
Postcondition	● The appointment form is successfully altered by the patient. ● The appointment details are updated in the appointment database

Table 7.0 Use case description for UC103: Extension Use Case <Alter Appointment>

2.2.7 UC104: Extension Use Case <Delete Appointment>

Use Case ID:	UC104
Extends	UC003 Make Appointment at <Alter Appointment>
Actor	1. Patient
Brief Description	This use case describes the process of a patient deleting an appointment
Pre-condition	● The patient has made an appointment ● The patient intends to delete scheduled appointments
Normal Flow	1. The use case starts when the user selected a scheduled appointment in the AppointmentList Page 2. The user taps the “Delete” button 3. The system deletes the corresponding appointment from the database 4. The system displays a success message 5. The use case end
Postcondition	● The appointment form is successfully deleted by the patient. ● The appointment details are updated in the appointment database

Table 8.0 Use case description for UC104: Extension Use Case <Delete Appointment>

2.2.8 UC105: Use Case <Decide Consultation>

Use Case ID:	UC105
Actor	1. Patient
Brief Description	This use case describes the process of a patient deciding on a consultation
Pre-condition	● The patient is ready to decide on a consultation type for his new appointment
Normal Flow	1. The use case starts after the user makes a new appointment successfully 2. The user chooses the consultation type by selecting provided options 3. If the user chooses “Synchronous Consultation” 3.1 The system processes the consultation info into meeting info 4. Else 4.1 The system processes the consultation info into questionnaire info 5. The use case end
Postcondition	● The consultation type is decided ● The consultation info is processed for the selected consultation type

Table 9.0 Use case description for UC105:Use Case <Decide Consultation>

2.2.9 UC004: Use Case <Diagnosis>

Use Case ID:	UC004
Use Case Name	Diagnosis
Extension point	<Prescribe Medicine> <Test Request>
Actor	1. Doctor 2. Patient
Brief Description	This use case describes the process of the diagnosis for the doctor and patient
Pre-condition	● The patient has passed the asynchronous or synchronous consultation
Normal Flow	1. The use case starts when the doctor selects the "Medical Assessment page" on the main page. 2. The doctor entered the patient's id. 3. If the patient's id is valid then 3.1 The system retrieves and displays the patient's personal information, including name, gender, age, phone number and lab test records 4. Else 4.1 Exception 1 is followed 5. <Prescribe medicine> 6. If the doctor requires laboratory test 6.1 <Test Request> 7. The use case ends.
Exception flow	1. The system displays an error message.
Postcondition	● The prescription is updated in the database. ● Lab test result is updated to the database
Alternative Flow 1	1. At any point, the User may leave the system.

Table 10.0 Use case description for UC004: Use Case <Diagnosis>

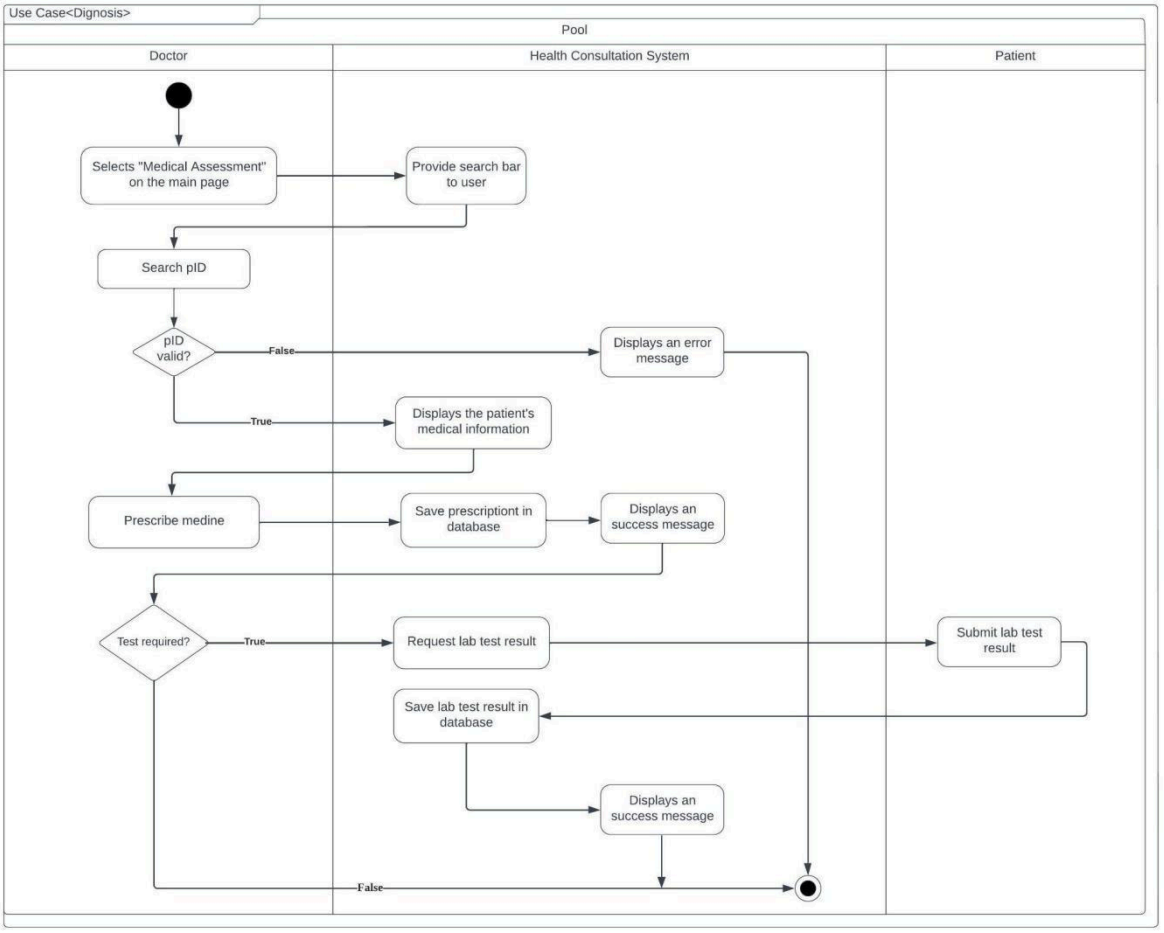


Figure 9.0: Activity Diagram for UC004: Use Case <Diagnosis>

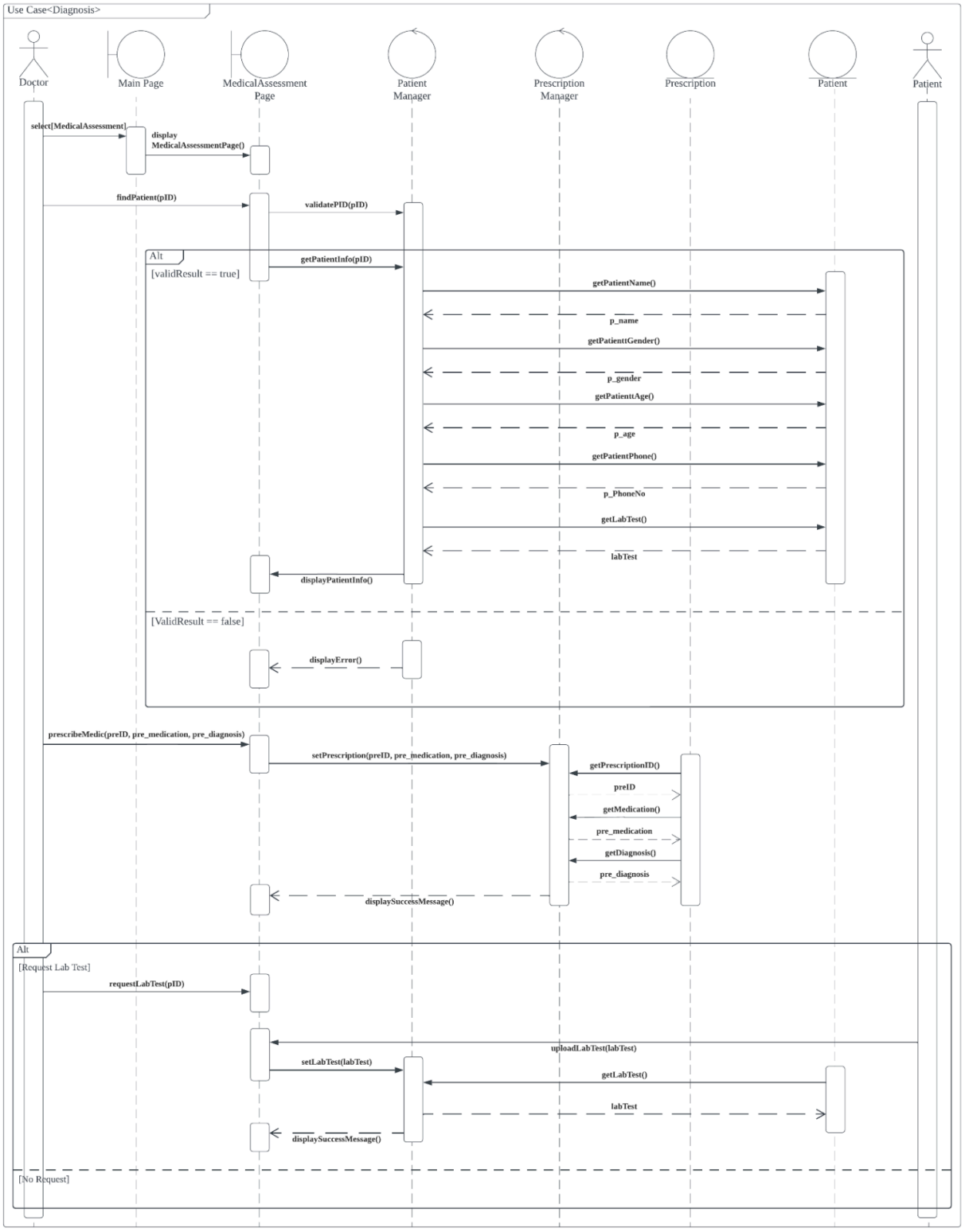


Figure 10.0: Sequence Diagram for UC004: Use Case <Diagnosis>

2.2.10 UC106: Extension Use Case <Prescribe Medicine>

Use Case ID:	UC106
Use Case Name	Prescribe Medicine
Extends	UC004 Diagnosis at <Prescribe Medicine>
Actor	1. Doctor
Brief Description	This use case describes the process of a doctor prescribing medicine
Pre-condition	• The patient has passed the asynchronous or synchronous consultation
Normal Flow	1. The use case starts after the doctor has checked the patient's information. 2. The doctor prescribes medications based on the patient's information. 3. Prescription then updated to the Prescription database 4. An updated successfully message is displayed on page 5. The use case ends.
Exception flow	
Postcondition	• The laboratory result is uploaded to the database

Table 11.0 Use case description for UC105: Extension Use Case <Prescribe Medicine>

2.2.11 UC107: Extension Use Case <Test Request>

Use Case ID:	UC107
Use Case Name	Test Request
Extends	UC004 Diagnosis at <Test Request>
Actor	1. Doctor 2. Patient
Brief Description	This use case describes the process of requesting lab test results from the doctor and submitting lab test results for the patient
Pre-condition	● The patient has passed the asynchronous or synchronous consultation ● The doctor has prescribed medicine for the patient
Normal Flow	1. The use case starts after the doctor prescribes medicine for the patient. 2. Doctor requests lab test results on the medical assessment page. 3. Patient sends the result via the same page. 4. Result then is updated in the Patient database. 5. An updated successful message is displayed on the page. 6. The use case ends.
Exception flow	
Postcondition	● Lab test result is updated to the database
Alternative Flow 1	1. At any point, the User may leave the system.

Table 12.0 Use case description for UC106: Extension Use Case <Test Request>

2.2.12 UC005: Use Case <Manage Prescription>

Use Case ID:	UC005
Use Case Name	Manage Prescription
Actor	1. Pharmacist
Brief Description	This use case describes the process of verification and delivery management of the medication following the prescription given by the doctor.
Pre-condition	● The pharmacist has a login to the system. ● The pharmacist can retrieve and review the prescription that has to be verified.
Normal Flow	1. The use case starts when the pharmacist logs in to the system. 2. The system will display the “Main Page”. 3. The pharmacist clicks the “Prescription” button to view the prescriptions list. 4. The pharmacist selects one of the prescriptions to view details 5. The pharmacist arranges the delivery. 6. If delivery is successful 6.1 The pharmacist will update the delivery status to “Complete” 7. Else 7.1 Exception 1 is followed 8. The use case ends.
Exception flow	1. The delivery is unsuccessful 1.1. The system will send the unsuccessful message to the pharmacist interface.
Postcondition	● The medication has been delivered to the patient. ● The complete delivery status of the prescription has been updated by the pharmacist.

