**A blue and white logo

Description automatically generatedCOMSATS University Islamabad,**

**Abbottabad, Pakistan**

**FurnishARt (An AR-Based Furniture Store)**

(Version 1.0)

***By***

**Muhammad Umer CIIT/SP21-BCS-016/ATD**

**Muhammad Noman CIIT/SP21-BCS-014/ATD**

**Hafiz Talha Nazir CIIT/SP21-BCS-007/ATD**

***Supervisor***

**Ma’am Bushra Mushtaq**

***Bachelor of Science in Computer Science (2021-2025)***

**The candidate confirms that the work submitted is their own and appropriate  
 credit has been given where reference has been made to the work of others**.

**A blue and white logo

Description automatically generated COMSATS University Islamabad, Pakistan**

**FurnishARt (An AR-Based Furniture Store)**

**A project presented to**

**COMSATS University Islamabad, Abbottabad**

**In partial fulfillment**

**of the requirement for the degree of**

***Bachelor of Science in Computer Science (2021-2025)***

***By***

**Muhammad Umer CIIT/SP21-BCS-016/ATD**

**Muhammad Noman CIIT/SP21-BCS-014/ATD**

**Hafiz Talha Nazir CIIT/SP21-BCS-007/ATD**

**DECLARATION**

We hereby declare that no part of or all this software has been taken from any source. It is also stated that we created this software and associated report fully through our own efforts. If any component of this project turns out to have been copied from another source or to be an exact copy of others. We will not back down from the consequences. No part of the work presented has been submitted in support of any other degree or accreditation available at this or any other university or institute of research.

Muhammad Umer Muhammad Noman Hafiz Talha Nazir

------------------------ ------------------------------ ------------------------

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (CS (Computer Science)) “FurnishARt (An AR-Based Furniture Store)” was Developed by **Muhammad Umer (CIIT/SP21-BCS-016), Muhammad Noman (****CIIT/SP21-BCS-014)** and **Hafiz Talha Nazir (CIIT/SP21-BCS-007)** under the supervision of “**Ma’am Bushra Mushtaq**” and that in their opinion; it is fully sufficient, in the scope and quality for the degree of Bachelors of Science in Computer Science.

---------------------------------------

**Ma’am Bushra Mushtaq**

**Supervisor**

---------------------------------------

**External Examiner**

---------------------------------------

**Dr. Iftikhar Ali**

**Head of Department**

**(Department of Computer Science)**

**EXECUTIVE SUMMARY**

The shift toward digital shopping has created demand for innovative solutions that seamlessly blend online and offline experiences, particularly in the furniture sector, where customers benefit significantly from visualizing products in their real-world spaces before making a purchase. Traditional e-commerce platforms often fall short in providing this immersive experience, leading to uncertainty in purchasing decisions and increased return rates. To address these challenges, furnishARt introduces an Augmented Reality (AR)-based platform that enables customers to view and interact with 3D furniture models both on the web and through an AR-capable mobile application.

furnishARt provides users with the ability to preview interactive 3D models of furniture on a web application. For an immersive AR experience, users can select the "AR preview" option on the web, which will redirect them to the mobile app, allowing them to place and view models in their own environment through their device’s camera. This integrated flow offers customers a clear sense of furniture dimensions, style, and fit within their space, helping them make informed purchasing decisions without needing an in-store visit.

The furnishARt system features modules such as 3D model management, AR preview capabilities, user registration and authentication, furniture catalog management, customer reviews, and secure payment processing. An admin panel allows for efficient management of furniture listings and reviews, while users benefit from a secure and user-friendly shopping experience. The project is being developed using an Agile methodology to support iterative progress and rapid response to feedback, and object-oriented principles for modularity and maintainability.

With furnishARt, furniture retailers can offer a unique and engaging shopping experience, giving them a competitive edge while enabling customers to make confident, personalized choices directly from their own homes.

**ACKNOWLEDGEMENT**

All praise belongs to Almighty Allah, who enabled us to complete this difficult activity by giving us a tiny fraction of His limitless wisdom.

We are grateful to our project supervisor, "**Ma’am Bushra Mushtaq**". The accomplishment of this project would have been uncertain without their personal supervision, advice, and important guidance. We are grateful to them for their support and encouragement throughout this process.

We are also grateful to our parents and loved ones for their constant encouragement and for teaching us the values of honesty and respect.

Muhammad Umer Muhammad Noman Hafiz Talha Nazir

------------------------ ------------------------------ ------------------------

**ABBREVIATIONS**

|  |  |
| --- | --- |
| **AR** | Augmented Reality |
|  |  |

Table of Contents

[1. Introduction 1](#_Toc181204904)

[1.1. Brief 1](#_Toc181204905)

[1.2. Relevance to Course Module 1](#_Toc181204906)

[1.3. Project Background 1](#_Toc181204907)

[1.4. Literature Review 1](#_Toc181204908)

[1.5. Analysis from Literature Review 1](#_Toc181204909)

[1.6. Methodology and Software Life Cycle 1](#_Toc181204910)

[1.6.1. Rational behind Selected Methodology 1](#_Toc181204911)

[2. Problem Statement 1](#_Toc181204912)

[1.7. Problem Statement 1](#_Toc181204913)

[1.8. Deliverable and Development Requirements 1](#_Toc181204914)

[3. Requirement Analysis 1](#_Toc181204915)

[1.9. Use Case Diagram(s) 1](#_Toc181204916)

[1.10. Detailed Use Case 1](#_Toc181204917)

[1.10.1. Registration 1](#_Toc181204918)

[1.10.2. Login 1](#_Toc181204919)

[1.10.3. Scan Prescription 1](#_Toc181204920)

[1.10.4. Edit Profile 1](#_Toc181204921)

[1.10.5. Search Pharmacy 1](#_Toc181204922)

[1.10.6. Search Doctor 1](#_Toc181204923)

[1.10.7. Manage Users 1](#_Toc181204924)

[1.10.8. Manage Doctors 1](#_Toc181204925)

[1.10.9. Logout 1](#_Toc181204926)

[1.11. Functional Requirements 1](#_Toc181204927)

[1.12. Non-Functional Requirements 1](#_Toc181204928)

[1.12.1. Security: 1](#_Toc181204929)

[1.12.2. Usability 1](#_Toc181204930)

[1.12.3. Performance 1](#_Toc181204931)

[1.12.4. Security 1](#_Toc181204932)

[1.12.5. Reliability 1](#_Toc181204933)

[1.12.6. Scalability 1](#_Toc181204934)

[4. Design and Architecture 1](#_Toc181204935)

[1.13. System Architecture 1](#_Toc181204936)

[1.14. Data Representation [Class Diagram] 1](#_Toc181204937)

[1.15. Process Flow/ Representation 1](#_Toc181204938)

[1.15.1. Patient Activity Diagram 1](#_Toc181204939)

[1.15.2. Admin Activity Diagram 1](#_Toc181204940)

[1.15.3. Firebase Tree Structure 1](#_Toc181204941)

[1.16. Design Model 1](#_Toc181204942)

[1.16.1. Sequence Diagram 1](#_Toc181204943)

[1.16.2. System Sequence Diagram (s) [User] 1](#_Toc181204944)

[1.16.3. System Sequence Diagram (s) [Admin] 1](#_Toc181204945)

[1.16.4. Package Diagram 1](#_Toc181204946)

[1.16.5. Deployment Diagram 1](#_Toc181204947)

[5. Requirement Analysis 1](#_Toc181204948)

[1.17. Algorithm 1](#_Toc181204949)

[1.17.1. User Account Management 1](#_Toc181204950)

[1.17.2. Pharmacy Locator 1](#_Toc181204951)

[1.17.3. Doctor Search 1](#_Toc181204952)

[1.17.4. Image Recognition & Translation Module 1](#_Toc181204953)

[1.17.5. Admin Notification System 1](#_Toc181204954)

[1.18. External APIs 1](#_Toc181204955)

[1.19. User Interfaces 1](#_Toc181204956)

[6. Testing & Evaluation 1](#_Toc181204957)

[1.20. Manual Testing 1](#_Toc181204958)

[1.20.1. System Testing 1](#_Toc181204959)

[1.20.2. Unit Testing 1](#_Toc181204960)

[1.20.3. Functional Testing 1](#_Toc181204961)

[1.20.4. Integration Testing 1](#_Toc181204962)

[7. Conclusion & Future Work 1](#_Toc181204963)

[1.21. Conclusion 1](#_Toc181204964)

[1.22. Future Work 1](#_Toc181204965)

[2. References 1](#_Toc181204966)

**LIST OF FIGURES**

[Fig 3. 1: Use Case Diagram 9](file:///F:\Final%20Year%20Project\EXTERNAL%20AND%20FINAL%20THESIS\AI-Based%20SPC%20Thesis.docx#_Toc156394275)

**LIST OF Tables**

[Table 1. 1: Literature Review 3](#_Toc156394368)

# Introduction

With the rapid advancement of augmented reality (AR) technology, the retail industry, especially e-commerce, has gained a powerful tool to elevate the online shopping experience. For furniture retail, AR brings a unique solution to a longstanding challenge—how to bridge the gap between online shopping and the in-person experience of seeing furniture in context.

furnishARt is designed to address this challenge by providing customers with an immersive AR shopping experience. Through the furnishARt web application, users can view interactive 3D models of furniture, exploring details and design from various angles. When ready for an AR preview, users can select the option to be redirected to the furnishARt mobile app, where they can view these models in their real environment, adjusting placement and angles to ensure the perfect fit.

This approach allows customers to make well-informed purchasing decisions, blending online convenience with a tactile sense of the product's presence in their homes. By incorporating an intuitive AR flow and easy-to-navigate 3D previews, furnishARt aims to revolutionize online furniture shopping, giving retailers a competitive edge and creating a more satisfying, user-centered shopping experience.

## Brief

furnishARt is an innovative AR-based online furniture store that allows customers to visualize furniture pieces in their physical space. Through an interactive web application and an AR-enabled mobile app, furnishARt empowers users to view 3D furniture models online and seamlessly transition to augmented reality for a true-to-life preview. By combining the convenience of online shopping with the immersive potential of AR, furnishARt enhances customer confidence in purchasing furniture online, reducing uncertainties and improving satisfaction.

## Relevance to Course Module

This system is built upon the foundation set up by many of the course modules we studied throughout the four-year program of BSCS. The courses modules that served as the building blocks for the project are:

**Software Engineering Concept**: We kept in mind the concepts we learned in software engineering Concept about how the project lifecycle progresses, which process model is used for what kind of project etc.

**Human Computer Interaction:** We kept in mind the major principles of HCI so that the interface is more user friendly and understandable by the end user.

**Software Project Management:** in this course we studied how to manage time, resources effectively and efficiently. How to make schedule for your project. Planning is also important we studied how to plan the development process and make the best use of available resources. Through this course we learn how to use the available tools for the planning like making Gantt chart on project plan.

**Web Technologies:** This project integrates essential web development principles by leveraging React.js, Next.js, and Three.js for creating interactive and responsive user interfaces that display 3D models online. RESTful APIs facilitate seamless data flow between the frontend and backend, aligning with the Web Technologies course focus on building dynamic, user-centric applications.

**Database Systems I:** With SQL database integration, this project applies core concepts from Database Systems I, including relational database design, data integrity, and transaction security. SQL stores and manages essential data such as user profiles, furniture details, and transactions, emphasizing efficient data retrieval and reliable storage. This supports high-volume operations typical in e-commerce platforms, directly tying into course topics on data management and security.

## Project Background

As online shopping grows, the need for enhanced customer experience is critical, especially in sectors like furniture retail where visualizing products in physical spaces plays a significant role in purchase decisions. Traditional e-commerce methods often fall short, offering only static images and leaving customers unsure of product fit and style within their own homes.

To address these limitations, augmented reality (AR) has emerged as a solution, allowing customers to view virtual furniture models in real-world environments. Leveraging AR, this project combines interactive 3D model viewing on a web platform with the option to preview furniture in AR via a mobile app, providing customers a highly immersive and realistic shopping experience. By enhancing the online shopping process with AR technology, this approach aims to bridge the gap between online and in-person shopping, ultimately leading to more informed purchasing decisions and customer satisfaction.

## Literature Review

Table 1. 1: Literature Review

|  |  |  |
| --- | --- | --- |
| **Application Name** | **Weakness** | **Proposed Project Solution** |
| **VSurface** | It lacks AR previews for furniture items, restricting its functionality to rugs only. | This system provides AR previews for a broader range of furniture items. |
| **IKEA Place** | The app's interface is complex or difficult to the end user – and IKEA is not available in Pakistan. | Our system will provide user friendly interface and this system will be launching in Pakistan. |
| **Wayfair Spaces** | Their Website is overwhelming and cluttered at times, with vast variety of products, categories, and subcategories to choose from. The website can be slow and laggy at times and Wayfair is not available in Pakistan. | We propose simplifying product categorization, optimizing the website for faster loading times, and refining subcategory structures. |

## Analysis from Literature Review

The literature review highlights the limitations and gaps in existing AR-based furniture and decor applications, particularly in usability, accessibility, and product range:

1. **Limited AR Functionality**  
   Applications like VSurface offer AR previews only for specific products, such as rugs, limiting their overall utility. Our platform will address this by providing a full range of AR-enabled furniture items, expanding the shopping experience.
2. **User Interface Complexity**  
   The IKEA Place app has a complex interface that may overwhelm users, and the service is not available in Pakistan. By focusing on a user-friendly design and accessibility in Pakistan, our system will offer a streamlined experience for a broader audience.
3. **Website Performance and Navigation**  
   Wayfair Spaces, while comprehensive, faces challenges with cluttered categories, subcategories, and loading speeds. We aim to overcome these issues by optimizing the website for faster performance, simplifying product categories, and ensuring efficient navigation for a smoother user experience.

## Methodology and Software Life Cycle

For the development of the project, an Agile software development methodology has been selected due to its emphasis on iterative progress and flexibility. Agile allows teams to adapt to changes quickly, making it particularly suitable for projects involving emerging technologies like augmented reality (AR). The Agile approach promotes collaboration among cross-functional teams and enables continuous feedback from stakeholders, ensuring that the final product aligns with user expectations and business goals.

The software life cycle in this project will consist of several iterative cycles, including planning, designing, developing, testing, and deploying the application. Each cycle will focus on delivering incremental improvements and features, allowing for regular updates and adjustments based on user feedback and testing results. This iterative nature supports rapid prototyping and encourages experimentation, which is vital for integrating AR functionality effectively into the online furniture store.

### Rational behind Selected Methodology

The Agile methodology was chosen for several reasons:

1. **Flexibility:** Agile allows for adjustments throughout the development process, accommodating changes in user requirements or market conditions. This adaptability is crucial in the fast-evolving field of AR technology.
2. **Customer-Centric Focus:** By prioritizing customer feedback and involving users in the development process, Agile ensures that the final product delivers maximum value and meets the actual needs of users. This focus on customer satisfaction is vital in the competitive e-commerce landscape.
3. **Iterative Development:** The Agile framework promotes incremental progress through sprints, enabling the team to release functional components of the application regularly. This approach allows for early detection of issues and continuous improvement based on real user interactions.
4. **Enhanced Collaboration:** Agile encourages collaboration among developers, designers, and stakeholders, fostering a culture of communication and shared ownership of the project. This collaborative environment enhances creativity and innovation, essential for integrating AR features.

# Problem Statement

## Problem Statement

Medical receipts contain important information about a patient's diagnosis, treatment plans, and medication details. However, patients often struggle to understand the complex medical terminologies used in these receipts, which can result in confusion and hinder their ability to make informed decisions about their healthcare. This can lead to medication non-adherence, medical errors, and poor healthcare outcomes. In addition, patients may have difficulty finding nearby pharmacies to purchase their medications or keeping track of their daily dosage routine. Moreover, patients may face difficulty in finding qualified doctors according to their disease type if the recommended prescription is not working in their favor. This can lead to delayed treatment and a decline in health outcomes. Additionally, patients may struggle with keeping track of their medication expenses, leading to unnecessary financial burdens. The proposed AI-based Smart Patient Companion aims to address these challenges by providing a comprehensive solution that enables patients to easily manage their healthcare and make informed decisions about their treatment plans.

