**HiLCoE  
School of Computer Science and Technology  
  
ALPHA Intelligent Health Companion and Monitoring System**

**Project Proposal**Prepared by:

HAFTOM TSEGAY  
NAHOM LELISSA  
YEABSIRA BELAY  
ALAZAR SHIMELIS  
  
Approved By:  
  
Advisor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Examiner: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_

**Executive Summary**

The ALPHA Intelligent Health Companion and Monitoring System is a web-based platform designed to help individuals manage their personal health and improve their wellbeing through AI-driven solutions. By collecting and analyzing diversified health-related data, the system provides personalized lifestyle guidance and context-aware insights. It does not offer direct diagnosis. The platform is especially useful in regions with limited healthcare access. It promotes preventive care and health awareness through secure, internet-accessible tools. This technology encourages proactive care, early risk detection, and well-informed lifestyle changes by converting fragmented health data into concise, context-aware insights. This strategy encourages timely, effective self-management, which not only gives people more control over their health but also reduces the burden on healthcare providers. The platform provides a workable solution with a focus on relevance and accessibility. By doing this, ALPHA promotes more sustainable healthcare practices, increases public awareness of health issues, and strengthens the role of individual health data as an opportunity for better health and national advancement.

Table of Contents

[Chapter 1 - Project Proposal 4](#_Toc207664067)

[1.1 Introduction / Background 4](#_Toc207664068)

[1.2 Statement of the Problem 4](#_Toc207664069)

[1.3 Objectives 5](#_Toc207664070)

[1.3.1 General Objective 5](#_Toc207664071)

[1.3.2 Specific Objectives 5](#_Toc207664072)

[•Plan the entire project phases and timeline 5](#_Toc207664073)

[• Collect relevant data from diverse groups of users and public health sources 5](#_Toc207664074)

[1.4 Scope of the Project and Limitations 6](#_Toc207664075)

[1.4.1 Scope 6](#_Toc207664076)

[1.4.2 Limitations 6](#_Toc207664077)

[1.5 Methodology / Approach 6](#_Toc207664078)

[1.5.1 Data Collection Methodologies 6](#_Toc207664079)

[1.5.2 System Development Methodology 7](#_Toc207664080)

[**Frontend Tools** 8](#_Toc207664081)

[**Backend Tools** 8](#_Toc207664082)

[**Modeling and Documentation Tools** 8](#_Toc207664083)

[**Deployment Tools** 8](#_Toc207664084)

[1.6 Significance and Beneficiaries 9](#_Toc207664085)

[1.7 Task Breakdown and Feasibility Analysis 9](#_Toc207664086)

[**Feasibility Analysis** 10](#_Toc207664087)

[1.8 Project Schedule/Timeline 10](#_Toc207664088)

[Chapter Two - System Software / System Requirements Analysis and Specification 11](#_Toc207664089)

[2.1Introduction / Overview 11](#_Toc207664090)

[2.2 Problem Context 11](#_Toc207664091)

[2.3 Purpose of the System 11](#_Toc207664092)

[2.4 Current System Description 12](#_Toc207664093)

[2.5 Proposed System Description 12](#_Toc207664094)

[2.5.1 Functional Requirements 12](#_Toc207664095)

[2.5.2 Non-Functional Requirements 13](#_Toc207664096)

[2.5.3 Business Rules 13](#_Toc207664097)

[2.6 Use Case Modeling 13](#_Toc207664098)

[2.6.1 Actors 13](#_Toc207664099)

[2.6.2 List of Main Use Cases 14](#_Toc207664100)

[2.6.3 Use Case Descriptions 14](#_Toc207664101)

[2.6.4 Use Case Diagram 21](#_Toc207664102)

[2.7 Sequence Diagrams 22](#_Toc207664103)

[2.8 Activity Diagram 29](#_Toc207664104)

[Chapter 3 – System Design and Architecture 40](#_Toc207664105)

[3.1 Introduction 40](#_Toc207664106)

[3.2 System Architecture Overview 40](#_Toc207664107)

[3.2.1 Architectural Layers 40](#_Toc207664108)

[3.2.2 Architectural Style 41](#_Toc207664109)

[3.2.3 Cross-Cutting Concerns 41](#_Toc207664110)

[3.3 Subsystem Decomposition 42](#_Toc207664111)

[3.4 Object Design Document (ODD) 44](#_Toc207664112)

[3.4.1 Trade-offs 44](#_Toc207664113)

[3.4.2 Package Diagram 45](#_Toc207664114)

[3.4.3 Key Classes 45](#_Toc207664115)

[3.4.4 Class Diagram 48](#_Toc207664116)

[3.5 Design Constraints and Quality Attributes 48](#_Toc207664117)

[References 49](#_Toc207664118)

# Chapter 1 - Project Proposal

## 1.1 Introduction / Background

The volume of personal health data collected daily has grown rapidly due to advancements in digital health technologies [3]. From electronic health records maintained by clinics to various forms of self-tracked data, individuals now generate a wide range of information. Despite this availability, most people lack the tools and knowledge to interpret their data effectively, especially in developing countries like Ethiopia [2]. This gap limits their ability to make informed decisions and manage their health proactively.

In resource-limited settings, individuals often struggle to understand fragmented health data and access reliable guidance for everyday health choices. Without proper interpretation, early signs of health issues go unnoticed, and opportunities for preventive care are missed [4]. The ALPHA Intelligent Health Companion and Monitoring System addresses this challenge by collecting, storing, and analyzing diversified health-related data. It translates raw inputs into personalized insights and reminders, offering accessible, non-diagnostic support through AI-driven solutions [1].

Healthy individuals strengthen society. When people maintain good health, they contribute more productively to the economy, engage actively in their communities, and reduce pressure on healthcare systems. This leads to improved national performance and better overall well-being [5].

Personalizing health data empowers individuals to take control of their health. By understanding their unique patterns, users respond more effectively to changes, manage chronic conditions with greater confidence, and avoid unnecessary medical visits. Personalized insights support smarter, faster decisions [6].

This project proposes a web-based system that bridges the gap between raw health data and meaningful action. Through AI-driven solutions, it delivers tailored lifestyle recommendations that promote preventive care and informed self-management [1].

## 1.2 Statement of the Problem

There is a growing number of internet access in Ethiopia, and the rapid growth of digital connectivity and the growing dependence on online platforms for daily tasks like communication, education, and health-related information are all contributing factors to Ethiopia's steadily rising internet access level [8]. However, most users struggle to use their health data effectively. People track symptoms, vital signs, and routines by using digital services, but they lack tools to interpret the data because they don't know what the numbers indicate. They often remain uncertain about what actions to take next and this leads to missed chances and early signs of illness go unnoticed. Small problems become serious and eventually preventable conditions get worse [4].

Most users lack basic medical knowledge. They don't know how to analyze their data. They don't get timely advice. They don't get support for daily decisions. You need a system that works with what you have, one that collects your data and analyzes it with AI. A system like this helps you give clear, personal advice. It does not make diagnosis on your health, more like guidance you can act on. This kind of system helps you make better decisions. It helps you manage conditions. It reduces pressure on hospitals and supports public health [1, 2].

## 1.3 Objectives

### 1.3.1 General Objective

The general objective of this project is to design and develop an Intelligent Health Companion and Monitoring System

### 1.3.2 Specific Objectives

To achieve the above general objective, the following specific activities will be carried out. These include:

•Plan the entire project phases and timeline

• Collect relevant data from diverse groups of users and public health sources

• Elicit system requirements through stakeholder interviews and research

• Design system architecture, user interface, and data flow

• Select development tools, frameworks, and platforms

• Build /implement a secure, user-friendly Progressive Web App for smartphones and desktops

• Design a structured database for storing health data including vitals, conditions, sleep, ovulation, medications, and symptoms

• Test or validate the system functionality, performance, and security

• Evaluate system effectiveness and refine based on users feedback

## 

## 1.4 Scope of the Project and Limitations

### 1.4.1 Scope

The Intelligent Health Companion and Monitoring System is a web-based platform designed to assist users in tracking, understanding, and managing their health through the collection and AI-driven analysis of diverse personal health inputs [1]. The system is built to serve individuals by offering non-diagnostic lifestyle guidance based on user-specific data, without replacing healthcare professionals. It includes core functionalities such as user health profile management, AI-driven symptom analysis, an ovulation cycle tracker, a prescription decoder, AI-generated lifestyle recommendations, personalized reminders for medications and health actions, structured health data storage and reporting, vital sign monitoring, and a responsive Progressive Web App interface.

### 1.4.2 Limitations

• Integration with hospital EHR systems may be limited initially due to lack of

standardized APIs and data-sharing policies.

• Predictive accuracy of the AI models will depend on the quality and frequency of

data collected by users.

• Continuous real-time monitoring may not be feasible for users without compatible

devices.

• Limited budgets may restrict the scale of initial pilot testing to a small sample group.

## 1.5 Methodology / Approach

### 1.5.1 Data Collection Methodologies

• **Interview**: Conduct structured interviews with target users, including patients, community health workers, and local clinicians. These sessions reveal practical challenges, expectations, and usability needs. Direct input helps shape features like symptom tracking, reminder systems, and language preferences.

• **Group meetings:** Hold regular team meetings to align technical constraints, user feedback, and deployment priorities. These sessions support fast decision-making and ensure the system reflects both engineering feasibility and public health relevance.

**• Document analysis:** Review national health guidelines, digital health policies, and existing survey reports. This helps identify regulatory requirements, infrastructure limitations, and integration opportunities. It ensures the system complies with PDPP [6] and supports local health strategies.

