ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCES DEPARTMENT OF COMPUTER SCIENCE

MERI ETHIOPIAN FITNESS AND NUTRITION WEB APP

COMPUTER SCIENCE: FINAL YEAR PROJECT

BY

NAME OF STUDENTS	ID
1.DURESA GUTA	NSE/7330/13
2.NATNAEL SEMA	NSE/7373/13
3.YEABKAL SOLOMON	NSE/4299/13
4.ESRAEL DANIEL	NSE/0919/13

PROJECT ADVISOR: ASHENAFI KASSAHUN

EXAMINER : NESREDIN

CERTIFICATE

I certify that this BSc Final project entitled “Meri Ethiopian Fitness and Nutrition Web App ” by:

1. Duresa Guta
2. Esrael Daniel
3. Natnael Sema
4. Yeabkal solomon

is approved by me for submission. I certify further that, to the best of my knowledge, the report represents work carried out by the students.

Feb 03 2025 Ashenafi Kassahun

_____ _____

Date Name and Signature of Supervisor

Acknowledgment

First of all, we would like to thank our almighty GOD for giving us strength to complete our project. Then we owe our deep gratitude and sincere esteem to our advisor Instructor Ashenafi for his constructive opinion and willingness to participate in each part of our project and his effective direction, assistance and guidance. We are extremely thankful for providing us such a nice support and guidance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

List of Abbreviations

API	Application Programming Interface
BMI	Body Mass Index
CSS	Cascading Style Sheet
DB	Database
FR	Functional Requirement
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
HTTP	HyperText Transfer Protocol
ID	Identify Content
IIS	Internet Information Service
JS	JavaScript
OS	Operating System
PHP	HyperText Preprocessor
QA	Quality Assurance
RAM	Random Access Memory
SQL	Structured Query Language
UAT	User Acceptance Testing
UML	Unified Modified Language
UX	User Experience

CHAPTER ONE

1.INTRODUCTION

In recent years, global awareness surrounding health and wellness has increased dramatically. People are becoming more conscious of their dietary choices and physical activity levels. However, this shift towards healthier living has not been adequately supported by digital tools tailored to specific cultural contexts, particularly in Ethiopia. The absence of comprehensive fitness and nutrition applications that resonate with Ethiopian food culture is a significant gap in the market.

Current fitness apps predominantly focus on Western dietary habits and exercise regimens, which may not align with Ethiopian eating patterns or lifestyle practices. For instance, traditional meals are often high in carbohydrates and low in processed foods, which can be beneficial when balanced with appropriate physical activity. Yet, without guidance on how to adapt these traditional diets to modern nutritional standards, many individuals may struggle to achieve their health goals.

In the contemporary fitness and nutrition landscape, most available apps are designed with a more Western-centric approach. They often overlook the unique dietary habits and fitness needs of non-Western societies, including Ethiopia. This creates a disconnect for Ethiopian users who wish to maintain their health and fitness using tools that resonate with their cultural background. Traditional Ethiopian meals, which are naturally nutritious and balanced, are not reflected in these global fitness apps, leading to an underrepresentation of Ethiopian food culture in the digital health space.

In conclusion, as Ethiopia navigates the complexities of modern health challenges, there is an urgent need for digital solutions that respect and incorporate its rich cultural tapestry. The "meri Ethiopian Fitness and Nutrition Web App" seeks to fill this void by fostering a holistic approach to fitness and nutrition that resonates with the Ethiopian populace.

1.1 Statement Of The Problem And Justification

The current market for fitness and nutrition apps is saturated with options that are predominantly Western-centric, offering solutions that often do not resonate with the cultural and dietary habits of non-Western users. For Ethiopian users, this presents a significant problem. The absence of a fitness and nutrition app that incorporates Ethiopian food culture results in several challenges that need to be addressed.

1.1.1 Statement Of Problem

The landscape of health and wellness in Ethiopia is evolving; however, it faces significant challenges due to the lack of culturally relevant digital resources. The primary problem lies in the absence of dedicated fitness and nutrition applications that cater specifically to Ethiopian dietary habits and lifestyle choices. This gap presents several issues:

- **Lack of Cultural Relevance**: The majority of fitness and nutrition apps fail to consider the unique dietary practices of Ethiopian cuisine. Traditional Ethiopian foods, which are rich in nutrients and culturally significant, are not represented in these apps. This lack of representation forces users to adapt their dietary habits to fit the app's suggestions, leading to a disconnect between their cultural food practices and their fitness plans.
- **Limited Access to Information**: Without localized resources, individuals seeking to improve their health may struggle to find reliable information about how traditional foods can fit into a balanced diet or how they can engage in physical activities that resonate with their cultural practices.
- **Accessibility and Affordability:** Many existing fitness and nutrition apps are not affordable for individuals with limited income and financial resources. Additionally, these apps may require access to gym equipment or other resources that are not readily available to all users, particularly those in rural areas.
- **Educational Gap:** There is a lack of educational content on how to incorporate traditional Ethiopian meals into a fitness and nutrition plan. Users are often unaware of the nutritional value of their traditional foods and how these can be used to achieve their fitness goals.

1.1.2 Justification Of The Work

Developing the meri Ethiopian Fitness and Nutrition Web App is crucial for several reasons:

- **Cultural Preservation and Promotion:** By integrating traditional Ethiopian foods into the app, we promote cultural heritage and ensure that users can maintain their dietary traditions while pursuing their fitness goals. This cultural relevance will enhance user engagement and satisfaction.
- **Affordability and Accessibility:** The app will be designed to be affordable and accessible to users with varying financial resources. It will include workout plans that do not require expensive gym equipment, making it feasible for users in different settings.
- **Educational Value:** The app will serve as an educational tool, providing users with valuable information on the nutritional content of traditional Ethiopian foods and how to incorporate these into their fitness journeys. This knowledge will empower users to make informed decisions about their health and wellness.
- **Filling a Market Gap:** The development of this app addresses a significant gap in the market. By offering a solution that is tailored to the Ethiopian context, we provide a unique value proposition that sets the app apart from generic fitness and nutrition apps.

  In conclusion, the meri Ethiopian Fitness and Nutrition Web App is a necessary and valuable project that will not only promote healthier lifestyles but also celebrate and preserve Ethiopian food culture. It will provide users with a comprehensive, culturally relevant tool to achieve their fitness goals, ensuring that their journey towards health and wellness is both effective and meaningful.

1.2 Project Objective

1.2.1General Objective of The System

The primary goal of our project is to develop a web application that integrates Ethiopian food culture with modern fitness and nutrition principles. This app will serve as a holistic resource for individuals seeking to improve their health, fitness, and nutritional knowledge while celebrating their cultural heritage.

1.2.2 Specific Objective Of The System

Software Activities

1. **Requirement Analysis**

Activities:

- Gather specific requirements through surveys and interviews with potential users to understand their fitness goals, dietary habits, and cultural preferences.

Output: Requirement Specification Document.

2. **Planning**

Activities:

- Define the scope, timeline, and resources (hardware, software, and team members).

Output: Project Plan

3. **Design**

Activities:

- Create a database schema to store Ethiopian food (Meal) items and their nutritional information.

Output: System Architecture.

4. **Implementation (Coding)**

Activities:

- Implement the frontend using tools like HTML, CSS, JavaScript, Develop the backend with Python (Django), Node.js, or PHP and Integrate the database (MySQL, PostgreSQL, or MongoDB).

  Output: Source Code.

5. Testing

Activities:

- Perform unit testing on each module to ensure individual functionality.

Output: Test Report.

6. Deployment

Activities:

- Monitor initial user feedback for any technical or usability issues.

Output: Deployment Guide.

7. Maintenance

Activities:

- Implement patches or updates for security vulnerabilities and performance enhancements.

Output: Updated Versions

1.3 Scope Of The Project

Scope defines the coverage areas of the project, activities and operations done by the system.

The project will focus on:

- Creating a system for Ethiopian foods, their nutritional values, and meal plans that fit fitness goals.
- Offering a selection of fitness programs that include cardio exercises and weight lifting routines.
- Providing tips and educational resources on fitness and nutrition in the context of Ethiopian food culture.

The project will not cover:

- Detailed diet plans for international food cultures.
- Medical fitness recommendations (the app will be a guide, not a substitute for professional advice).

1.4 System Development Methodology

1.4.1 Data collection tools and techniques

Data collection is the most important part of the project to find the main requirement of the system and to understand how the system does and also to understand how the overall organization work flow looks like.

Conduct surveys or interviews with potential users to understand preferences regarding diet, fitness goals, and cultural expectations.

Research Ethiopian dietary customs, meal structures, and fitness practices to inform the app's design.

1.5 System Analysis And Design

Objective: To design an intuitive user interface and define the technical architecture that will best support the app's functionality.

Activities: Develop system architecture, including a database structure to store Ethiopian food data and fitness plans.

Design user flows to ensure seamless navigation and positive user experience (UX).

1.6 System Development

Our project will use an Agile development model to ensure flexibility and adaptability throughout the development process. This allows for iterative testing and feedback incorporation, ensuring the final product meets user needs.

1.7 System Development Tools

To support the development of the web app, the following tools will be utilized

Hardware Requirements:

Development Machine: A reliable computer with at least 8GB RAM and a modern processor (e.g., Intel i5) for development tasks.

Software Requirements:

Programming Languages: HTML, CSS, JavaScript for front-end development; Python, Node.js, or PHP for back-end development.

Frameworks and Libraries: React or Angular for front-end; Django or Express.js for back-end development.

Database: MySQL, PostgreSQL, or MongoDB for data storage.

1.8 Significance Of Project

The meri Ethiopian Fitness and Nutrition Web App holds significant importance for several reasons:

- **Cultural Relevance**: The app promotes and preserves Ethiopian food culture by integrating traditional dietary practices into modern fitness and nutrition plans.
- **Health and Wellness:** By providing fitness and nutrition plans, the app encourages healthier lifestyles and improves overall well-being for Ethiopian users.
- **Educational Value:** The app serves as an educational tool, raising awareness about the nutritional value of traditional Ethiopian foods and providing guidance on healthy living.
- **Accessibility**: The app is designed to be affordable and accessible to a wide range of users, ensuring that everyone, regardless of financial resources or geographic location, can benefit from its features.