Table 13.0 Use case description for UC005: Use Case <Manage Prescription>

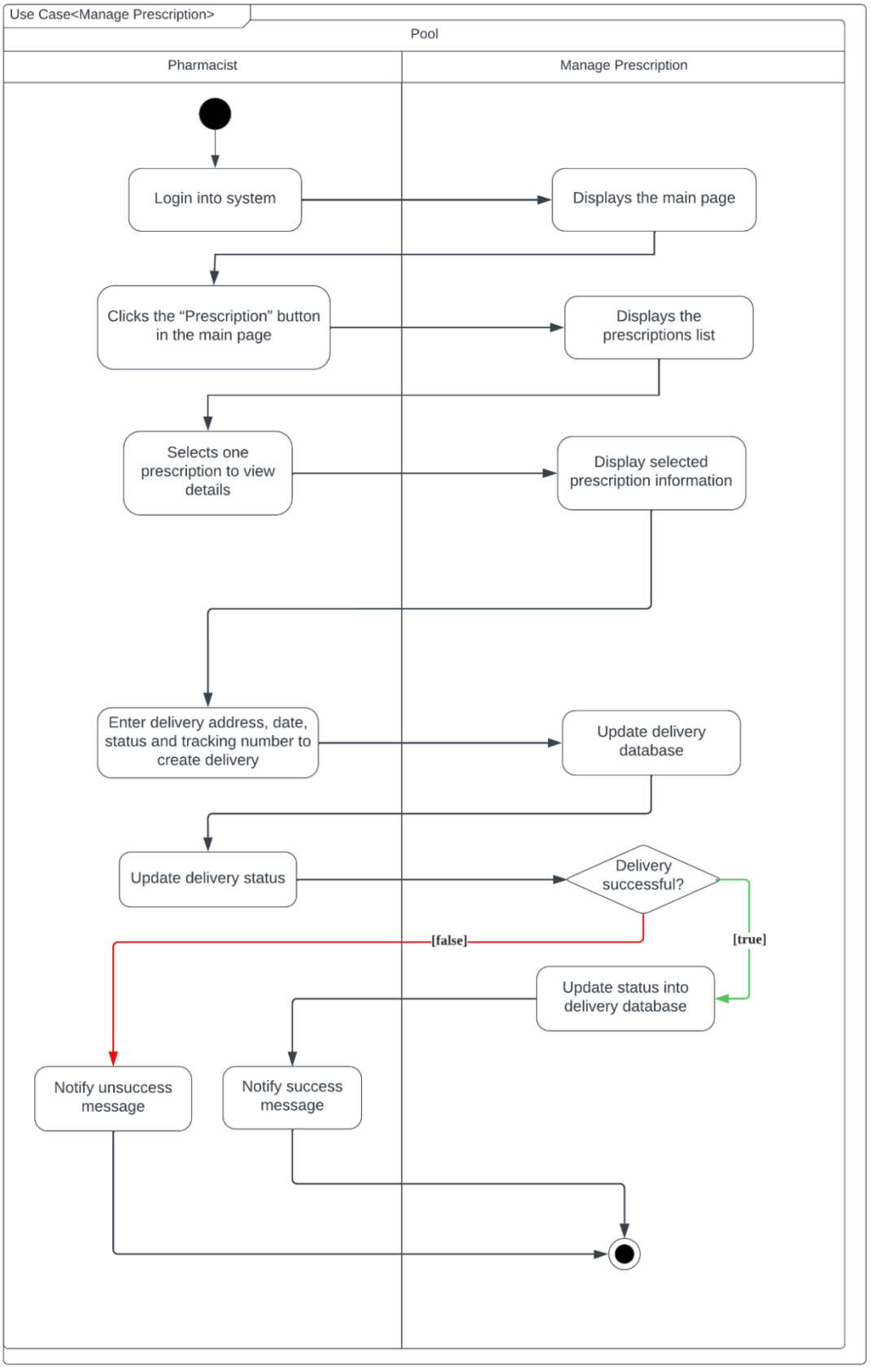


Figure 11.0: Activity Diagram for UC005: Use Case <Manage Prescription>

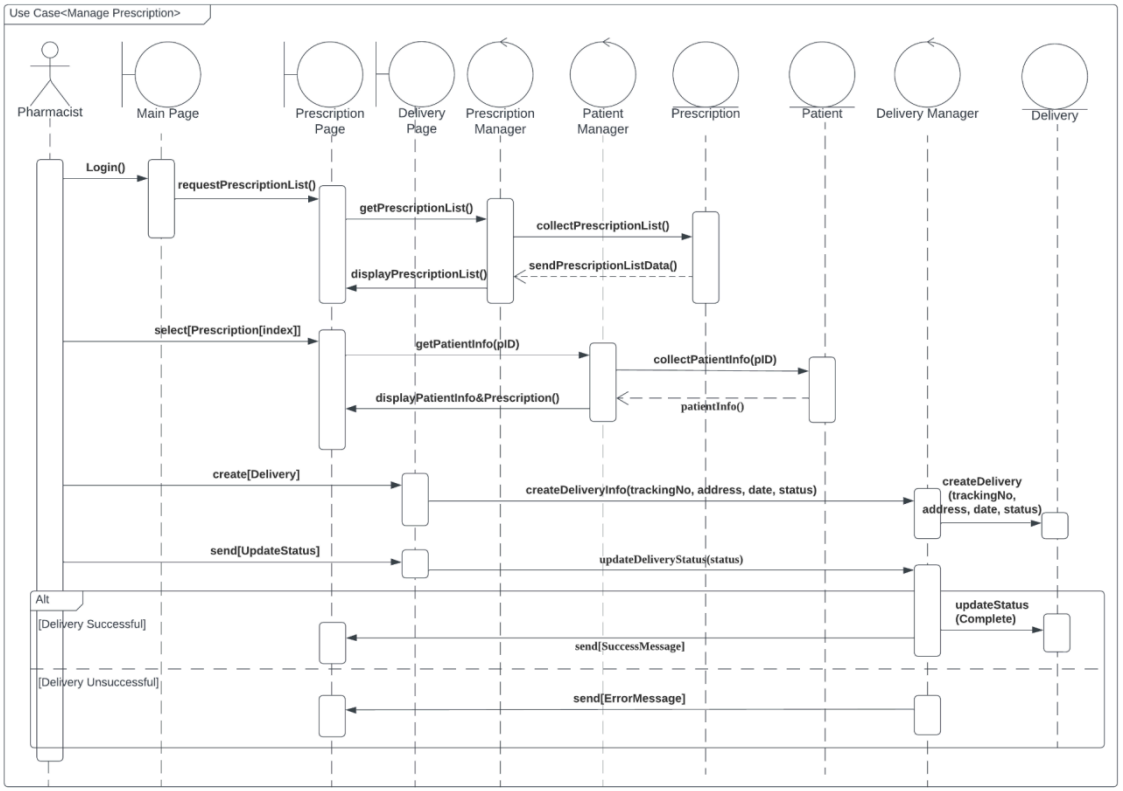


Figure 12.0: Sequence Diagram for UC005: Use Case <Manage Prescription>

1.1.1 UC006 Use Case <Track Parcel>

Use Case ID:	UC006
Use Case Name	Track Parcel
Actor	1. Patient
Brief Description	This use case is called when the user wants to track the status of the parcel.
Pre-condition	• The medication has been shipped by the pharmacist. • Pharmacist had updated the tracking number in the system.
Normal Flow	1. The use case starts when the user clicks the “Track Parcel” button. 2. The system will redirect the user to a tracking parcel website. 3. The use case ends
Exception flow	
Alternative Flow	At any point, the user may leave the tracking screen.
Postcondition	

Table 14.0 Use case description for UC006: Use Case <Track Parcel>

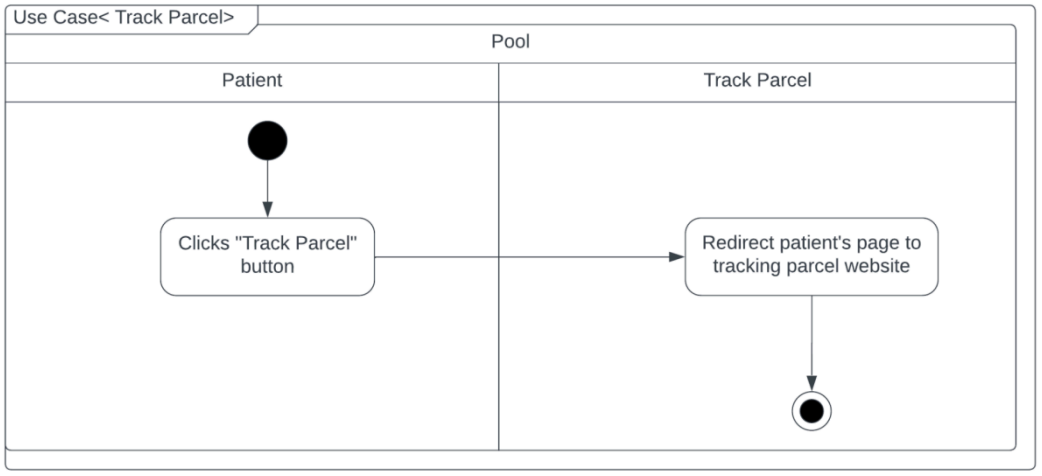


Figure 13.0: Activity Diagram for UC006: Use Case <Track Parcel>

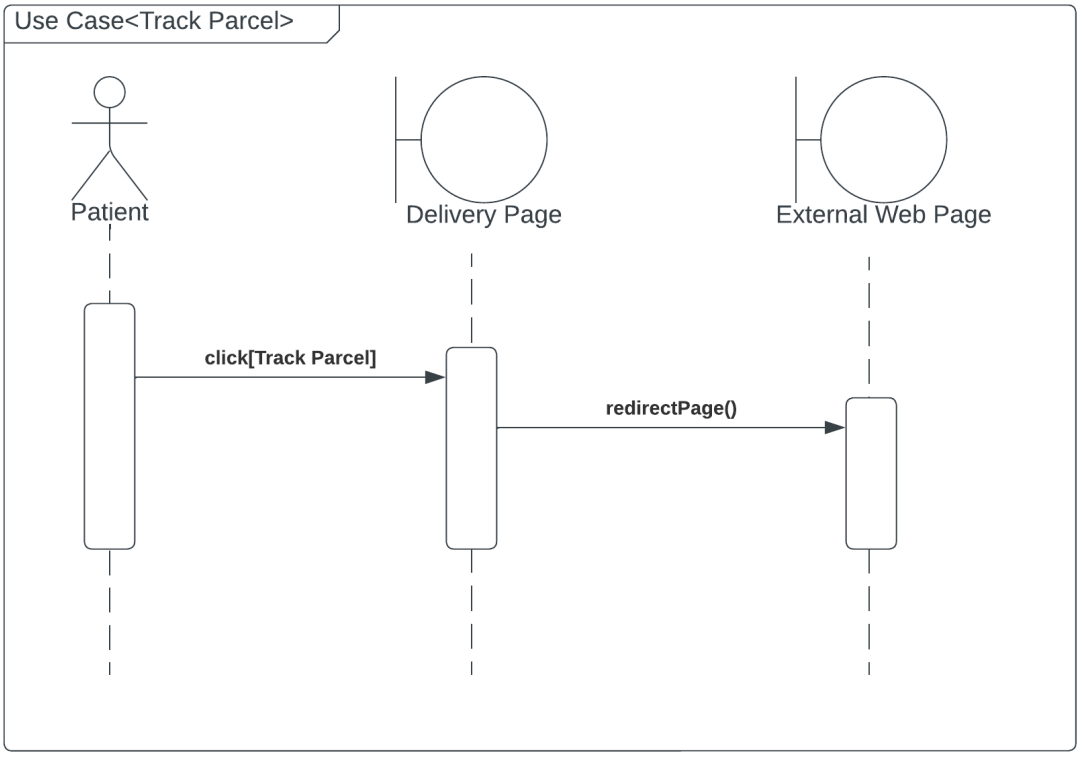


Figure 14.0: Sequence Diagram for UC006: Use Case <Track Parcel>

2.3 Software System Attributes, Performance and Other Requirements

Software System Attributes:

Usability: The system must have an intuitive and user-friendly interface for patients, doctors, and pharmacists to easily navigate and use.

Reliability: The system must be able to perform its functions correctly and consistently, and ensure patient and medical data is secure and confidential.

Maintainability: The system should be easy to modify, repair, and enhance to accommodate changes in medical procedures, regulations, or technologies.

Portability: The system should be able to run on different hardware and software platforms to allow flexibility for healthcare providers and patients.

Compatibility: The system must work seamlessly with other healthcare systems and components to ensure effective patient care.

Performance:

Response Time: The system must respond promptly to user requests for appointments, consultations, or prescription refills within 3 seconds.

Throughput: The system should be able to handle a high volume of consultations and prescriptions per day (at least 500) to accommodate the needs of patients and healthcare providers.

Capacity: The system should be able to handle a large number of patients and medical records (At least 10000), and support multiple consultations and prescriptions simultaneously.

Availability: The system should be operational and accessible during weekdays from 9 am to 5 pm, and have at least 99% uptime during those hours.

Other Requirements:

Security: The system must protect patient data and medical records from unauthorized access or malicious attacks, and ensure secure communication channels between patients, doctors, and pharmacists.

Safety: The system must operate safely to avoid any harm to patients or healthcare providers during consultations or medication delivery.

Legal and Regulatory: The system must comply with healthcare laws, regulations, and standards to ensure patient safety and privacy.

Environmental: The system should minimize its impact on the environment through efficient use of resources and waste reduction measures.

2.4 Design Constraints

Environmental constraints:

The system must be designed to operate in temperatures ranging from 10°C to 30°C.

The system must be resistant to dust, water, and other environmental hazards commonly found in healthcare facilities.

Hardware constraints:

The system must be designed to run on a minimum of 2 CPU cores and 16GB of RAM.

The system must be able to store a minimum of 10TB of data on a secure and reliable storage solution.

The system must be designed to use a solid-state drive (SSD) for data storage to ensure fast read and write speeds.

The system must be designed to use a high-speed internet connection to support real-time video consultations.

Security constraints:

The system must include role-based access control to ensure that sensitive information is only accessible to authorized users.

The system must be designed to support multi-factor authentication and strong password policies to prevent unauthorized access.

Compatibility constraints:

The system must be designed to be compatible with popular web browsers such as Chrome, Firefox, and Safari, to ensure accessibility for a wide range of users.

The system must be designed to be compatible with mobile devices, such as smartphones and tablets, to allow for remote consultations.

The system must be compatible with common operating systems such as Windows 10 and above, MacOS, and Linux.

Performance constraints:

The system must be designed to support a minimum of 50 concurrent users, with a response time of less than 3 seconds, to ensure efficient use by healthcare providers.

The system must be designed to handle a minimum of 500 consultations and prescription requests per day, to ensure that patient's needs are met in a timely manner.

Maintenance constraints:

The system must be designed to facilitate easy updates and bug fixes, with minimal disruption to normal system operation.

The system must be designed to allow for easy replacement of hardware components, such as hard drives or network cards, without requiring extensive reconfiguration.

The system must be designed to generate useful diagnostic logs and reports to aid in troubleshooting and maintenance activities.

The system must be designed to provide adequate documentation and training materials to support maintenance personnel in their tasks.

3. System Architectural Design

3.1 Architecture Pattern and Rationale

The described architecture pattern follows the Model-View-Controller (MVC) pattern, where the view components (RegistrationPage, LoginPage, ReviewPage, etc.) handle the user interface, the controller components (RegisterHandler, LoginHandler, UserHandler, etc.) mediate user interactions and business logic, and the model components (User, Patient, Doctor, etc.) manage the data and underlying operations. This pattern promotes separation of concerns and facilitates modular and maintainable applications, allowing changes in one component without affecting the others. MVC provides a structured approach to organizing and coordinating different parts of the application for effective communication and collaboration.