• **Public datasets:** Use structured health data from WHO [4], Kaggle [5], and Fitbit [5] to simulate inputs for AI training. These datasets provide baseline patterns for symptom analysis, lifestyle trends, and model validation [3].

• **Pilot testing:** Deploy early versions to volunteer users. Collect feedback on usability, clarity, and recommendation accuracy. This real-world input guides iterative improvements and ensures relevance in low-resource settings.

### 1.5.2 System Development Methodology

Based on the scope and requirements of our system, The Agile software development methodology would be the most suitable approach. The Agilemethodology is an iterative and flexible approach to software development that focuses on delivering functional software in small, incremental releases rather than a single final product. Agile emphasizes collaboration, adaptability, and continuous improvement throughout the development process. Given the nature of our project, Agile is the most suitable approach to ensure the system remains responsive to real-world needs. Here are the benefits of Agile for this project:

• Agile delivers the system in small, incremental releases. This helps us test features early and adjust based on user input.

• Agile supports cross-functional teams. Developers, health experts, and users work together. This improves communication and problem-solving.

• Requirements will change as we gather feedback. Agile adapts to these changes without delay.

• Frequent feedback loops help us catch issues early. This ensures the system meets user expectations.

• Continuous integration and delivery allow steady progress. We build, test, and deploy in short cycles.

• Agile helps us manage complexity and respond to real-world needs. It is ideal for health systems in low-resource settings.

Agile is ideal for this project because it effectively manages complexity by breaking down the health system’s diverse stakeholder needs into manageable, prioritized iterations. Its user-centric approach ensures continuous feedback from health workers and administrators who may have limited technical expertise guaranteeing an intuitive and practical system. Frequent testing and incremental releases help mitigate risks by catching issues early, avoiding costly late-stage failures, while Agile’s scalability allows us to launch with a minimum viable product (MVP) and expand features based on real-world demand and performance. This adaptability makes Agile perfectly suited for dynamic, resource-constrained healthcare environments.

The following requirement software tools are used to implement our project work

### 

### **Frontend Tools**

• **HTML:** Builds the structure of web pages

• **CSS:** Styles the layout and appearance

**• Bootstrap:** Creates responsive, mobile-friendly design

**• JavaScript:** Adds interactivity and dynamic behavior

**• Supported Browsers:** Google Chrome, Mozilla Firefox, Opera, Internet Explorer

### 

### **Backend Tools**

**• PostgreSQL:** Manages structured health data

### **Modeling and Documentation Tools**

**• PlantUML:** Designs UML diagrams for system architecture

**• Microsoft Word 2016:** Prepares project documentation

**IDE**

**Visual Studio Code:** Used for writing and editing code

### **Deployment Tools**

• Laptop computer

## 1.6 Significance and Beneficiaries

**Significance**:

• Improves personal health management through data-driven insights

• Promotes early detection of health risks

• Encourages preventive care and lifestyle adjustments

• Reduces pressure on healthcare facilities

• Supports digital health innovation in Ethiopia

• Aligns with national health and technology goals

**Beneficiaries**:

• **Individual users**: Receive personalized, non-diagnostic health guidance

• **Healthcare providers:** Engage with better-informed patients

• **Public health authorities:** Benefit from improved awareness and behavior change

• **Developers and researchers:** Gain access to real-world data and use cases

## 1.7 Task Breakdown and Feasibility Analysis

Table 01: Task Breakdown and Deliverables

|  |  |
| --- | --- |
| **Task** | **Deliverables** |
| Requirement Elicitation | Stakeholder needs, use cases, feature list, functional and non-functional needs |
| **.** Data Collection | Raw health data samples, stakeholder interviews, surveys |
| **.** Requirement Analysis | Functional and non-functional requirements, sample data. |
| **.** Development Model | Agile methodology with iterative milestones |
| System Design | Architecture diagrams, data models, UI mock-ups |
| Back-End Development | REST API, data storage, authentication |
| Front-End Development | Web dashboard, |
| Integration & Testing | Test cases, bug tracking, user feedback logs |
| Deployment & Documentation | Final system, user manual, final report |

## **Feasibility Analysis**

* Technical: Open-source tools (TensorFlow, Tesseract) to minimize costs.
* Economic: Core system development and testing can be completed with minimal

financial resources by leveraging student developer time and free datasets.

## 1.8 Project Schedule/Timeline

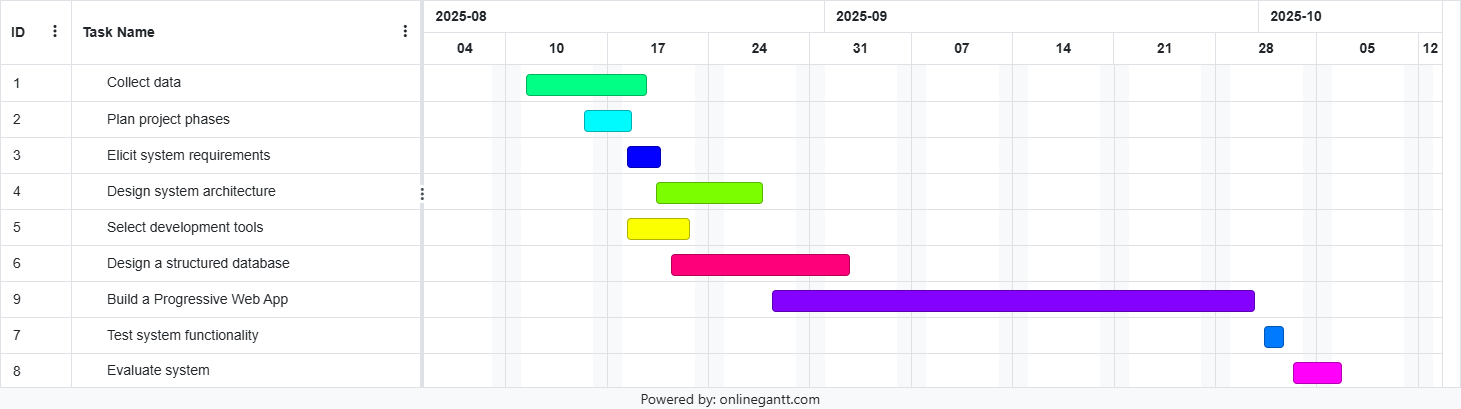


Figure 1.1 Gantt Chart

# Chapter Two - System Software / System Requirements Analysis and Specification

## 2.1Introduction / Overview

This chapter presents the results of the system requirements analysis and specification phase for the ALPHA Intelligent Health Companion and Monitoring System. The purpose is to fully define the functionality, quality attributes, and analysis-level models that describe the system. It builds upon the ideas introduced in the proposal document and provides a more structured breakdown of how the system is expected to perform and what it must deliver.

This system aims to support users in managing their day-to-day health by combining health data like vital signs and symptoms with intelligent recommendations powered by AI. The system includes modules for symptom logging, ovulation tracking, medication decoding, and health trend analysis.

## 2.2 Problem Context

In Ethiopia and similar low-to-middle-income settings, access to regular medical consultations is often limited due to economic or systemic constraints [1]. People ultimately rely more on self-diagnosis or unverified home remedies, which often result in recurrent chronic diseases, elevated symptoms, and delayed treatment.

At the same time, there is a growing use of internet services, yet existing health applications remain fragmented, non-localized, and lack proper integration with intelligent systems. Most of these tools fail to deliver personalized insights, reminders, or feedback, especially not in a way that considers the user's health history and local context [2].

This project addresses that gap by creating a unified, AI-assisted health companion that is designed to provide personalized lifestyle guidance, promote preventive care, and empower individuals with meaningful interpretations of their health data while respecting privacy and minimizing clinical overreach [3].

## 2.3 Purpose of the System

The purpose of the ALPHA Intelligent Health Companion and Monitoring System is to empower individuals to take an active role in managing their personal health. Users can track symptoms, check vital signs, understand medications, and, if necessary, follow their reproductive cycles with the help of the system's features. The system builds a basis for more relevant and personalized insights that promote healthy everyday choices by collecting and organizing these data sources.

The system does more than just save health data; it uses artificial intelligence to evaluate user information and provide personalized, non-clinical lifestyle recommendations. To keep users informed and involved in their health management, it also includes features like medication reminders and wellness routines. Although the system is a helpful companion, it is not intended to take the place of expert medical advice; rather, it is intended to raise users' awareness and provide support so they can get the right care when needed.

## 2.4 Current System Description

Currently, individuals rely on fragmented health tracking applications or manual notetaking to monitor symptoms, medications, and lifestyle routines. There are specific applications for things like tracking ovulation, medicine reminders, and symptom logging, but they don't share data or offer integrated insights. These approaches lack intelligence, personalization, and integration across different health aspects. For example, Symptom tracking apps often offer only static dropdowns and no real recommendations. Ovulation trackers are isolated from other health metrics like sleep or stress. Medication reminders lack intelligence about interactions or timing. These disjointed systems lead to poor health self-management, especially in underserved regions where access to physicians is expensive or limited.

These methods suffer from a major absence of intelligence and personalization, and this fragmentation results in inefficient self-management. Without the ability to analyze data, these systems cannot identify trends, make proactive suggestions, or notify users of any irregularities based on their individual medical histories. People are forced to sort through multiple data points on their own, frequently turning to questionable sources on the internet, which can lead to missed early warning signs and delayed treatment. By allowing preventable problems to become worse, this fragmented system ultimately puts extra burden on clinical facilities while falling short to empower users.