1.9 Beneficiaries

- **General Public**: Ethiopian individuals seeking culturally relevant fitness and nutrition guidance.
- **Health Enthusiasts**: Users interested in maintaining a healthy lifestyle with a focus on Ethiopian food culture.
- **Healthcare Providers**: Doctors and dietitians may recommend the app to patients seeking culturally suitable health tools.
- **Health and Fitness Professionals**: Nutritionists, fitness coaches, and wellness experts who can use the app as a tool to provide tailored advice and support to their clients.

1.10 Time Schedule Of The Project

Gantt Chart

A Gantt chart is a visual project management tool that illustrates a project schedule. Project manager is also responsible for monitoring and controlling the project development based on the schedule shown below.

	SEMISTER 1				SEMISTER 2			
WEEKS	Oct30 - Nov6	Nov6- Nov13	Nov13- Nov20	Nov20- Dec4	Dec18 -Jan15	Jan15 - Jan29	Jan29- Feb12	Feb12 - Feb26
PHASES (PROJECT I)								
Proposal Stage								
Requirement Gathering								
Requirement Analysis								
Design								
FINAL PROJECT II								
Coding								
Testing								
Deployment (Install)								
Maintenance								

Table 1 : gantt chart

This Gantt chart may not be exact so the time is estimated or predicted.

CHAPTER TWO

2. REQUIREMENT ANALYSIS AND SPECIFICATION

2.1 Purpose Of The System

The purpose of our system is to develop a comprehensive web application that integrates Ethiopian food culture with fitness and nutrition plans. This system aims to fill the existing gap in the fitness and nutrition app market by offering culturally tailored workout routines and dietary recommendations, enabling users to achieve their health goals without compromising their traditional dietary habits. The app will serve as a bridge between modern fitness needs and Ethiopian cultural practices, promoting a healthier lifestyle in a way that resonates with the local community.

2.2 Current System

2.2.1 Overview

Currently, digital tools and apps focused on fitness, nutrition, and health are largely generic, catering to a global audience. While there are numerous apps offering meal planning, workout routines, and calorie tracking, these tools lack cultural specificity, particularly when it comes to Ethiopian food culture and lifestyle. Existing fitness apps do not include Ethiopian dietary ingredients, recipes, or cultural dietary preferences. Furthermore, workout programs are not adapted to the unique lifestyles, environments, and physical goals typical in Ethiopian contexts, limiting their relevance and effectiveness.

Limitations

- **Lack of Cultural Relevance**: Existing apps do not include traditional Ethiopian foods and recipes.
- **Generic Fitness Plans**: Fitness plans are not tailored to Ethiopian lifestyles or body goals.
- **Limited Awareness**: There is a gap in awareness and accessibility to culturally relevant health and fitness information.

2.3 Requirement Gathering

To develop a user-friendly and culturally relevant fitness and nutrition app, it is essential to gather detailed requirements that reflect the needs and preferences of the target audience. This includes understanding the dietary habits, popular Ethiopian dishes, traditional meal preparation methods, and specific fitness plans of users. Additionally, gathering information on existing gaps in current fitness apps will help shape the features and functionalities of the new platform, ensuring it addresses the unique challenges faced by Ethiopian users.

2.4 Requirement Gathering Methodologies

To effectively gather requirements for the fitness and nutrition app, some methods will be employed to gather requirements for the system. This includes conducting interviews and surveys with fitness enthusiasts, nutritionists, and the general public in Ethiopia to gain insights into their fitness routines and dietary practices. Additionally, a thorough analysis of existing fitness apps will be conducted to identify gaps and areas for improvement, ensuring the new platform offers a unique value proposition.

2.5 Results Found

The results found from the requirement gathering process indicate a strong demand for a fitness and nutrition app that incorporates Ethiopian food culture. Users expressed a need for meal plans featuring traditional Ethiopian dishes, personalized fitness routines that align with their lifestyles, and features that allow them to track their progress. The feedback highlighted the importance of cultural relevance in maintaining user engagement and achieving fitness plans.

2.6 Proposed System

2.6.1 Overview

The proposed system is a user-friendly web app that integrates Ethiopian food culture with personalized fitness and nutrition plans. The app will cater specifically to the dietary preferences, fitness plans, and cultural context of Ethiopian users.

Features

- **Cultural Integration**: Inclusion of traditional Ethiopian foods and recipes.
- **Personalized Plans**: Customized fitness and nutrition plans based on user preferences and plans.
- **Educational Content**: Articles and tips on health, fitness, and nutrition within the Ethiopian context.
- **User Engagement**: Interactive features to keep users motivated and engaged.

2.7 Functional Requirements

Functional requirements define what a product must do. Functional requirements are product features or functions that developers must implement to enable user to accomplish their tasks**.**

The main features of the app are as follows:

User

Fr1: The system shall allows new users to create an account by providing necessary details.

Fr2: The system shall allows users to to login into their accounts.

Fr3: The system shall allows users to calculate their BMI based on their height and weight.

Fr4: The system shall allows users to access fitness plans.

Fr5: The system shall allows users to access nutrition plan.

Fr6: The system shall allows users to submit feedback .

Administrator

Fr7: The system shall allows administrators to view, create, update, or delete accounts.

Fr8: The system shall allows administrators to manage the system.

Fr9: The system shall allows administrators to resove issues and receive feedbacks.

Fr10: The system shall allows administrators to login to the page.

Workout planner

Fr11: The system shall allows workout planner to add,delete and modify fitness plans .

Fr12: The system shall allows workout planner to login to the page.

Nutritionist

Fr13: The system shall allows nutritionist to add, delete and modify nutritions .

Fr14: The system shall allows nutritionist to login to the page.

2.8 Nonfunctional Requirements

Non-functional requirements describe how the system works. Non-functional requirements are often called qualities of a system. The following are non-functional requirements associated with the new system: -

2.8.1 Usability

- The app should be intuitive and easy to use, with a simple design that accommodates all age groups.
- Text should be clear and readable, with options for English and Amharic.

2.8.2 Reliability

The application must be available at least 99% of the time, with backup systems to minimize downtime. User data should be stored securely and accurately.

2.8.3 Performance

- **Response Time**: The app should have quick response times for user actions.
- **Scalability**: The app should be able to handle a growing number of users without performance degradation.

2.8.4 Supportability

Regular updates should be easy to implement for adding new recipes, workout routines, or health guidelines.

2.8.5 Packaging

- The app should have a user-friendly installation and configuration process.
- The app should be easy to deploy on various platforms and devices.

2.8.6 Interface

- A clean and culturally relevant design that resonates with Ethiopian users.
- The app should have a responsive design to adapt to desktops, tablets, and smartphones.

2.9 User Interface And Human Factors

The system we are going to develop follows a good interface principle and it will provide easy, attractive, simple, and interactive interfaces for the user of our system. The user of the system should have a computer to connect to the internet and access the system.

2.10 Documentation

Our system will have well-defined documents which help to easily maintain the system. We will also prepare a short and understandable file for users on how to use the system. And the development process will be provided for the user to read to know about the process and what type of model is used to develop the system.

2.11 Hardware Consideration

Our system will be developed by considering hardware requirements. The system will support computers and secondary memory to store all databases to provide the services for the users.

2.12 Performance Characteristics

The proposed system will have easy and efficient code manipulation and a clear database.

Response Time: Upon the request for user query the system under normal conditions will display results as quickly as possible.

Processing Time: Since the system will be developed with efficient programming language and database upon request for users' activity. The system under normal conditions will process the request as quickly as possible.

Concurrent Processing: since the proposed system is web-based it concurrently deals with the requests rather than waiting for a previous one to be completed, the system will support multiple users at a time.

2.13 Error Handling And Extreme Conditions

Our System will Response to user error and the undesired situation should be taken care of to ensure that the system operate without halting. The system should be able to avoid or tackle catastrophic behavior. Throw exceptions instead of returning an error code. Validate the system by ensuring that our system will meet user's needs and functions according to its intended use.

2.14 Quality Issues

The users will be supported by a feedback mechanism in which they can give comments on the system for quality assessment. The functions of the system are appropriate, implemented correctly, and handle data securely. It will be user-friendly, so the usability of the system will be clear and understandable by users and do not require much effort. Failure and fault can be identified and will be fixed quickly.

2.15 System Modifications

our system will have a modular architecture to facilitate easy updates and modifications.

2.16 Physical Environment

The app is designed to function effectively in various environments, especially in urban and rural settings within Ethiopia.

2.17 Security Issues

Some of the factors that are identified to protect the software from accidental or malicious access, use, modification, destruction, or disclosure are described below.

- Assign certain functions to different modules
- Restrict communications between some areas of the program
- Check data integrity for critical variables

2.18 Resource Issues

Our System can save our resource in order to do the requirements they satisfy the time efficiency , storage space efficiency and communication band width and speed and user can do any operation in a short period of time.

2.19 System Model

2.19.1 Scenario

Scenarios are an instance of a use case explaining a concrete major set of action. Scenario or use case realizations are just a graphical sequence of events or an instance of a use case. These realizations are depicted as either sequence or collaboration diagram.

Scenario name: Daily use

Participant actor: User

To use app daily. Ayele (user) logs into the app. The system displays a list of fitness plans and Ethiopian nutrition guides. ayele selects a fitness plan and a nutrition guide from the available options. The app provides detailed instructions for the chosen plan, including workout routines and meal recommendations. ayele begins following the plan and works daily.

Scenario name: Admin manages accounts

Participant actor: Admin

Tesfaye(admin) logs into the system using his credentials (username and password). After logging in, tesfaye accesses the " user management " section of the admin dashboard. Tesfaye searches for users using filters like user ID , name , or email. Tesfaye selects account to view detailed information such as name , email , workout plans and nutrition plans . if the user has reported incorrect details , tesfaye edits their profile , such as updating email or fixing a typo in their name . A confirmation message appears asking if tesfaye wants to proceed with changes . After confirming, the account updates successfully, and a success message appears.

2.20 Use Case Model

Actors: An actor is a person, organization, or external system that plays a role in one or more interactions with your system. In our system the following actors are involved: -

User: Regular user of the app.

Admin: Administrator with special privileges.

Workout planner / expert : Develop fitness programs targeting strength, weight loss, or muscle building.

Nutritionist : Manage meal plans .