## Deliverable and Development Requirements

The deliverables of the AI-based Smart Patient Companion include the following:

* A user-friendly mobile application for android smartphones.
* Implementation of an image recognition module to identify medical terms on uploaded receipts and prescriptions.
* Integration of natural language processing (NLP) techniques to translate complex medical jargon into plain and understandable language.
* Development of features for patients to record and manage their daily dosage routines.
* Implementation of medication reminders to ensure timely medication intake.
* Integration of a feature allowing users to find nearby pharmacies to purchase their prescribed medications.
* Development of a doctor search functionality based on disease type or user preferences.
* The system should provide doctor recommendations in case the prescribed treatment is ineffective.

The development requirements of the proposed AI-based Smart Patient Companion include the following:

* Proficiency in developing mobile applications for Android Platforms.
* Expertise in image recognition technology and natural language processing for medical terminology translation.
* Ability to create features for medication tracking, reminders, and expense management.
* Knowledge of database systems to store user data securely.
* Skills in designing a user-friendly and visually appealing interface.
* Experience in deploying applications to app stores and managing the submission process.

Table 2. 1: Development Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **Tools**  **And**  **Technologies** | **Tools** | **Version** | **Rationale** |
| Android Studio | 2021.3.1 | IDE |
| Flutter SDK | 3.3.4 | UI toolkit |
| Firebase |  | Real-time DBMS |
| Star UML | 3.2.2 | System Modeling |
| MS Word | Office 365 | Documentation |
| MS Power Point | Office 365 | Presentation |
| Canva | 1.62.0.0 | Mockups Creation |
| **Technology** | **Version** | **Rationale** |
| Dart language, Python | 2.1.0, 3.11.2 | Programming language |
| UML | 2.5 | Modeling Language |
| Flutter | 3.3.4 | Mobile Application Development |

# Requirement Analysis

Following parts of the SRS are included:

## Use Case Diagram(s)

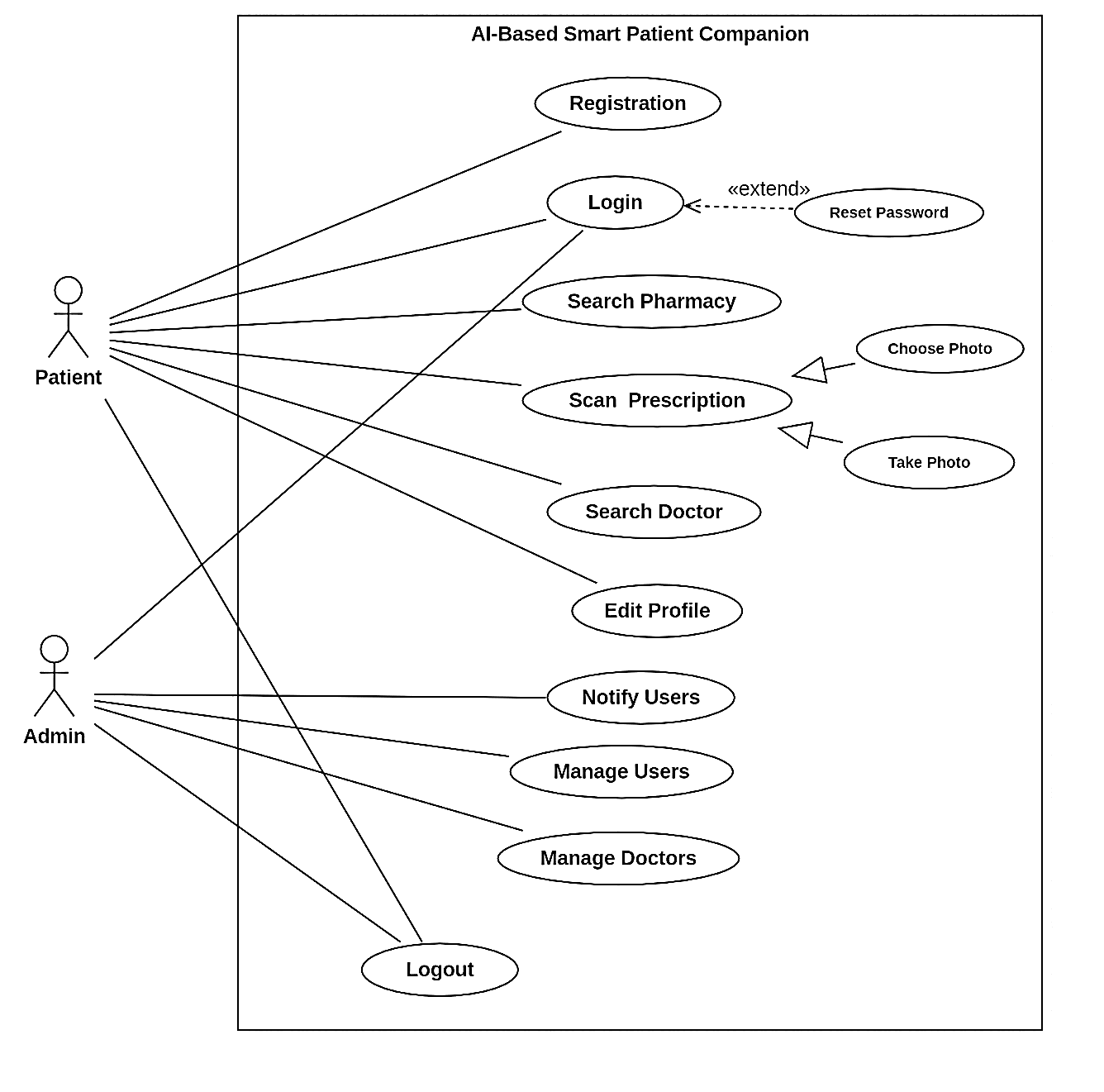


Fig 3. 1: Use Case Diagram

A diagram of a relationship between a group of individuals

Description automatically generated**Legends**

## Detailed Use Case

Detailed use cases of project provide a comprehensive description of how users interact with sytem.

### Registration

Table 3. 1: UC-1 Register Account

|  |  |
| --- | --- |
| **Use Case ID:** | UC-1 |
| **Use Case Name:** | Registration |
| **Actors:** | User (Patient) |
| **Description:** | User registers for the AI-Based Smart Patient Companion application by providing the required information in the sign-up form. |
| **Trigger:** | A user clicks the Sign-Up link and indicates a desire to create a new account. |
| **Preconditions:** | PRE-1. User has installed AI-Based Smart Patient Companion application on their android device. |
| **Postconditions:** | POST-1. User is successfully logged in and navigate to the Dashboard.  POST-2. User is shown all his medicines. |
| **Normal Flow:** | 1. User Opens the application.  2. The user is prompted to provide the following information in the sign-up form First Name , Last Name, Email Address, Password, Age, Gender.  3. The user fills in the required information accurately.  4. The user submits the sign-up form.  5. The system validates the provided information.  6. The user can now log in to the application using the registered email and password. |
| **Alternative Flows:** | [Alternative Flow 1 – Existing User  1.1 If the user tries to register with an email that is already associated with an existing account, they are prompted to log in with their existing credentials or choose a different email for registration.]  [Alternative Flow 2 – Google Signup  2.1 Users have the option to sign up using their social media accounts, such as Google or Facebook. The application redirects them to the chosen social media platform's authentication page, verifies their credentials, and creates a new account using the received information.] |
| **Exceptions:** | E-1. Invalid or missing information entered in the sign-up form.  1. The system displays an error message indicating the issue.  2. The user is prompted to correct the invalid or missing information. |
| **Special Requirements (NFRs)** | NFR-1: User credentials must be securely stored and transmitted.  NFR-2: The registration form should have a user-friendly interface that is easy to understand and navigate. |

### Login

Table 3. 2: UC-2 Login

|  |  |
| --- | --- |
| **Use Case ID:** | UC-2 |
| **Use Case Name:** | Login |
| **Actors:** | User (Patient), Admin |
| **Description:** | A user accesses the system by providing their credentials in the login form. |
| **Trigger:** | A user clicks the Login button and indicates to enter the system. |
| **Preconditions:** | PRE-1. User has installed AI-Based Smart Patient Companion application on their android device.  PRE-2. User is already registered to access the application and its features. |
| **Postconditions:** | POST-1. The user is successfully registered and can log in to the application using their credentials in login form. |
| **Normal Flow:** | 1. User Opens the application.  2. User is prompted to enter Email and Password  3. The User successfully accesses his/her account after entering correct Email and Password. |
| **Alternative Flows:** | [Alternative Flow 1 – New User Sign In  1.1 New User login credentials  1.2 Application requests to create new account using Sign Up form.  1.3 After User is successfully registered.  1.4 Return to step 2 of normal flow.]  [Alternative Flow 2 – Forget User Password  2.1 User is denied access after entering wrong email or password.  2.2 Forget your password option is used to reset User’s password.  2.3 New passwords will be entered upon using the reset password link in the email.  2.4 Return to step 2 of normal flow.] |
| **Exceptions:** | E-1. Wrong Credentials entered in the given fields.  1. The System will display a warning dialogue box.  2. If User enters the right details, system will login, then continue with step 2 of normal flow.  3. Else if the user will be shown with a warning message. |
| **Special Requirements (NFRs)** | NFR-1: Implement measures to protect against unauthorized access attempts.  NFR-2: Display clear and concise error messages for incorrect login attempts.  NFR-3: The login functionality should be available and accessible to users consistently. |

### Scan Prescription

Table 3. 3: UC-3 Scan Prescription.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-3 |
| **Use Case Name:** | Scan Prescription |
| **Actors:** | User (Patient) |
| **Description:** | The user scans a prescription using the AI-Based Smart Patient Companion application to store the prescription information. |
| **Trigger:** | The user clicks the camera button to scan prescription or choose a prescription from gallery. |
| **Preconditions:** | PRE-1. The user has installed the AI-Based Smart Patient Companion application on their Android device.  PRE-2. The user is logged into the application. |
| **Postconditions:** | POST-1. The prescription is successfully scanned and stored in the user's account. |
| **Normal Flow:** | 1. User Opens the application.  2. The user navigates to the "Scan Prescription" section.  3. The user activates the device camera or selects an existing image from their gallery.  4. The application captures or selects the image of the prescription.  5. The application processes the scanned image to extract relevant information.  6. The extracted prescription information is displayed to the user for verification.  7. The user reviews the scanned prescription details and confirms their accuracy.  8. The application stores the scanned prescription in the user's account, associating it with their profile. |
| **Alternative Flows:** | [Alternative Flow 1 – New User Sign In  1.1 If the user determines that the scanned image is not clear or legible, they have the option to retake the image by activating the camera again. The process of capturing a new image and verifying the scanned prescription details is repeated before storing the prescription in the user's account.] |
| **Exceptions:** | E-1. Scanning issues or incorrect extraction of prescription details. |
| **Special Requirements (NFRs)** | NFR-1: The application should have a high accuracy rate in extracting relevant information from scanned prescriptions.  NFR-2: Implement advanced image processing algorithms to enhance the accuracy of prescription extraction.  NFR-3: The scanning process should be intuitive and user-friendly, guiding the user to capture clear and legible images of prescriptions. |

### Edit Profile

|  |  |
| --- | --- |
| **Use Case ID:** | UC-4 |
| **Use Case Name:** | Edit Profile |
| **Actors:** | User (Patient) |
| **Description:** | The user manages their profile information in the AI-Based Smart Patient Companion application, including updating personal details, preferences, and profile picture. |
| **Trigger:** | The user accesses the profile management section. |
| **Preconditions:** | PRE-1. The user has installed the AI-Based Smart Patient Companion application on their Android device.  PRE-2. The user is logged into the application. |
| **Postconditions:** | POST-1. The user's profile information is successfully updated and saved in the application. |
| **Normal Flow:** | 1. The user opens the application.  2. The user navigates to the "Profile" section.  3. The application displays the user's current profile information.  4. The user modifies the desired fields.  5. The user submits the updated profile information.  6. The application validates the modified fields |
| **Alternative Flows:** | [Alternative Flow 1 – Cancel Profile Update  1.1 Before submitting the updated profile information, the user decides to cancel the update and discards the modifications.] |
| **Exceptions:** | E-1. Invalid or missing information entered in the profile fields.  1. The system displays an error message indicating the issue.  2. The user is prompted to correct the invalid or missing information. |
| **Special Requirements (NFRs)** | NFR-1: The application should ensure the confidentiality and integrity of user profile information.  NFR-2: The system should efficiently handle user profile updates, ensuring minimal delays in processing and saving changes.  NFR-3: The profile management section should have a user-friendly interface, making it easy for users to navigate and update their profile  information. |

Table 3. 4: UC-5 Edit Profile

.

### Search Pharmacy

Table 3. 5: UC-5 Search Pharmacy

|  |  |
| --- | --- |
| **Use Case ID:** | UC-5 |
| **Use Case Name:** | Search Pharmacy |
| **Actors:** | User (Patient) |
| **Description:** | The user searches for nearby pharmacies using the AI-Based Smart Patient Companion application to locate nearby pharmacies. |
| **Trigger:** | The user accesses the Pharmacy Search section and select nearby pharmacy as per their location. |
| **Preconditions:** | PRE-1. The user has installed the AI-Based Smart Patient Companion application on their Android device.  PRE-2. The user is logged into the application.  PRE-3. The locations services at user android device are on. |
| **Postconditions:** | POST-1. The user is presented with a location of nearby pharmacies based on their current locations.  POST-2. The user can see the distance of all the nearby pharmacies. |
| **Normal Flow:** | 1. The user opens the application.  2. The user navigates to the "Pharmacy Search" section.  3. The application displays all the nearby pharmacies.  4. The user can select the specific pharmacy.  5. The system provide the distance and route direction to the user. |
| **Alternative Flows:** | None |
| **Exceptions:** | E-1. No Pharmacies found nearby the user. |
| **Special Requirements (NFRs)** | NFR-1: The application should provide accurate location of nearby pharmacies.  NFR-2: The pharmacy search functionality should have fast response times, allowing users to quickly retrieve information about nearby pharmacies.  NFR-3: The application should seamlessly integrate with map services or APIs to provide accurate route directions to nearby pharmacies. |

### Search Doctor

Table 3. 6: UC-6 Search Doctor

|  |  |
| --- | --- |
| **Use Case ID:** | UC-6 |
| **Use Case Name:** | Search Doctor |
| **Actors:** | User (Patient) |
| **Description:** | The user searches for doctors using the AI-Based Smart Patient Companion application to find healthcare professionals based on their specific requirements. |
| **Trigger:** | The user initiates a search for doctors |
| **Preconditions:** | PRE-1. The user has installed the AI-Based Smart Patient Companion application on their Android device.  PRE-2. The user is logged into the application. |
| **Postconditions:** | POST-1. The user is presented with a list of doctors based on their search criteria.  POST-2. The user can access detailed information about the selected doctor, including their profile, expertise, availability, and contact details. |
| **Normal Flow:** | 1. The user opens the application.  2. The user navigates to the "Search Doctor" section.  3. The user enters their search criteria.  4. The user submits the search query.  5. The application processes the search query and retrieves a list of doctors based on the provided criteria.  6. The user can select a specific doctor from the list. |
| **Alternative Flows:** | None |
| **Exceptions:** | E-1. No doctors found based on the search criteria. |
| **Special Requirements (NFRs)** | NFR-1: The system should utilize reliable data sources and employ effective search algorithms to minimize irrelevant or inaccurate search results.  NFR-2: The doctor search functionality should have fast response times, allowing users to quickly retrieve a list of doctors based on their search criteria. |