## 2.5 Proposed System Description

The proposed system is a modular web application that serves as a personal health companion. It collects user data on vital signs, medications, symptoms, sleep, and reproductive cycles, and uses AI integrations to generate non-diagnostic, lifestyle-oriented feedback.

### 2.5.1 Functional Requirements

* The system should allow users to register, log in securely, and manage their profile.
* The system shall accept and store health data including vitals, symptoms, medications, Sleep patterns, allergies, and chronic conditions.
* The system shall process user data with machine learning models to identify patterns, anomalies, and trends to generate summaries and recommendations based on AI output in natural language and dashboards.
* The system shall allow users to set and receive reminders for medications and lifestyle activities.
* The system shall provide calendar-based ovulation tracking for women.
* The system shall decode symptoms, medications and supplements into readable advice using GPT.
* The system shall store all health data securely and comply with data privacy guidelines.

### 2.5.2 Non-Functional Requirements

* The system should retain user data and logs even after logout or connection loss.
* Personal data must be encrypted, at rest and during transmission.
* The architecture should allow adding new health modules without major rewrites.
* The systems scalable architecture should support more users and records as it grows.
* The system should be available at least 90% of the time during testing phase.

### 2.5.3 Business Rules

* The system should not provide medical diagnoses. All recommendations must be labeled as non-clinical advice.
* All AI-generated recommendations shall include a disclaimer: ‘This is not medical advice. Consult a healthcare provider for professional diagnosis.’
* Vital sign inputs outside medically accepted ranges should trigger warnings but not diagnoses.
* Symptom analysis responses should include a disclaimer: "Consult a healthcare professional for diagnosis."
* Medication reminders should persist until manually dismissed or confirmed as taken.
* All users must consent to data collection during registration.
* The system shall comply with Ethiopia’s PDPP (2024) for data protection.
* All users must consent to data collection during registration.
* Users must complete a basic health profile before accessing AI recommendations.

## 2.6 Use Case Modeling

This section outlines the key use cases for the ALPHA Intelligent Health Companion. The use cases are based on system functionality identified in the requirement analysis and describe how users interact with each component of the platform. Each use case captures a functional interaction between the system and its actors.

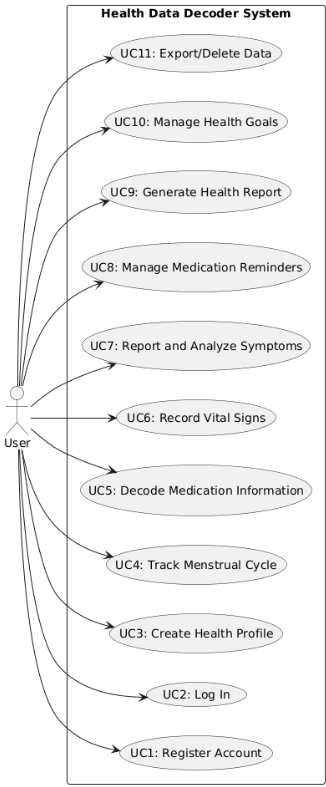
### 2.6.1 Actors

* **New User:** Unregistered individual creating an account.
* **User**: Registered and authenticated person who interacts with the system**.**
* **Female User:** Specialization of *User* with access to menstrual/ovulation tracking features.

### 2.6.2 List of Main Use Cases

* UC1 – Register Account
* UC2 – Log In
* UC3 – Create Health Profile
* UC4 – Track Menstrual Cycle (Female User only)
* UC5 – Decode Medication Information
* UC6 – Record Vital Signs
* UC7 – Report and Analyze Symptoms
* UC 8 – Manage Medication Reminders
* UC9 – Generate Health Report
* UC10 – Manage Health Goals
* UC11 – Export/Delete Data

### 2.6.3 Use Case Diagram



*Figure 01: Use case diagram*

### 2.6.4 Use Case Descriptions

| **Use Case** | **UC1 – Register Account** |
| --- | --- |
| **Actor** | **New User** |
| **Goal** | **Create a verified account to enable secure access.** |
| **Preconditions** | **User has a valid email or phone number.** |
| **Main Flow** | **1. User selects “Register.”**  **2. System displays registration form.**  **3. User enters email/phone, password, and basic info.**  **4. System validates input.**  **5. System creates account and sends verification code/link.**  **6. User verifies account.**  **7. System confirms success.** |
| **Alternative Flows** | **• Invalid information → System shows error.**  **• Verification fails → User may request resend.** |

| **Use Case** | **UC2 – Log In** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Authenticate to access personalized features.** |
| **Preconditions** | **Registered account exists.** |
| **Main Flow** | **1. User enters credentials.**  **2. System authenticates.**  **3. Dashboard is displayed.** |
| **Alternative Flows** | **• Invalid credentials → Error message.**  **• Forgotten password → Password recovery flow.** |

| **Use Case** | **UC3 – Create/Update Health Profile** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Establish a health baseline for insights.** |
| **Preconditions** | **User is logged in.** |
| **Main Flow** | **1. User opens “My Health Profile.”**  **2. System shows form (vitals, allergies, conditions, meds, sleep, ovulation if female).**  **3. User enters/updates information.4. System validates and saves data.5. System confirms success.** |
| **Alternative Flows** | **• Invalid data (e.g., unrealistic BP) → Warning message.** |

| **Use Case** | **UC4 – Track Menstrual Cycle** |
| --- | --- |
| **Actor** | **Female User** |
| **Goal** | **Predict and monitor menstrual cycles and fertile windows.** |
| **Preconditions** | **User logged in, identified as female.** |
| **Main Flow** | **1. User selects “Cycle Tracker.”**  **2. Marks start/end dates.**  **3. May add symptoms (cramps, mood).**  **4. System analyzes and predicts future cycles.**  **5. Predictions displayed with reminders.** |
| **Alternative Flows** | **• Insufficient history → System asks for more data before predictions.** |

| **Use Case** | **UC5 – Decode Medication Information** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Provide plain-language info about medications.** |
| **Preconditions** | **User logged in.** |
| **Main Flow** | **1. User selects “My Medications.”**  **2. System lists medications.**  **3. User selects or enters new one.**  **4. System displays: purpose, side effects, interactions, instructions.** |
|  |  |
| **Use Case** | **UC6 – Record Vital Signs** |
| **Actor** | **User** |
| **Goal** | **Detect health trends and anomalies.** |
| **Preconditions** | **User logged in.** |
| **Main Flow** | **1. User selects “Record Vitals.”**  **2. Enters data (BP, HR, temp, glucose, weight) or syncs from wearable.**  **3. System validates and stores.**  **4. System analyzes for trends.**  **5. Alerts user if abnormal.** |
| **Alternative Flows** | **• Critical values → Urgent alert generated.** |

| **Use Case** | **UC6 – Record Vital Signs** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Detect health trends and anomalies.** |
| **Preconditions** | **User logged in.** |
| **Main Flow** | **1. User selects “Record Vitals.”**  **2. Enters data (BP, HR, temp, glucose, weight) or syncs from wearable.**  **3. System validates and stores.**  **4. System analyzes for trends.**  **5. Alerts user if abnormal.** |
| **Alternative Flows** | **• Critical values → Urgent alert generated.** |

| **Use Case** | **UC7 – Report and Analyze Symptoms** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Provide non-clinical symptom insights.** |
| **Preconditions** | **User logged in.** |
| **Main Flow** | **1. User selects “Report Symptoms.”**  **2. Enters symptoms in natural language.**  **3. Submits.4. System analyzes and matches patterns.**  **5. Recommendations displayed (lifestyle changes, when to seek care).** |
| **Alternative Flows** | **• Emergency-like symptoms → Strong alert to seek immediate care.** |

| **Use Case** | **UC8 – Generate Health Report** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Summarize health history for reflection or sharing.** |
| **Preconditions** | **Sufficient data exists.** |
| **Main Flow** | **1. User selects “Generate Report.”**  **2. System asks for period (week/month).**  **3. User selects.4. System compiles trends, symptoms, meds, goals.**  **5. Displays report with charts.** |

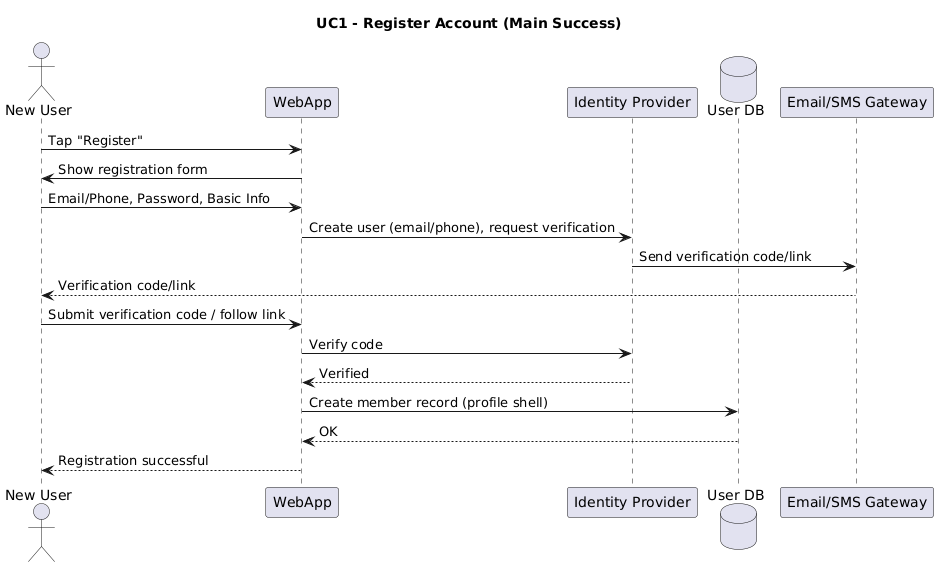
| **Use Case** | **UC9 – Manage Health Goals** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Set, track, and adapt personal wellness goals.** |
| **Preconditions** | **User logged in.** |
| **Main Flow** | **1. User selects “Set Health Goals.”**  **2. System displays categories (fitness, sleep, nutrition).**  **3. User creates or customizes goal.**  **4. System validates realism.**  **5. System begins tracking.**  **6. Sends reminders and progress updates.** |
| **Alternative Flows** | **• Unrealistic goal → Suggest adjustment.**  **• Consistent failure → Recommend revision.** |

| **Use Case** | **UC10 – Manage Medication Reminders** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Ensure adherence to medication schedule.** |
| **Preconditions** | **User logged in.** |
| **Main Flow** | **1. User adds medication + schedule.**  **2. System schedules reminders.**  **3. At reminder time, alert is sent.**  **4. User marks “Taken”/ “Skipped.”**  **5. System logs and tracks adherence.**  **6. Alerts user if refill is due.** |
| **Alternative Flows** | **• No response → Follow-up alert.** |