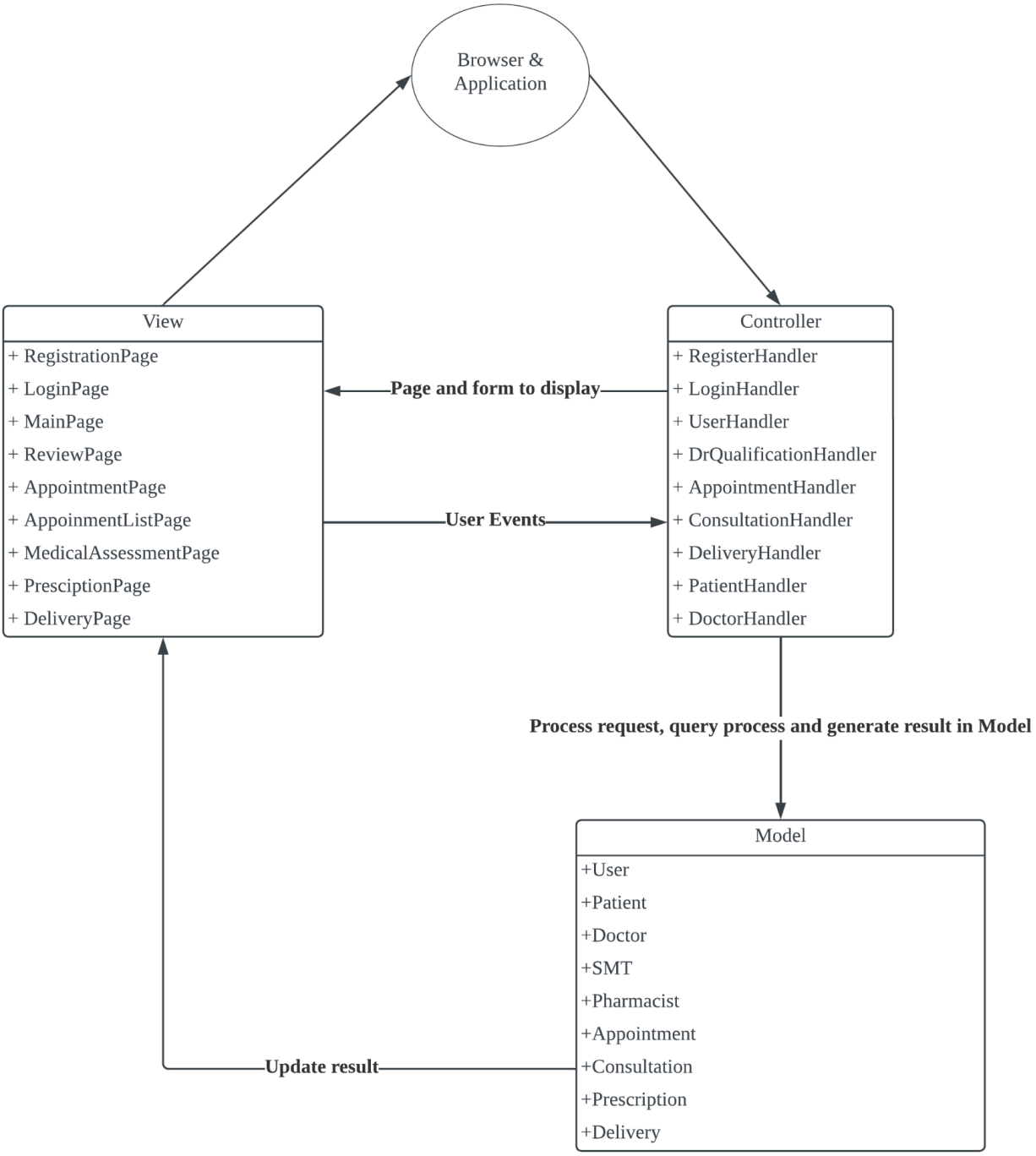


Figure 3.1: Architecture Diagram for <Health Consultation System>

3.2 Component Model

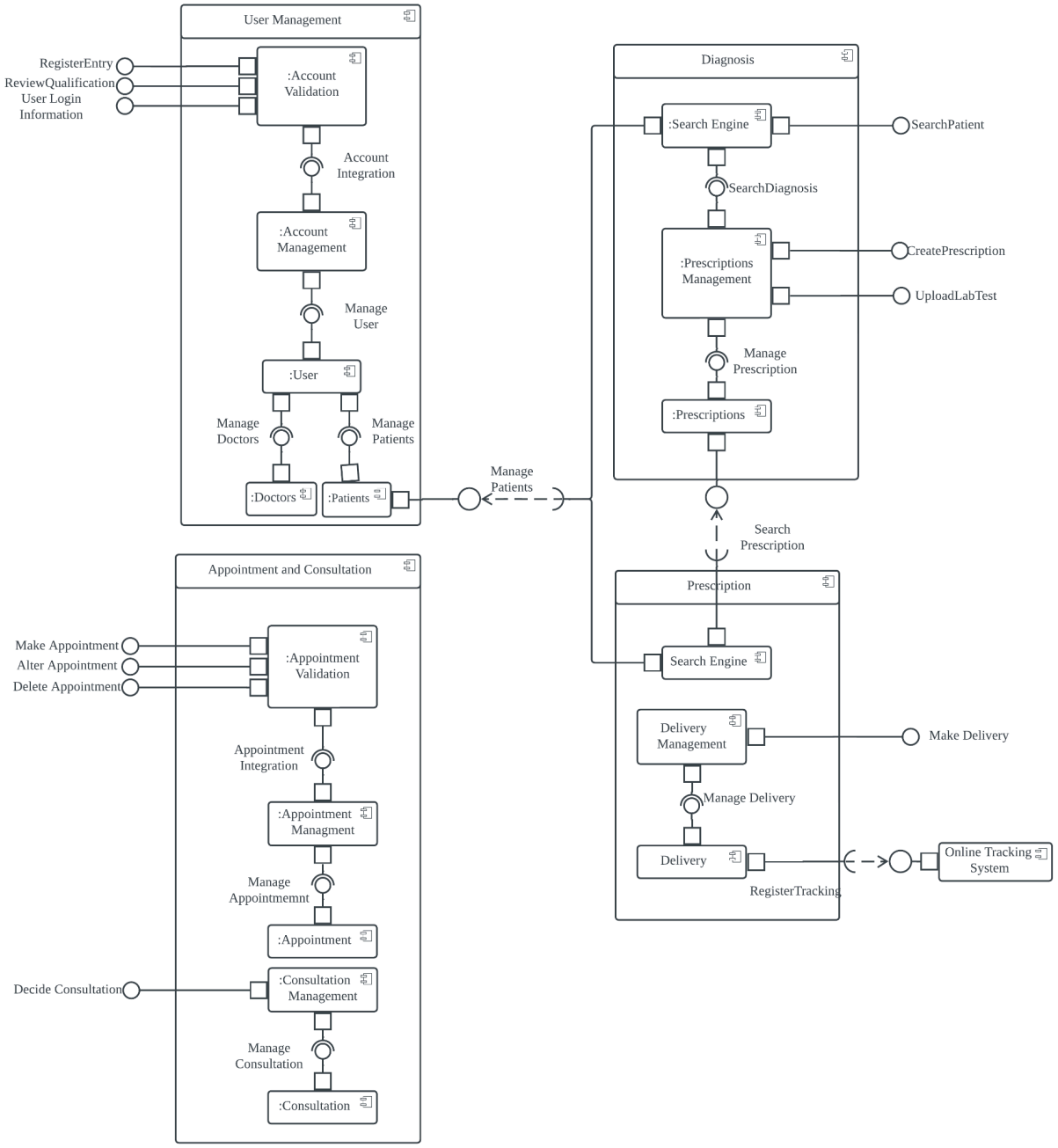


Figure 3.2: Component Diagram of <Health Consultation System>

4. Detailed Description of Components

4.1 Complete Package Diagram

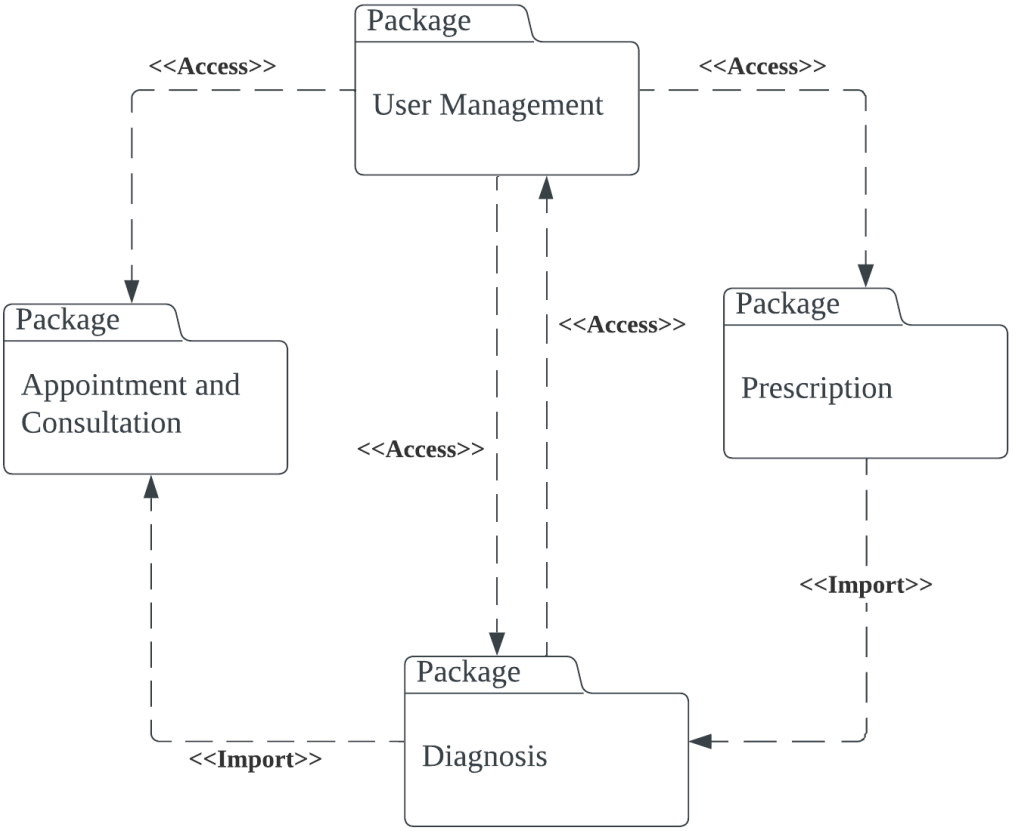


Figure 4.1 Package Diagram for <Health Consultation System>

4.2 Detailed Description

4.2.1 P001: <User Management> Subsystem

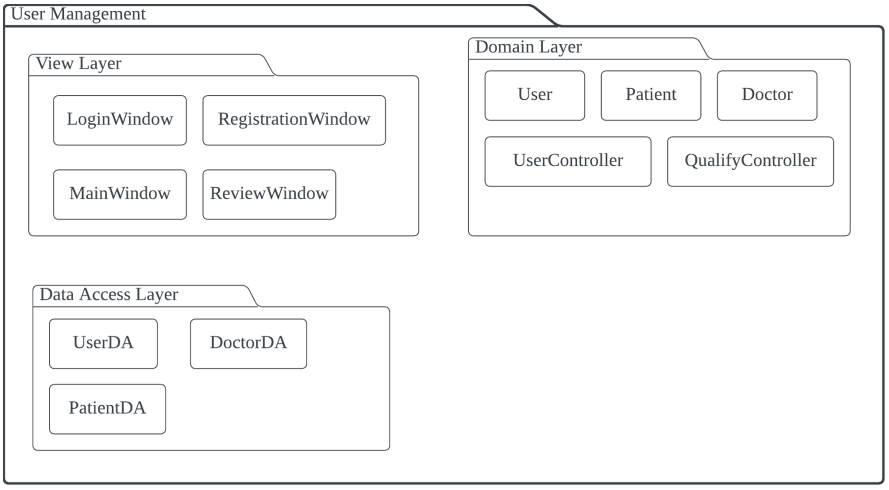


Figure 4.2: Package Diagram for <User management> Subsystem

4.2.1.1 Class Diagram

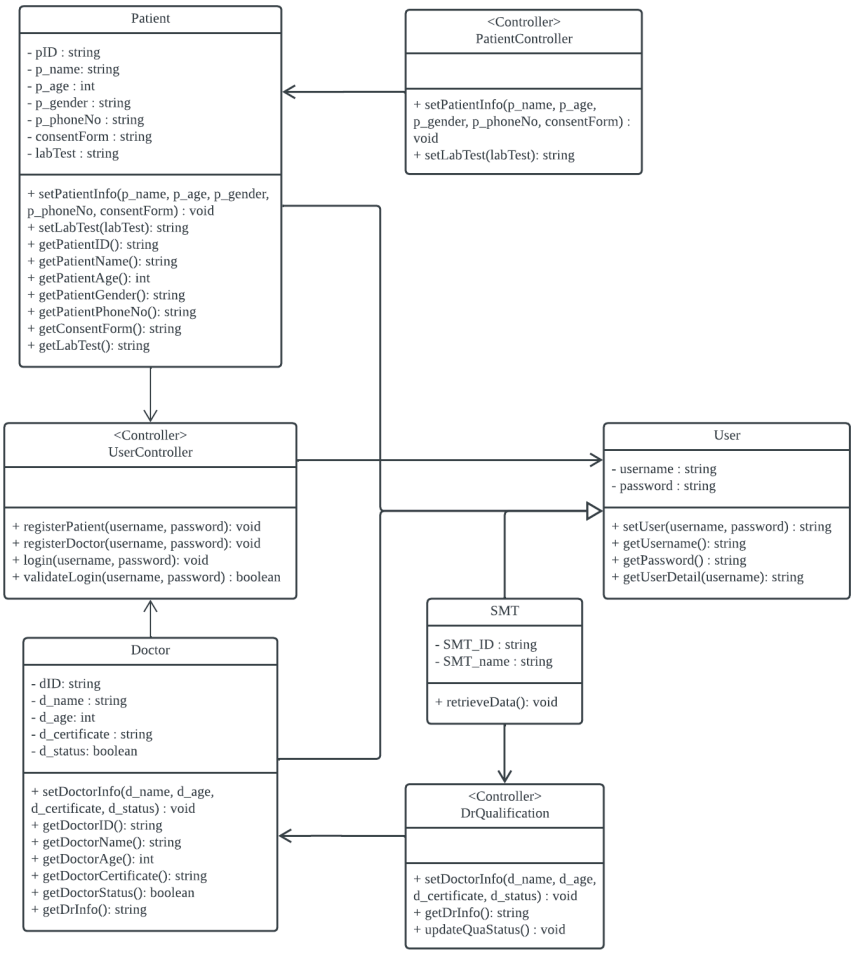


Figure 4.3: Class Diagram for <User management> Subsystem

Entity Name	Patient
Method Name	setPatientInfo
Input	p_name, p_age, p_gender, p_phoneNo, consentForm
Output	pID
Algorithm	1. Start 2. Display textarea to fill in personal information 3. Sign the consent form 4. Read all input and pass it to the system 5. Create pID 6. End

Entity Name	Doctor
Method Name	setDoctorInfo
Input	d_name, d_age, d_specialist, d_certificate
Output	-
Algorithm	1. Start 2. Display textarea to fill in personal information especially qualification 3. Submit input to system 4. Read all input and pass it to the system 5. Create dID 6. End

Entity Name	UserController
Method Name	registerPatient
Input	username, password
Output	Username, password

Algorithm	1. Start 2. Read username and password entered by user, pass into system 3. Set these information identified into data access 4. End

Entity Name	UserController
Method Name	login
Input	username, password
Output	-
Algorithm	1. Start 2. Read username and password 3. System give feedback for account confirmation 4. User success login account if feedback is true, else stop user from logging into system 5. End

Entity Name	DrQualification
Method Name	updateQuaStatus
Input	-
Output	-
Algorithm	1. Start 2. Request the doctor info from data access 3. Display it doctor's information 4. Update approve status of qualification manually either "True" or "False" 5. End

Entity Name	UserController
Method Name	registerDr

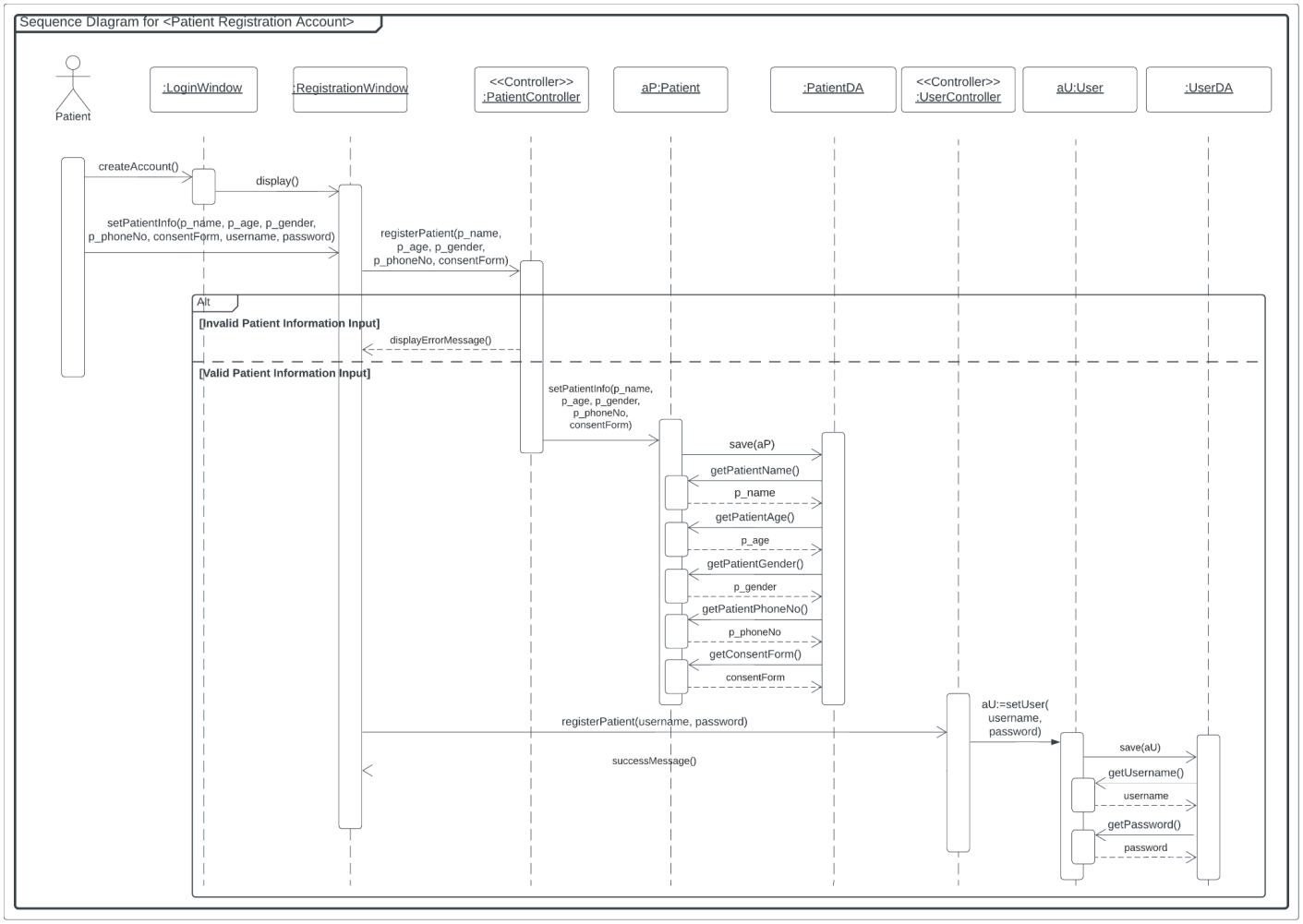
Input	username, password
Output	-
Algorithm	1. Start 2. Read username and password entered by doctor, pass into system 3. Check approval status 4. If status is “True”, set username and password into data access 5. else, call “deleteDoctor()” method to free the space 6. End