2.21 Use Case Diagram

Use case diagram shows use cases, actors, and their interrelationships. A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. A use case model describes the proposed functionality of a new system.

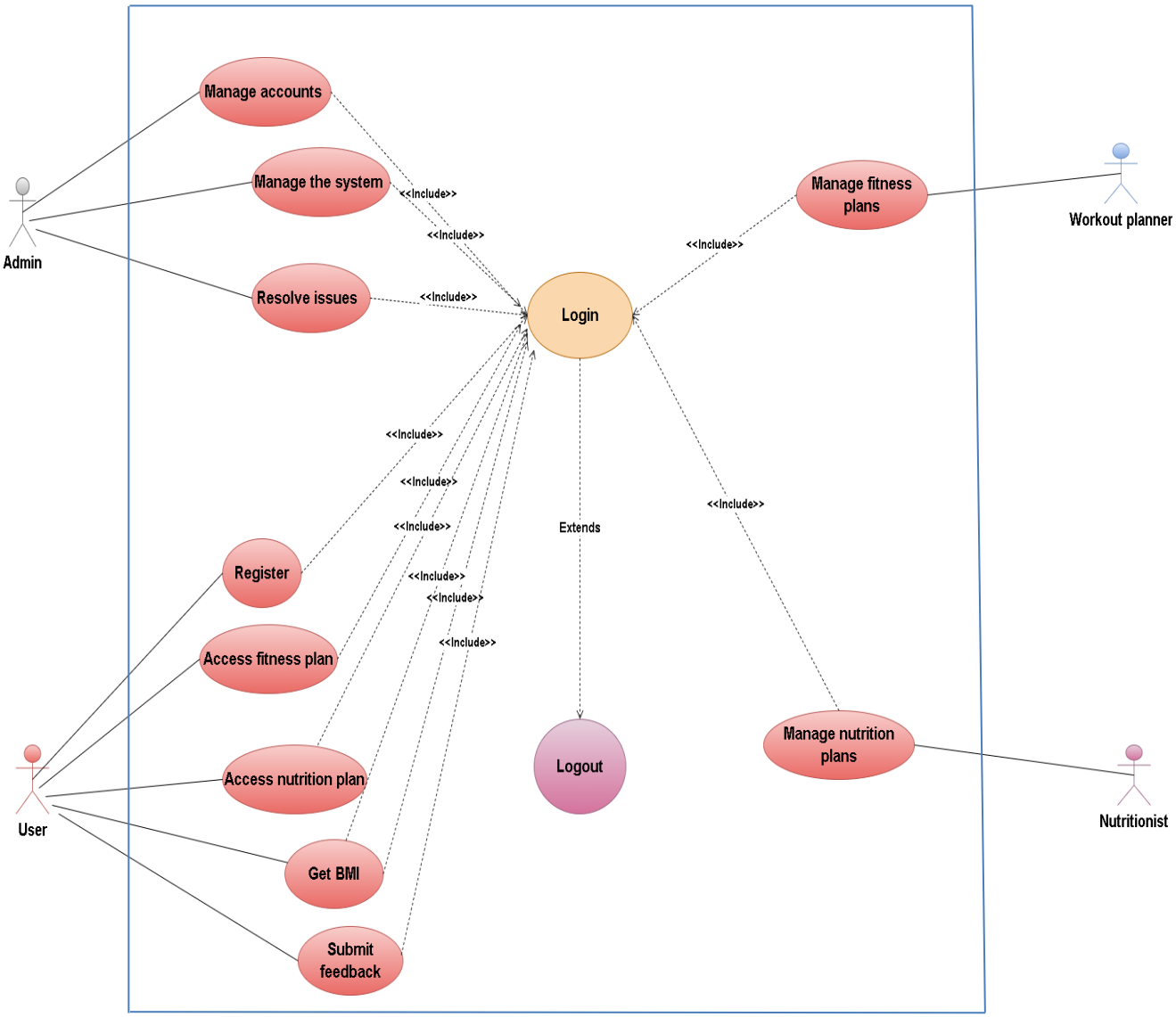

Figure 1: use case diagram

2.22 Description Of Use-Case Model

use case is an interaction between users and a system. It captures the plan of the users and the responsibility of the system to its users. It is the functionality of the system or the service provided by the system.

Table 2: Register use case description

Use Case Name	Register
Actor	User
Description	Allows new users to create an account by providing necessary details.
Precondition	1. The user must have access to the registration page. 2. The user must have a valid email address or phone number.
Basic course of action	1. User navigates to the registration page. 2. User enters personal details (name, email, password, etc.). 3. System validates the input and creates a new account.
Alternative course of action	2.1 If required fields are missing or invalid , 2.2 the system displays an error and prompts the user to fix the issues.
Post condition	The user's account is successfully created, and they can log in.

Table 3: Login use case description

Use Case Name	Login
Actor	User , Administrator, workout planner,nutritionist .
Description	Allows actors to sign in and access the system .
Precondition	The actor should have a valid account.
Basic course of action	1. Actor navigates to the login page. 2. Actor enters personal details (name, email, password, etc.). 3. System validates the input and logged them.
Alternative course of action	2.2 If required fields are missing or invalid , 2.2 the system displays an error and prompts the actor to fix the issues.
Post condition	The Actor logged in successfully.

Table 4: Get BMI use case description

Use Case Name	Get BMI
Actor	User
Description	Allows user to calculate their body mass index (BMI) by entering their height and weight .
Precondition	1. The user must be logged into the system. 2. Height and weight data must be entered in valid units (e,g., centimeters/meters for height and kilograms for weight)
Basic course of action	1. User navigates to the BMI calculation section . 2. User enters their height and weight . 3. System validates the input values . 4. System calculates the BMI using the formula : BMI = $\frac{\text{Weight (kg)}}{\text{Height (m)}^2}$ 5. System displays the calculated BMI .
Alternative course of action	2.1 If the users enters invalid data , 2.2 The system displays an error message and prompts the user to correct the input .
Post condition	The user receives their BMI value .

Table 5: Access Fitness plans use case description

Use Case Name	Access Fitness plans
Actor	User
Description	Allows users to access fitness plans such as weight loss or strength training.
Precondition	1. The user must be logged in. 2. The system must have access to fitness plans.
Basic course of action	1. User navigates to the fitness plan section. 2. User selects or customizes a fitness plan. 3. System saves the selected plan and provides plans.
Alternative course of action	2.1 If a fitness plan isn't available for the chosen plan , 2.2 the system suggests alternatives.
Post condition	The fitness plan is set, and the user receives their fitness plan.

Table 6: Access nutrition plans use case description

Use Case Name	Access nutrition plans
Actor	user
Description	This use case allows the user access nutrition plans.
Precondition	The user must be logged in.
Basic course of action	1. The user navigates to the nutrition plans page. 2. The system fetches nutrition plans.
Alternative course of action	2.1 If the food item is not found , 2.2 the system suggests alternatives .
Post condition	The user gets their nutrition plan.

Table 7: Submit feedback use case description

Use Case Name	Submit feedback
Actor	user
Description	This use case allows the user to submit their feedback .
Precondition	The user must be logged in.
Basic course of action	1. The user navigates to the about us page. 2. The system submits the feedback.
Alternative course of action	2.1 If the submit feedback page isn't available, 2.2 the system suggests alternatives .
Post condition	The user submits their feedback successfully.

Table 8: Manage accounts use case description

Use Case Name	Manage accounts
Actor	Administrator
Description	Allows administrators to view, update, or delete accounts.
Precondition	1. Administrator must be logged in. 2. User data must be available in the database.
Basic course of action	1. Administrator accesses the user management panel. 2. Admin selects account to view or modify. 3. System processes the request and updates the database as necessary.
Alternative course of action	2.1 If the selected user data is corrupted or unavailable , 2.2 the system displays an error message.
Post condition	The account is updated, deleted, or displayed as required.

Table 9: Manage the system use case description

Use Case Name	Manage the system
Actor	Administrator
Description	Allows administrators to manage the overall system including roles, permissions and system settings.
Precondition	1. Administrator must be logged in. 2. Administrator must have the necessary permissions to perform .
Basic course of action	1. Administrator navigates to the the admins login page. 2. Administrator logs into the system using their credentials . 3. Administrator navigates to system management section. 4. Administrator assign roles and sets permissions . 5. The system processes the request and updated as necessary.
Alternative course of action	2.1 If required fields are missing or invalid , 2.2 the system displays an error and prompts the actor to fix the issues
Post condition	System settings are updated and applied.

Table 10: Resolve issues use case description

Use Case Name	Resolve issues
Actor	Administrator
Description	Allows administrators to resolve issues reported by users and collecting feedbacks.
Precondition	1. Administrator must be logged in. 2. Administrator must have the necessary permissions to perform . 3. users must have submitted issues or feedbacks.
Basic course of action	1. Administrator navigates to the the admins login page. 2. Administrator logs into the system using their credentials. 3. Administrator navigates to issues or feedback section . 4. Administrator views a list of reported issues and feedbacks. 5. Administrator resolved the issue and analyzes feedbacks.
Alternative course of action	2.1 If no issues or feedbacks are reported , 2.2 the system displays no issues or feedback found .
Post condition	Reported issues are resolved and user feedbacks are analyzed .

Table 11: Manage fitness plans use case description

Use Case Name	Manage fitness plans
Actor	Workout planner
Description	Allows workout planner to add, modify, or delete fitness plans.
Precondition	1. Workout planner must be logged in. 2. Workout planner must have the necessary permissions to perform .
Basic course of action	1. Workout planner navigates to the the workout planner login page. 2. Workout planner logs into the system using their credentials. 3. Workout planner navigates to manage fitness plans section . 4. Workout planner add, modify or delete workout plans.
Alternative course of action	2.1 If the database update fails , 2.2 the system logs the error and notifies the workout planner.
Post condition	finess plans updated successfully .

Table 12: Manage nutrition plans use case description

Use Case Name	Manage nutrition plans
Actor	Nutritionist
Description	Allows Nutritionist to add, modify, or delete nutrition plans.
Precondition	1. Nutritionist must be logged in. 2. Nutritionist must have the necessary permissions to perform .
Basic course of action	1. Nutritionist navigates to the the Nutritionist login page. 2. Nutritionist logs into the system using their credentials. 3. Nutritionist navigates to manage nutrition plans section . 4. Nutritionist add, modify or delete nutrition plans.
Alternative course of action	2.1 If the database update fails , 2.2 the system logs the error and notifies the Nutritionist .
Post condition	nutrition plans updated successfully .