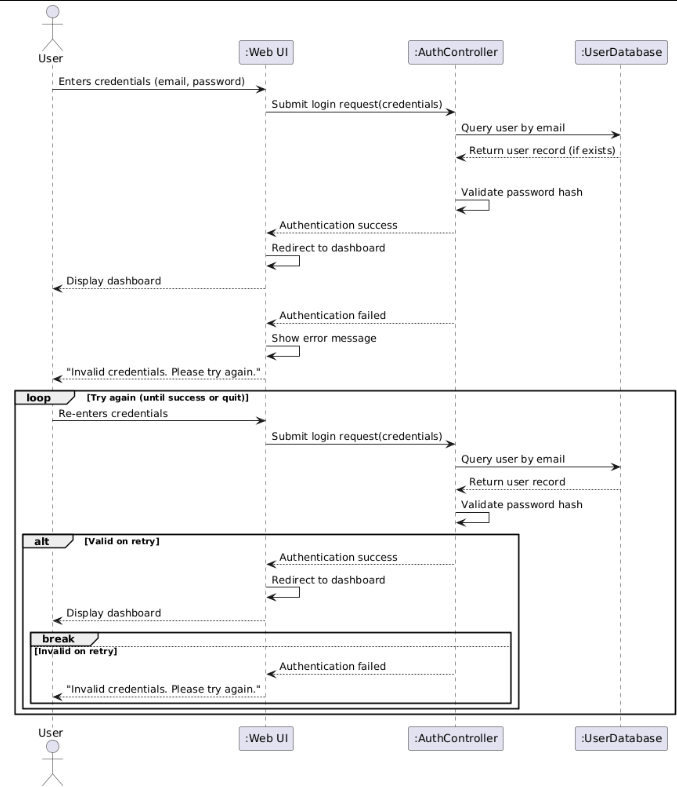
| **Use Case** | **UC11 – Export/Delete Data** |
| --- | --- |
| **Actor** | **User** |
| **Goal** | **Empower the user to control their personal data (e.g., GDPR compliance).** |
| **Preconditions** | **User is logged in.** |
| **Main Flow** | **1. User selects “Account & Data.”**  **2. System offers Export or Delete option.**  **3. Export: system compiles JSON/CSV archive → user downloads.**  **4. Delete: system requests re-authentication → user confirms → system deletes all data and revokes access.** |
| **Alternative Flows** | **• Export request fails → User retries.**  **• Delete canceled → Account remains active.** |

## 2.7 Sequence Diagrams

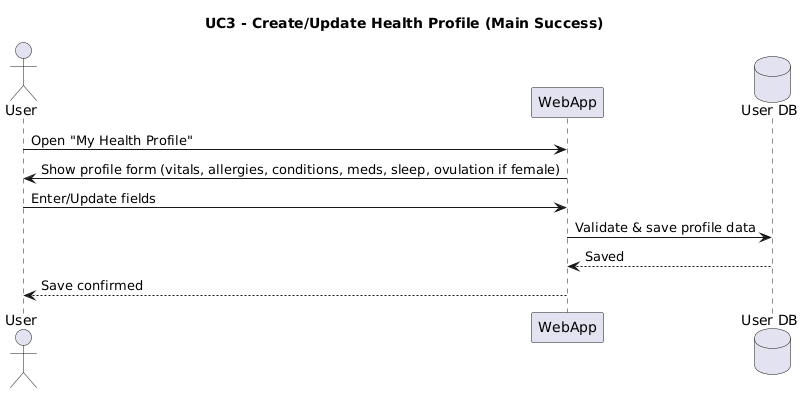
A sequence diagram is a type of interaction diagram that shows the interactions between objects or components of a system in chronological order. It shows the sequence of messages exchanged between different components or objects in the system to accomplish a specific task or scenario. The following sequence diagrams for the Alpha Intelligent Health Companion and Monitoring System project illustrate key use cases, providing a clear step-by-step visualization of system interactions.



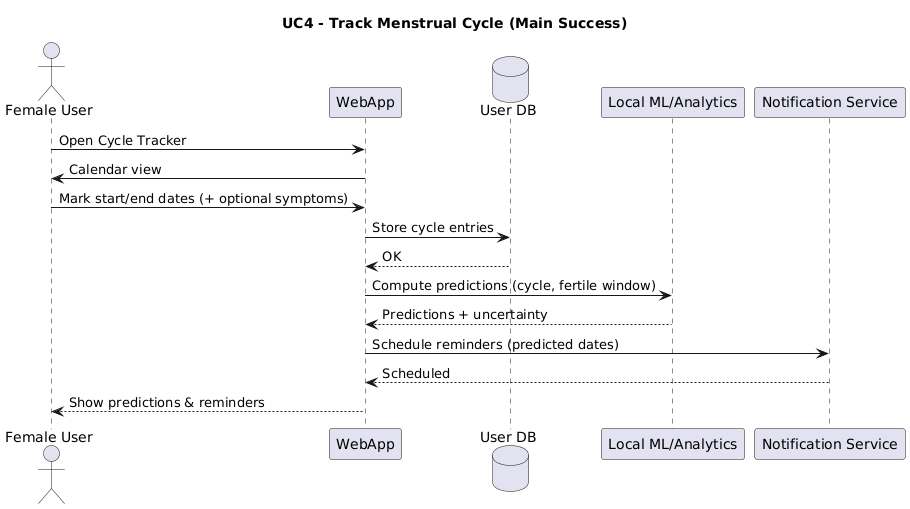
*Figure 02: Sequence Diagram for Register Account*



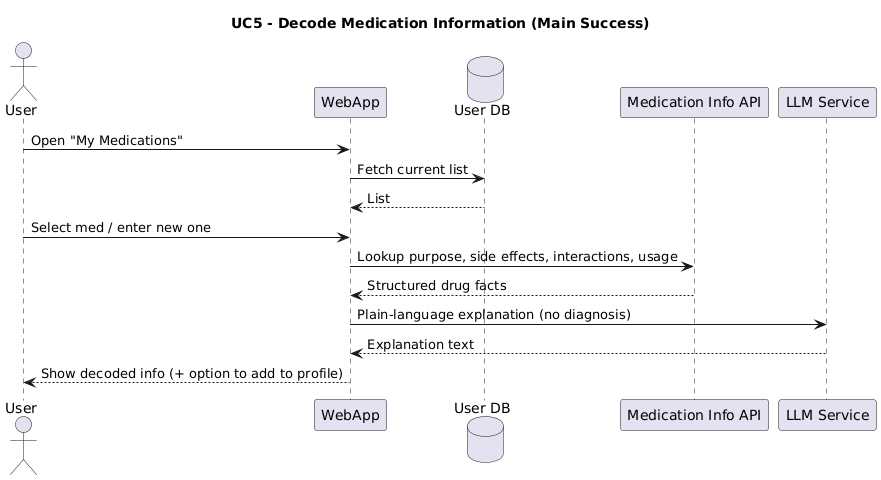
*Figure 03: Sequence Diagram for Log in*



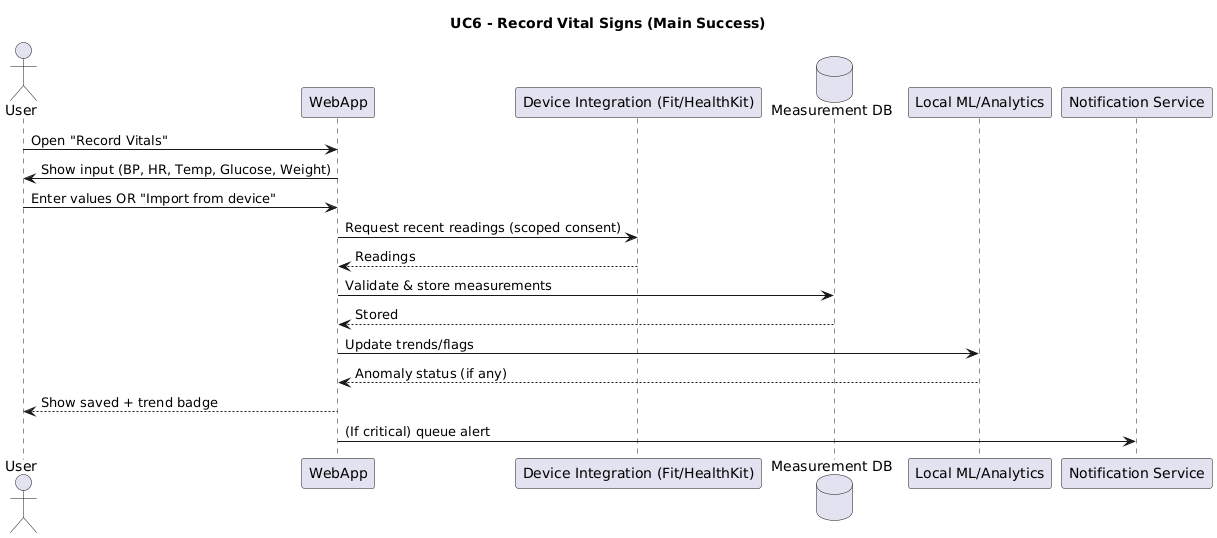
*Figure 04: Sequence Diagram for Create/Update Health Profile*