Entity Name	UserController
Method Name	validateLogin
Input	username, password
Output	-
Algorithm	1. Start 2. Read username and password, pass into system 3. Request system controller to validate the user input 4. End

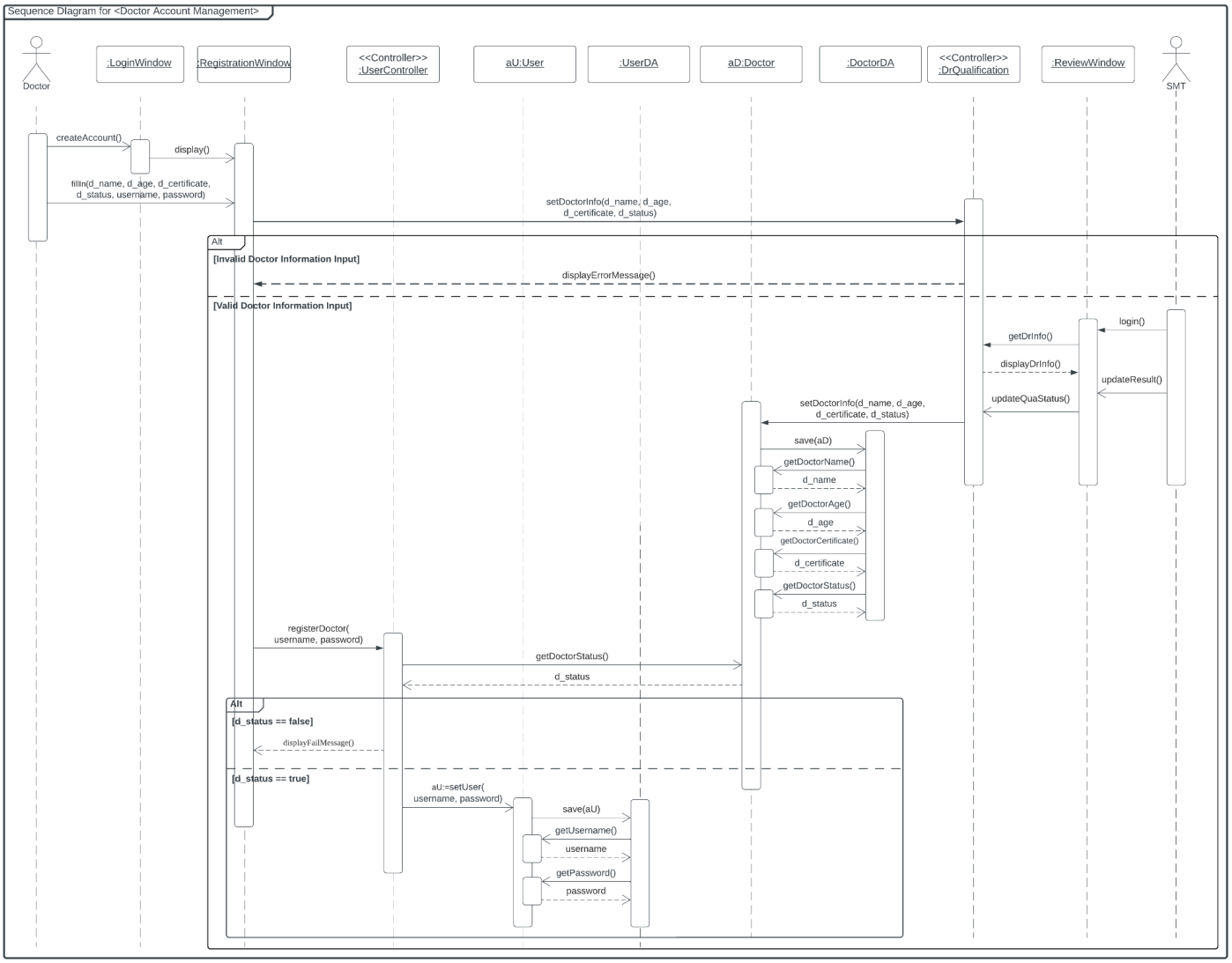
Entity Name	User
Method Name	getUserDetail
Input	username
Output	-
Algorithm	1. Start 2. Read username, pass to system 3. Retrieve the information which is username and password 4. System give a response for account validation 5. User success login account into main page if response is true, else stop user from logging into system 6. End

4.2.1.2 Sequence Diagram

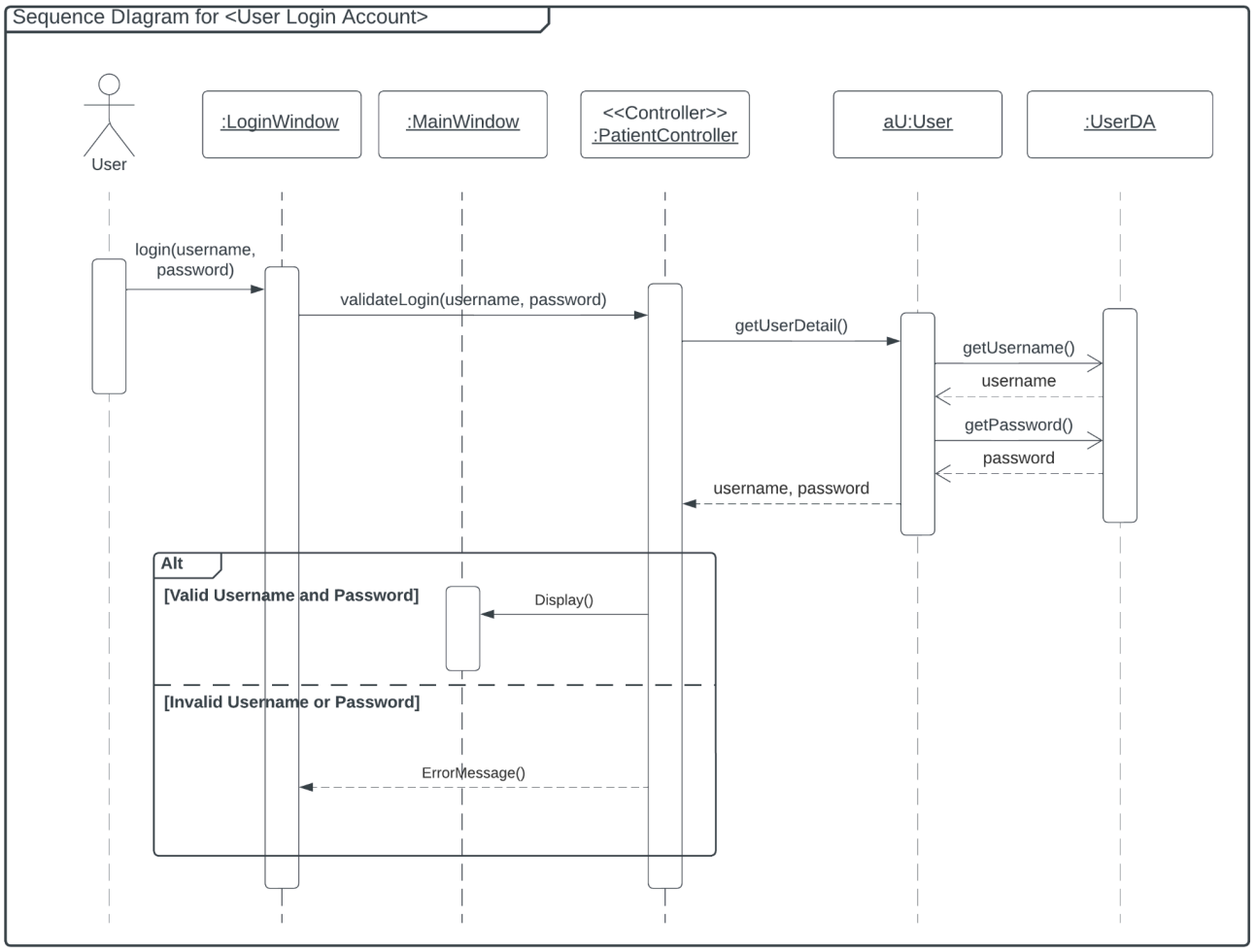
a) SD001: Sequence diagram for User Register System



b) SD002: Sequence diagram for Doctor Register and Login System



c) SD003: Sequence diagram for User Login System



4.2.2 P002: <Consultation and Appointment> Subsystem

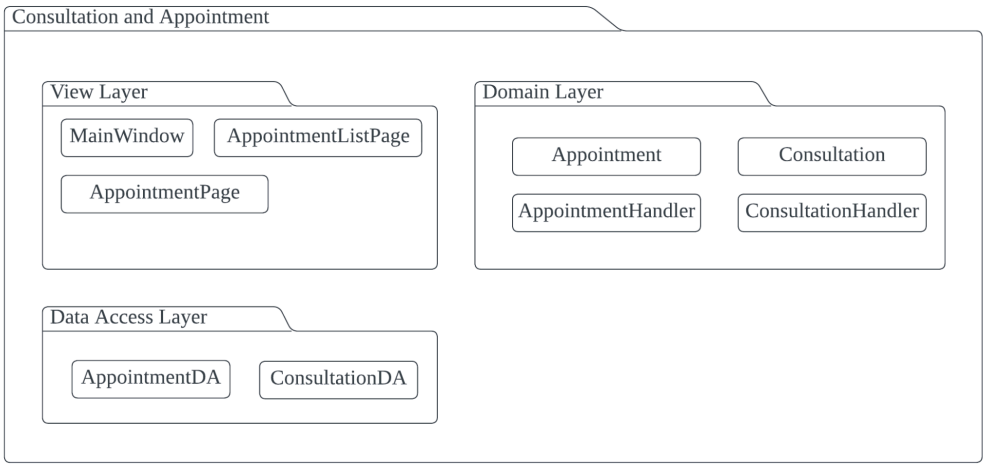


Figure 4.4: Package Diagram for <Consultation and Appointment> Subsystem

4.2.2.1 Class Diagram

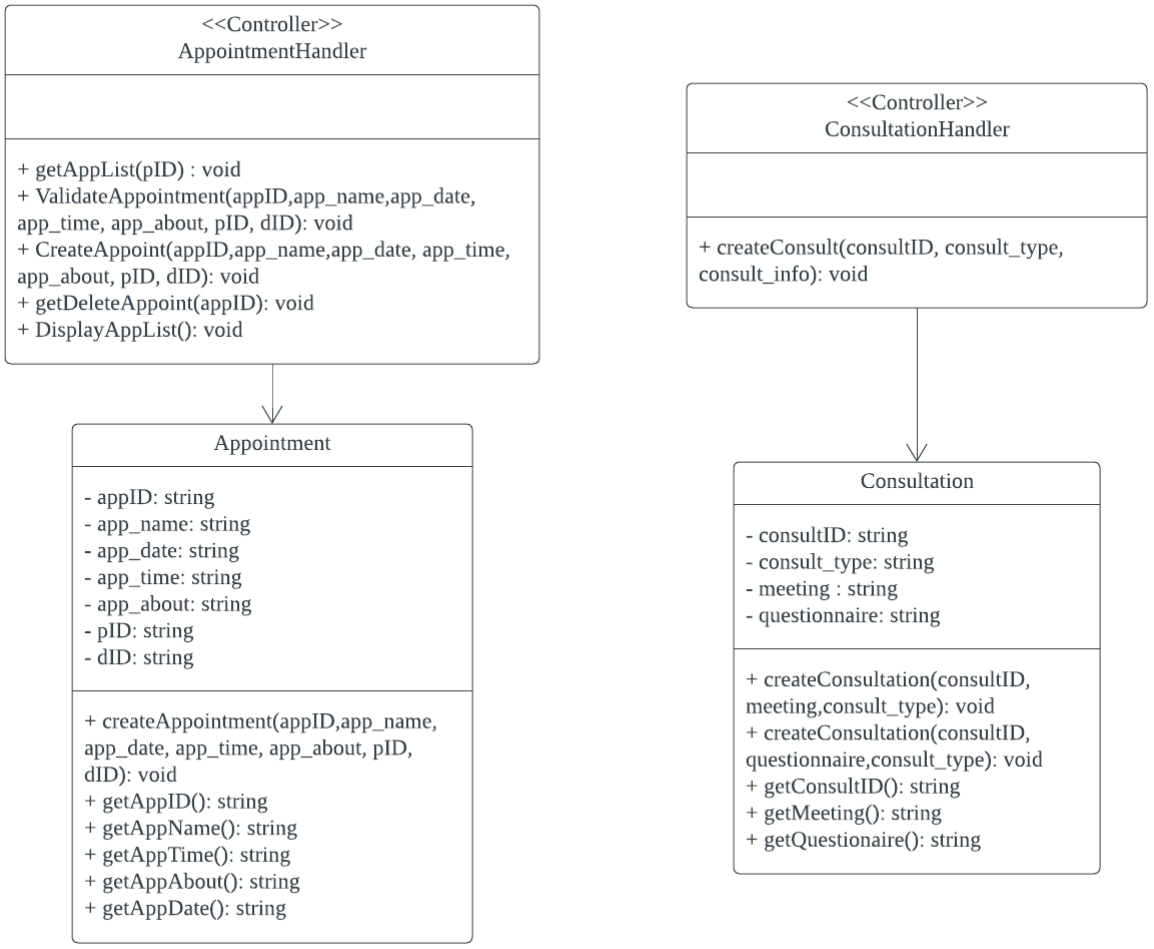


Figure 4.5: Class Diagram for <Consultation and Appointment> Subsystem

Entity Name	AppointmentHandler
Method Name	getAppList
Input	pID
Output	Retrieve a list of scheduled appointment that had been made by the users for the display in the AppointmentList Page
Algorithm	1. Start 2. Retrieve the tuple of appointment that is corresponding to patient ‘s pID 3. Download the tuple retrieved into a file 4. Call the method “displayAppList” from its own entity 5. End

Entity Name	AppointmentHandler
Method Name	displayAppList
Input	-
Output	AppListFile
Algorithm	1. Start 2. Return the path of the file that contains the list of scheduled appointment of the patient 3. End

Entity Name	AppointmentHandler
Method Name	ValidateAppointment
Input	appID, app_name, app_date, app_time, app_about, pID, dID
Output	ValidApp
Algorithm	1. Start 2. Check Input Appointment Details 3. If valid