### Manage Users

Table 3. 7: UC-7 Manage Users

|  |  |
| --- | --- |
| **Use Case ID:** | UC-7 |
| **Use Case Name:** | Manage Users. |
| **Actors:** | Admin |
| **Description:** | The admin user manages the users of the AI-Based Smart Patient Companion application by performing various actions such as creating new user accounts, updating user information, or deactivating user accounts. The user management is handled through the Firebase database. |
| **Trigger:** | The admin user indicates a desire to manage the application users. |
| **Preconditions:** | PRE-1. The admin has the necessary privileges and is logged in to the AI- Based Smart Patient Companion admin module.  PRE-2. The admin panel is connected to firebase database. |
| **Postconditions:** | POST-1. The user management actions, such as creating, updating, or deactivating user accounts, are successfully performed, and reflected. |
| **Normal Flow:** | 1. The admin logged into the admin panel.  2. The admin user selects the "Manage Users".  3. The admin retrieves the list of users from the Firebase database.  4. The admin user can view the list of users, including relevant details such as:  4.1. User’s Name.  4.2. User’s Email.  4. The admin user can search for a specific user using filters such as name, email.  5. The admin user can perform various actions on the selected user. |
| **Alternative Flows:** | None |
| **Exceptions:** | E-1. Invalid or missing information entered in the news or update form. |
| **Special Requirements (NFRs )** | NFR-1: The user management functionality should enforce access control and authorization mechanisms to ensure that only authorized admin users can perform actions such as creating, updating, or deactivating user accounts.  NFR-2: The user management operations should be performed efficiently,  even with a large number of users in the system. |

### Manage Doctors

Table 3. 8: UC-8 Manage Doctors

|  |  |
| --- | --- |
| **Use Case ID:** | UC-8 |
| **Use Case Name:** | Manage Doctors. |
| **Actors:** | Admin |
| **Description:** | The admin user manages the doctors of the AI-Based Smart Patient Companion application by performing various actions such as adding new doctors, updating doctor information, or removing doctors from the system. |
| **Trigger:** | The admin user initiates the doctor management process by accessing the "Manage Doctors" section within the AI-Based Smart Patient Companion admin panel. |
| **Preconditions:** | PRE-1. The admin has the necessary privileges and is logged in to the AI- Based Smart Patient Companion admin module.  PRE-2. The admin panel is connected to firebase database. |
| **Postconditions:** | POST-1. The doctor management actions, such as adding, updating, or removing doctors, are successfully performed and reflected in the application's doctor database.. |
| **Normal Flow:** | 1. The admin logged into the admin panel.  2. The admin user selects the "Manage Doctors".  3. The admin retrieves the list of doctors from database.  4. The admin user can perform various actions on the selected doctor.  4. The admin user can search for a specific user using filters such as name, email.  5. The admin user can perform various actions on the selected user like add, remove or update doctors details. |
| **Alternative Flows:** | None |
| **Exceptions:** | E-1. Invalid or missing information entered in the news or update form. |
| **Special Requirements (NFRs )** | NFR-1: The doctor management functionality should enforce access control and authorization mechanisms to ensure that only authorized admin users can perform actions such as adding, updating, or removing doctors.  NFR-2: The doctor management actions, such as adding, updating, or removing doctors, should ensure data integrity and consistency |

### Logout

Table 3. 9: UC-9 Logout

|  |  |
| --- | --- |
| **Use Case ID:** | UC-9 |
| **Use Case Name:** | Logout |
| **Actors:** | User (Patient),Admin |
| **Description:** | The user or admin logs out from the AI-Based Smart Patient Companion application/system, ending their current session and ensuring the protection of their account and data. |
| **Trigger:** | The user or admin selects the logout option. |
| **Preconditions:** | PRE-1. The user or admin is currently logged in to the AI-Based Smart Patient Companion system. PRE-2. The admin panel is connected to firebase database. |
| **Postconditions:** | POST-1. The user or admin is successfully logged out, and their session is terminated.  POST-2. The user or admin is redirected to the login screen. |
| **Normal Flow:** | 1. The user or admin locates and selects the "Logout".  2. The admin user selects the "Manage Doctors".  3. The application/system prompts for confirmation to ensure the intentional action.  4. The user or admin confirms their choice to log out.  5. The system terminates the current session and clears any user-specific data stored locally.  6. The user or admin is redirected to the login screen |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Special Requirements (NFRs )** | NFR-1: The system should ensure that all user or admin session data is securely cleared upon logout to prevent unauthorized access.  NFR-2: User and admin credentials should not be stored or cached locally after logout to maintain privacy. |

## Functional Requirements

1. System shall allow patient to register his/herself.
2. Patient shall enter valid username and password for login to system.
3. Patient shall be able to reset their password if forgotten.
4. Patient shall be able to edit their profile.
5. Patient shall be able to scan their medical prescription.
6. Patient shall be able to import their medical prescription from gallery.
7. Patient shall be able to search nearby pharmacies.
8. Patient shall be able to search for qualified doctors.
9. Patient shall be able to set reminder for their dosage.
10. Admin shall be able to add or remove doctor.
11. Admin shall be able to notify users.

Table 3. 10: FR-1 Registration/ Signup

|  |  |
| --- | --- |
| **Identifier** | FR-1 |
| **Title** | Registration/ Signup |
| **Requirement** | The user shall be able to create a new account by providing the required information and completing the signup process. |
| **Source** | User (Patient) |
| **Rationale** | To allow new users to access the application's features and functionalities and personalize their experience. |
| **Business Rule (if required)** | Users must be able to sign up at all hours of the day. |
| **Dependencies** | None |
| **Priority** | High |

Table 3. 11: FR-2 Login

|  |  |
| --- | --- |
| **Identifier** | FR-2 |
| **Title** | Login |
| **Requirement** | The user shall be able to securely log into the AI-Based Smart Patient Companion application using their credentials. |
| **Source** | User (Patient) |
| **Rationale** | To ensure that only authorized users can access their personalized account and utilize the application's features. |
| **Business Rule (if required)** | Users must be able to login at all hours of the day. |
| **Dependencies** | None |
| **Priority** | High |

Table 3. 12: FR-3 Forgot Password

|  |  |
| --- | --- |
| **Identifier** | FR-3 |
| **Title** | Forgot Password |
| **Requirement** | The user shall have the ability to reset their password in case they forget it. |
| **Source** | User (Patient) |
| **Rationale** | To provide a secure and convenient way for users to regain access to their account if they cannot remember their password. |
| **Business Rule (if required)** | None |
| **Dependencies** | None |
| **Priority** | Medium |

Table 3. 13: FR-4 Edit Profile

|  |  |
| --- | --- |
| **Identifier** | FR-4 |
| **Title** | Edit Profile |
| **Requirement** | The user shall be able to view and manage their profile information within the AI-Based Smart Patient Companion application. |
| **Source** | User (Patient) |
| **Rationale** | To allow users to update their personal details, contact information, and preferences as needed, ensuring their profile remains up to date. |
| **Business Rule (if required)** | None |
| **Dependencies** | None |
| **Priority** | Medium |

Table 3. 14: FR-5 Scan Prescription

|  |  |
| --- | --- |
| **Identifier** | FR-5 |
| **Title** | Scan Prescription |
| **Requirement** | The user shall be able to scan and upload images of their medical prescriptions for processing within the AI-Based Smart Patient Companion application. |
| **Source** | User (Patient) |
| **Rationale** | To enable users to conveniently digitize and store their medical prescriptions for easy access and reference. |
| **Business Rule (if required)** | None |
| **Dependencies** | None |
| **Priority** | High |

Table 3. 15: FR-6 Search Pharmacy

|  |  |
| --- | --- |
| **Identifier** | FR-6 |
| **Title** | Search Pharmacy |
| **Requirement** | The user shall be able to search for nearby pharmacies within the AI-Based Smart Patient Companion application. |
| **Source** | User (Patient) |
| **Rationale** | To assist users in finding and locating nearby pharmacies for purchasing medicines and fulfilling their prescription requirements. |
| **Business Rule (if required)** | None |
| **Dependencies** | None |
| **Priority** | Medium |

Table 3. 16: FR-7 Search Doctor

|  |  |
| --- | --- |
| **Identifier** | FR-7 |
| **Title** | Search Doctor |
| **Requirement** | The user shall be able to search for qualified doctors within the AI-Based Smart Patient Companion application based on their disease or medical condition. |
| **Source** | User (Patient) |
| **Rationale** | To provide users with a convenient way to find and connect with doctors specializing in their specific healthcare needs. |
| **Business Rule (if required)** | None |
| **Dependencies** | None |
| **Priority** | Medium |

## Non-Functional Requirements

Following are non-functional requirements of our systme

### Security:

The system should be designed with robust security features to ensure the confidentiality, integrity, and availability of user data. Measures such as encryption, access control, and regular security audits can help to mitigate security risks.

### Usability

Usability requirements deal with ease of learning, ease of use, error avoidance and recovery, efficiency of interactions, and accessibility.

**USE-1:** The AI-based Smart Patient Companion application shall provide a user-friendly and intuitive interface, allowing patients to easily understand and navigate the features and functionalities of the system.

**USE-2:** The system shall have a simple and straightforward onboarding process, guiding users through the initial setup and configuration of their profile and preferences.

**USE-3:** The system shall provide error prevention mechanisms, such as validation checks and confirmation prompts, to help users avoid making mistakes or entering incorrect data.

**USE-4:** The application shall support efficient interactions, minimizing the number of steps required to complete common tasks, such as uploading images, viewing prescriptions, and searching for doctors or pharmacies.

### Performance

**PER-1:** The AI-based Smart Patient Companion application shall load and display the user interface within 5 seconds of launching the application on a standard Android device with average hardware specifications.

**PER-2:** The system shall scan and process medical prescriptions within 5 seconds, providing the translated information to the user in real-time.

**PER-3:** The system shall retrieve nearby pharmacy search results within 5 seconds, presenting the information to the user without significant delays or latency.

**PER-4:** The system shall send medication and appointment reminders to users promptly, with a delivery time of less than 1 second from the scheduled reminder time.

**PER-5:** The application shall provide search results for qualified doctors within 3 seconds, allowing patients to quickly find doctor based on their search criteria.

### Security

**SER-1:** The AI-based Smart Patient Companion application shall ensure secure user authentication and authorization by leveraging Firebase Authentication.

**SER-2:** The system shall utilize Firebase Realtime Database with appropriate security rules and permissions, ensuring that only authorized users can access and modify the relevant data.

**SER-3:** The application enforces access control, ensuring that users have appropriate permissions and privileges within the application.

### Reliability

**REL-1:** The AI-based Smart Patient Companion application shall have a high level of availability, aiming for a minimum uptime of 99.9% during regular operation, excluding planned maintenance periods.

**REL-2:** The system shall handle concurrent user requests and data transactions without experiencing performance degradation or system failures, ensuring reliable and uninterrupted access to the application.

### Scalability

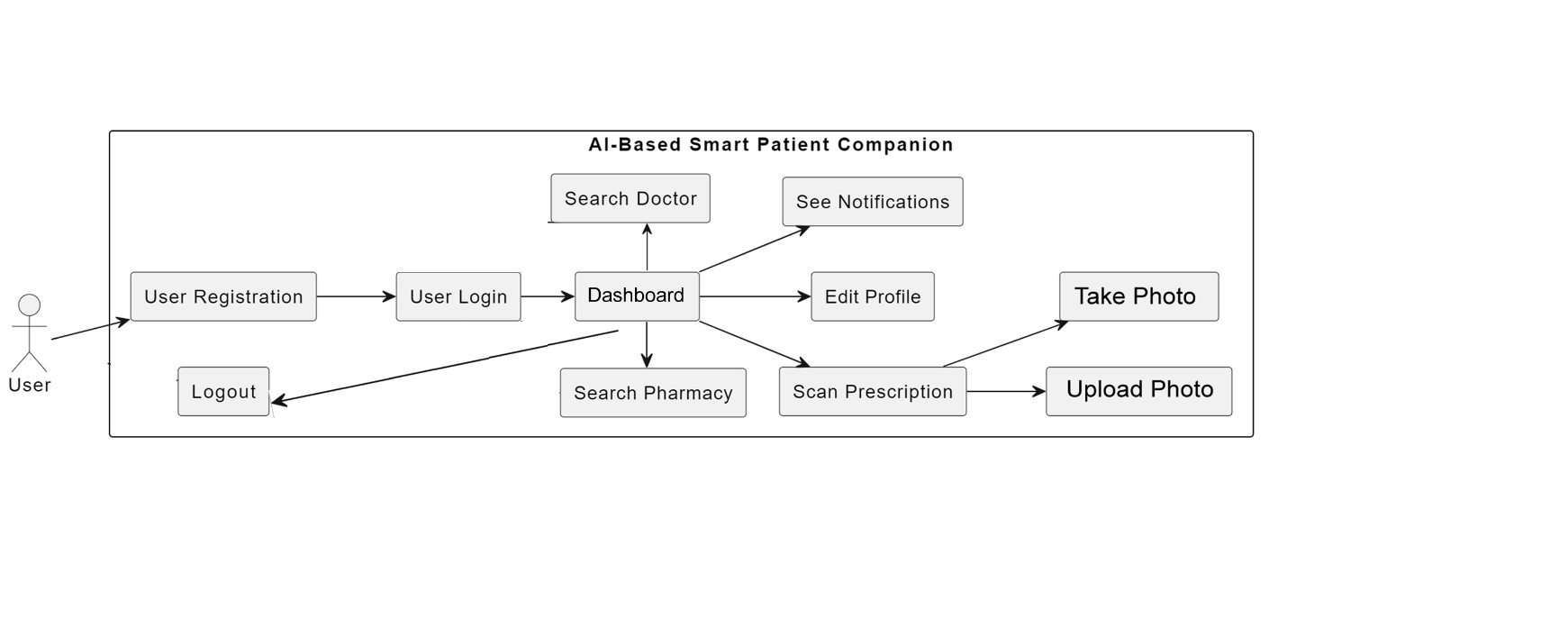
**SCT-1:** The AI-based Smart Patient Companion application shall be designed and implemented using modular and loosely coupled architecture, allowing for easy integration of new features and functionalities in the future.

**SCT-2:** The system shall support scalable infrastructure and accommodate increasing user demands or growing datasets without compromising performance or usability

# Design and Architecture

In this chapter, Design and Architecture of our system AI-Based Smart Patient Companion is explained.