2.23 Dynamic Model

Dynamic model refers to a representation of the behavior and interactions of a system over time. It is a tool used to understand and describe how the system's components and processes evolve and change during its operation. In our system we will discuss as sequence diagram, activity diagram and state diagram.

2.24 Sequence Diagram

Sequence diagrams are used to model the logic of usage scenario, enable you to visually model the logic of the system. It may also be one entire pass through a use case, such as a logic described by the basic course of action or portion of the basic course of action plus one or more alternate scenarios. Objects, classes and actors are can be depicted in sequence diagrams.

A sequence diagram is used to:

- Represent the details of a UML use case.
- Model the logic of a sophisticated procedure, function, or operation.
- See how objects and components interact with each other to complete a process.
- Plan and understand the detailed functionality of an existing or future scenario.

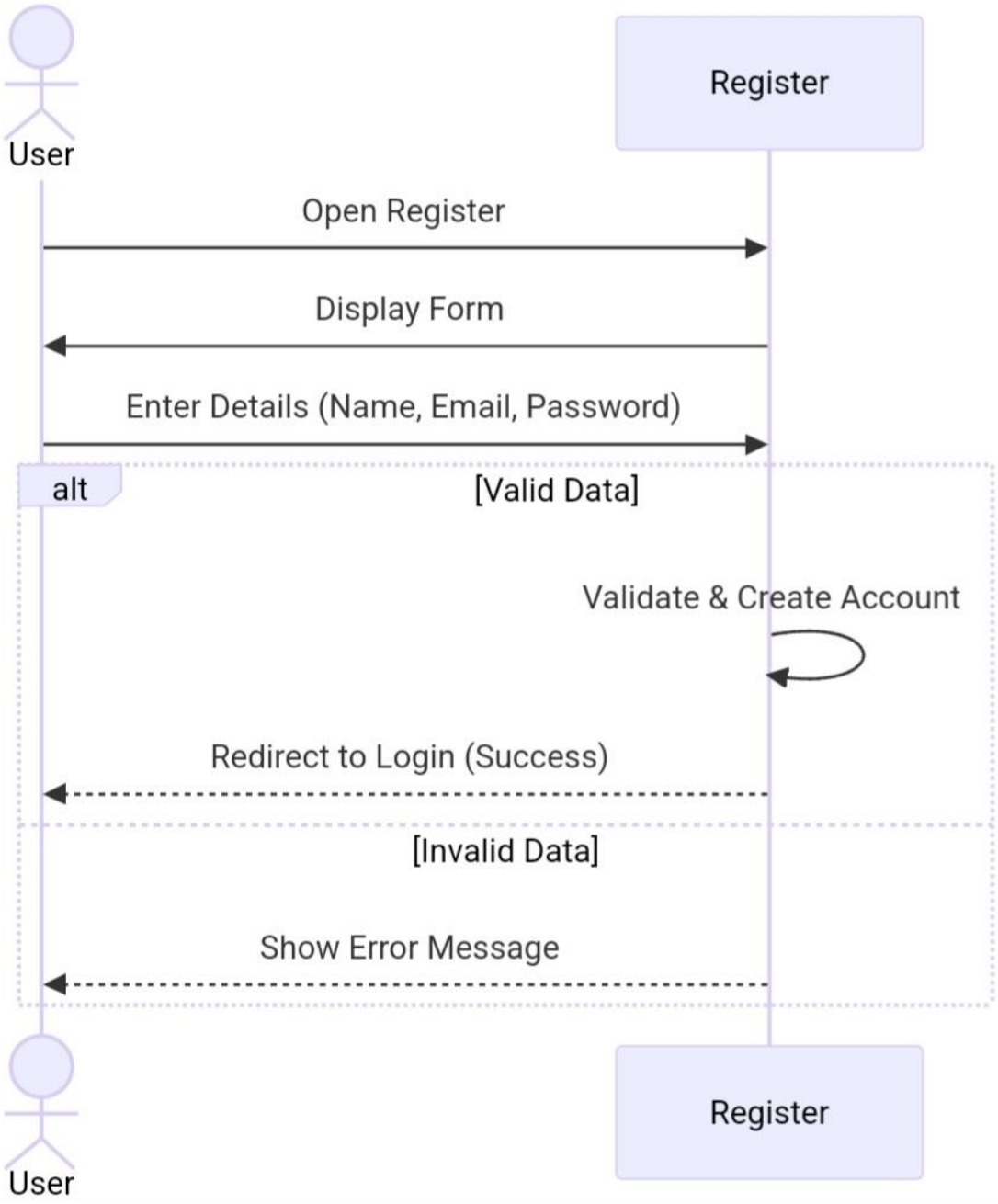

Figure 2 : Sequence diagram for register

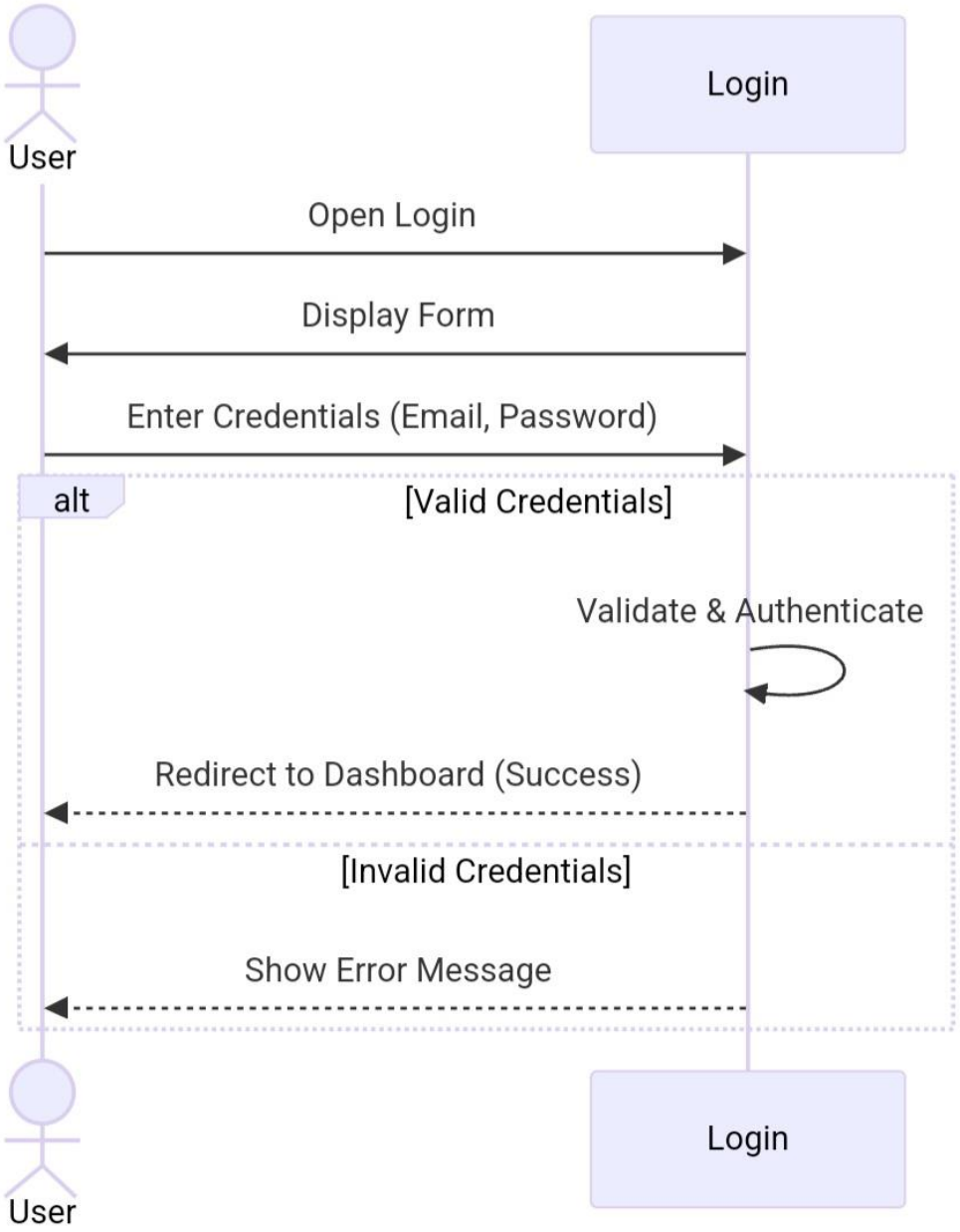

Figure 3: Sequence diagram for login

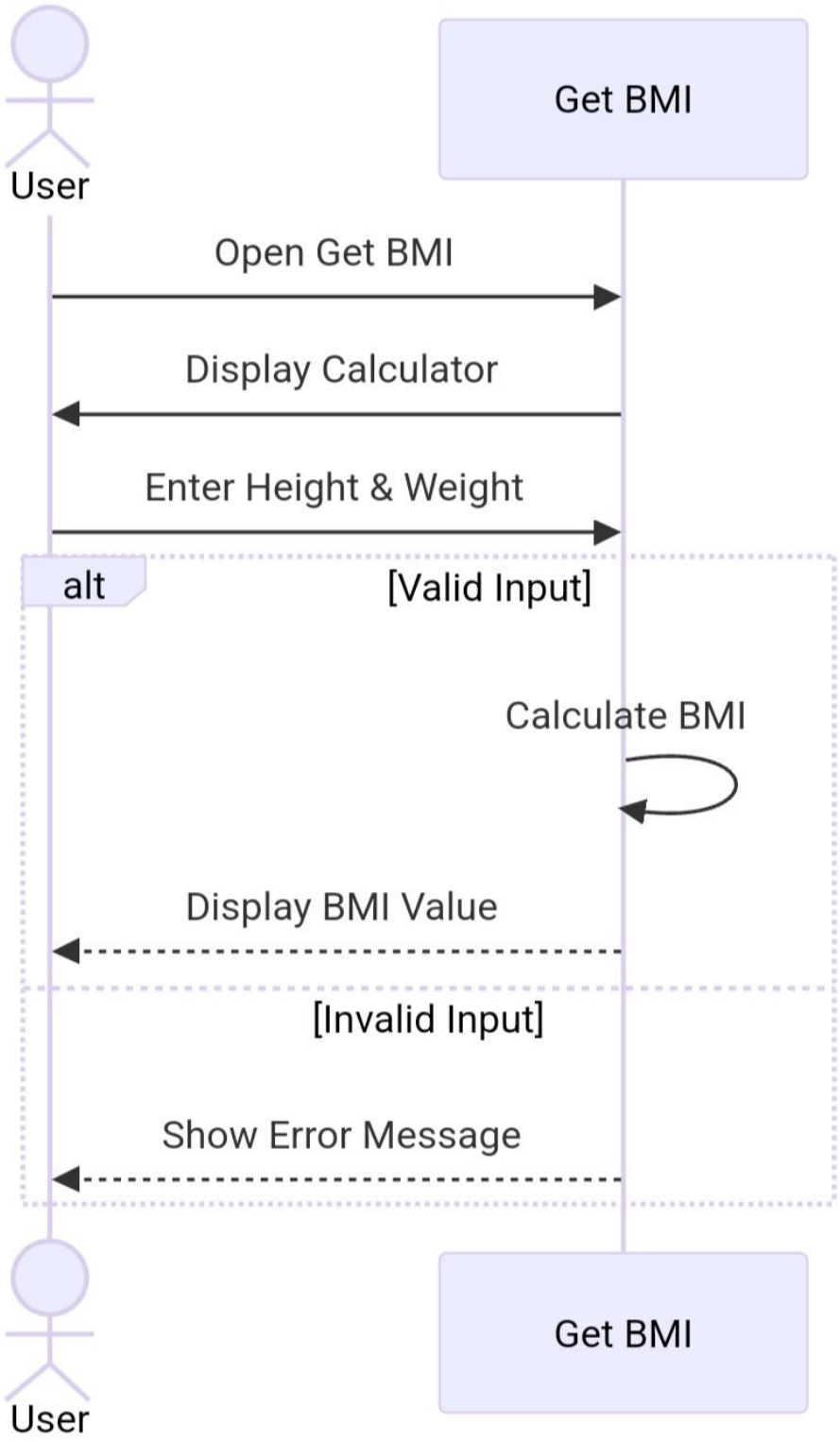