	3.1. ValidApp = true 4. If invalid 4.1. ValidApp = false 5. End

Entity Name	AppointmentHandler
Method Name	CreateAppoint
Input	appID,app_name,app_date,app_time,app_about,pID,dID
Output	● Create an entity to store the information of the appointment that the user made with in the current session ● Display error message to notify if the submitted appointment form is invalid
Algorithm	1. Start 2. Assign aA which is an object of the Appointment Entity with the new object that is created by calling createAppointment method from Appointment entity 3. If ValidApp == false 3.1. Display error message 4. End

Entity Name	AppointmentHandler
Method Name	getDeleteAppoint
Input	appID
Output	● Delete the selected appointment of the patient from the scheduled appointment list ● Display success message
Algorithm	1. Start 2. Delete the selected appointment data from the file that contains scheduled appointment list of the patient 3. Update the deleted appointment tuple in the appointment database by keeping pID empty 4. End

Entity Name	Appointment
Method Name	createAppointment
Input	appID,app_name, app_date,app_time, app_about,pID,dID
Output	● Create a new object of Appointment Entity by using the inputs ● The corresponding appointment tuple is updated ● Display success message
Algorithm	1. Start 2. Create a new object of Appointment Entity with the inputs 3. Update the appointment tuple that is corresponding to the appID with the attributes of aA in the database 4. Display success message 5. End

Entity Name	ConsultationHandler
Method Name	CreateConsult
Input	consultID, consult_info, consult_type
Output	Create an entity to store the information of the consultation that the user made with in the current session
Algorithm	1. Start 2. If consult_type = “Synchronous” 2.1. meeting = consult_info 2.2. Assign aC which is an object of the Consultation Entity with the new object that is created by calling createConsultation method from Consultation entity which accept meeting as one of its parameter 3. Else 3.1. questionnaire = consult_info 3.2. Assign aC which is an object of the Consultation Entity with the new object that is created by calling createConsultation method from Consultation entity which accept questionnaire as one of its parameter

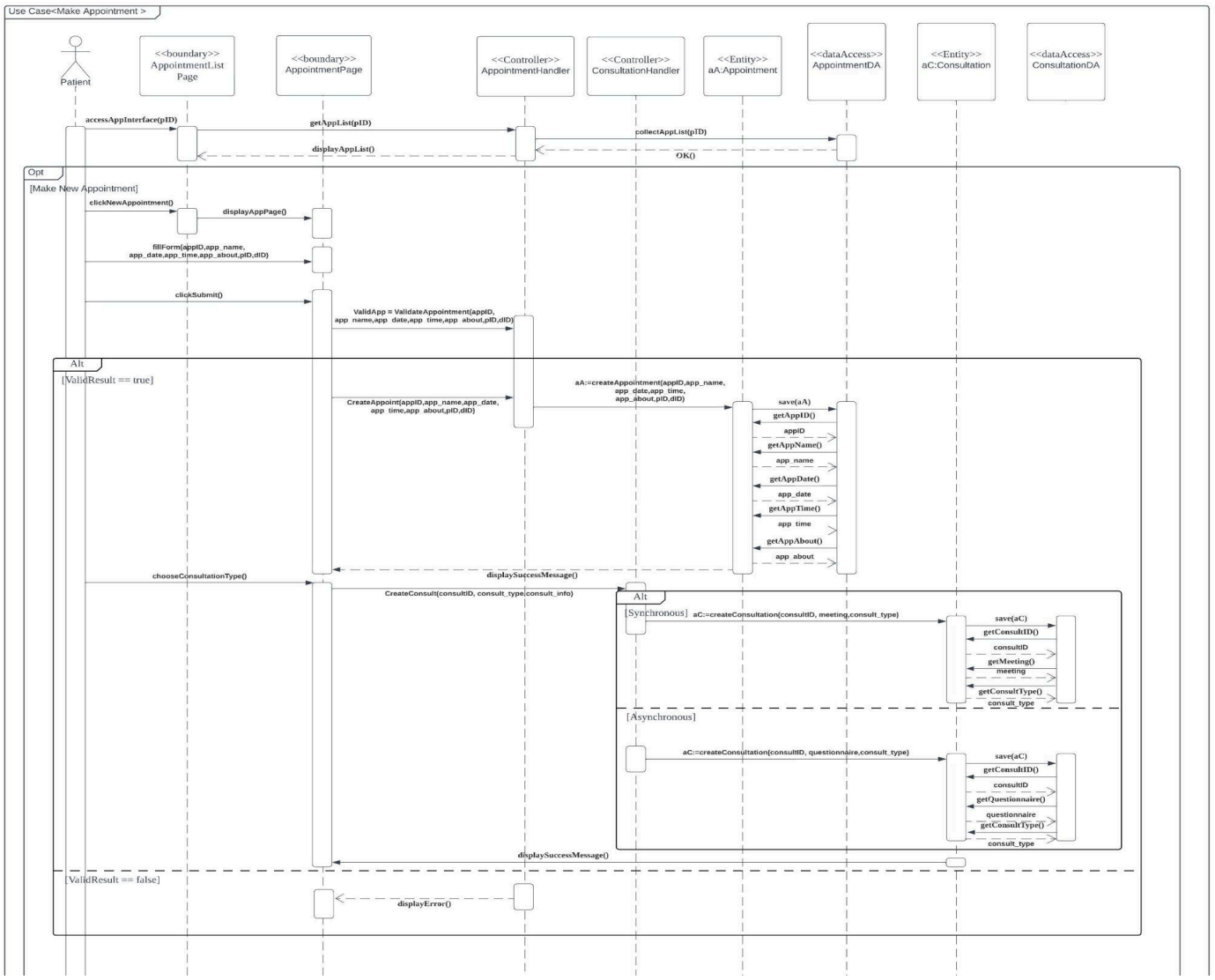
	4. End

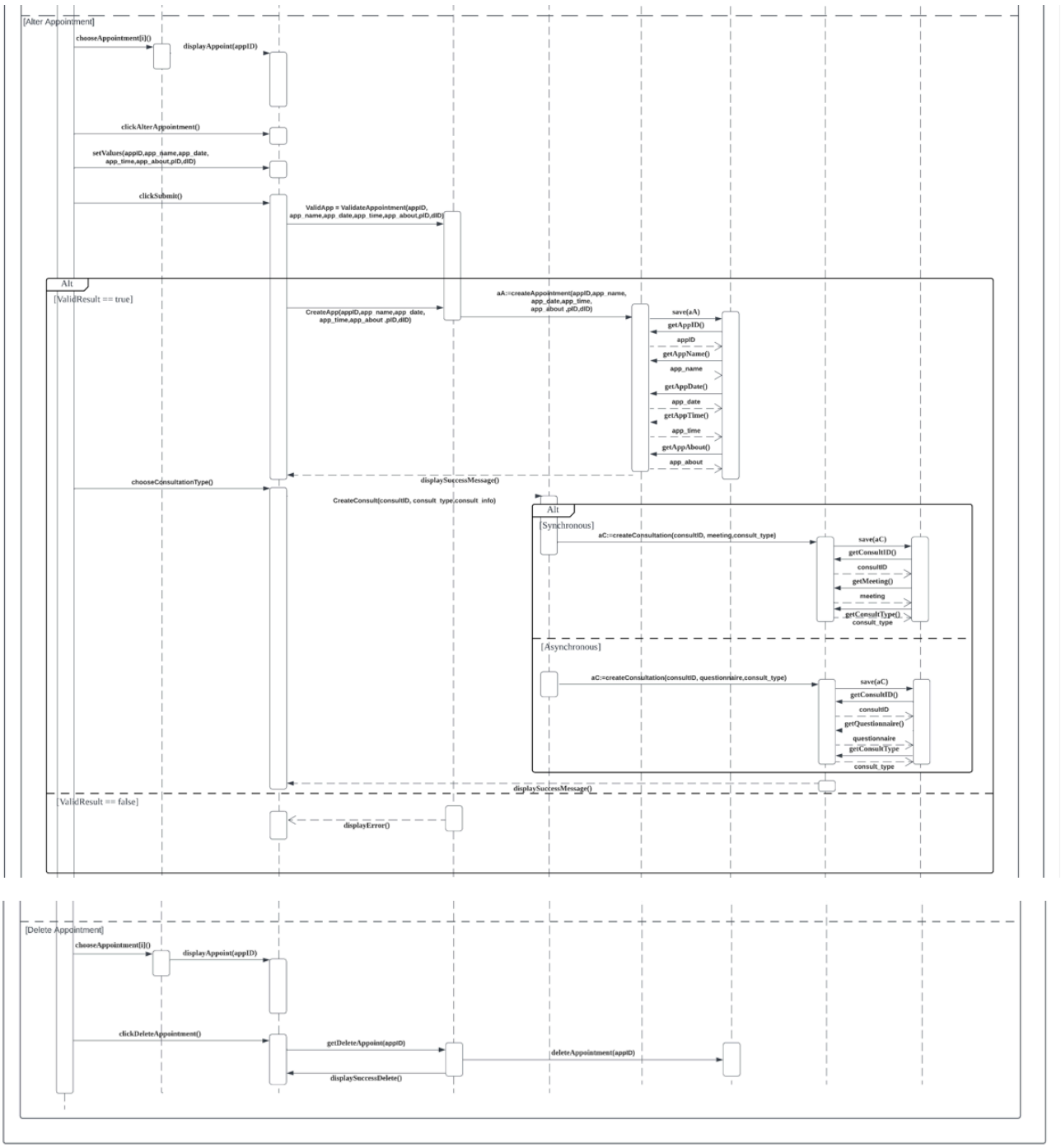
Entity Name	Consultation
Method Name	createConsultation
Input	consultID, meeting, consult_type
Output	Create a new object of Consultation Entity by using the inputs
Algorithm	1. Start 2. Create a new object of Consultation Entity with the inputs 3. Display success message 4. End

Entity Name	Consultation
Method Name	createConsultation
Input	consultID, questionnaire, consult_type
Output	Create a new object of Consultation Entity by using the inputs
Algorithm	1. Start 2. Create a new object of Consultation Entity with the inputs 3. Display success message 4. End

4.2.2.2 Sequence Diagram

SD004: Sequence Diagram for <Make Appointment>




[Alter Appointment]
chooseAppointment[i]()
displayAppoint(appID)
clickAlterAppointment()
setValues(appID,app_name,app_date,
app_time,app_about,pID,dID)
clickSubmit()
ValidApp = ValidateAppointment(appID,
app_name,app_date,app_time,app_about,pID,dID)
Alt
[ValidResult == true]
CreateApp(appID,app_name,app_date,
app_time,app_about ,pID,dID)
aA:=createAppointment(appID,app_name,
app_date,app_time,
app_about ,pID,dID)
save(aA)
getAppID()
appID
getAppName()
app_name
getAppDate()
app_date
getAppTime()
app_time
getAppAbout()
app_about
displaySuccessMessage()
chooseConsultationType()
CreateConsult(consultID, consult_type,consult_info)
Alt
[Synchronous]
aC:=createConsultation(consultID, meeting,consult_type)
save(aC)
getConsultID()
consultID
getMeeting()
meeting
getConsultType()
consult_type
[Asynchronous]
aC:=createConsultation(consultID, questionnaire,consult_type)
save(aC)
getConsultID()
consultID
getQuestionnaire()
questionnaire
getConsultType
consult_type
displaySuccessMessage()
[ValidResult == false]
displayError()
[Delete Appointment]
chooseAppointment[i]()
displayAppoint(appID)
clickDeleteAppointment()
getDeleteAppoint(appID)
deleteAppointment(appID)
displaySuccessDelete()

4.2.3 P003: <Diagnosis> Subsystem

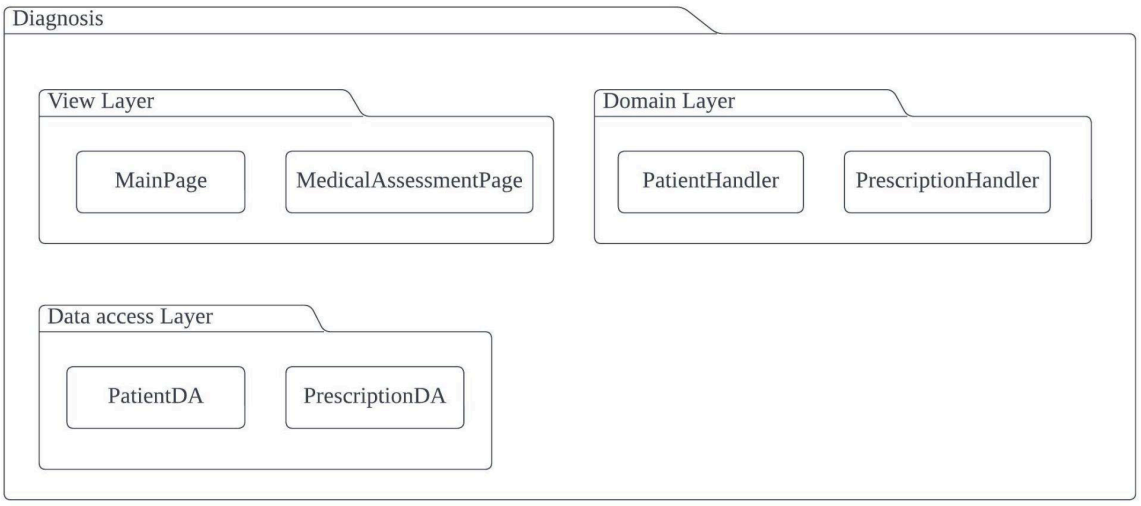


Figure 4.6: Package Diagram for <Diagnosis> Subsystem

4.2.3.1 Class Diagram

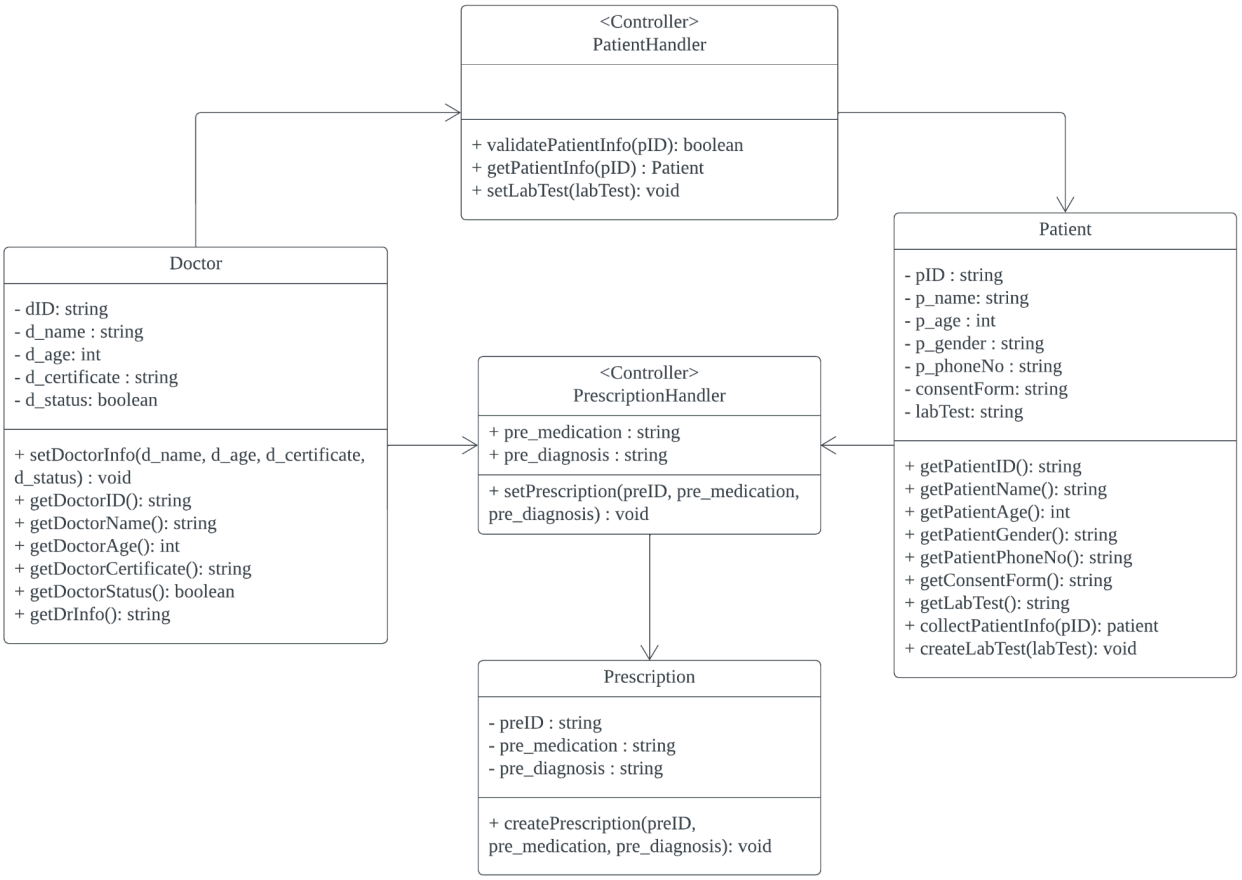


Figure 4.7: Class Diagram for <Diagnosis> Subsystem

Entity Name	PatientHandler
Method Name	validatePatientInfo
Input	pID
Output	Boolean value (validity of pID)
Algorithm	1. Start 2. Check if the pID entered exists and is valid 3. If the pID is valid then 3.1 Return true 4. If the pID is invalid 4.1 Return false 5. End