## System Architecture



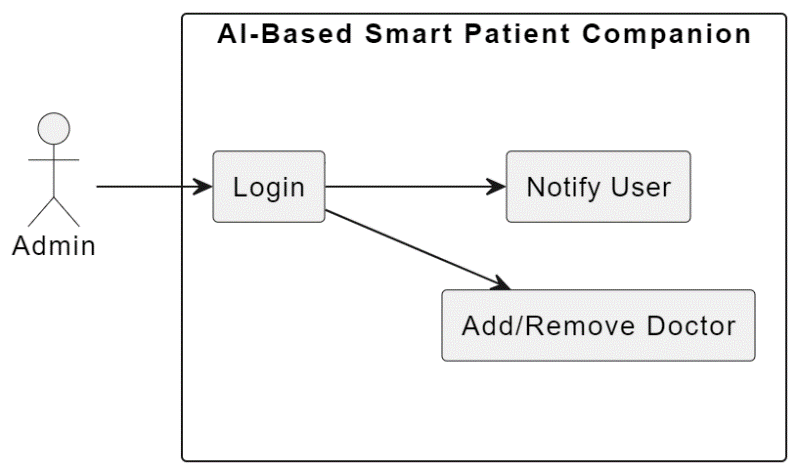


Fig 4. 1: Block Diagram

## Data Representation [Class Diagram]

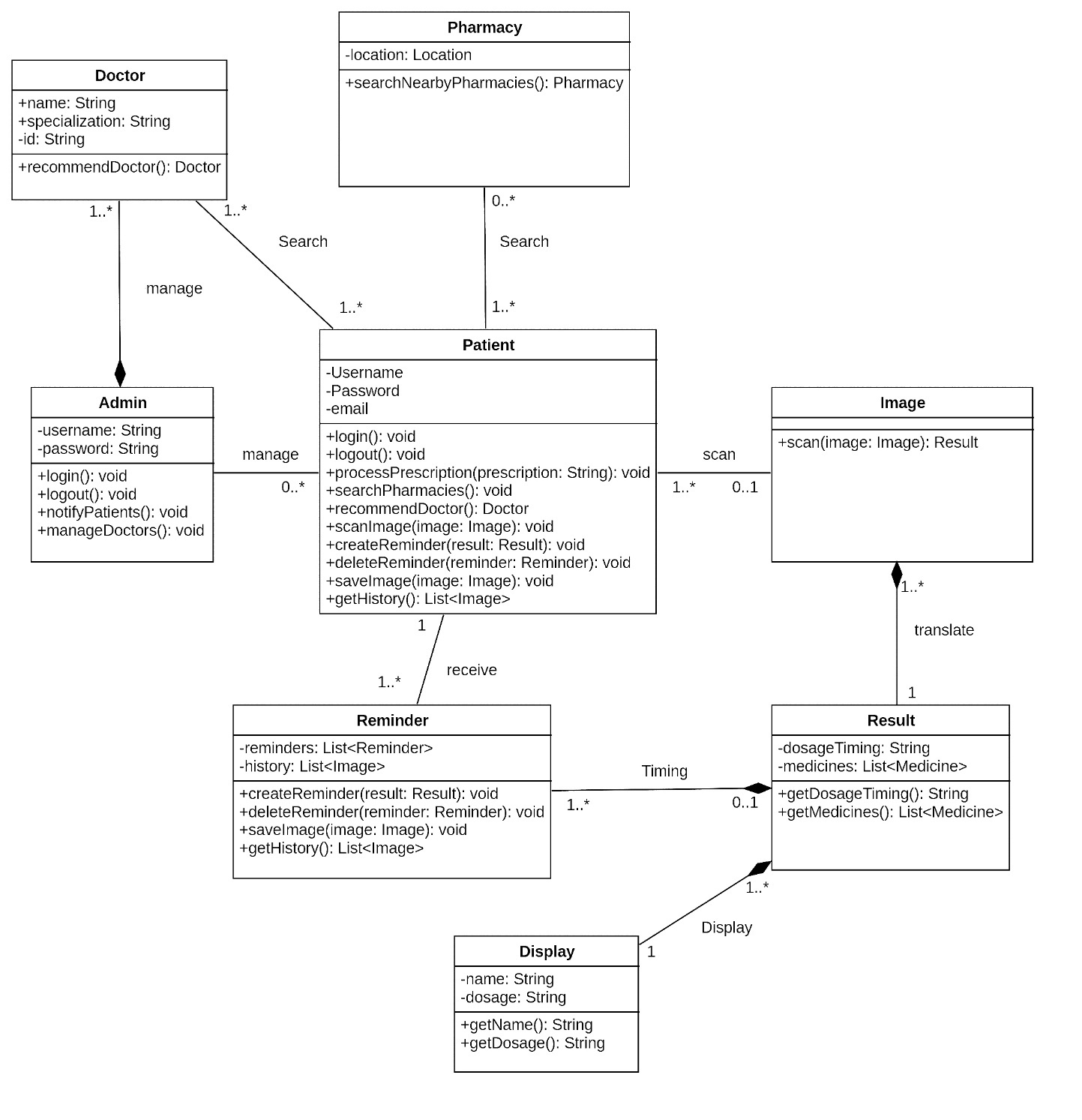
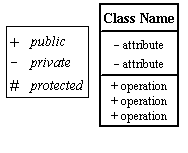
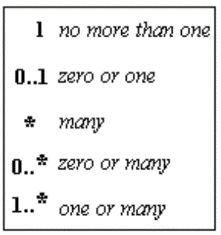


Fig 4. 2: Class Diagram

**Legends**



Composition

Association

## Process Flow/ Representation

Process Flow are given as follow:

### Patient Activity Diagram

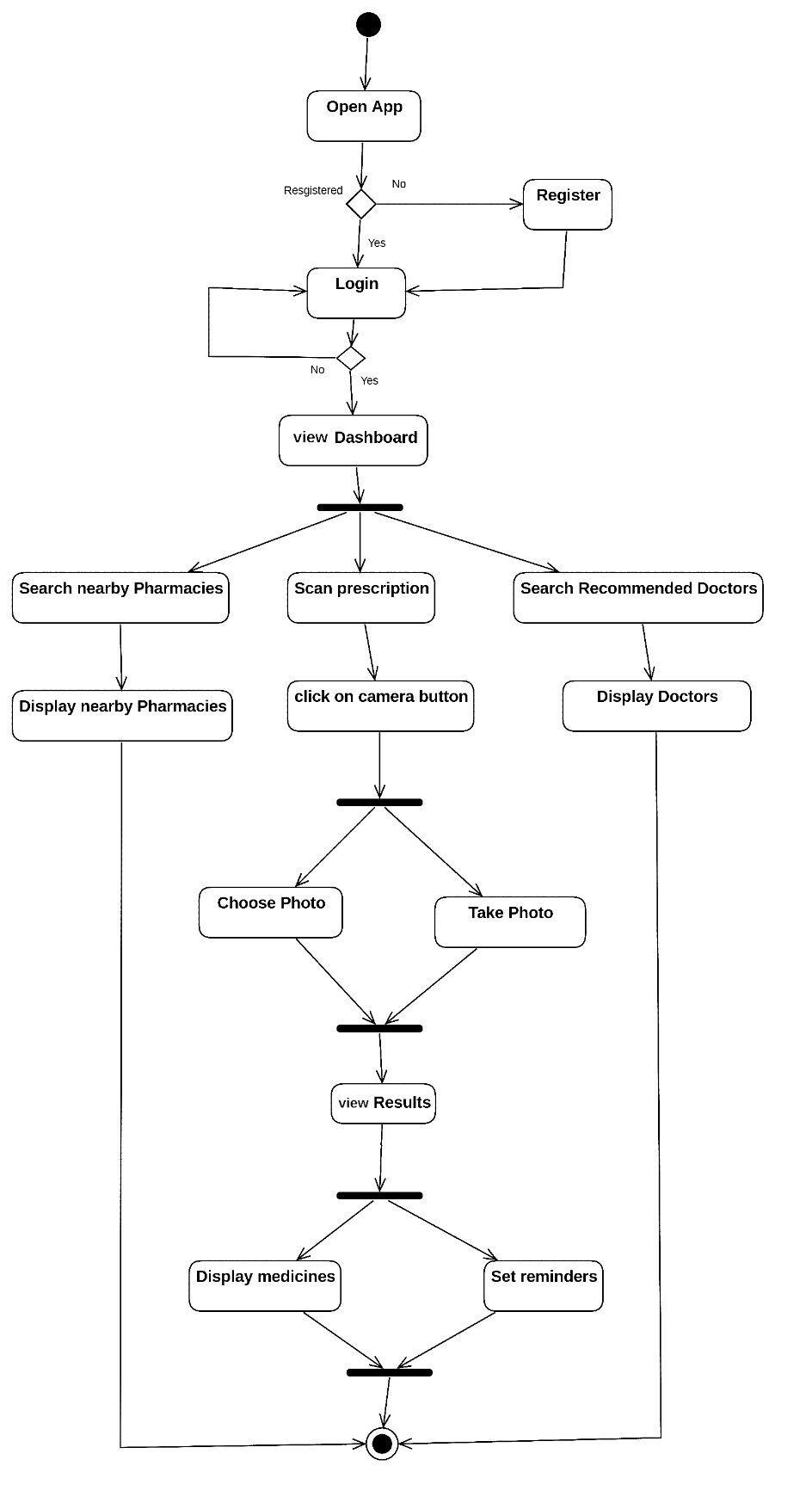
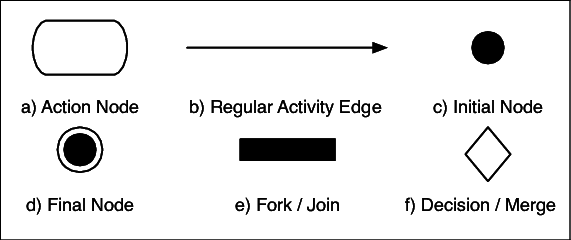