Figure 4: Sequence diagram for Get BMI

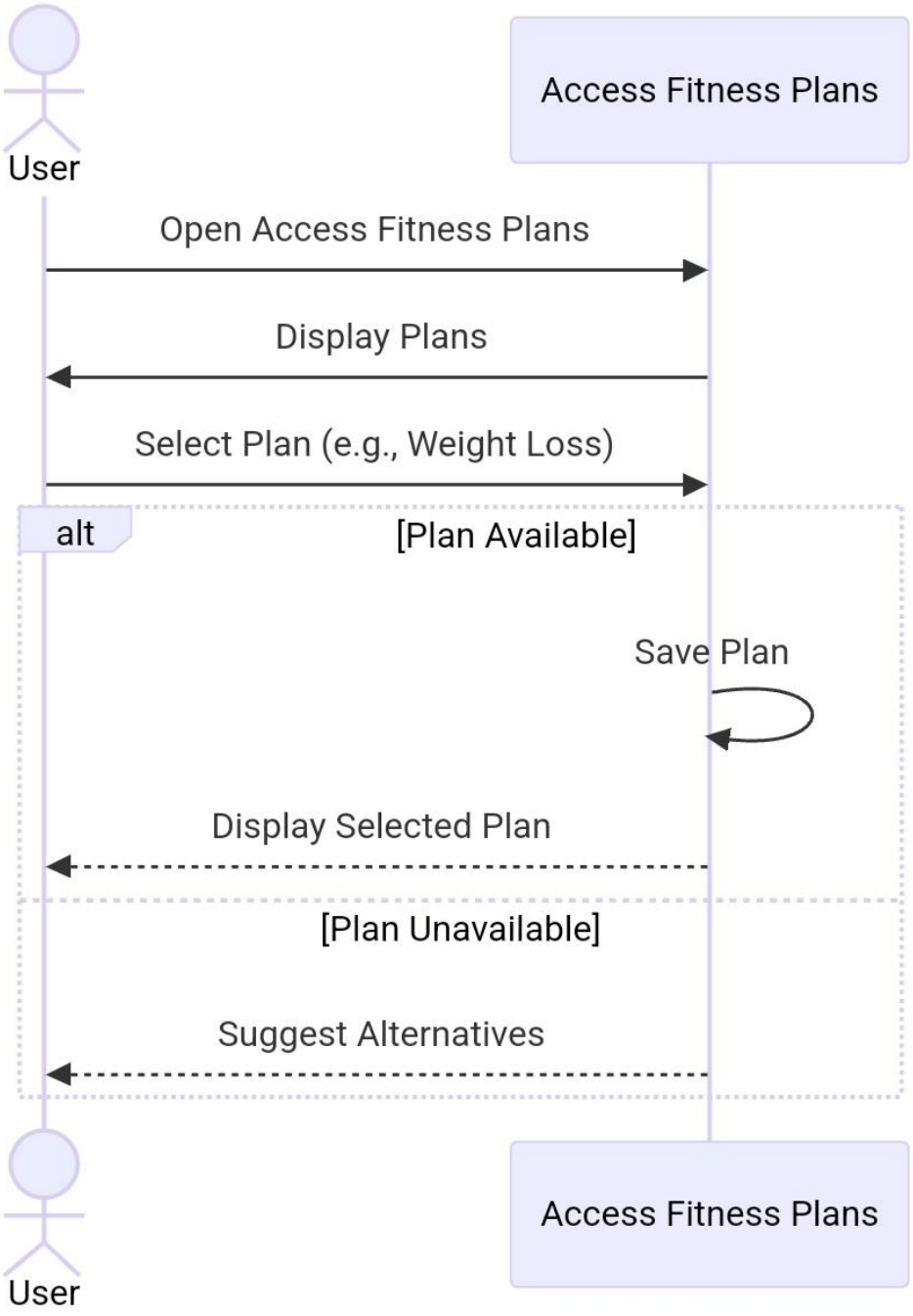

Figure 5: Sequence diagram for Access fitness plans

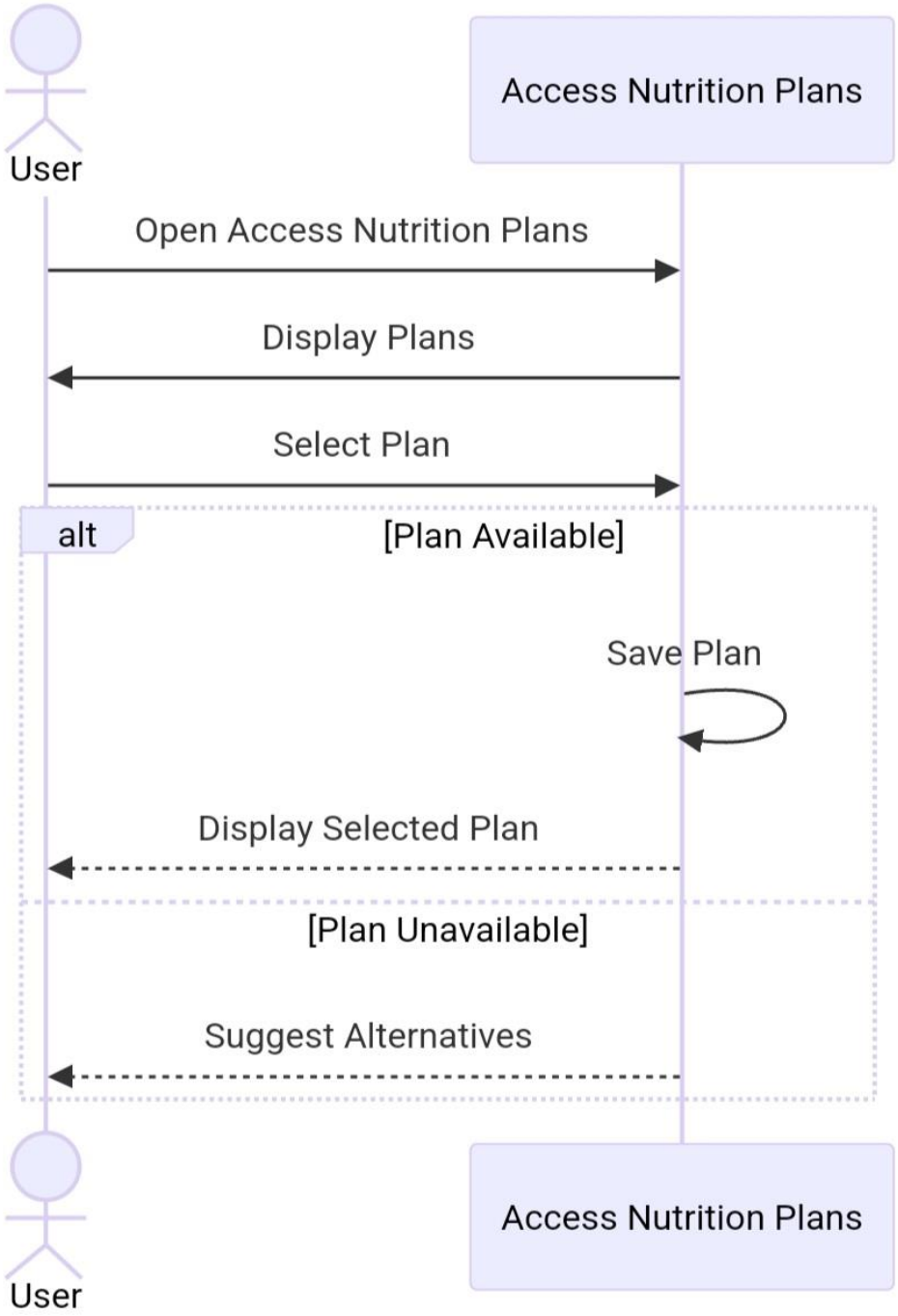

Figure 6: Sequence diagram for Access nutrition plans

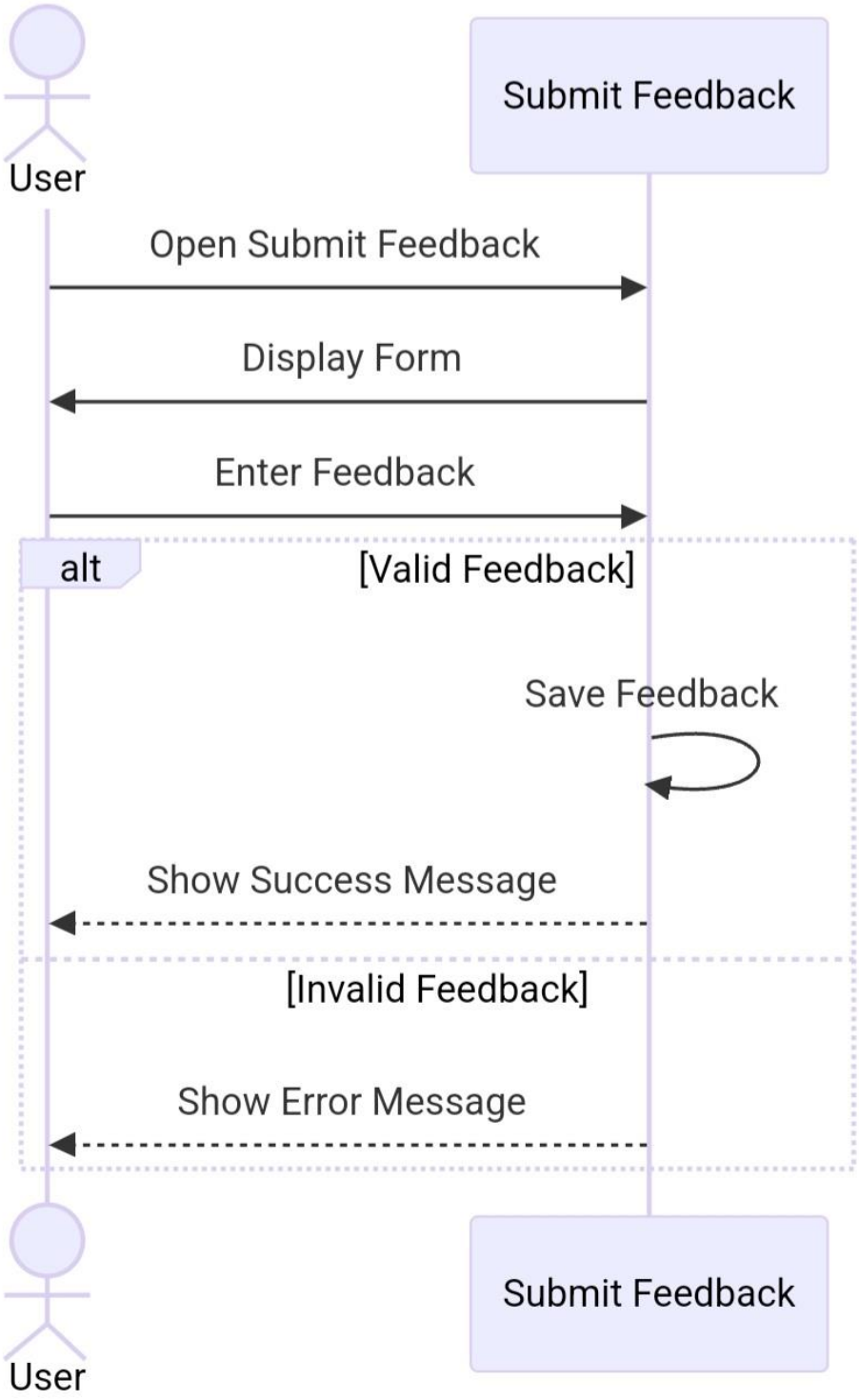

Figure 7: Sequence diagram for Submit feedback

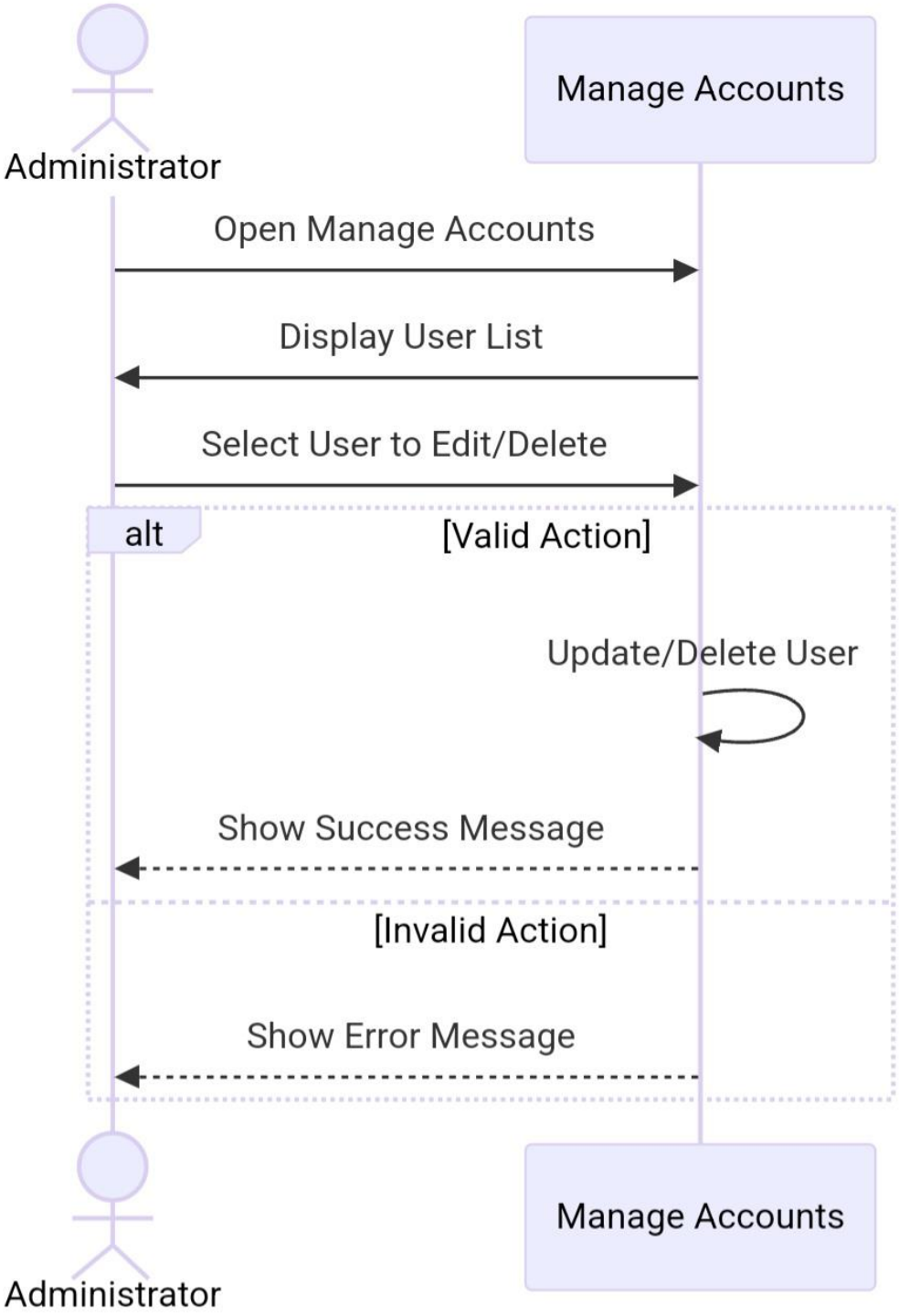

Figure 8: Sequence diagram for manage accounts

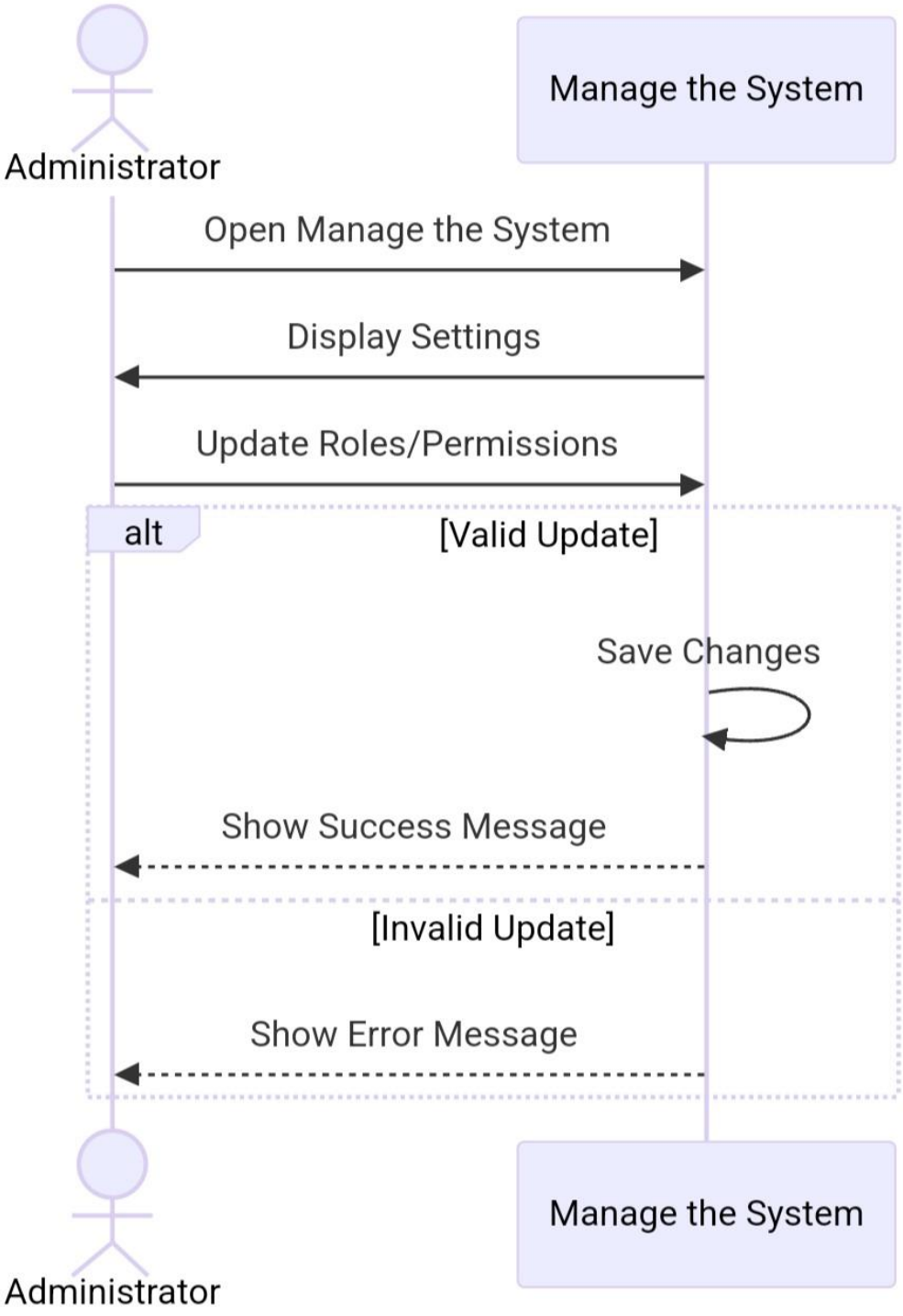

Figure 9: Sequence diagram for manage the system

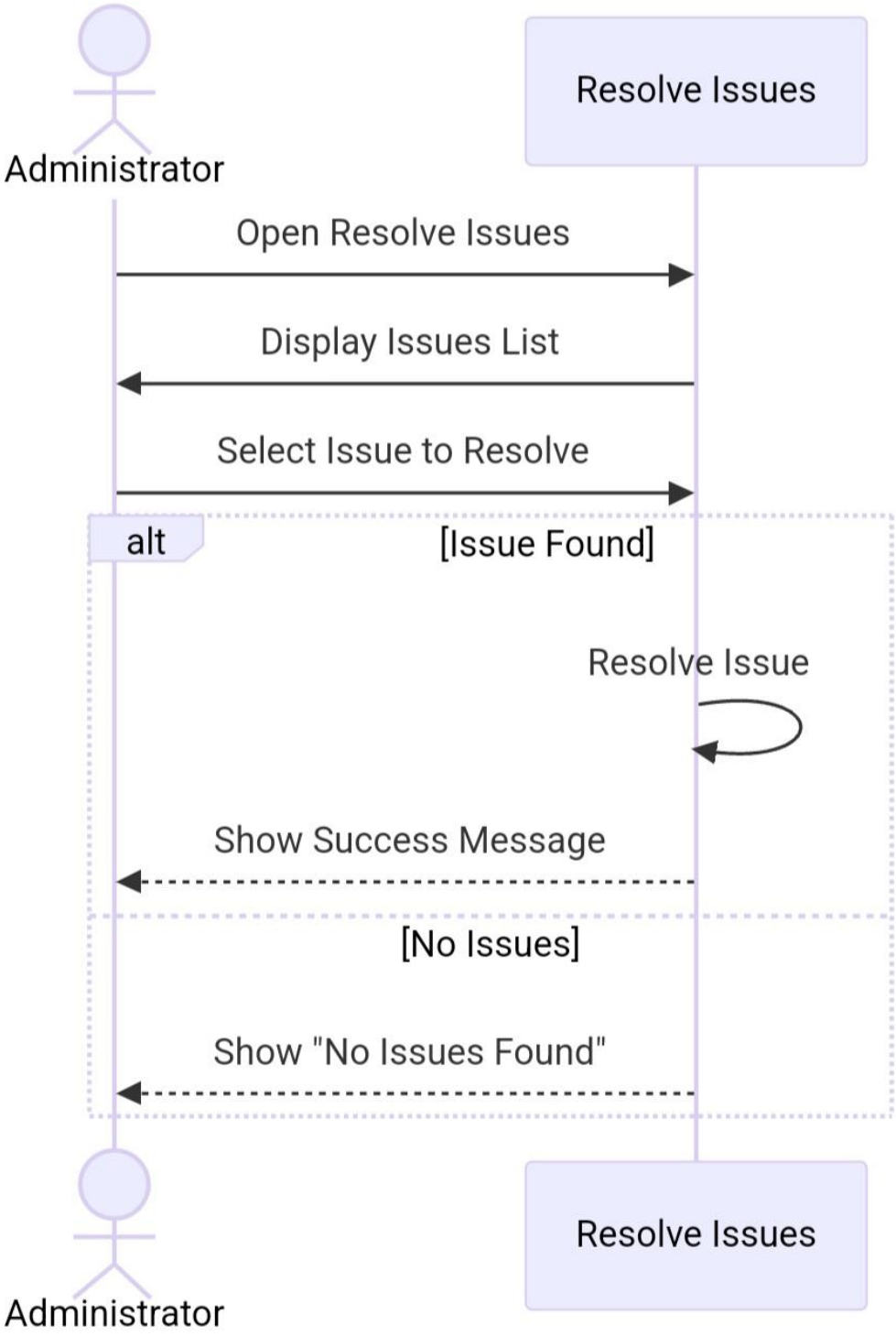

Figure 10: Sequence diagram for resolve issues

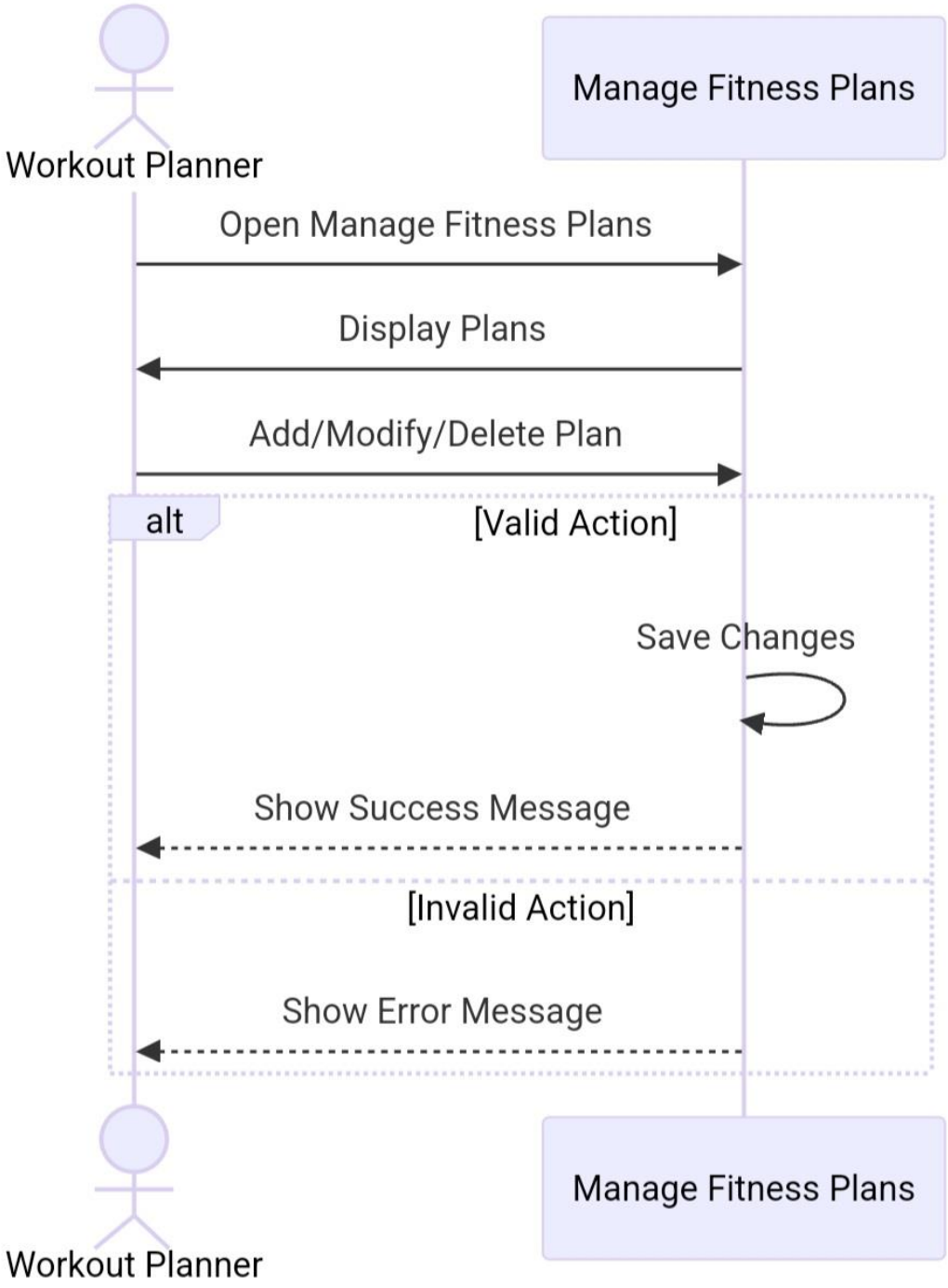