Entity Name	PatientHandler
Method Name	getPatientInfo
Input	pID
Output	Patient information
Algorithm	6. Start 7. Input the pID 8. Retrieve the patient information from the Patient data access 9. End

Entity Name	PatientHandler
Method Name	setLabTest()
Input	labTest
Output	-

Algorithm	1. Start 2. Input the labTest 3. Save the labTest into Patient data access 4. End

Entity Name	PrescriptiontHandler
Method Name	setPrescription
Input	preID, pre_medication, pre_diagnosis
Output	-
Algorithm	1. Start 2. Input preID, pre_medication, pre_diagnosis 3. Save the preID, pre_medication, pre_diagnosis into Prescription data access 4. End

Entity Name	Patient
Method Name	collectPatientInfo
Input	pID
Output	-
Algorithm	1. Start 2. Get p_name, p_age, p_gender, p_phoneNo, labTest from Patient data access 3. End

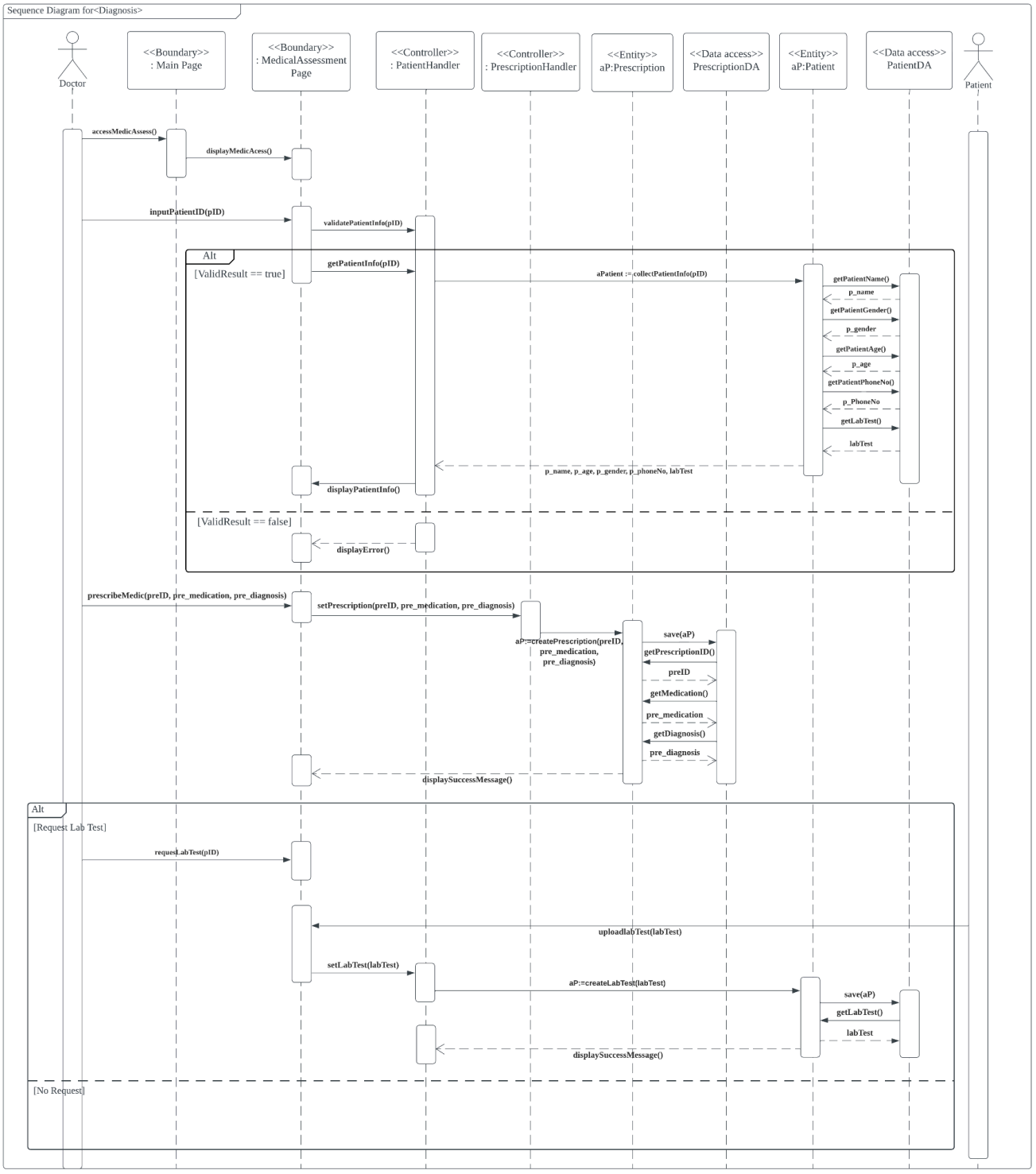
Entity Name	Patient
Method Name	createLabTest
Input	labTest
Output	-

Algorithm	1. Start 2. Create labTest record into Patient data access 3. End

Entity Name	Prescription
Method Name	createPrescription
Input	preID, pre_medication, pre_diagnosis
Output	-
Algorithm	1. Start 2. Create preID, pre_medication, pre_diagnosis records into Prescription data access 3. End

4.2.3.2 Sequence Diagram

SD005: Sequence Diagram for <Diagnosis>



4.2.4 P004: <Prescription> Subsystem

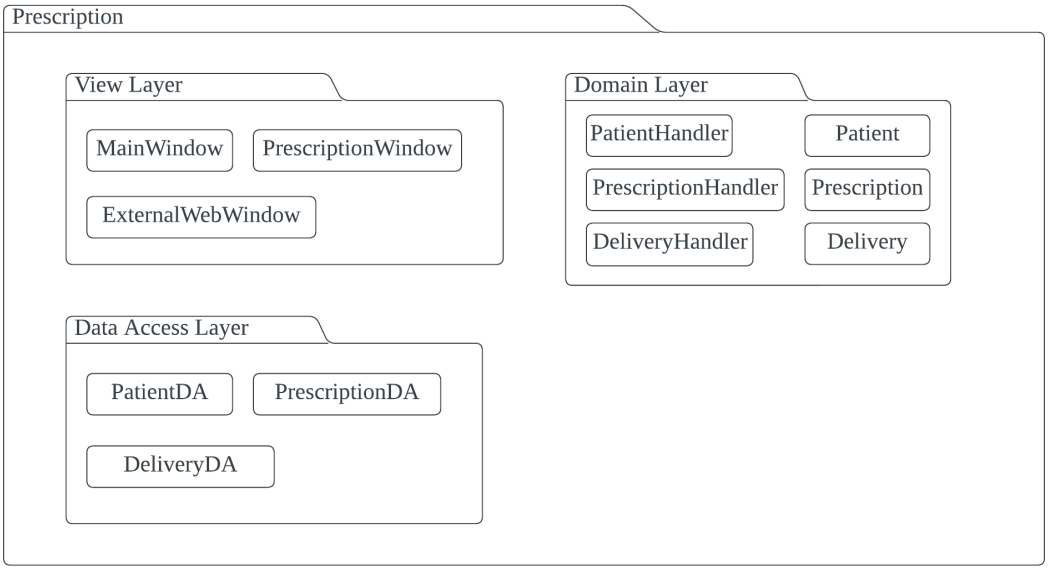


Figure 4.8: Package Diagram for <Prescription> Subsystem

4.2.4.1 Class Diagram

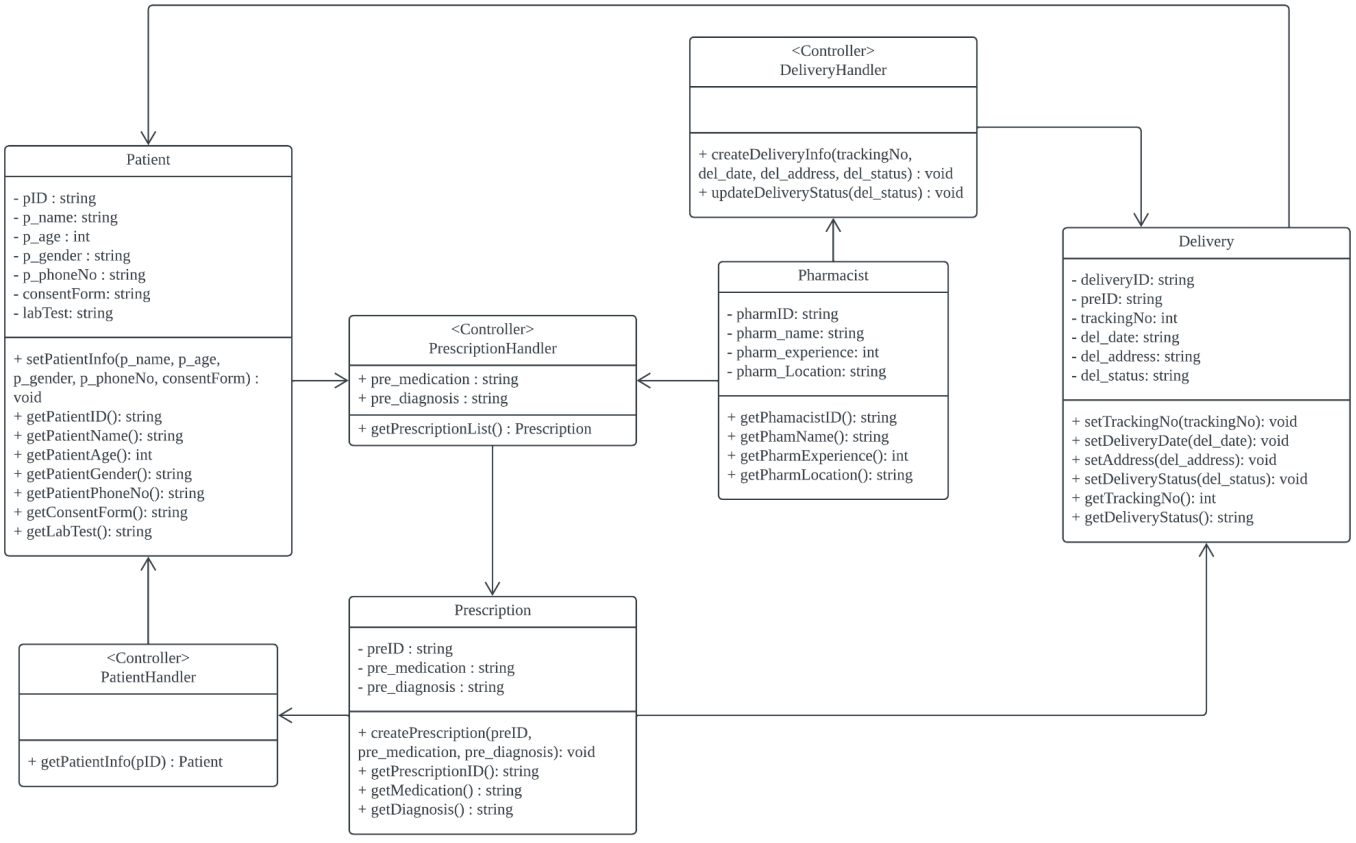


Figure 4.9: Class Diagram for <Prescription> Subsystem

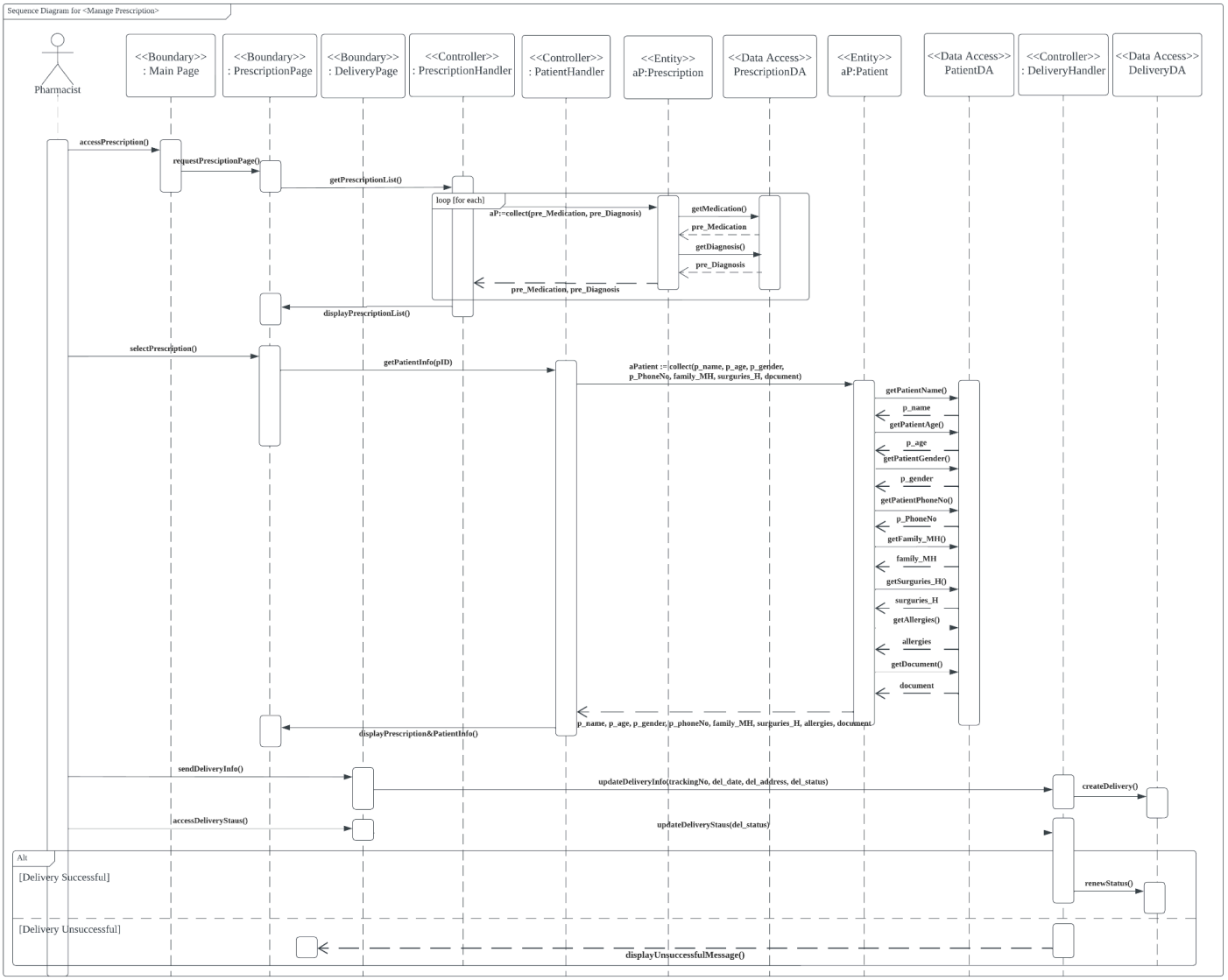
Entity Name	PrescriptionHandler
Method Name	getPrescriptionList
Input	-
Output	Prescription List
Algorithm	1. Start 2. For each 2.1 Get Prescription Information 3. Display Prescription List 4. End