Fig 4. 3: Patient Activity Diagram

**Legends**

### Admin Activity Diagram

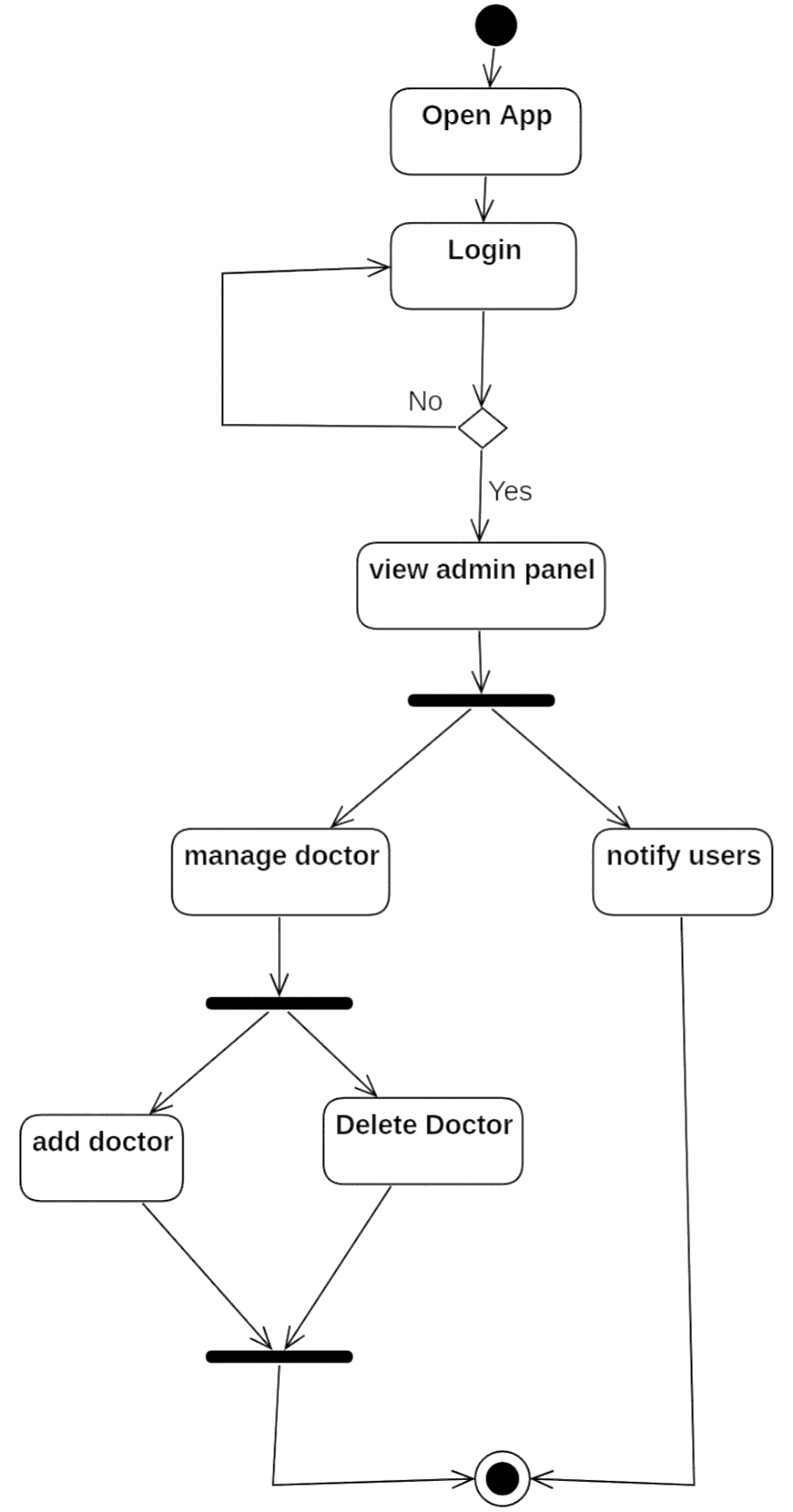


Fig 4. 4: Admin Activity Diagram

### Firebase Tree Structure

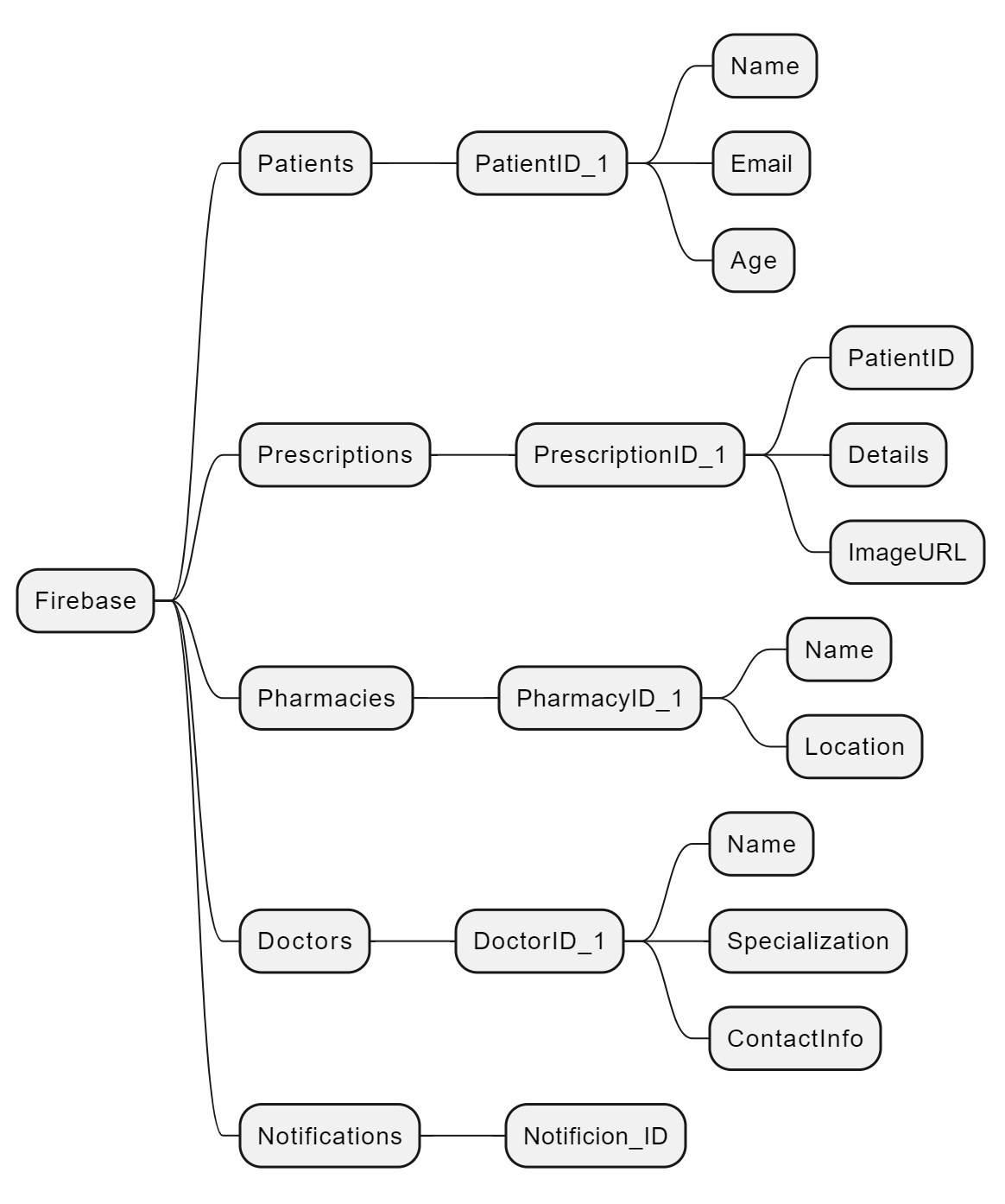


Fig 4. 5: Firebase Tree Structure

## Design Model

Design model of our system is given in this section

### Sequence Diagram

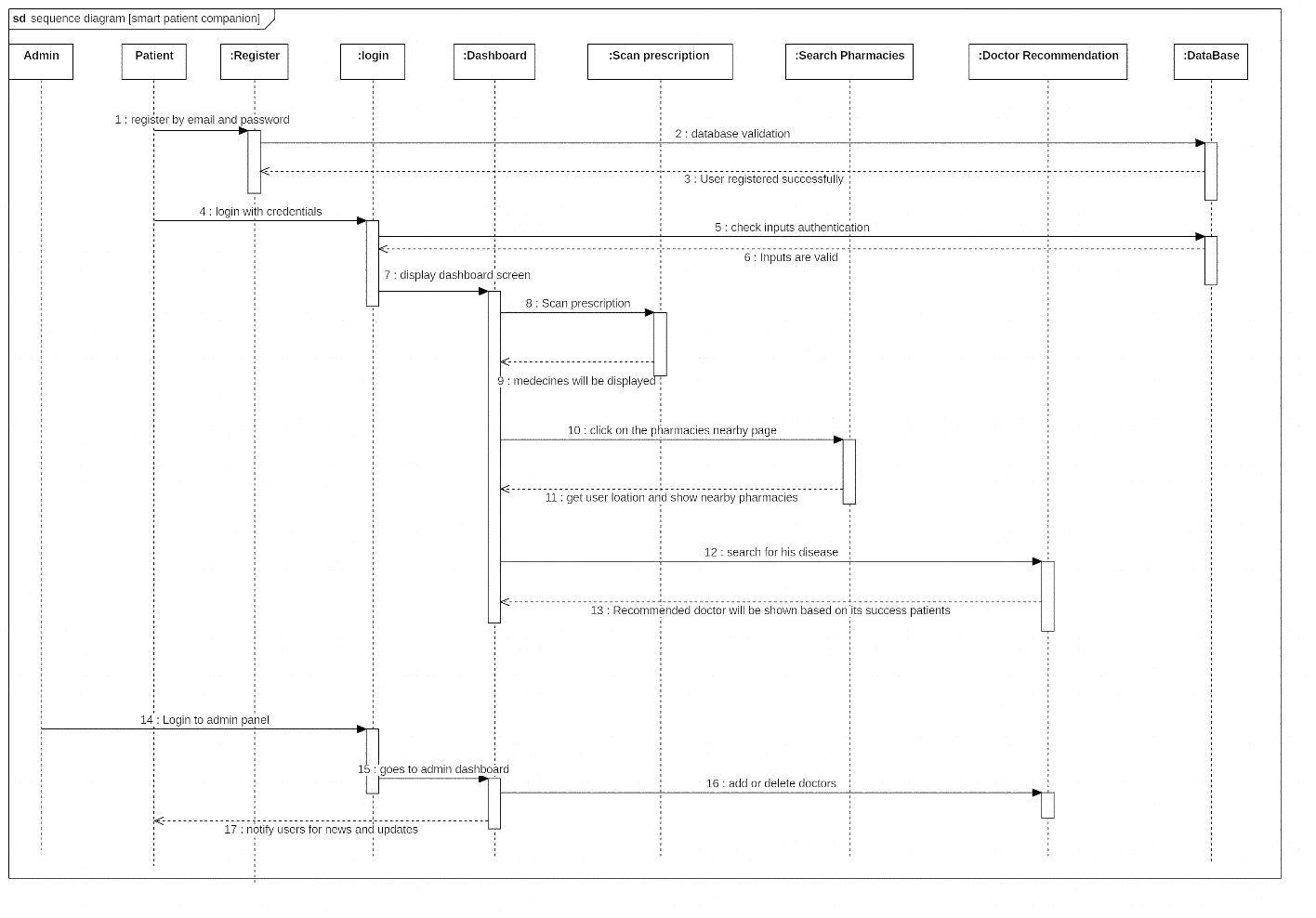


Fig 4. 6: Sequence Diagram

### System Sequence Diagram (s) [User]

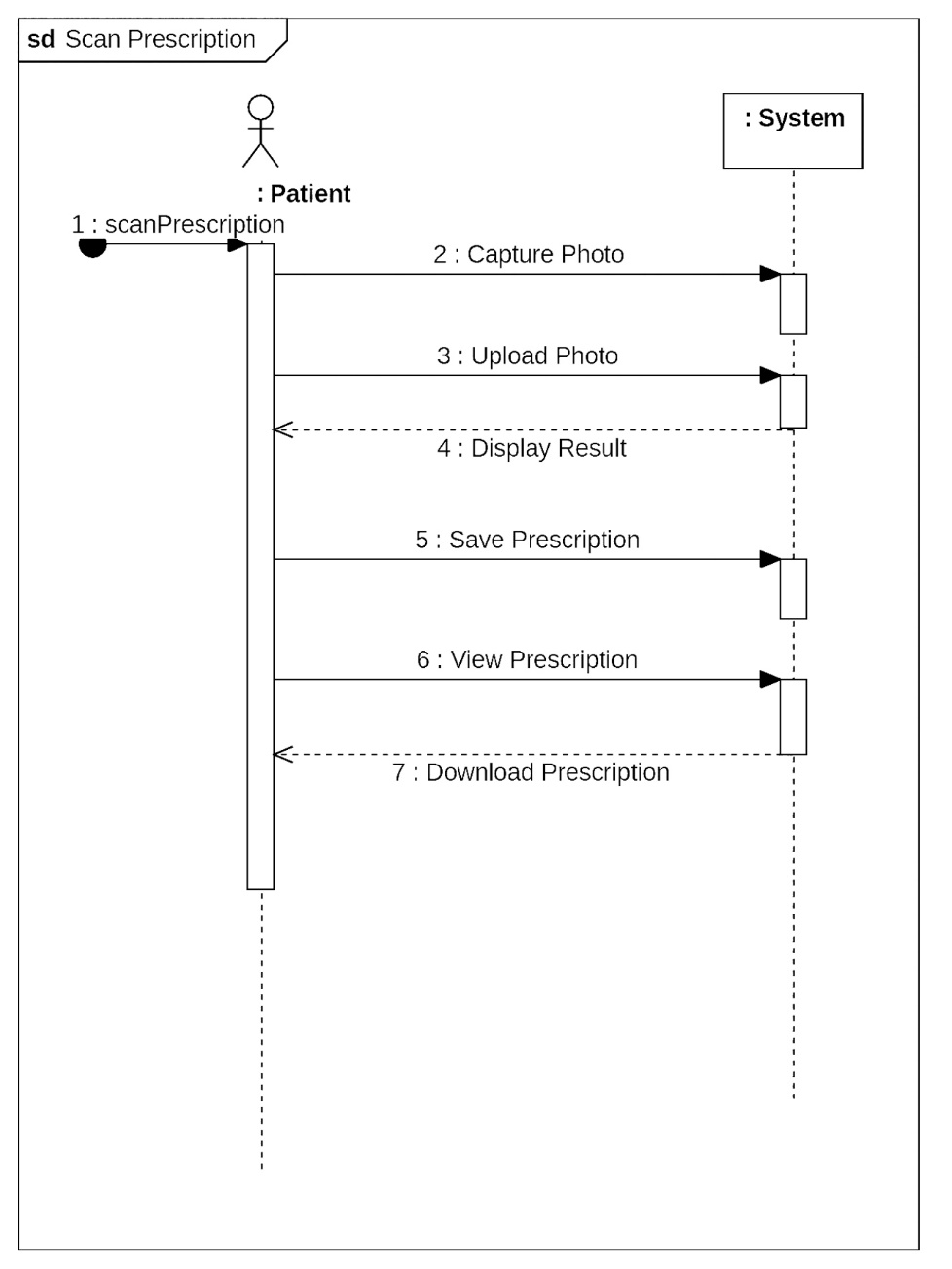


Fig 4. 7: Scan Prescription SSD

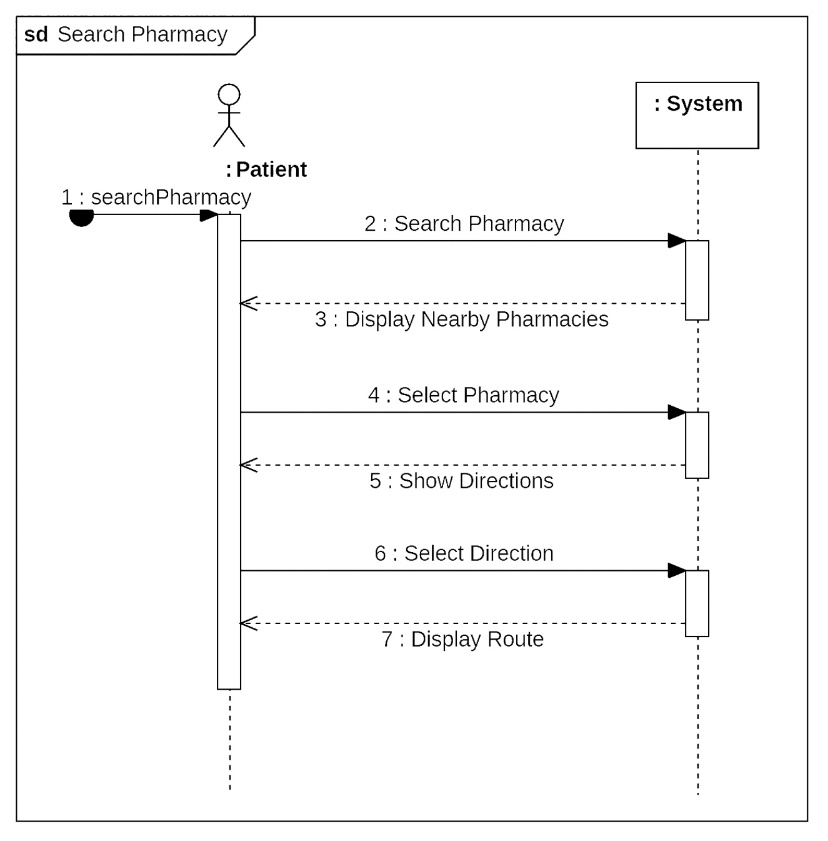


Fig 4. 8: Search Pharmacy SSD

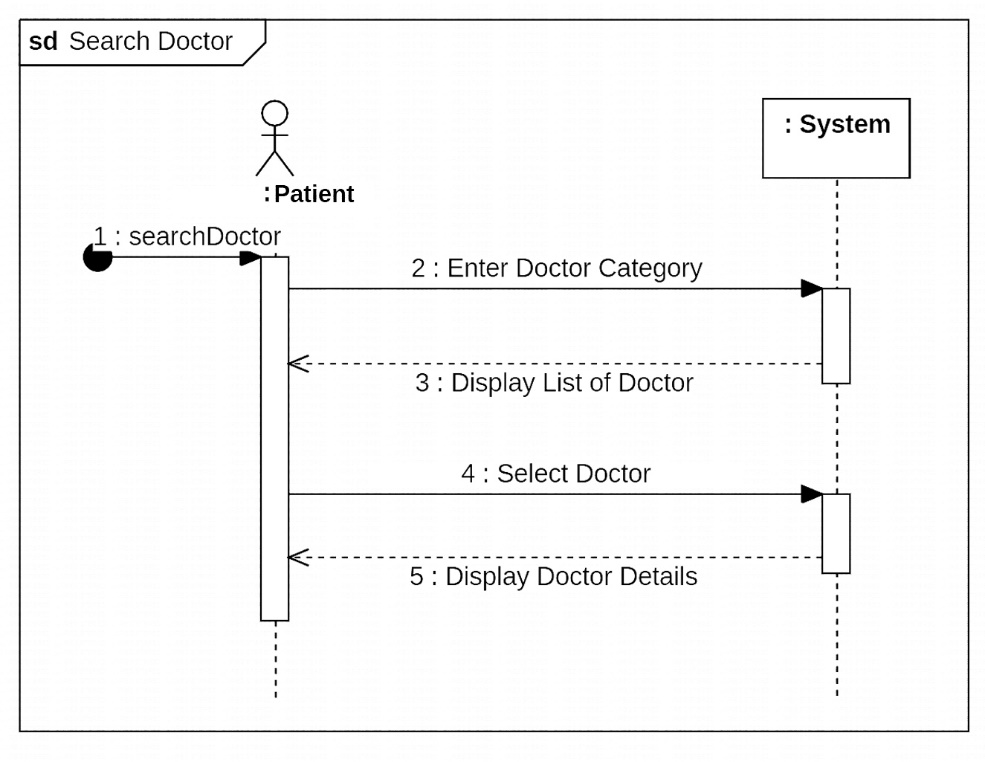


Fig 4. 9: Search Pharmacy S

### System Sequence Diagram (s) [Admin]

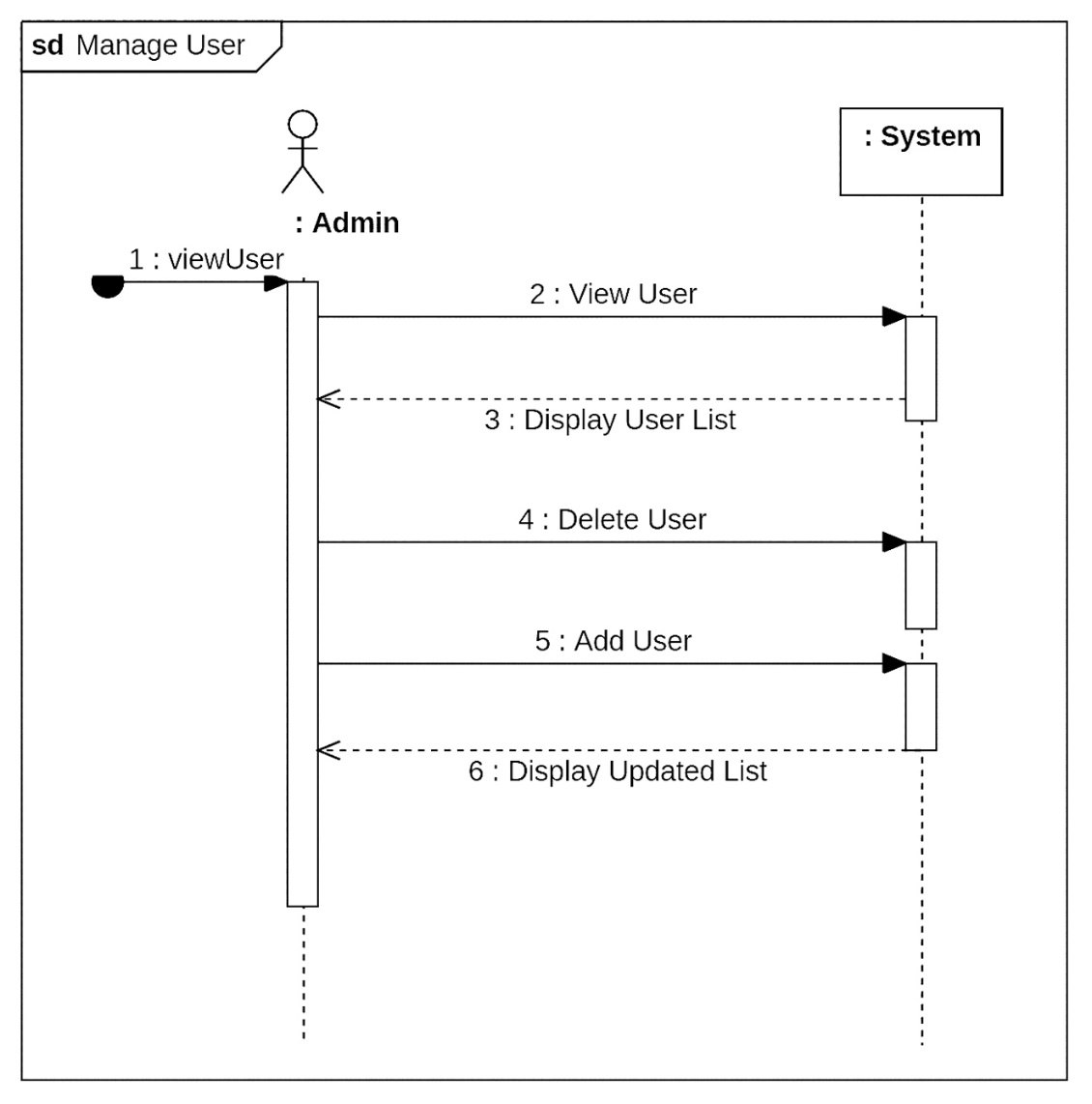


Fig 4. 10: Manage User SSD

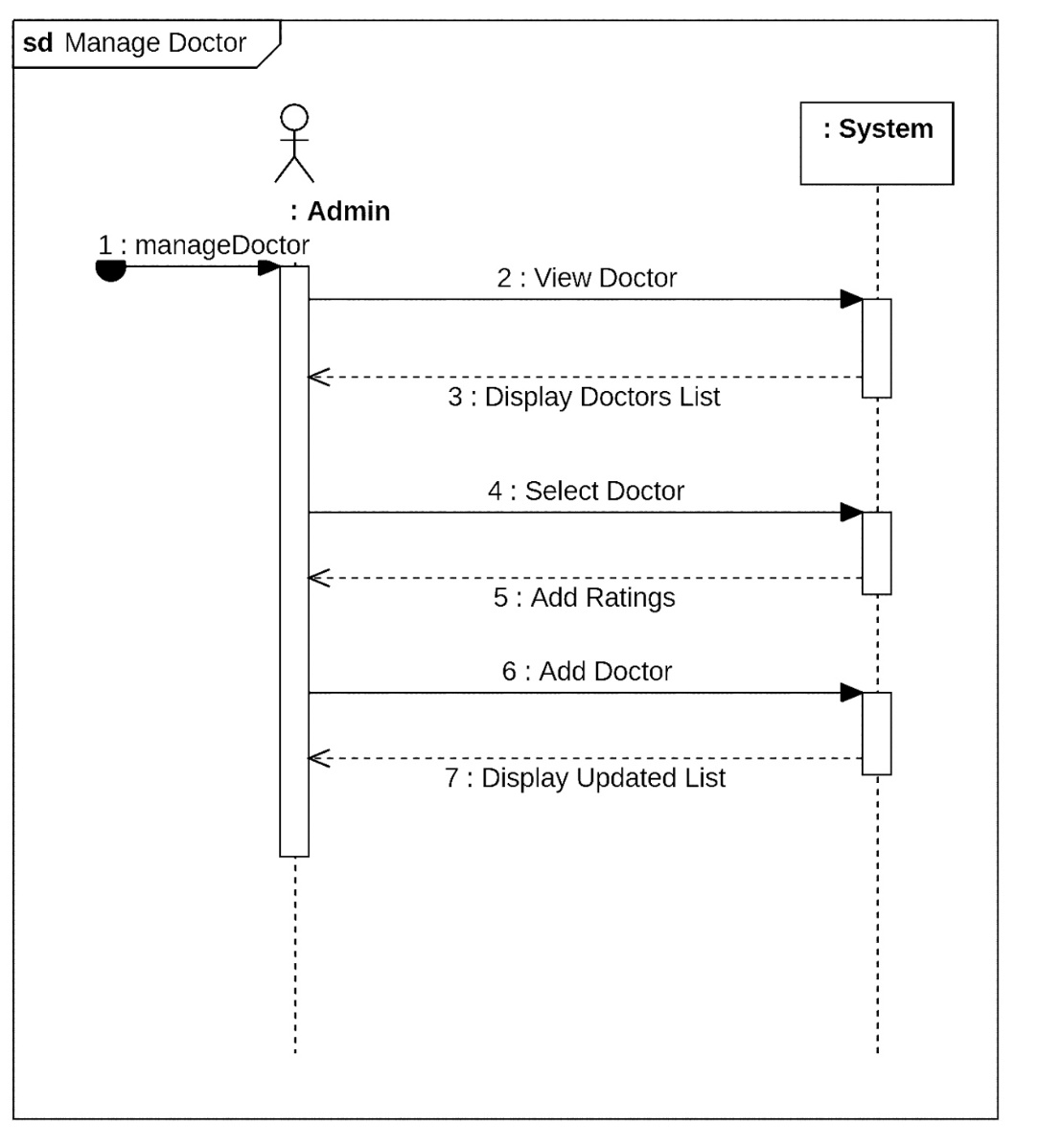


Fig 4. 11: Manage Doctor SSD

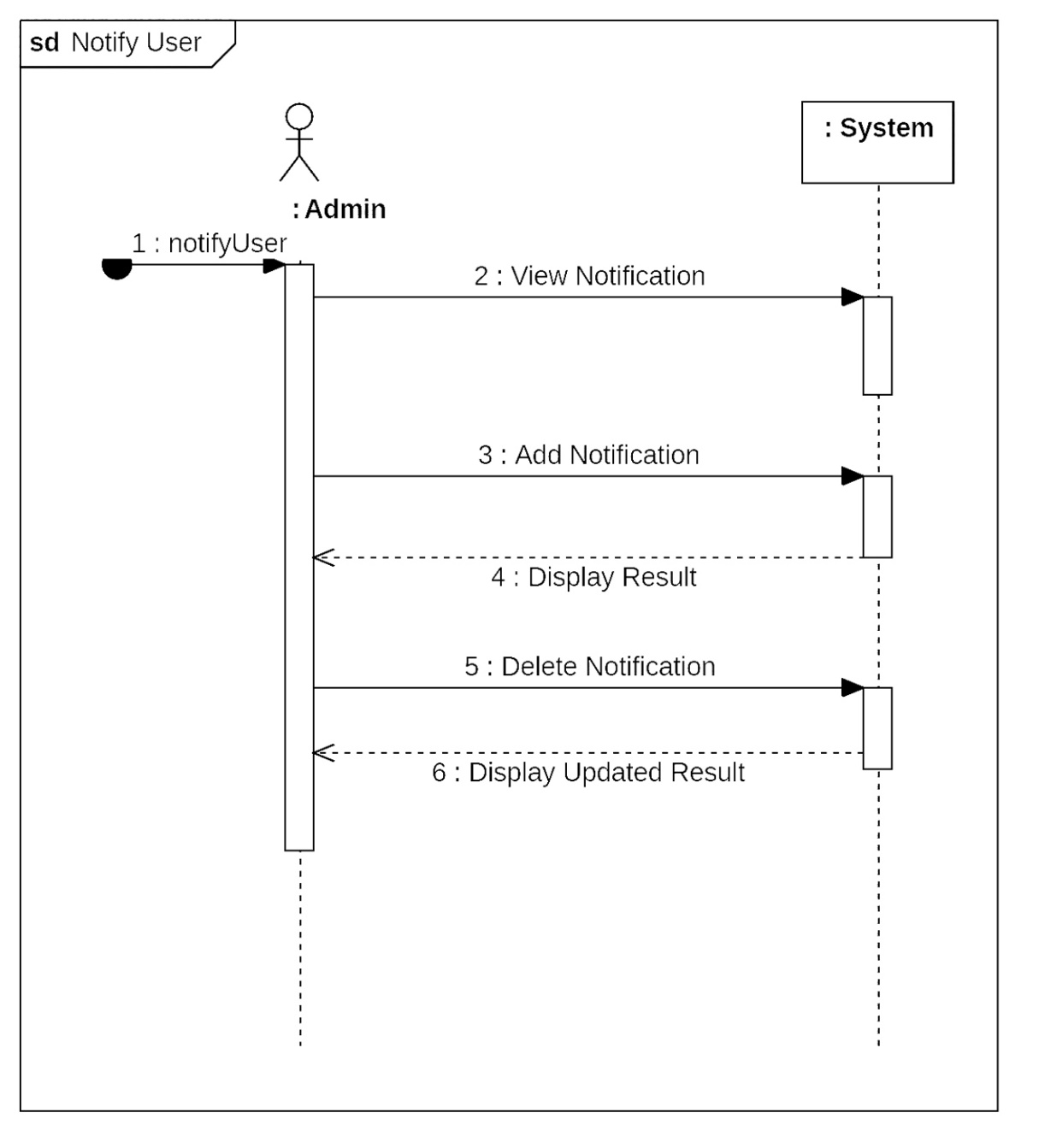


Fig 4. 12: Notify User SSD

### Package Diagram

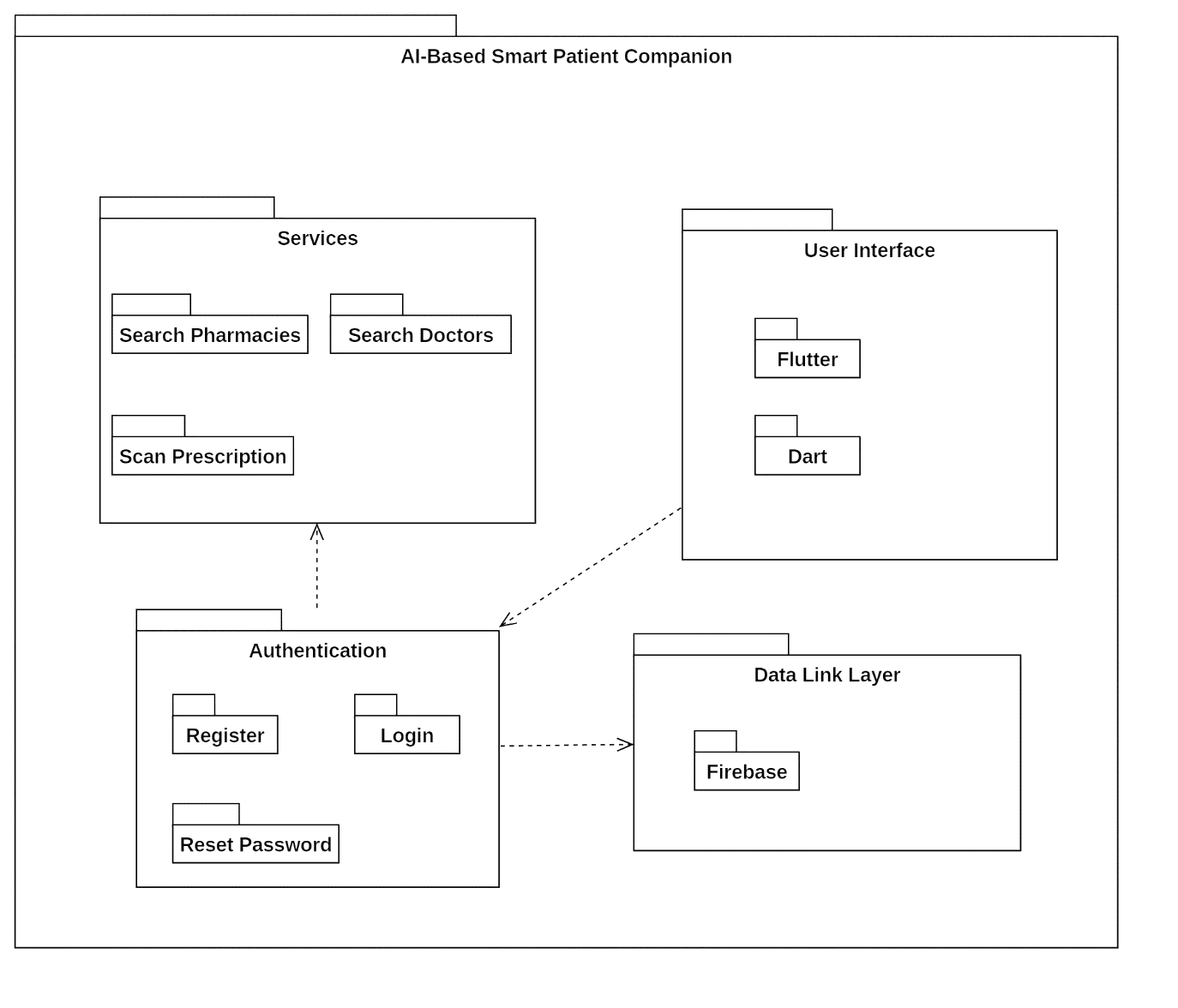


Fig 4. 13: Package Diagram

### Deployment Diagram

A diagram of a diagram

Description automatically generated

Fig 4. 14: Deployment Diagram

# Requirement Analysis

This chapter explains requirement analysis of AI-Based Smart Patient Companion

## Algorithm

### User Account Management

*Objective:* Create and manage user profiles securely.

**Algorithm Steps:**

* + - * **User Registration:** Allow users to register by providing necessary information.
      * **Profile Creation:** Generate and store user profiles securely.
      * **Authentication:** Implement a secure authentication process for user logins.
      * **Data Encryption:** Encrypt sensitive user data for privacy.

### Pharmacy Locator

*Objective:* Enable users to find nearby pharmacies.

**Algorithm Steps:**

* **User Location Retrieval:** Obtain the user's location through the device geolocation sensor.
* **Google Maps API Integration:** Utilize the Google Maps API to query nearby pharmacies based on the user's location.
* **Display Result:** Display nearby pharmacies within the radius of 5 kilometres on the application with information retrieved from the Google Maps API.
* **Navigation:** Display the navigation to the desired pharmacy.

### Doctor Search

*Objective:* Facilitate users in finding qualified doctors based on specialization.

**Algorithm Steps:**

* **Specialization Input:** Users input doctor specialization.
* **Doctor Database Query:** Query a database of qualified doctors based on specialization.
* **Output:** Display a list of all the requested doctors sorted by their ranking.

### Image Recognition & Translation Module

*Objective:* Identify medicine names in prescriptions and translate them into plain language.

**Algorithm Steps:**

* **Image Captured or Uploaded:** Users capture the picture or upload images of medical prescription.