Figure 11: Sequence diagram for manage fitness plans

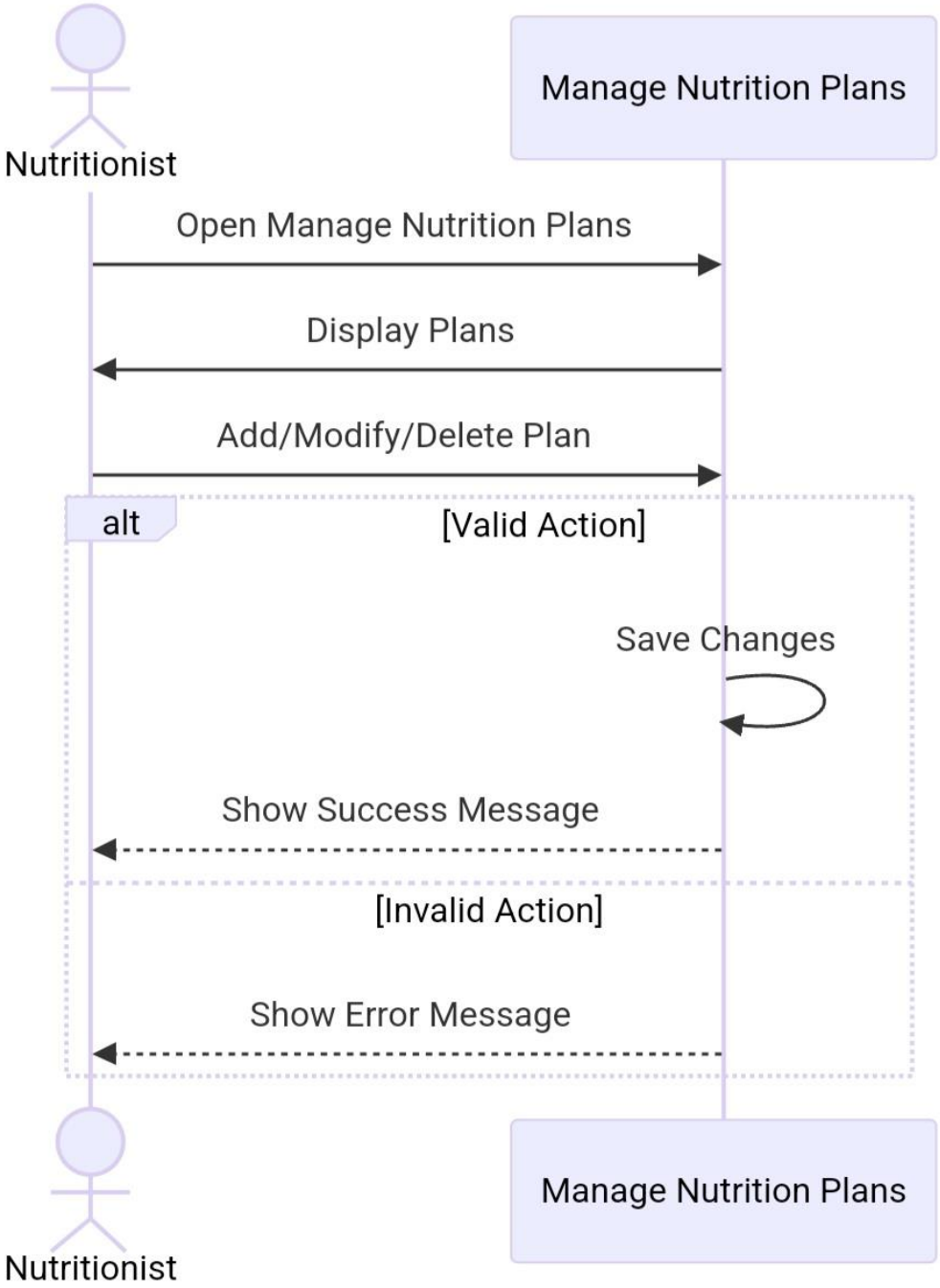

Figure 12: Sequence diagram for manage nutrition plans

2.25 Activity Diagram

Activity diagram is the object oriented equivalent of flow charts and data follow diagrams from the structured development. Flowchart is a graphical way of expressing the steps needed to solve a problem. A flow chart is a schematic (diagrammatic description) representation of a process. Used to document the logic of a single operation/method, use case or the flow of logic of a business process in the form of flow chart. Describe