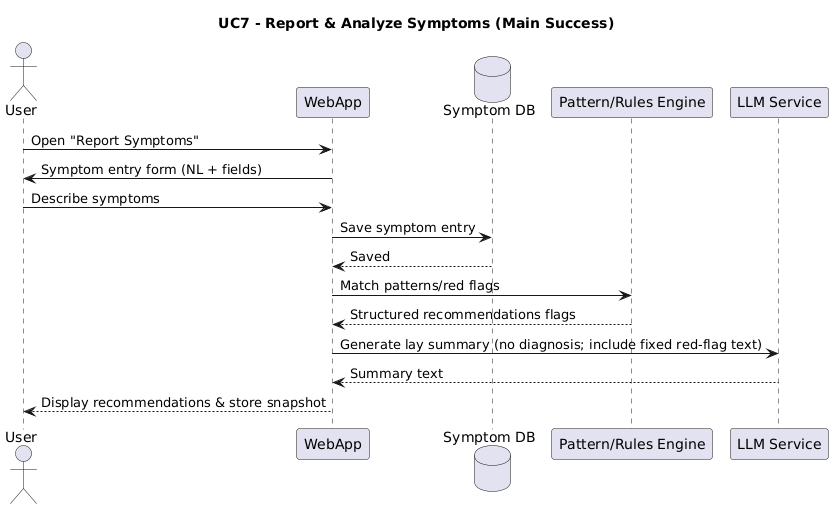
*Figure 05: Sequence Diagram for Track Menstrual Cycle*



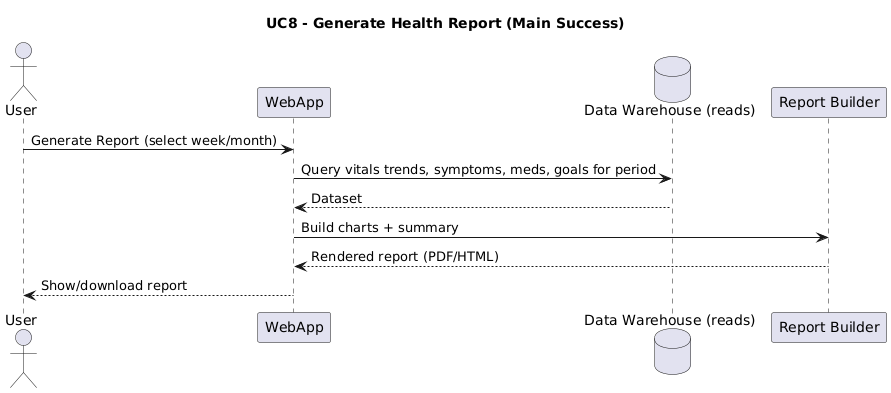
*Figure 06: Sequence Diagram for Decode Medical Information*



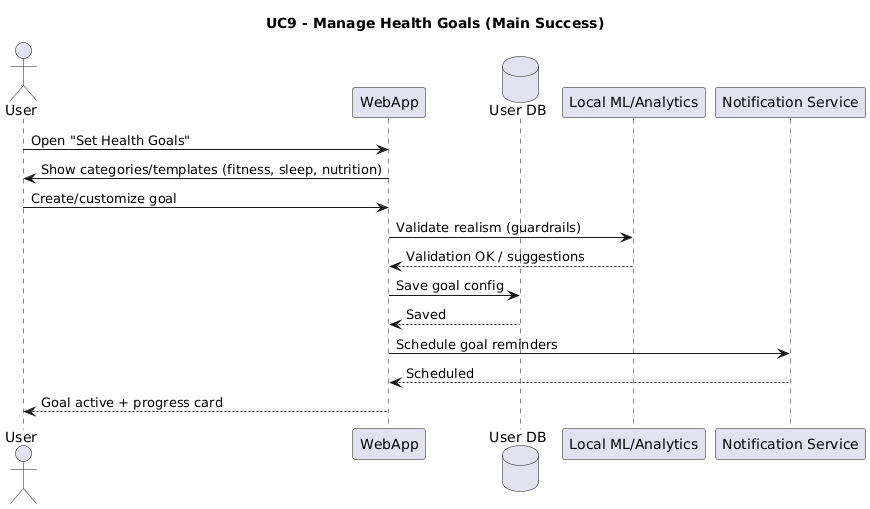
*Figure 07: Sequence Diagram for Record Vital Signs*



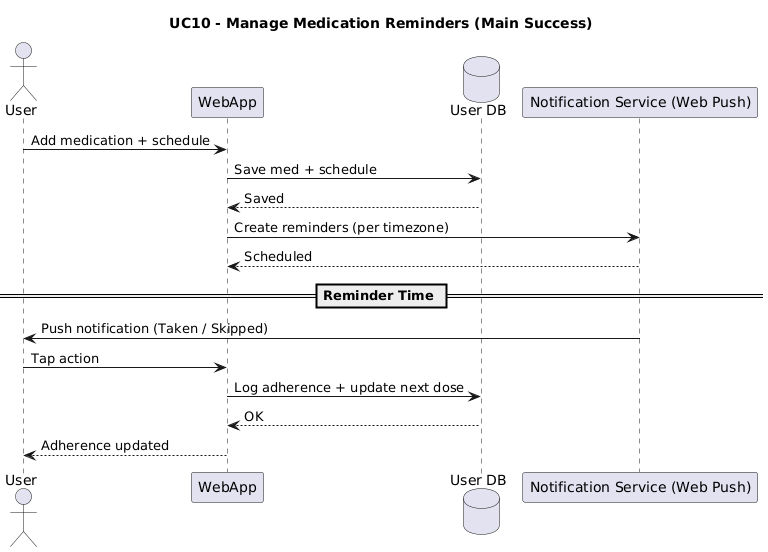
*Figure 08: Sequence Diagram for Report & Analyze Symptoms*



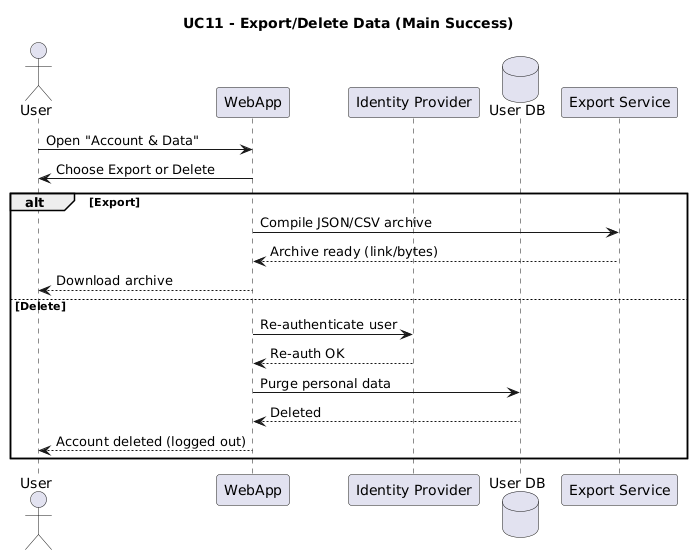
*Figure 09: Sequence Diagram for Generate Health Report*