* **Medical Term Identification:** Implement an algorithm to identify medical terms using natural language processing (NLP) techniques.
* **Translation:** Translate identified medical terms into plain language.
* **Output:** Display translated information to users.

### Admin Notification System

*Objective:* Allow users to receive notifications from the admin regarding important updates, health-related information, or system alerts.

**Algorithm Steps:**

* **Admin Notification Setup:** Admin sets up notifications through the administrative dashboard.
* **Notification Trigger:** Admin triggers notifications based on system updates, health tips, or other relevant information.
* **Notification Delivery:** When triggered, notifications are sent to users.
* **Notification Storage:** Store notifications in the database for historical reference.
* **Output:** Display notifications to users in a timely and non-intrusive manner.

## External APIs

Following external APIs are used in our system

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of API** | **Description of API** | **Purpose of Usage** | **Function/Classes in which used** |
| **Firebase** | Firebase APIs are a set of tools and services provided by Google's Firebase platform, designed to help developers build scalable and feature-rich applications more easily. Firebase offers a wide range of APIs that cover various aspects of app development, including authentication, database management, cloud messaging, analytics. | Flutter has excellent integration with Firebase services, providing APIs for authentication (firebase\_auth), cloud messaging (firebase\_messaging), real-time databases (firebase\_database), and much more. These APIs enable you to leverage the powerful features of Firebase in your Flutter application. | Used in  Signup.dart, SignIn.dart, Home.dart, Login.dart, scanPrescription.dart,  searchDoctor.dart,  dashBoard.dart, searchPharmacy.dart, |
| **Google Maps** | Google Maps API is mapping service provided by Google, offering APIs for embedding maps into applications. The API facilitates the location of nearby pharmacies based on user input or device location. | Locating nearby pharmacies | Used in searchPharmacy.dart |
| **Flask** | Flask is a micro web framework for Python applications. It facilitates the creation of web services and acts as an interface between the application and external functionalities. In the Smart Patient Companion, Flask is used to connect the frontend with the backend model for medical term matching. | Integrates the machine learning model with the application | Used in scanPrescription.dart |

Table 5. 1: External APIs

## User Interfaces

The following section presents a visual exploration of the user interface (UI) design for our AI-Based Smart Patient Companion application.