Entity Name	DeliveryHandler
Method Name	updateDeliveryInfo
Input	trackingNo, del_date, del_address, del_status
Output	-
Algorithm	1. Start 2. Read trackingNo, del_date, del_address, del_status 3. Update delivery information to database 4. End

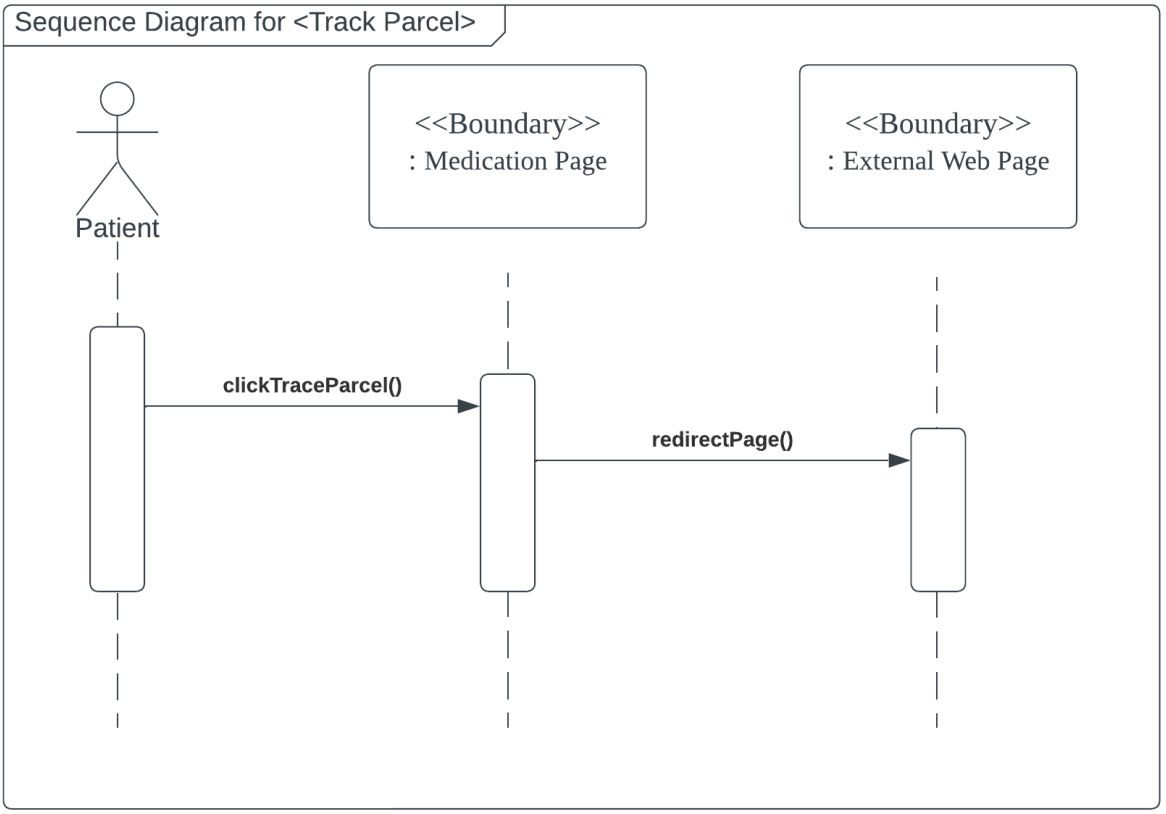
Entity Name	DeliveryHandler
Method Name	updateDeliveryStatus
Input	del_status
Output	-
Algorithm	1. Start 2. Read del_status 3. Update delivery status in database 4. End

4.2.4.2 Sequence Diagram

SD006: Sequence Diagram for <Manage Prescription>



SD007: Sequence Diagram for <Track Parcel>



5. Data Design

5.1 Data Description

The major data or systems entities are stored into a relational database named HealthSystem database, processed and organized into *8* entities as listed in Table 5.1.

Table 5.1: Description of Entities in the Database

No.	Entity Name	Description
1.	User	Store user information
2.	Patient	Store patient information
3.	Doctor	Store doctor information
4.	SMT	Store SMT information
5.	Appointment	Store appointment information
6.	Consultation	Store consultation information
7.	Prescription	Store prescription including diagnosis and medication information
8.	Delivery	Store delivery information

5.2 Data Dictionary

5.2.1 Entity: <User>

Attribute Name	Type	Description
username	VARCHAR2(10)	User account name (PRIMARY KEY)
password	VARCHAR2(10)	User account password
email	VARCHAR2(20)	User email

5.2.2 Entity: <Patient>

Attribute Name	Type	Description
pID	VARCHAR2(10)	Patient ID (PRIMARY KEY)
p_name	VARCHAR2(20)	Patient name
p_age	Integer	Patient age
p_gender	VARCHAR2(5)	Patient gender
p_phoneNo	VARCHAR2(15)	Patient phone number
consentForm	VARCHAR2(100)	Patient consent form
labTest	VARCHAR2(100)	Patient Lab Test Result

5.2.3 Entity: <Doctor>

Attribute Name	Type	Description
dID	VARCHAR2(10)	Doctor ID (PRIMARY KEY)
d_name	VARCHAR2(10)	Doctor name
d_age	Integer	Doctor age
d_certificate	VARCHAR2(200)	Doctor certificate
d_status	Boolean	Doctor qualification approval status

5.2.4 Entity: <SMT>

Attribute Name	Type	Description
SMT_ID	VARCHAR2(10)	SMT ID (PRIMARY KEY)
SMT_name	VARCHAR2(20)	SMT name

5.2.5 Entity: <Appointment>

Attribute Name	Type	Description
appID	VARCHAR2(10)	Appointment ID (PRIMARY KEY)
app_name	VARCHAR2(20)	Appointment name
app_date	DATE	Appointment date
app_time	TIME	Appointment time
app_about	VARCHAR(200)	Appointment details
pID	VARCHAR2(10)	Patient ID
dID	VARCHAR2(10)	Doctor ID

5.2.6 Entity: <Consultation>

Attribute Name	Type	Description
consultID	VARCHAR2(10)	Consultation ID (PRIMARY KEY)
consult_type	VARCHAR2(20)	Appointment name
meeting	VARCHAR(200)	Synchronous consultation information
questionnaire	VARCHAR(200)	Asynchronous consultation information

5.2.7 Entity: <Prescription>

Attribute Name	Type	Description
preID	VARCHAR2(10)	Prescription ID (PRIMARY KEY)
pre_medication	VARCHAR(50)	Prescription medication
pre_diagnosis	VARCHAR(50)	Prescription diagnosis

5.2.8 Entity: <Delivery>

Attribute Name	Type	Description
deliveryID	VARCHAR2(10)	Delivery ID (PRIMARY KEY)
preID	VARCHAR2(10)	Prescription ID (PRIMARY KEY)
trackingNo	Integer	Tracking number
del_date	DATE	Delivery date
del_time	TIME	Delivery time
del_status	VARCHAR(20)	Delivery status

6. Requirements Traceability Matrix

Package item	Use Case ID	Use Case Description	Sequence Diagram ID	Sequence Diagram Description	Test Case ID
Package 1	UC001	Manage Registration	SD001	Register Patient	TC001_01_01
					TC001_01_02
			SD002	Register Doctor	TC001_02_01
					TC001_02_02
					TC001_02_03
					TC001_02_04
	UC002	Login Account	SD003	User Login	TC002_01_01
					TC002_01_02
Package 2	UC003	Make Appointment	SD004	Make/Alter/Delete Appointment	TC003_01_01
					TC003_01_02
					TC003_01_03
					TC003_01_04
					TC003_01_05
Package 3	UC004	Diagnosis	SD005	Diagnosis	TC004_01_01
					TC004_01_02
					TC004_01_03
					TC004_01_04
Package 4	UC005	Manage Prescription	SD006	Manage Prescription	TC005_01_01
					TC005_01_02
	UC006	Track Parcel	SD007	Track Parcel	TC006_01_01

Table 6.1: RTM for <Health Consultation System>

7. Test Cases

7.1 TC001: Test <User Management> Subsystem: <Manage Registration (UC001)>

This test contains the following test cases:

(a) TC001_01: Test <Scenario of Patient Registration Account (SD001)>

(b) TC001_02: Test <Scenario of Doctor Account Management (SD002)>

(c) TC001_03: Test <Scenario of User Login Account (SD003)>

7.1.1 TC001_01: Test <Scenario of Patient Registration Account (SD001)>

This test contains the following scenarios:

(a) TC001_01_01: Test <normal scenario of Patient Registration Account (SD001)>

Test Case ID	T001_01_01	Test Case Description	Test normal flow of patient registration		
Created by	Gan Heng Lai	Reviewed By	Ng Kai Zheng	Version	1.0
QA Tester's Log					
Tester's Name	Gan Heng Lai	Date Tested	14-June-2023	Test case(Pass/Fail/...)	Pass

S#	Prerequisties:
1	Access to Chrome Browser
2	Patient have not register account before
3	
4	

S#	Test Data
1	username = ali123
2	password = jvk@123
3	p_name = ali muhammad
4	p_age = 30
5	p_gender = Male
6	p_phoneNo = 012-34576784

Table 7.1.1a: TC001_01_01 - <Normal Scenario of Patient Registration Account (SD001)>

(b) TC001_01_02: Test <exception scenario of Patient Registration Account (SD001)>

Test Case ID	T001_01_02	Test Case Description	Test exception flow of patient registration		
Created by	Gan Heng Lai	Reviewed By	Ng Kai Zheng	Version	1.0
QA Tester's Log					
Tester's Name	Gan Heng Lai	Date Tested	14-June-2023	Test case(Pass/Fail/...)	Pass

S#	Prerequisties:
1	Access to Chrome Browser
2	Patient have not register account before
3	
4	

S#	Test Data
1	username = ali amir
2	password = ali123
3	p_name = 2334
4	p_age = -10
5	p_gender = 12345
6	p_phoneNo = abcd

Test Scenario	Verify the handling of exceptions during the patient registration process			
Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Navigate to the registration page	Site should open		
2	Enter invalid information in all the required fields	Required information can enter by user with some error.		
3	Click on the "Create Account" button	The system shows the corresponding error messages for each missing or incorrect field.		

Table 7.1.1b: TC001_01_02 - <Exception Scenario of Patient Registration Account (SD001)>

7.1.2 TC001_02: Test <Scenario of Doctor Account Management (SD002)>

(a) TC001_02_01: Test <normal scenario of Doctor Registration Account (SD002)>

Test Case ID	T001_02_01	Test Case Description	Test normal flow of doctor registration		
Created by	Gan Heng Lai	Reviewed By	Ng Kai Zheng	Version	1.0
QA Tester's Log					
Tester's Name	Gan Heng Lai	Date Tested	14-June-2023	Test case(Pass/Fail/...)	Pass

S#	Prerequisties:
1	Access to Chrome Browser
2	Doctor have not register account before
3	
4	

S#	Test Data
1	username = drjohnsmith
2	password = Dr@Pass456
3	p_name = Dr. John Smith
4	p_age = 45
5	d_certificate: "mycertificate.pdf"

Test Scenario	Verify that a user can successfully create a patient registration account

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Navigate to the registration page	Site should open		
2	Enter valid information in all the required fields	Required information can enter by user without any error.		
3	Upload certificate	The doctor can upload the updated certificate file.		
4	Click on the "Create Account" button	If the data is valid, the account creation process is initiated and user required waiting account qualification approvement.		

Table 7.1.2a: TC001_02_01 - <Normal Scenario of Doctor Registration Account (SD002)>

(b) TC001_02_02: Test <alternative scenario of Doctor Registration Account (SD002)>

Test Case ID	T001_02_02	Test Case Description	Test alternate flow of SMT Approving Qualification		
Created by	Gan Heng Lai	Reviewed By	Ng Kai Zheng	Version	1.0
QA Tester's Log					
Tester's Name	Gan Heng Lai	Date Tested	14-June-2023	Test case(Pass/Fail/...)	Pass

S#	Prerequisties:
1	Access to Chrome Browser
2	New qualification arrived
3	
4	

S#	Test Data
1	username = smt_reviewer1
2	password = ExpertSMT2023
3	dID = P005
4	d_certificate: "mycertificate.pdf"

Test Scenario	Verify the normal flow of approving the qualification of a doctor by the SMT

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Log in to the SMT reviewer account	The SMT reviewer successfully logs in and is directed to the SMT dashboard		
2	Review the qualification documents and information	The SMT reviewer can thoroughly assess the provided qualification documents		
3	Verify the authenticity and accuracy of the qualification	The SMT reviewer approves the qualification and system updates the doctor information into database		

Table 7.1.2b: TC001_02_02 - <Alternative Scenario of Doctor Registration Account (SD002)>

(c) TC001_02_03: Test <exception scenario of Doctor Registration Account (SD002)>

Test Case ID	T001_02_03	Test Case Description	Test alternate flow of SMT Rejecting Qualification		
Created by	Gan Heng Lai	Reviewed By	Ng Kai Zheng	Version	1.0
QA Tester's Log					
Tester's Name	Gan Heng Lai	Date Tested	14-June-2023	Test case(Pass/Fail/...)	Pass

S#	Prerequisties:
1	Access to Chrome Browser
2	New qualification arrived
3	
4	

S#	Test Data
1	username = smt_reviewer1
2	password = ExpertSMT2023
3	dID = P006
4	d_certificate: "dradamcertificate.pdf"

Test Scenario	Verify the normal flow of rejecting the qualification of a doctor by the SMT

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Log in to the SMT reviewer account	The SMT reviewer successfully logs in and is directed to the SMT dashboard		
2	Review the qualification documents and information	The SMT reviewer can thoroughly assess the provided qualification documents		
3	Verify the authenticity and accuracy of the qualification	The SMT reviewer rejects the qualification and system send a notification to doctor and display rejection message		

Table 7.1.2c: TC001_02_03 - <Alternative Scenario of Doctor Registration Account (SD002)>

(d) TC001_02_04: Test <exception scenario of Doctor Registration Account (SD002)>

Test Case ID	T001_02_04	Test Case Description	Test exception flow of doctor registration		
Created by	Gan Heng Lai	Reviewed By	Ng Kai Zheng	Version	1.0
QA Tester's Log					
Tester's Name	Gan Heng Lai	Date Tested	14-June-2023	Test case(Pass/Fail/...)	Pass

S#	Prerequisties:
1	Access to Chrome Browser
2	Doctor have not register account before
3	
4	

S#	Test Data
1	username = Dr. John Smith
2	password = password
3	p_name = 213431
4	p_age = 0
5	d_certificate: "mycertificate.png"

Test Scenario	Verify the handling of exceptions during the doctor registration process

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Navigate to the registration page	Site should open		
2	Enter invalid information in all the required fields	Required information can enter by user with some error.		
3	Upload certificate	The doctor can upload the new certificate file.		
4	Click on the "Create Account" button	The system shows the corresponding error messages for each missing or incorrect field.		