*Figure 10: Sequence Diagram for Manage Health Goals*

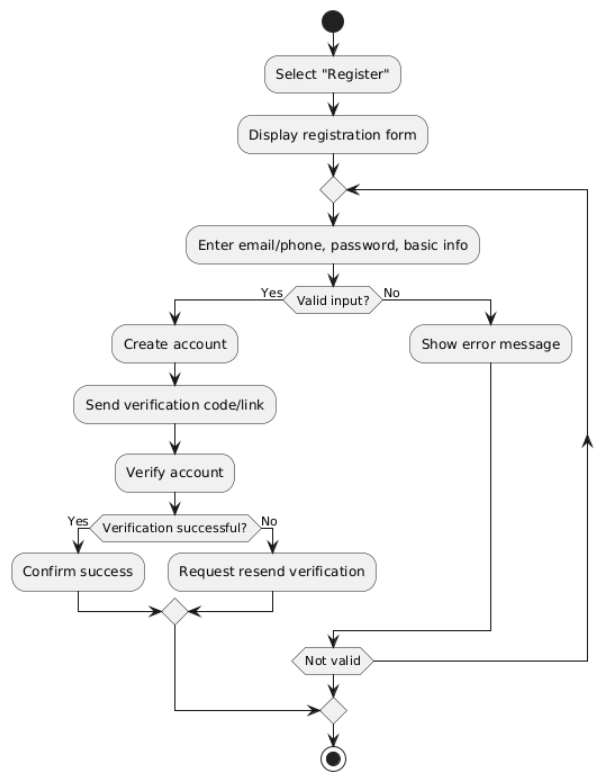


*Figure 11: Sequence Diagram for Manage Medication Reminders*

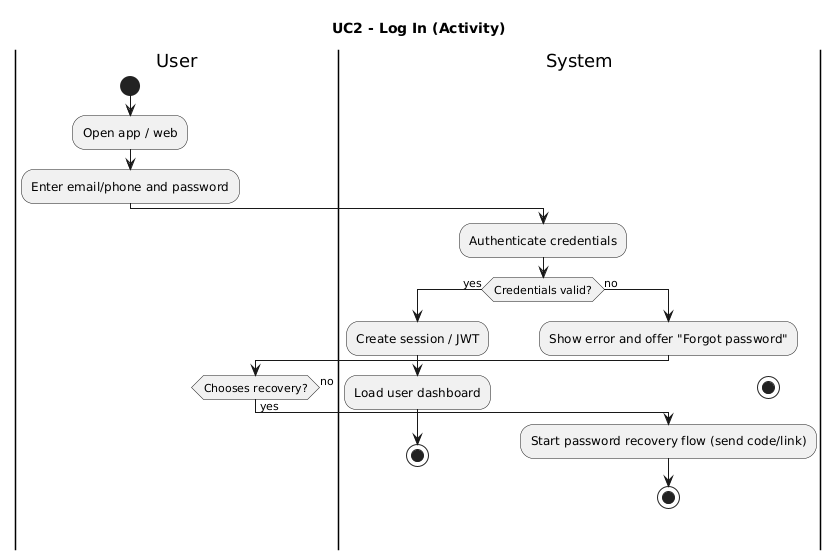


*Figure 12: Sequence Diagram for Export/Delete Data*

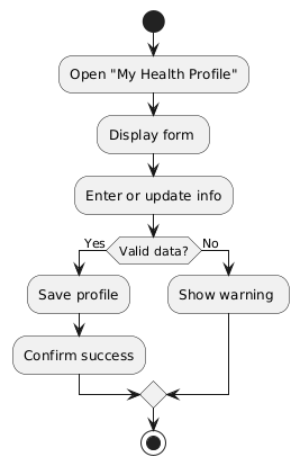
## 2.8 Activity Diagram



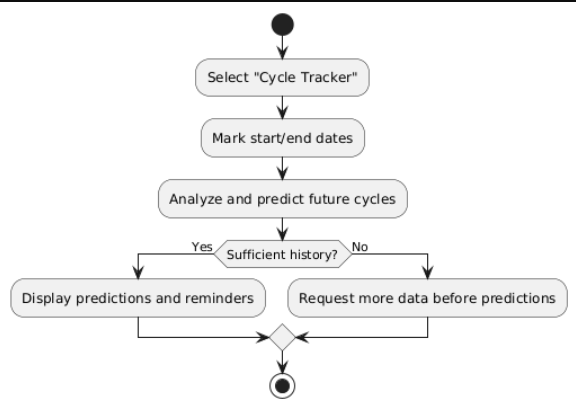
*Figure 13: Activity Diagram for Register Account*



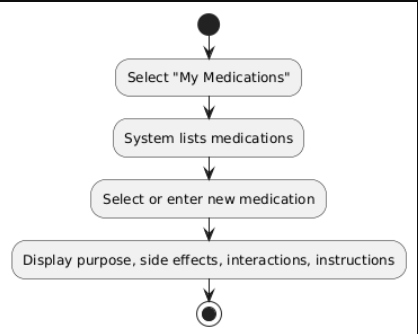
*Figure 14: Activity Diagram for Log in*



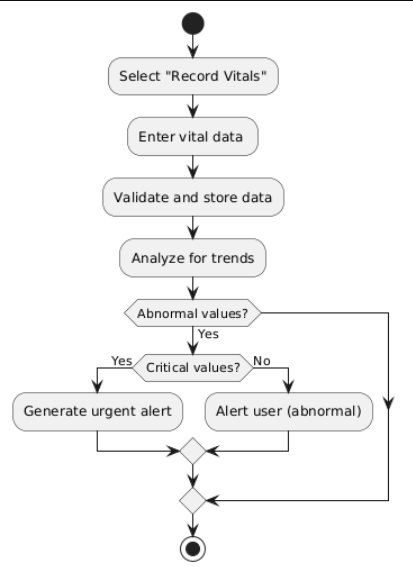
*Figure 15: Activity Diagram for Create/Update Health Profile*



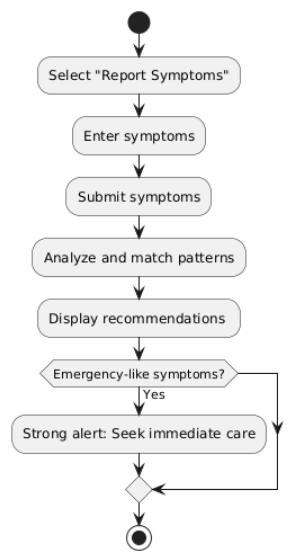
*Figure 16: Activity Diagram for Track Menstrual Cycle*



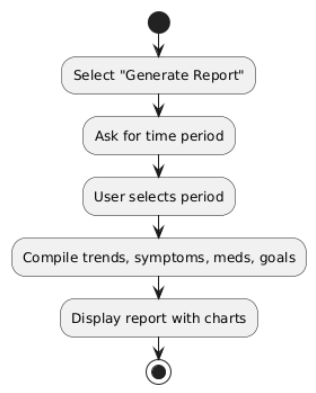
*Figure 17: Activity Diagram for Decode Medication Information*



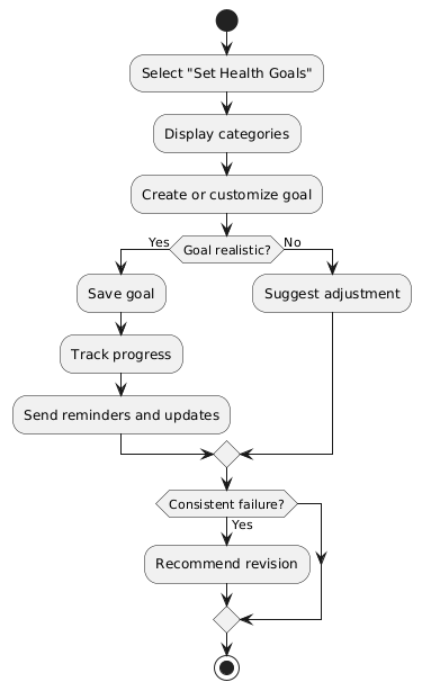
*Figure 18: Activity Diagram for Record Vital Signs*



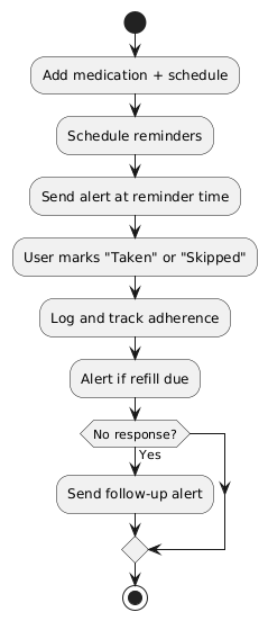
*Figure 19: Activity Diagram for Report and Analyze Symptoms*



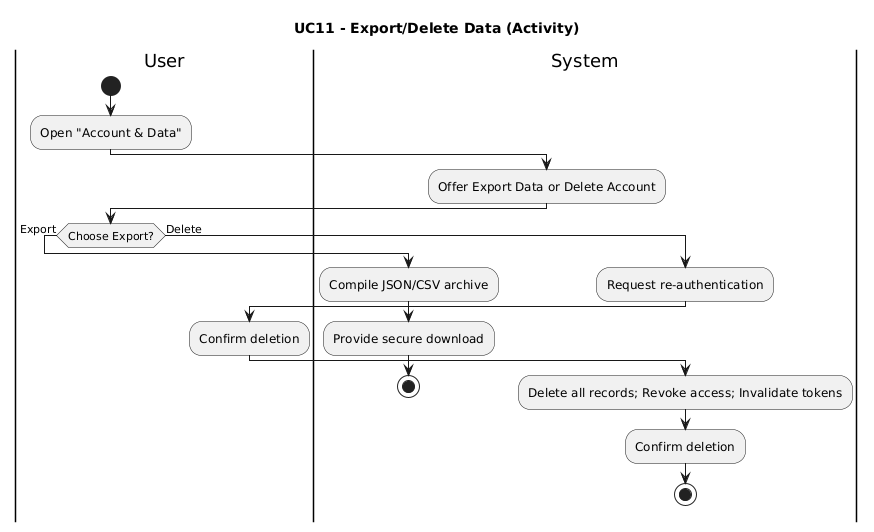
*Figure 20: Activity Diagram for Generate Health Report*



*Figure 21: Activity Diagram for Manage Health Goals*



*Figure 22: Activity Diagram for Manage Medication Reminders*



*Figure 23: Activity Diagram for Export/Delete Data*

# Chapter 3 – System Design and Architecture

## 3.1 Introduction

This chapter details the design and architecture of the ALPHA Intelligent Health Companion and Monitoring System. The transition from requirements (Chapter 2) to design is a critical step in software engineering, as it provides the structural and behavioral blueprint necessary for implementation. The design incorporates both high-level architectural decisions and detailed object-oriented specifications to ensure that the system meets its functional, non-functional, and business rule requirements.  
  