Fig 5. 1: Splash Screen UI

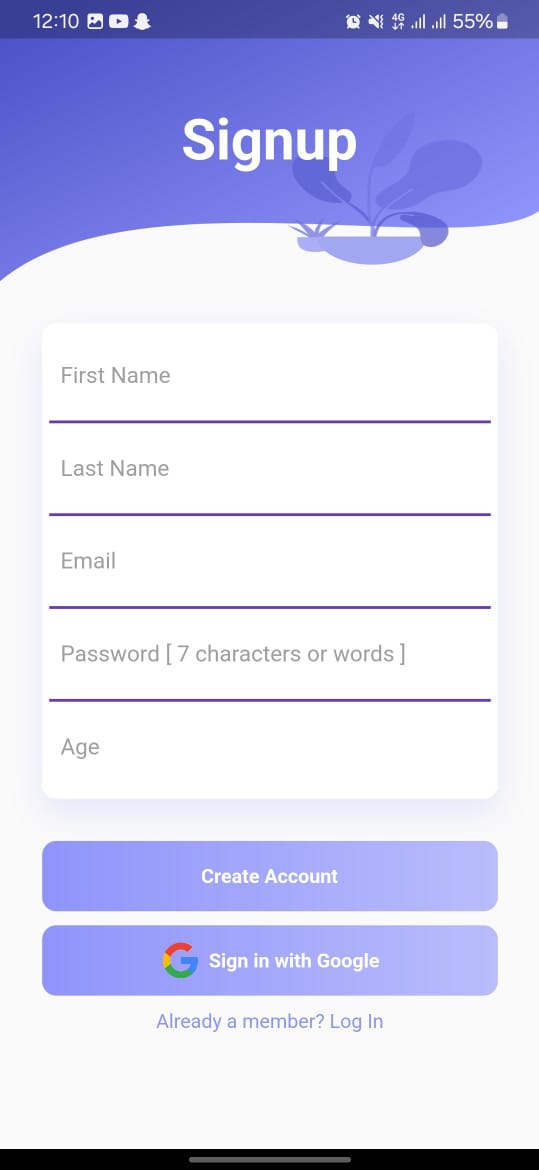


Fig 5. 2: Signup UI

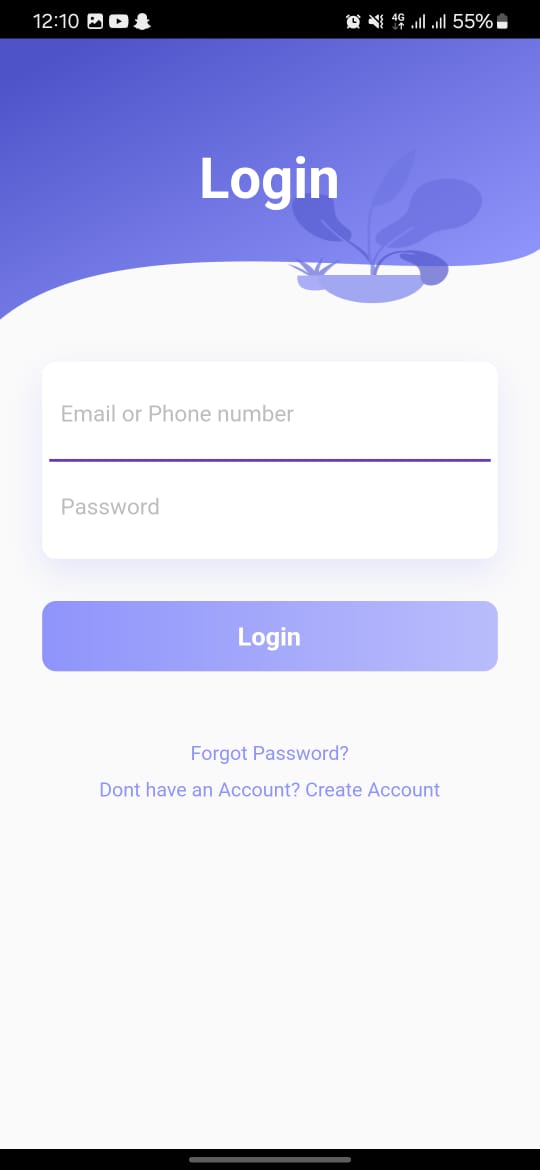


Fig 5. 3: Login UI

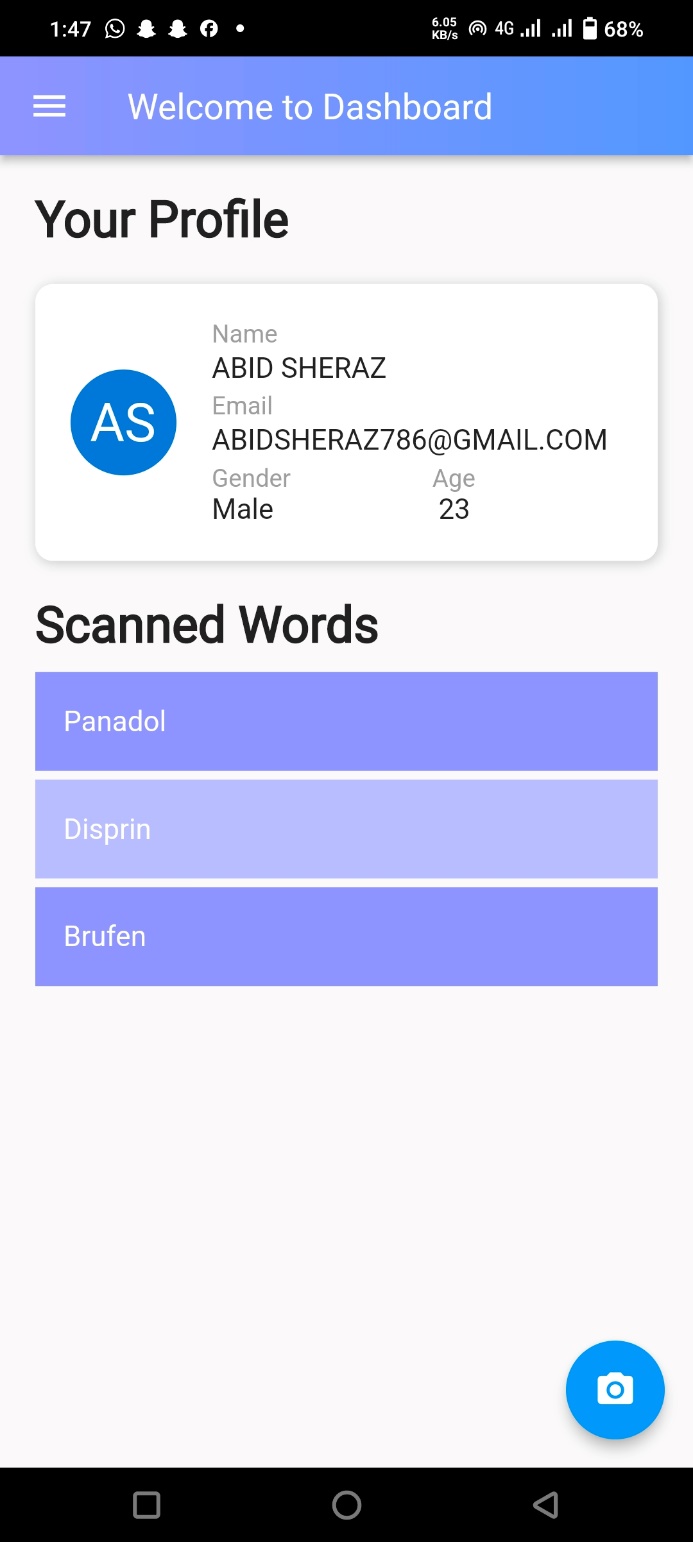


Fig 5. 4: Dashboard UI



Fig 5. 5: Search Pharmacy UI

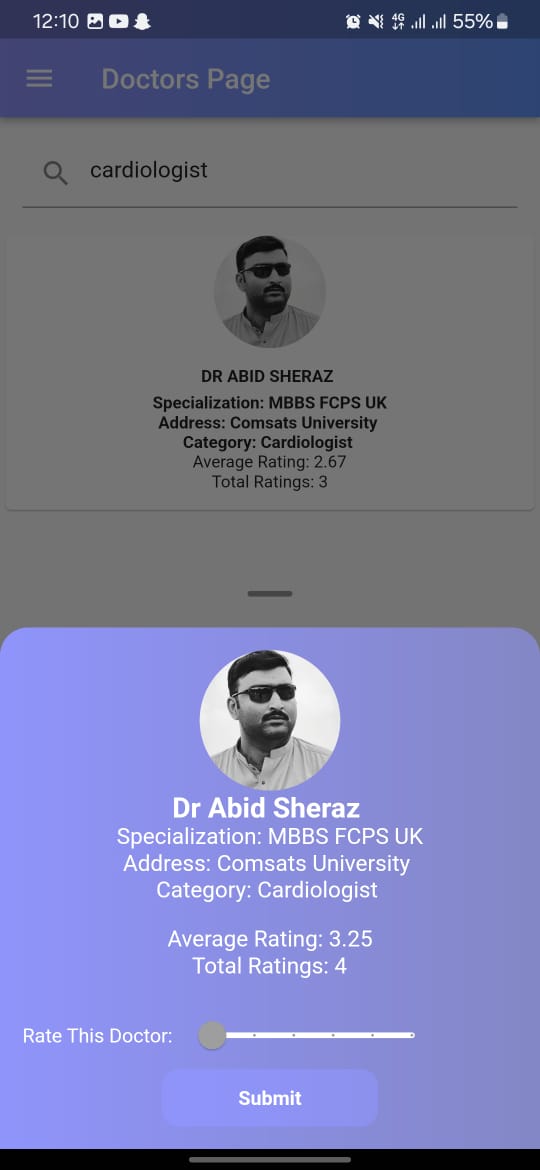


Fig 5. 6: Search Doctor UI

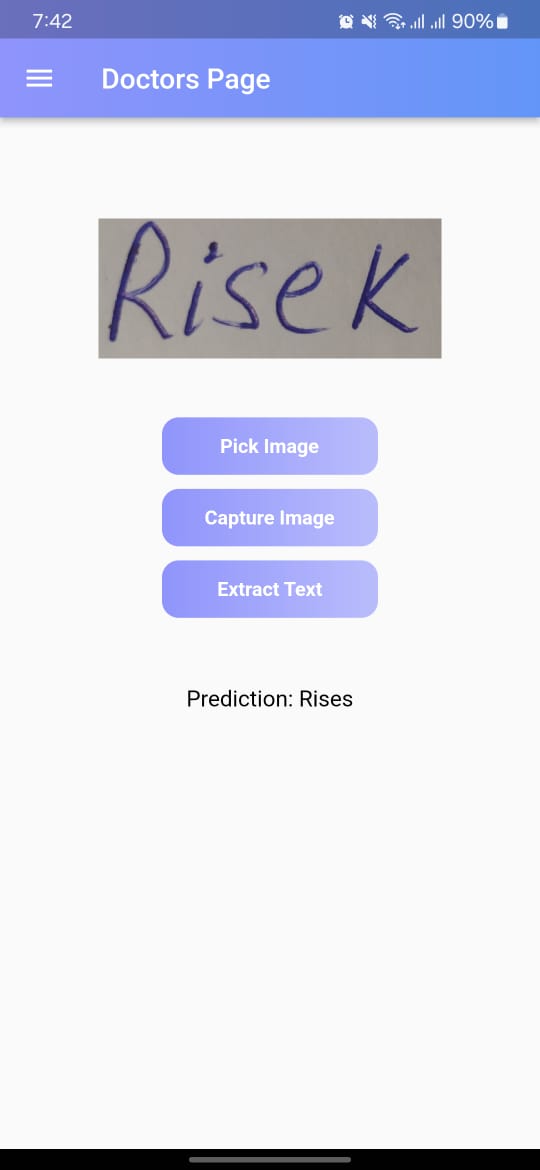


Fig 5. 7: Scan Prescription UI

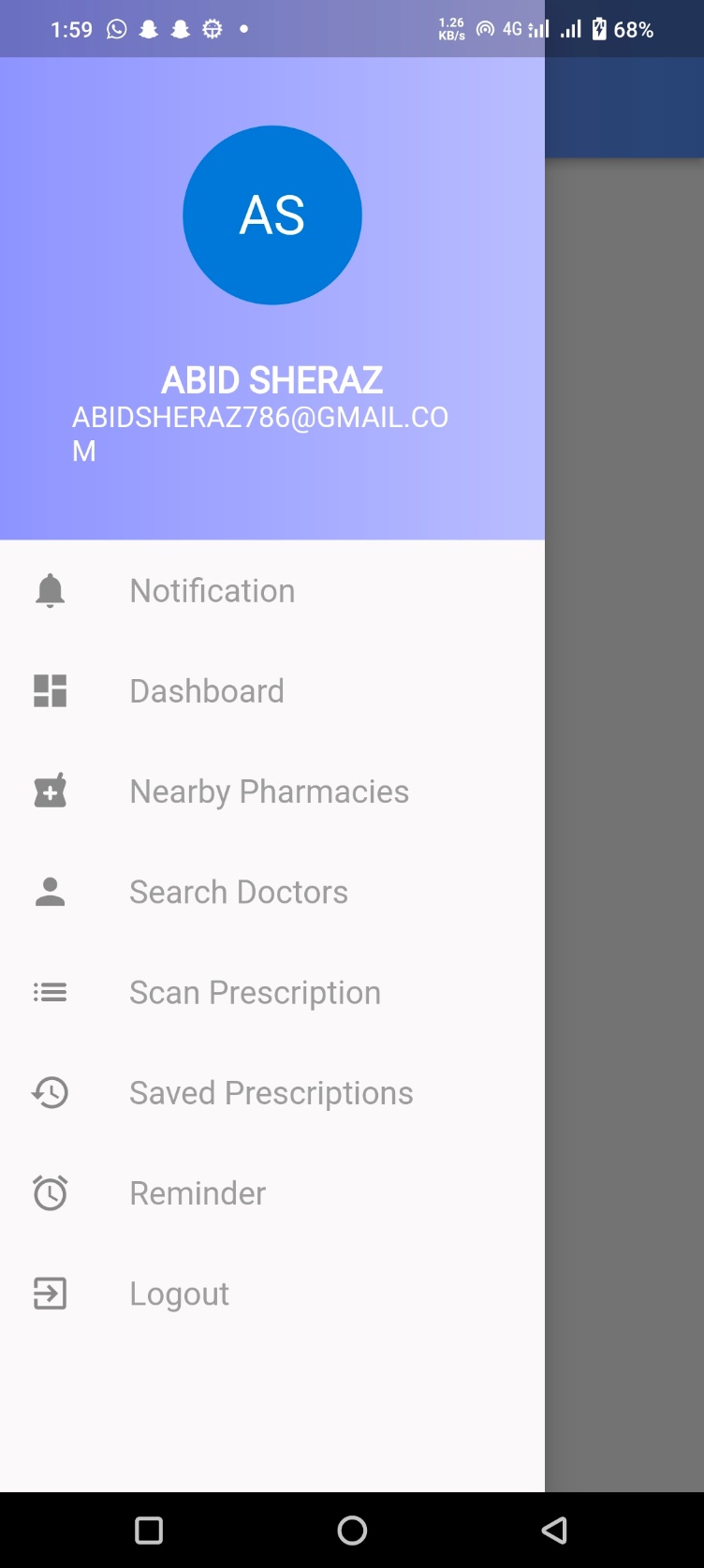


Fig 5. 8: Navigation Drawer UI

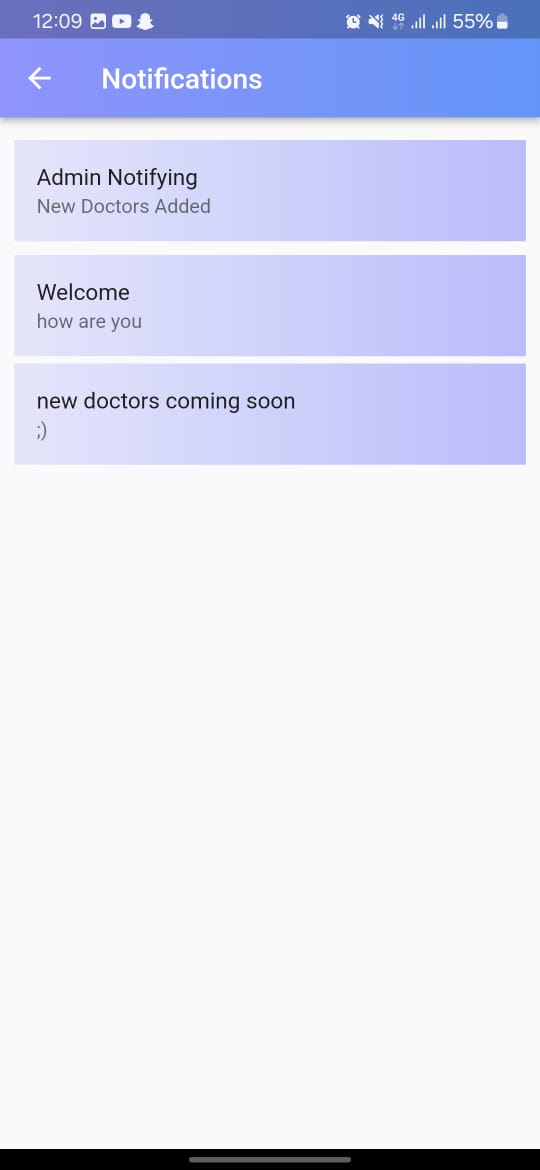


Fig 5. 9: Notification UI

# Testing & Evaluation

This chapter explain testing of our system.