Table 7.1.2d: TC001_02_04 - <Exception Scenario of Doctor Registration Account (SD002)>

7.2 TC002: Test <User Management> Subsystem: <Login Account (UC002)>

This test contains the following test cases:

(a) TC002_01: Test <Scenario of User Login Account (SD003)>

7.2.1 TC002_01: Test <Scenario of User Login Account (SD003)>

This test contains the following scenarios:

(a) TC002_01_01: Test <normal scenario of User Login Account (SD003)>

Test Case ID	T002_01_01	Test Case Description	Test normal flow of user login		
Created by	Gan Heng Lai	Reviewed By	Ng Kai Zheng	Version	1.0
QA Tester's Log					
Tester's Name	Gan Heng Lai	Date Tested	14-June-2023	Test case(Pass/Fail/...)	Pass

S#	Prerequisties:
1	Access to Chrome Browser
2	User has registered an account before
3	
4	

S#	Test Data
1	username = johndoe
2	password = jhn@2023
3	
4	

Test Scenario	Verify the user entered username and password and successful login process for a user.

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Navigate to the login page	Site should open		
2	Enter username and password	Credential can be entered		
3	Click submit	User is logged in		

Table 7.2.1a: TC002_01_01 - <Normal Scenario of User Login Account (SD003)>

(a) TC002_01_02: Test <exception scenario of User Login Account (SD003)>

Test Case ID	T002_01_02	Test Case Description	Test exception flow of user login		
Created by	Gan Heng Lai	Reviewed By	Ng Kai Zheng	Version	1.0
QA Tester's Log					
Tester's Name	Gan Heng Lai	Date Tested	14-June-2023	Test case(Pass/Fail/...)	Pass

S#	Prerequisties:
1	Access to Chrome Browser
2	User has registered an account before
3	
4	

S#	Test Data
1	username = johndoe
2	password = amd@2331
3	
4	

Test Scenario	Verify the user entered username and password and successful login process for a user.

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Navigate to the login page	Site should open		
2	Enter username and password	Credential can be entered		
3	Click submit	An appropriate error message is displayed indicating the login failure		

Table 7.2.1b: TC002_01_02 - <Exception Scenario of User Login Account (SD003)>

7.3 TC003: Test <Consultation and Appointment> Subsystem: <Make Appointment (UC003)>

This test contains the following test cases:

(a) TC003_01: Test <Scenario of Make Appointment (SD004)>

7.3.1 TC003_01: Test <Scenario of Make Appointment(SD004)>

This test contains the following scenarios:

(a) TC003_01_01: Test <normal scenario of Make Appointment (SD004)>

Test Case ID	TC003_01_01	Test Case Description	Test the normal flow of Making Appointment and Consultation		
Created by	Yeo Chun Teck	Reviewed By		Version	4.0
QA Tester's Log					
Tester's Name	Yeo Chun Teck	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User is logged into the system as patient
3	Existence of valid patient IDs in the database
4	The patient intends to make a new appointment

S#	Test Data
1	pID: P012
2	appID: A035
3	app_name: First Consultation with Dr. Smith
4	app_date: 14/6/2023
5	app_time: 10:00AM
6	app_about: Dental Cleaning and Check-up
7	dID: D001

Test Scenario	Verify inputs of Appointment Form,then make a new appointment and select the mode for the consultation

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click "Make New Appointment" option	A new appointment form is displayed		
2	Fill in the Appointment Form	Valid Info is filled in the appointment form		
3	Click submit	Appointment data is updated in the Appointment database		
4	Decide Consultation Mode	Questionnaire info or meeting info is updated in the consultation database based on the decision		

Table 7.3.1a: TC003_01_01 - <Normal Scenario of Make Appointment(SD004)>

(b) TC003_01_02: Test <alternate scenario 1 of Make Appointment (SD004) (Alter Appointment)

Test Case ID	TC003_01_02	Test Case Description	Test the alternate flow of Making Appointment and Consultation		
Created by	Yeo Chun Teck	Reviewed By		Version	4.0
QA Tester's Log					
Tester's Name	Yeo Chun Teck	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User is logged into the system as patient
3	Existence of valid patient IDs in the database
4	The patient intends to alter a scheduled appointment
5	The patient has at least a scheduled appointment which is not expired

S#	Test Data
1	pID: P012
2	appID: A035
3	app_name: First Consultation with Dr. Smith
4	app_date: 14/6/2023
5	app_time: 10:00AM
6	app_about: Dental Cleaning and Check-up
7	dID: D001

Test Scenario	Verify Altered Inputs to Scheduled Appointment,then alter the scheduled appointment and alter the mode for the consultation

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click one of scheduled appointments in the list	A scheduled appointment info is displayed		
2	Click Alter	An appointment form with booking info is displayed		
3	Alter the booking info in the appointment form	The booking info in appointment form is altered and verified		
4	Click Submit	Appointment data is updated in the Appointment database		
4	Alter Consultation Mode	Questionnaire info or meeting info is updated in the consultation database based on the decision		

Table 7.3.1b: TC003_01_02 - <Alternate Scenario 1 of Make Appointment(SD004)>

(c) TC003_01_03: Test <alternate scenario 2 of Make Appointment (SD004) (Delete Appointment)

Test Case ID	TC003_01_03	Test Case Description	Test the alternate flow of Making Appointment and Consultation		
Created by	Yeo Chun Teck	Reviewed By		Version	4.0
QA Tester's Log					
Tester's Name	Yeo Chun Teck	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User is logged into the system as patient
3	Existence of valid patient IDs in the database
4	The patient intends to delete a scheduled appointment
5	The patient has at least a scheduled appointment which is not expired

S#	Test Data
1	appID: A035

Test Scenario	Delete a scheduled appointment

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click one of scheduled appointments in the list	A scheduled appointment info is displayed		
2	Click Delete	The appointment data is updated by deleting booking info		

Table 7.3.1c: TC003_01_03 - <Alternate Scenario 2 of Make Appointment(SD004)>

(d) TC003_01_05: Test <exceptional scenario 2 of Make Appointment (SD004)

Test Case ID	TC003_01_04	Test Case Description	Test the exceptional flow of Making Appointment and Consultation		
Created by	Yeo Chun Teck	Reviewed By	Ng Kai Zheng	Version	4.0
QA Tester's Log					
Tester's Name	Yeo Chun Teck	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User is logged into the system as patient
3	Existence of valid patient IDs in the database
4	The patient intends to make a new appointment

S#	Test Data
1	pID: P012
2	appID: A035
3	app_name: First Consultation with Dr. Smith
4	app_date:
5	app_time: 10:00AM
6	app_about: Dental Cleaning and Check-up
7	dID: D001

Test Scenario	Verify inputs of Appointment Form,then make a new appointment and select the mode for the consultation

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click "Make New Appointment" option	A new appointment form is displayed		
2	Fill in the Appointment Form	Doesn't fill in the form completely		
3	Click submit	An error message is displayed		

Table 7.3.1d: TC003_01_04 - <Alternate Scenario 2 of Make Appointment(SD004)>

(e) TC003_01_05: Test <exceptional scenario 2 of Make Appointment (SD004)

Test Case ID	TC003_01_05	Test Case Description	Test the exceptional flow of Altering Appointment and Consultation		
Created by	Yeo Chun Teck	Reviewed By	Ng Kai Zheng	Version	4.0
QA Tester's Log					
Tester's Name	Yeo Chun Teck	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User is logged into the system as patient
3	Existence of valid patient IDs in the database
4	The patient intends to make a new appointment

S#	Test Data
1	pID: P012
2	appID: A035
3	app_name: First Consultation with Dr. Smith
4	app_date:
5	app_time: 10:00AM
6	app_about: Dental Cleaning and Check-up
7	dID: D001

Test Scenario	Verify inputs of Appointment Form,then make a new appointment and select the mode for the consultation

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click "Alter Appointment" option	A new appointment form is displayed		
2	Fill in the Appointment Form	Doesn't fill in the form completely		
3	Click submit	An error message is displayed		

Table 7.3.1e: TC003_01_05 - <Exceptional Scenario 2 of Make Appointment(SD004)>

7.4 TC004: Test <Diagnosis> Subsystem: <Diagnosis (UC004)>

This test contains the following test cases:

(a) TC004_01: Test <Scenario of Diagnosis (SD010)>

7.4.1 TC004_01: Test <Scenario of Diagnosis(SD005)>

This test contains the following scenarios:

(a) TC004_01_01: Test <normal scenario of Diagnosis (SD005)>

Test Case ID	TC004_01_01	Test Case Description	Test normal flow of Diagnosis		
Created by	Ng Kai Zheng	Reviewed By		Version	4.0
QA Tester's Log					
Tester's Name	Ng Kai Zheng	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User doctor is logged in to the system
3	Existence of valid patient IDs in the database
4	

S#	Test Data
1	pID: P012 pre_medication: Paracetamol 500mg (3 times a day) pre_diagnosis: Fever
2	pID: P035 pre_medication: Omeprazole 20mg (Once daily before breakfast) pre_diagnosis: Acid Reflux
3	pID: P068 pre_medication: Simvastatin 20mg (Once daily at bedtime) pre_diagnosis: Hypercholesterolemia
4	

Test Scenario	Verify input patient ID and prescription medication, finally update prescribe into Prescription database

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Enter valid patient ID	ID can be entered		
2	Click submit	Patient Information is displayed		
3	Prescribe Medication (Enter medication, dosage and frequency)	Can enter prescription medication		
4	Click submit	An updated successfully message is displayed		

Table 7.4.1a: TC004_01_01 - <Normal Scenario of Diagnosis(SD005)>

(b) TC004_01_02: Test <alternate scenario 1 of Diagnosis (SD005)>

Test Case ID	TC004_01_02	Test Case Description	Test alternative flow of Diagnosis		
Created by	Ng Kai Zheng	Reviewed By		Version	4.0
QA Tester's Log					
Tester's Name	Ng Kai Zheng	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User doctor is logged in to the system
3	Existence of valid patient IDs in the database
4	

S#	Test Data
1	pID: P012 pre_medication: Paracetamol 500mg (3 times a day) pre_diagnosis: Fever
2	pID: P035 pre_medication: Omeprazole 20mg (Once daily before breakfast) pre_diagnosis: Acid Reflux
3	pID: P068 pre_medication: Simvastatin 20mg (Once daily at bedtime) pre_diagnosis: Hypercholesterolemia
4	

Test Scenario	Verify input patient ID and prescription medication, hence update prescribe into Prescription database, additionally request and upload lab test result

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Enter valid patient ID	ID can be entered		
2	Click submit	Patient Information is displayed		
3	Prescribe Medication (Enter medication, dosage and frequency)	Can enter prescription medication		
4	Click submit	An updated successfully message is displayed		
5	Click request lab test result	Patient received request lab test message		

Table 7.4.1b: TC004_01_02 - <Alternate Scenario 1 of Diagnosis(SD005)>

(c) TC004_01_03: Test <alternate scenario 2 of Diagnosis (SD005)>

Test Case ID	TC004_01_03	Test Case Description	Test alternative flow 2 of Diagnosis		
Created by	Ng Kai Zheng	Reviewed By		Version	4.0
QA Tester's Log					
Tester's Name	Ng Kai Zheng	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User patient is logged in to the system
3	
4	

S#	Test Data
1	labTest: "patient035_bloodtest.pdf"
2	labTest: "patient036_mri.jpg"
3	labTest: "patient035_xray.png"
4	

Test Scenario	Successfully upload and store lab test into database

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	upload lab test result	The lab test result file is successfully uploaded and stored in the system, hence successfully upload message is displayed		

Table 7.4.1c: TC004_01_03 - <Alternate Scenario 2 of Diagnosis(SD005)>

(d) TC004_01_04: Test <exception scenario of Diagnosis (SD005)>

Test Case ID	TC004_01_04	Test Case Description	Test exception flow of Diagnosis		
Created by	Ng Kai Zheng	Reviewed By		Version	4.0
QA Tester's Log					
Tester's Name	Ng Kai Zheng	Date Tested	14-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:
1	Access to Chrome Browser
2	User doctor is logged in to the system
3	Patient ID does not exist in the database
4	

S#	Test Data
1	pID: P012
2	pID: P035
3	pID: P068
4	

Test Scenario	Fail to verify on search valid patient id, displayed an error message

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Enter patient ID	ID can be entered		
2	Click submit	Displayed an error message		

Table 7.4.1d: TC004_01_04 - <Exception Scenario of Diagnosis(SD005)>

7.5 TC005: Test <Prescription> Subsystem: <Manage Prescription (UC005)>

This test contains the following test cases:

(a) TC005_01: Test <Scenario of Manage Prescription (SD006)>

7.5.1 TC005_01: Test <Scenario of Manage Prescription (SD006)>

This test contains the following scenarios:

(a) TC005_01_01: Test <normal scenario of Manage Prescription (SD006)>

Test Case ID	TC005_01_01	Test Case Description	Test normal flow of manage prescription		
Created by	Lew Chin Hong	Reviewed By	Ng Kai Zheng	Version	1
QA Tester's Log					
Tester's Name	Lew Chin Hong	Date Tested	15-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:	S#	Test Data
1	Access to Chrome Browser	1	tracking no:T015 del_date: 1 April 2023 del_address:42 Maple Avenue, Oakville del_status: delivering
2	User pharmacist has logged in to the system	2	tracking no:T019 del_date: 4 April 2023 del_address:15 Elm Street, Riverside del_status: delivering
3		3	tracking no:T020 del_date: 16 April 2023 del_address:27 Pine Street, Meadowville del_status: delivering
4		4	del_status: Complete

Test Scenario	Verify input patient ID and delivery information

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Clicks "Prescription"	Prescription lists is displayed		
2	Clicks one of the prescription from the list	Selected prescription's details is displayed		
3	Enter tracking number, delivery address and delivery status	Tracking number, delivery address and delivery status is entered		
4	Click "submit"	Tracking number, delivery address and delivery status is updated		
5	Click "Complete"	delivery status is updated to "Complete"		

(a) TC005_01_02: Test <exception scenario of Manage Prescription (SD006)>

Test Case ID	TC005_01_02	Test Case Description	Test exception flow of manage prescription		
Created by	Lew Chin Hong	Reviewed By	Ng Kai Zheng	Version	1
QA Tester's Log					
Tester's Name	Lew Chin Hong	Date Tested	15-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:	S#	Test Data
1	Access to Chrome Browser	1	del_status: Complete
2	User pharmacist has logged in to the system		
3	Tracking number, delivery address and delivery status is updated		
4			

Test Scenario	update delivery status to "Complete" when delivery has not been done

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Clicks "Prescription"	Prescription lists is displayed		
2	Clicks one of the prescription from the list	Selected prescription's details is displayed		
3	Click "Complete"	An error message is displayed		

7.6 TC006: Test <Prescription> Subsystem: <Track Parcel (UC006)>

This test contains the following test cases:

(a) TC006_01: Test <Scenario of Track Diagnosis(SD007)>

7.6.1 TC006_01: Test <Scenario of Track Diagnosis(SD007)>

This test contains the following scenarios:

(a) TC006_01_01: Test <normal scenario of Track Diagnosis(SD007)>

Test Case ID	TC006_01_01	Test Case Description	Test normal flow of track parcel		
Created by	Lew Chin Hong	Reviewed By	Ng Kai Zheng	Version	1
QA Tester's Log					
Tester's Name	Lew Chin Hong	Date Tested	15-June-2023	Test case(Pass/Fail/...)	

S#	Prerequisties:	S#	Test Data
1	Access to Chrome Browser	1	
2	User patient has logged in to the system		
3			
4			

Test Scenario	update delivery status to "Complete" when delivery has not been done

Step#	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Clicks "Track Parcel"	User redirected to an external web page		