The design is presented in two major parts: the System Design Document (SDD), which covers architectural decisions, subsystem decomposition, and component interactions, and the Object Design Document (ODD), which defines classes, packages, and their interfaces. By following the Object-Oriented Software Engineering (OOSE) approach, the design emphasizes modularity, extensibility, and traceability.

Designing health-related systems demands careful attention to privacy, compliance, and usability. The ALPHA system, targeting resource-constrained environments such as Ethiopia, must be lightweight, accessible via low-bandwidth connections, and capable of running on widely available devices. Furthermore, it must comply with Ethiopia’s Personal Data Protection Proclamation (PDPP, 2024), which enforces stringent rules regarding data collection, storage, consent, and deletion.

## 3.2 System Architecture Overview

The ALPHA system adopts a layered, service-oriented architecture that supports modular development and incremental deployment. The architecture not only separates concerns but also ensures that new functionalities such as additional health trackers or integration with national health systems can be incorporated without major rewrites.  
  
The architecture follows the 4+1 view model, which describes the system from logical, process, physical, and developmental perspectives. Each view provides stakeholders with insights into how the system fulfills its requirements.

### 3.2.1 Architectural Layers

The architecture is divided into four primary layers:

* Presentation Layer: Implemented as a Progressive Web Application (PWA), it provides responsive user interfaces accessible from smartphones and desktops. It supports offline caching, push notifications, and installability. This layer ensures accessibility even in areas with intermittent connectivity.
* Application Layer: Implements the business logic through RESTful services. Each service corresponds to a subsystem such as authentication, health data collection, symptom analysis, medication decoding, goals, reminders, and reporting. The application layer is containerized, ensuring that each subsystem can be scaled independently.
* Data Layer: A PostgreSQL relational database stores structured health data including vitals, symptoms, medications, and profiles. The database design enforces referential integrity, indexing for performance, and encryption for security. An object store (e.g., MinIO or S3-compatible storage) manages large report artifacts and exported data files.
* Integration Layer: Provides connectors to external systems, such as wearable device APIs (Google Fit, HealthKit) and third-party datasets. This layer also accommodates potential future integration with national electronic health record (EHR) systems, though initial deployment may restrict such integrations due to lack of standardized APIs.

### 3.2.2 Architectural Style

The architectural style combines a client-server model with service orientation. Clients (PWAs) interact with the system through REST APIs, which expose well-defined interfaces for each subsystem. The backend is designed as a set of loosely coupled services that can evolve independently. This choice ensures that as new health functionalities are needed (e.g., nutrition tracking), they can be added without disturbing existing modules.

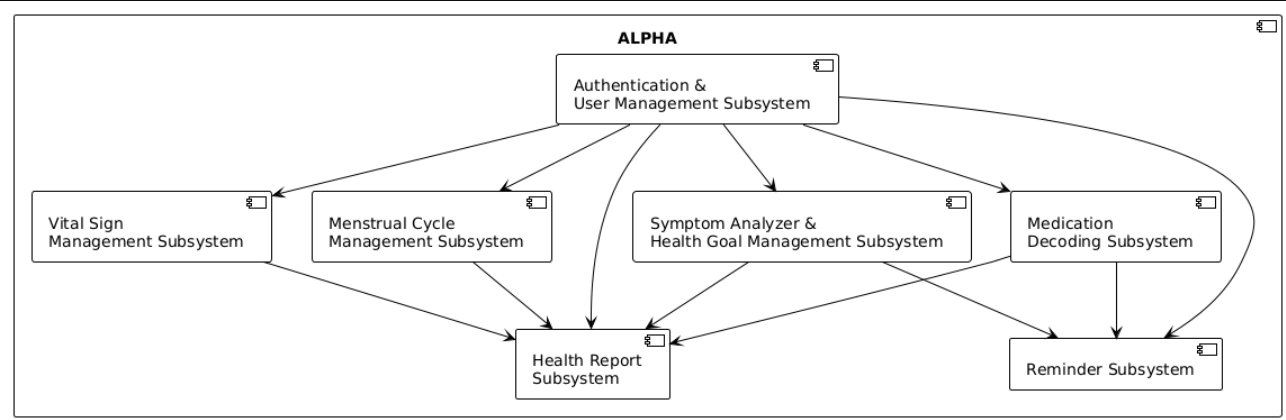
### 3.2.3 Cross-Cutting Concerns

* Security and Privacy: All communications use TLS. Sensitive data such as passwords are hashed using PBKDF2. Audit logs capture access events. Consent management ensures compliance with PDPP.
* Compliance: Features such as explicit consent prompts, disclaimers, data export, and data deletion are integral. These ensure compliance not only with PDPP but also with international standards like GDPR.
* Performance: API endpoints are optimized to respond within 2 seconds. Asynchronous processing is used for heavy tasks like report generation.
* Scalability: Containerized microservices allow independent scaling of subsystems like symptom analysis or reporting.
* Usability: The UI is designed with low-literacy users in mind. Icons, color coding, and simple messages aid accessibility.

## 3.3 Subsystem Decomposition

The ALPHA system is divided into ten major subsystems. Each subsystem is described with its responsibilities, interfaces, dependencies, and rationale. These various subsystems interact to ensure smooth operations and efficient management. The system we have developed has the following subsystems:

* Authentication & User Management Subsystem
* Vital Sign Management Subsystem
* Symptom Analyzer & Health Goal Management Subsystem
* Menstrual Cycle Management Subsystem
* Medication Decoding Subsystem
* Reminder Subsystem
* Health Report Subsystem



1. Authentication & User Management Subsystem

* Responsibilities: Manages user registration, login, password recovery, and multi-factor authentication, as well as storing and updating user demographics, allergies, chronic conditions, and baseline vitals.
* Interfaces: /auth/\*, /profiles/\*
* Dependencies: Provides identity verification and profile data to all other subsystems.
* Notes: Security hardening is applied. User data supports AI-driven personalization.

1. Vital Sign Management Subsystem

* Responsibilities: Handles manual entry and device-synced vital signs. Validates ranges, logs history, and supports longitudinal tracking.
* Interfaces: /vitals/\*, /devices/\*
* Dependencies: Relies on user data for baseline validation and provides data to the Health Report Subsystem.

1. Symptom Analyzer & Health Goal Management Subsystem

* Responsibilities: Accepts free-text input for symptoms, applies NLP, flags red alerts, stores symptom history, and allows users to define and monitor goals (fitness, nutrition, sleep). Tracks adherence and progress.
* Interfaces: /symptoms/\*, /goals/\*
* Dependencies: Integrates with Vital Signs for contextual tracking and with the Health Report Subsystem for summaries.

1. Menstrual Cycle Management Subsystem

* Responsibilities: Predicts and monitors menstrual cycles and fertile windows, logging related symptoms.
* Interfaces: /cycletracker/\*
* Dependencies: Provides cycle and fertility data to the Health Report Subsystem.

1. Medication Decoding Subsystem

* Responsibilities: Provides plain-language explanations of prescribed medications. May integrate external APIs/LLMs for advanced decoding.
* Interfaces: /meds/\*
* Dependencies: Shares medication data with Reminder and Health Report Subsystems.

1. Reminder Subsystem

* Responsibilities: Schedules and manages reminders for medications and health goals. Supports push notification APIs for timely alerts.
* Interfaces: /reminders/\*
* Dependencies: Integrates with Medication and Goal subsystems for context-specific reminders.

1. Health Report Subsystem

* Responsibilities: Aggregates data from all subsystems into weekly and monthly reports. Supports export in PDF/CSV formats.
* Interfaces: /reports/\*
* Dependencies: Relies on Vitals, Symptoms/Goals, Cycle Management, and Medication subsystems for input data.

## 3.4 Object Design Document (ODD)

The Object Design Document specifies packages and classes, along with trade-offs made during design. It provides detailed structures for developers.

### 3.4.1 Trade-offs

Several trade-offs were considered:  
- Composition vs. Inheritance: Composition chosen for flexibility.  
- Privacy vs. Usability: Explicit consent prompts versus user convenience.  
- Consistency vs. Latency: Reports generated asynchronously for consistency, vital entries immediate for low latency.

### 3.4.2 Package Diagram

**Core provider**

1. **Authentication & User Management**
   * Provides identity + profile context to all other subsystems.
   * No upstream dependencies called out.

**Application subsystems**

1. **Vital Sign Management**
   * Depends on → Authentication & User Management (baseline/user context).
   * Provides data to → Health Report.
   * Interfaces: /vitals/\*, /devices/\*.
2. **Symptom Analyzer & Health Goal Management**
   * Depends on → Vital Sign Management (contextual tracking).
   * Provides summaries to → Health Report.
   * (Also mentions an NLP engine, but that’s an external concern, not a package here.)
   * Interfaces: /symptoms/\*, /goals/\*.
3. **Menstrual Cycle Management**
   * Depends on → Authentication&User Management (demographics).
   * Provides data to → Health Report.
   * Interface: /cycletracker/\*.
4. **Medication Decoding**
   * Provides data to → Reminder, HealthReport.
   * Interface: /meds/\*.
5. **Reminder**
   * Depends on → Medication Decoding and Goals (from Symptom/Goal subsystem) for context-specific reminders; also interacts with User Management for user targeting.
   * Interface: /reminders/\*.
6. **Health Report**
   * Depends on → Vital Sign Management, Symptom Analyzer& Health GoalManagement, MenstrualCycleManagement, MedicationDecoding for input data aggregation.
   * Interface: /reports/\*.