## Manual Testing

### System Testing

Table 6. 1: System Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case** | **Description** | **Expected Result** | **Actual Result** | **Status** |
| User Registration | Verify successful user registration | User registration completes without errors, and user data is stored securely in the database. | Registration Process Successful Completed and user redirect to the dashboard. | Pass |
| Pharmacy Locator | Test pharmacy locator functionality | The application should correctly identify and display nearby pharmacies based on user location. | Pharmacy Locator Successful: Nearby pharmacies displayed accurately. | Pass |
| Doctor Search | Test the functionality of searching for doctors based on their specialisation | Users input their qualified doctors accordingly. | Search Successful: Relevant doctors displayed based on specialisation and ranking type. | Pass |
| Prescription Upload | Test prescription image upload functionality | Users should be able to upload images of their medical prescriptions. | Upload Successful: Prescription image successfully uploaded. | Pass |
| Scan Medicine Name | Test the functionality to scan medicine names from a prescription image | Users should be able to upload an image of a prescription, and the system should accurately scan and identify the medicine names. | Scan Successful: Medicine names accurately identified from the uploaded prescription image. | Pass |
| Manage Doctor Records | Ensure access and accuracy of doctor records | Admin can access and update doctor records. | Doctor Record Successfully Up-to-date | Pass |
| Data Security | Verify data security and privacy measures | Sensitive data is stored securely, with compliance to privacy standards. | Data security protocols are effective; no breaches detected. | Pass |
| Notification Interaction | Verify user interaction with notifications | Users should be able to interact with notifications (e.g., view details, dismiss). | Interaction Successful: Users can interact with notifications as intended. | Pass |
| Performance Testing | Test system performance under load | The system maintains performance and response times under high load. | System performance is stable even under peak usage. | Pass |

### Unit Testing

**1. Signup**

Table 6. 2: Signup Unit Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1. | Signup as a new user. | Username: [abidsheraz786@gmail.com](mailto:abidsheraz786@gmail.com) Password: abid123 | User account is successfully created. | Pass |
| 2. | Signup with existing email. | Username: [abidsheraz786@gmail.com](mailto:abidsheraz786@gmail.com) Password: abid123 | Display message "Email already in use. Please use a different email." | Pass |
| 3. | Signup with invalid email format. | Username:  abidsheraz.gmail.com Password: abid123 | Display message "Invalid email format. Please enter a valid email address." | Pass |

**2. Login**

Table 6. 3: Login Unit Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1. | Login with valid credentials. | Username: [abidsheraz786@gmail.com](mailto:abidsheraz786@gmail.com) Password: abid123 | User successfully logs in, and the dashboard is loaded. | Pass |
| 2. | Login with incorrect password. | Username: [abidsheraz786@gmail.com](mailto:abidsheraz786@gmail.com) Password: wrongpassword | Display message "Incorrect password. Please try again." | Pass |
| 3. | Login with non-existent email. | Username: [nonexistemail@gmail.com](mailto:nonexistemail@gmail.com) Password: anypassword | Display message "Email not found. Please check your email or sign up." | Pass |
| 4. | Login with blank email. | Username: "" Password: anypassword | Display message "Please enter your email." | Pass |
| 5. | Login with blank password. | Username: [abidsheraz786@gmail.com](mailto:abidsheraz786@gmail.com) Password: "" | Display message "Please enter your password." | Pass |

**3. Dashboard**

Table 6. 4: Dashboard Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1. | Verify initial dashboard loading. | Onclick on Login Button, user logged in successfully. Dashboard is displayed. | Dashboard loads with relevant user information and functionalities. | Pass |
| 2. | Display Medicine & User Details | On main dashboard user details and name of medicines are displayed. | User details and name of medicines are displayed on main dashboard | Pass |
| 3. | Drawer Search Pharmacies. | Open drawer. Click on "Search Pharmacies." | Nearby Pharmacies page is displayed. | Pass |
| 4. | Drawer Navigation - Search Doctors. | Open drawer. Click on "Search Doctors." | Search Doctors page is displayed. | Pass |
| 5. | Logout Functionality. | Open drawer. Click on "Logout." | User is logged out, and the login page is displayed. | Pass |
| 6. | Verify successful prescription scan. | Upload a valid prescription image. | System successfully recognizes and processes the prescription. | Pass |

### Functional Testing

**Functional Testing 1:** Login functional testing.

Table 6. 5: Login Functional Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1. | Login with valid credentials. | Username: [user@gmail.co](mailto:user@gmail.co) Password: password123 | Dashboard is loaded with user-specific information. | Pass |
| 2. | Login with incorrect password. | Username: [user@gmail.com](mailto:user@gmail.com) Password: wrongpassword | Display message "Incorrect password. Please try again." | Pass |
| 3. | Login with non-existent email. | Username: [nonexistemail@gmail.com](mailto:nonexistemail@gmail.com) Password: anypassword | Display message "Email not found. Please check your email or sign up." | Pass |

**Functional Testing 2:** Dashboard functional testing.

Table 6. 6: Dashboard Functional Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1. | Verify initial dashboard loading. | On click on login Button, user logged in successfully. Dashboard is displayed. | Dashboard loads with relevant user information and functionalities. | Pass |
| 2. | Display Medicine & User Details | On main dashboard user details and name of medicines are displayed. | User details and name of medicines are displayed on main dashboard | Pass |
| 3. | Drawer Search Pharmacies. | Open drawer. Click on "Search Pharmacies." | Nearby Pharmacies page is displayed. | Pass |
| 4. | Drawer Navigation - Search Doctors. | Open drawer. Click on "Search Doctors." | Search Doctors page is displayed. | Pass |
| 5. | Logout Functionality. | Open drawer. Click on "Logout." | User is logged out, and the login page is displayed. | Pass |
| 6. | Verify successful prescription scan. | Upload a valid prescription image. | System successfully recognizes and processes the prescription. | Pass |

**Functional Testing 3:** Search Pharmacy functional testing.

Table 6. 7: Search Pharmacy Functional Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1. | Request Geolocation Permission | User grants geolocation permission | System requests and receives permission to access device geolocation. | Pass |
| 2. | Deny Geolocation Permission | User denies geolocation permission | Display message "Location access denied. Please enable location services to find nearby pharmacies." | Pass |
| 3. | Search Pharmacies with Permission | User grants geolocation permission and initiates search | System automatically detects user location and displays nearby pharmacies. | Pass |
| 4. | Search Pharmacies without Permission | User denies geolocation permission and initiates search | Display message "Location access denied. Please enable location services to find nearby pharmacies." | Pass |
| 5. | Search Pharmacies Multiple Attempts | User grants geolocation permission and performs multiple searches | System consistently updates and displays nearby pharmacies based on the latest location. | Pass |
| 6. | Search Pharmacies No Results | User grants geolocation permission, but no nearby pharmacies found | Display message "No nearby pharmacies found. Please try again later or manually enter your location." | Pass |
| 7. | Search Pharmacies Invalid Location | User manually enters an invalid location | Display message "Invalid location. Please enter a valid location or enable location services." | Pass |

**Functional Testing 4:**

Search Doctor functional testing.

Table 6. 8: Search Doctor Functional Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1. | Search Doctors by Specialization | User selects a medical specialization and initiates search | System displays a list of doctors specializing in the selected field. | Pass |
| 2. | Search Doctors without Specialization | User initiates search without selecting a specialization | Display message "Please select a specialization to find doctors." | Pass |
| 3. | Invalid Specialization | User enters an invalid or non-existent specialization | Display message "Invalid specialization. Please select a valid medical specialization." | Pass |
| 4. | Multiple Search Attempts | User performs multiple searches with different specializations | System consistently updates and displays doctors based on the latest specialization. | Pass |
| 5. | Search Doctors No Results | User selects a valid specialization, but no doctors found | Display message "No doctors found in this specialization. Please try another specialization or check again later." | Pass |
| 6. | View Doctor Details | User clicks on a doctor in the search results | System displays detailed information about the selected doctor, including contact information and reviews. | Pass |

### Integration Testing

Table 6. 9: Integration Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1. | Verify Successful Login and Dashboard Loading | Modules Integrated: Login, Dashboard | Procedure: Login with valid credentials | User is redirected to the dashboard, and relevant information is loaded. |
| 2. | Verify Logout Integration | Modules Integrated: Login, Dashboard | Procedure: Log in and then log out | After logout, the system redirects to the login page. |
| 3. | Verify Dashboard Interaction with Scan Prescription | Modules Integrated: Dashboard, Scan Prescription | Procedure: Navigate to the dashboard and initiate a prescription scan | The scan prescription process seamlessly integrates into the dashboard. |
| 4. | Verify Dashboard Interaction with Search Pharmacy | Modules Integrated: Dashboard, Search Pharmacy | Procedure: Navigate to the dashboard and initiate a search for nearby pharmacies | The search pharmacy process seamlessly integrates into the dashboard. |
| 5. | Verify Dashboard Interaction with Search Doctors | Modules Integrated: Dashboard, Search Doctors | Procedure: Navigate to the dashboard and initiate a search for doctors based on specialization | The search doctors process seamlessly integrates into the dashboard. |
| 6. | Verify Search Pharmacy Interaction with Search Doctors | Modules Integrated: Search Pharmacy, Search Doctors | Procedure: Search for nearby pharmacies and then initiate a search for doctors | The search doctors process utilizes the user's location information from the search pharmacy module. |
| 7. | Verify Search Doctors Interaction with Scan Prescription | Modules Integrated: Search Doctors, Scan Prescription | Procedure: Search for doctors and initiate a prescription scan | The scan prescription process utilizes information about the selected doctor. |

# Conclusion & Future Work

## Conclusion

The AI-based Smart Patient Companion project is a transformative endeavor aimed at revolutionizing how individuals manage their healthcare. In response to the intricate medical jargon present in healthcare documents, we set out to develop an innovative solution using cutting-edge technologies and a holistic approach. The project's motivation stemmed from the evolving healthcare landscape, where patients are increasingly taking charge of their well-being amidst rapid technological advancements. Recognizing the challenges faced by patients in deciphering complex medical terminologies, the Smart Patient Companion was conceptualized to empower users through automatic translation and understanding of medical prescriptions. Throughout the project, we leveraged the knowledge and skills acquired during our four-year BSE program, incorporating principles from software engineering, human-computer interaction, software quality assurance, testing, project management, and various other courses. The system architecture was built upon the foundations of object-oriented design, utilizing Flutter for the front-end and Laravel for the back-end to enhance code maintainability and scalability. The Agile methodology was chosen for the Software Development Life Cycle (SDLC), allowing for adaptability to changing client requirements and facilitating parallel development efforts. In our literature review, we compared the Smart Patient Companion with existing applications, identifying its strengths in providing a comprehensive solution by addressing medical terminology understanding, healthcare management, and patient empowerment. The system's capabilities include translating medical jargon, tracking medication usage, finding nearby pharmacies, searching for qualified doctors, and offering voice reading functionality for accessibility. External APIs played a crucial role in the project's success. Firebase API was utilized for secure and scalable data storage, while Flask API served as a bridge between the application and the medical term matching model. Google Maps API facilitated the location of nearby pharmacies, enhancing the overall user experience. The algorithm implementation documentation detailed the step-by-step processes for key features such as image recognition and translation, medication management, pharmacy locator, doctor search, voice reading, user account management, reporting, analytics, and admin notifications. These algorithms were carefully designed to ensure seamless functionality and user interaction. Automated testing was a critical aspect of our development process, utilizing tools such as Flutter Test Framework, Firebase Test Lab. These tools ensured the robustness and reliability of the application across various scenarios and platforms. AI-based Smart Patient Companion represents a significant stride towards empowering individuals in managing their healthcare effectively. By combining advanced technologies, comprehensive algorithms, and a user-centric design, the application not only addresses existing challenges but also paves the way for a more informed, engaged, and healthier community. The successful integration of various components, adherence to software engineering principles, and rigorous testing underscore the project's commitment to delivering a high-quality, user-friendly, and impactful solution for the healthcare domain.

## Future Work

The AI-based Smart Patient Companion has laid a solid foundation for empowering individuals in managing their healthcare. As we look ahead, there are several avenues for future work and enhancements to further enrich the application and its impact on users' well-being:

1. **Integration of Additional Healthcare Services:** Expand the range of healthcare services integrated into the application. This could include features like telemedicine consultations, health monitoring through wearable devices, or integration with electronic health records (EHR) systems.
2. **Machine Learning Refinement:** Continuously refine and expand the machine learning models used in medical term matching. Incorporate more sophisticated natural language processing techniques to enhance the system's ability to accurately identify and translate diverse medical terminologies.
3. **Advanced Medication Adherence Solutions:** Implement advanced medication adherence solutions, such as integrating with smart pill dispensers or leveraging Internet of Things (IoT) devices to enhance the tracking and reminders for medication routines.
4. **Enhanced Reporting and Analytics:** Further enhance reporting and analytics features to provide users with deeper insights into their health trends, medication adherence patterns, and overall well-being. This could involve data visualization techniques for a more intuitive understanding.
5. **Personalized User Recommendations:** Develop a recommendation engine that provides personalized suggestions to users based on their health history, medication usage, and preferences. This could include lifestyle recommendations, health tips, or suggestions for nearby healthcare facilities.
6. **Social Integration and Community Building:** Introduce social integration features that allow users to connect with others facing similar health challenges. Building a supportive community within the application can foster collaboration, information sharing, and emotional support.
7. **Gamification for Health Engagement:** Implement gamification elements to make health management more engaging. Introduce challenges, rewards, and interactive features to motivate users to actively participate in their healthcare journey.
8. **Continuous User Feedback and Usability Studies:** Conduct regular user feedback sessions and usability studies to gather insights on user experiences and preferences. This iterative feedback loop will be invaluable in refining the user interface, enhancing features, and addressing evolving user needs.
9. **Integration with Emerging Technologies:** Explore integration with emerging technologies such as augmented reality (AR) or virtual reality (VR) to create immersive and interactive health education experiences for users.
10. **Cross-Platform Development:** Extend the application's compatibility by exploring cross-platform development frameworks. This could involve adapting the application for other platforms, such as web browsers, to reach a broader audience.
11. **Globalization and Multilingual Support:** Enhance the application's accessibility by providing multilingual support and considering the cultural nuances of healthcare practices. This will make the Smart Patient Companion more inclusive for users around the world.
12. **Security and Privacy Enhancements:** Place a continuous focus on security and privacy measures. Regularly update and reinforce security protocols to protect user data, and stay compliant with evolving data protection regulations.

# References

Al-Shahmeh, M. M. (2023). *Flutter Apps Development: Build Cross-Platform Flutter Apps with Trust.* Damascus, Syria: Independantly Published.

Boté, J.-J. (Mar 2019). Dataset Management as a Special Collection. *Collection Management*, 259-276.

Esraa Hassan, H. T. (January 2021). Medical Prescription Recognition using Machine Learning. *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)* (pp. 0973-0979). NV, USA: IEEE.

Galen, D. (2018). *Software Quality, Concepts and Practice.* Hoboken, New Jersey: Wiley, IEEE Press.

Karl Wiegers, J. B. (2013). *Software Requirements, Third edition.* Unterschleissheim, Germany: Microsoft Press.

Marco Ortu, G. D. (May 2015). Measuring and Understanding the Effectiveness ofJIRA Developers Communities. *Research Gate*, 8-9.

Pavel Hamet, J. T. (April 2017). Artificial intelligence in medicine. *Metabolism*, S36-S40.

Payne, R. (December 2019). *Beginning App Development with Flutter: Create Cross-Platform Mobile Apps.* New York: Apress.

Phillip A. Laplante, M. K. (2022). *Requirements Engineering for Software and Systems.* New York: Auerbach Publications.

Raschka, S. (September 2015). *Python Machine Learning.* Birmingham, UK: Packt Publising Ltd.