### 3.4.3 Key Classes

In The ALPHA system, the class interfaces play a vital role in defining the interactions and functionalities of various components. Here is an overview of each class, their dependencies, public attributes, operations, and potential exceptions they can raise:

Authentication and User Management Classes

* Interfaces: Define interfaces for user registration, login, password recovery, multi-factor authentication, and profile management.
* Dependencies: Provides identity verification and profile data to all other subsystems (Vitals, Symptoms, Reports, etc.).
* Public Attributes: Username, password (hashed), recovery email, demographics, allergies, chronic conditions, baseline vitals.
* Operations: RegisterUser(), Login(), RecoverPassword(), EnableMFA(), CreateProfile(), UpdateProfile(), DeleteProfile(), GetProfile().
* Exceptions: AuthenticationFailedException, UserAlreadyExistsException, InvalidRecoveryTokenException, ProfileNotFoundException, InvalidProfileDataException.

Vital Sign Management Classes

* Interfaces: Define interfaces for recording and retrieving vital signs, including manual entry and device-synced data.
* Dependencies: Interacts with User Management for baseline ranges, and Report Classes for aggregation.
* Public Attributes: Vital type (e.g., blood pressure, heart rate), value, timestamp.
* Operations: RecordVital(), GetVitalsHistory(), SyncDeviceVitals().
* Exceptions: InvalidVitalRangeException, VitalNotFoundException.

Symptom Analyzer and Health Goal Management Classes

* **Interfaces:** Define interfaces for analyzing free-text symptom inputs, generating health alerts, and managing user goals such as fitness, nutrition, and sleep.
* **Dependencies:** Interacts with NLP engine, Vitals subsystem for progress tracking, and Reports for summaries.
* **Public Attributes:** Symptom text, severity level, flagged alerts, goal type, target value, progress status.
* **Operations:** AnalyzeSymptom(), GetSymptomHistory(), CreateGoal(), UpdateGoal(), TrackGoalProgress(), GetGoals().
* **Exceptions:** InvalidSymptomInputException, SymptomAnalysisException, GoalNotFoundException, InvalidGoalException.

Menstrual Cycle Management Classes

* Interfaces: Define interfaces for predicting and tracking menstrual cycles and fertile windows.
* Dependencies: Interacts with Report Classes for monthly summaries and User Management for demographics.
* Public Attributes: Cycle start date, predicted fertile window, notes.
* Operations: PredictCycle(), LogCycleData().
* Exceptions: CycleDataNotFoundException, InvalidCycleDataException.

Medication Decoding Classes

* Interfaces: Define interfaces for decoding medical prescriptions into plain language.
* Dependencies: May integrate with external APIs/LLMs, interacts with Report Classes for medication summaries.
* Public Attributes: Medication code, name, description, dosage.
* Operations: DecodeMedication(), GetPlainLanguageDescription().
* Exceptions: MedicationNotFoundException, InvalidMedicationCodeException.

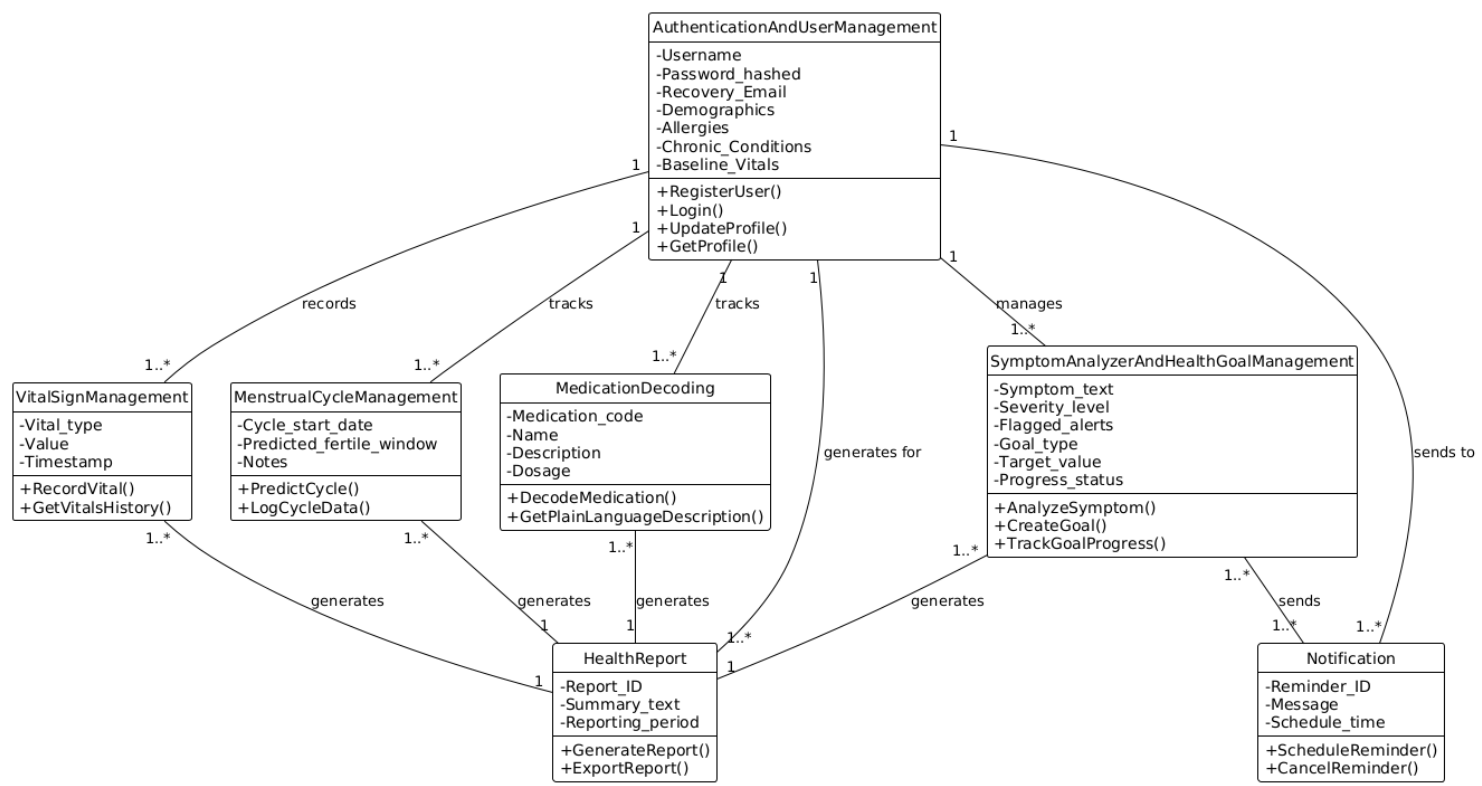
Notification Classes

* Interfaces: Define interfaces for creating, scheduling, and cancelling reminders for medications and goals.
* Dependencies: Interacts with User Management and Goal Classes for context-specific reminders.
* Public Attributes: Reminder ID, message, schedule time.
* Operations: ScheduleReminder(), CancelReminder(), GetReminders().
* Exceptions: ReminderNotFoundException, InvalidReminderScheduleException.

Health Report Classes

* Interfaces: Define interfaces for aggregating and exporting user health data into reports.
* Dependencies: Interacts with Vitals, Symptoms, Goals, and Cycle Management for data aggregation.
* Public Attributes: Report ID, summary text, reporting period (weekly/monthly).
* Operations: GenerateReport(), ExportReport().
* Exceptions: ReportNotFoundException, ReportExportException.

### 3.4.4 Class Diagram



## 3.5 Design Constraints and Quality Attributes

The design must satisfy quality attributes across security, scalability, performance, and compliance.

* Security: Encryption in transit and at rest, secure coding practices, audit logging.
* Scalability: Ability to add new modules without disrupting existing services.
* Performance: API responses within 2 seconds; reports processed asynchronously.
* Compliance: Explicit support for PDPP (2024), including user rights for export and deletion.
* Reliability: Offline caching, retries, and redundant infrastructure.

# References

[1] WHO, “Global Health Observatory Data Repository,” [Online]. Available: https://www.who.int/data/gho. [Accessed: 16-Jul-2025].

[2] GSMA, “The Mobile Economy Sub-Saharan Africa 2024,” GSMA Intelligence, 2024.

[3] J. Brownlee, “A Gentle Introduction to LSTM Recurrent Neural Networks,” Machine Learning Mastery, 2021. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/. [Accessed: 16-Jul-2025].

[4] TensorFlow Developers, “TensorFlow Guide,” [Online]. Available: https://www.tensorflow.org/guide. [Accessed: 16-Jul-2025].

[5] Kaggle, “Fitbit Fitness Tracker Data,” [Online]. Available: https://www.kaggle.com/datasets. [Accessed: 16-Jul-2025].

[6] Government of Ethiopia, “Personal Data Protection Proclamation (PDPP) 2024,” Federal Negarit Gazeta, 2024.

[7] PhysioNet, “MIMIC-III Clinical Database,” [Online]. Available: https://physionet.org/content/mimiciii/1.4/. [Accessed: 16-Jul-2025].

Here’s the full reference for the internet access statistics in Ethiopia:

**[8] DataReportal.** (2024). Digital 2024: Ethiopia. We Are Social & Meltwater. Retrieved from [DataReportal – Digital 2024: Ethiopia](https://datareportal.com/reports/digital-2024-ethiopia).