the sequence from one activity to another. Activity Diagrams are also useful for analyzing a use case by describing what actions need to take place and when they should occur, describing a complicated sequential algorithm and modeling applications with parallel processes. Some uses of activity diagram is:-

- Draw the activity flow of the system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and simultaneous flow of the system's activity.

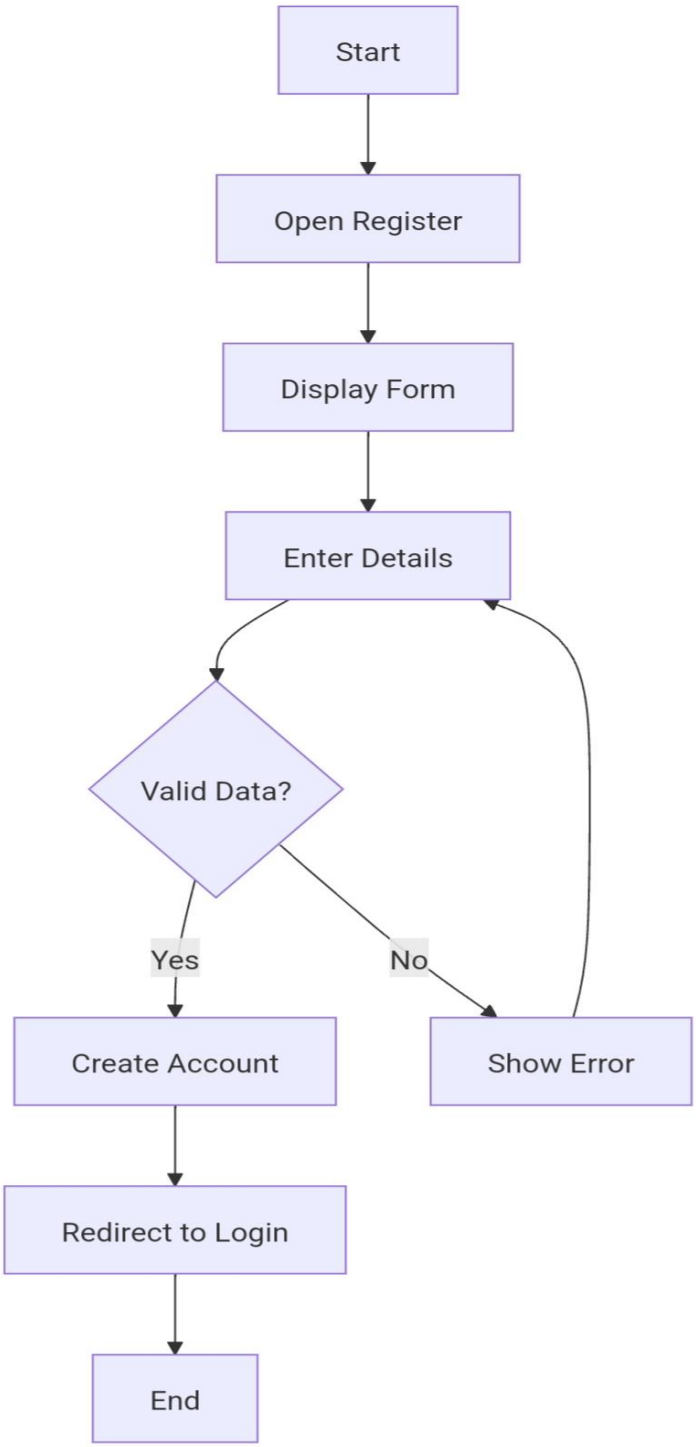

Figure 13 : Activity diagram for register

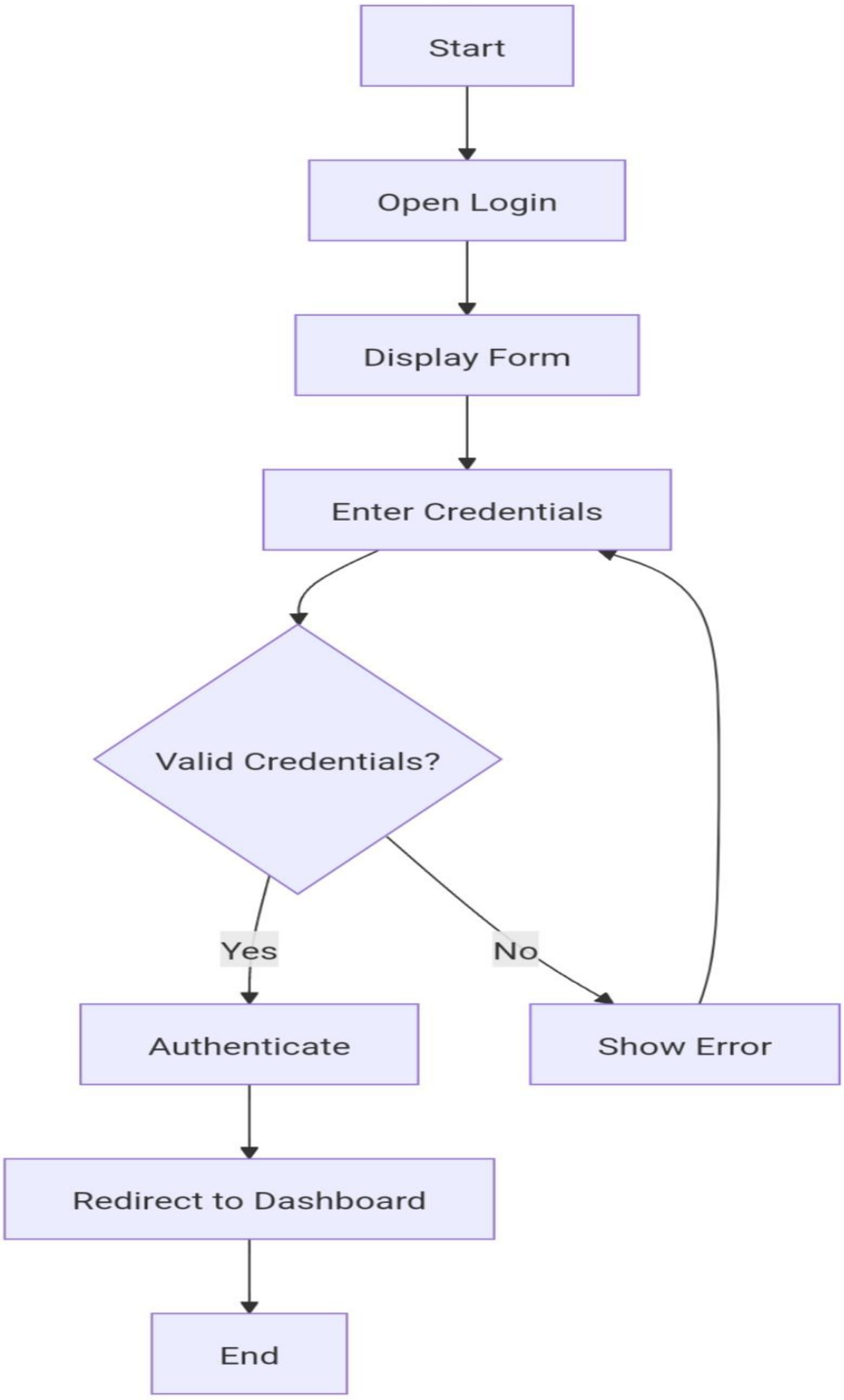

Figure 14 : Activity diagram for login

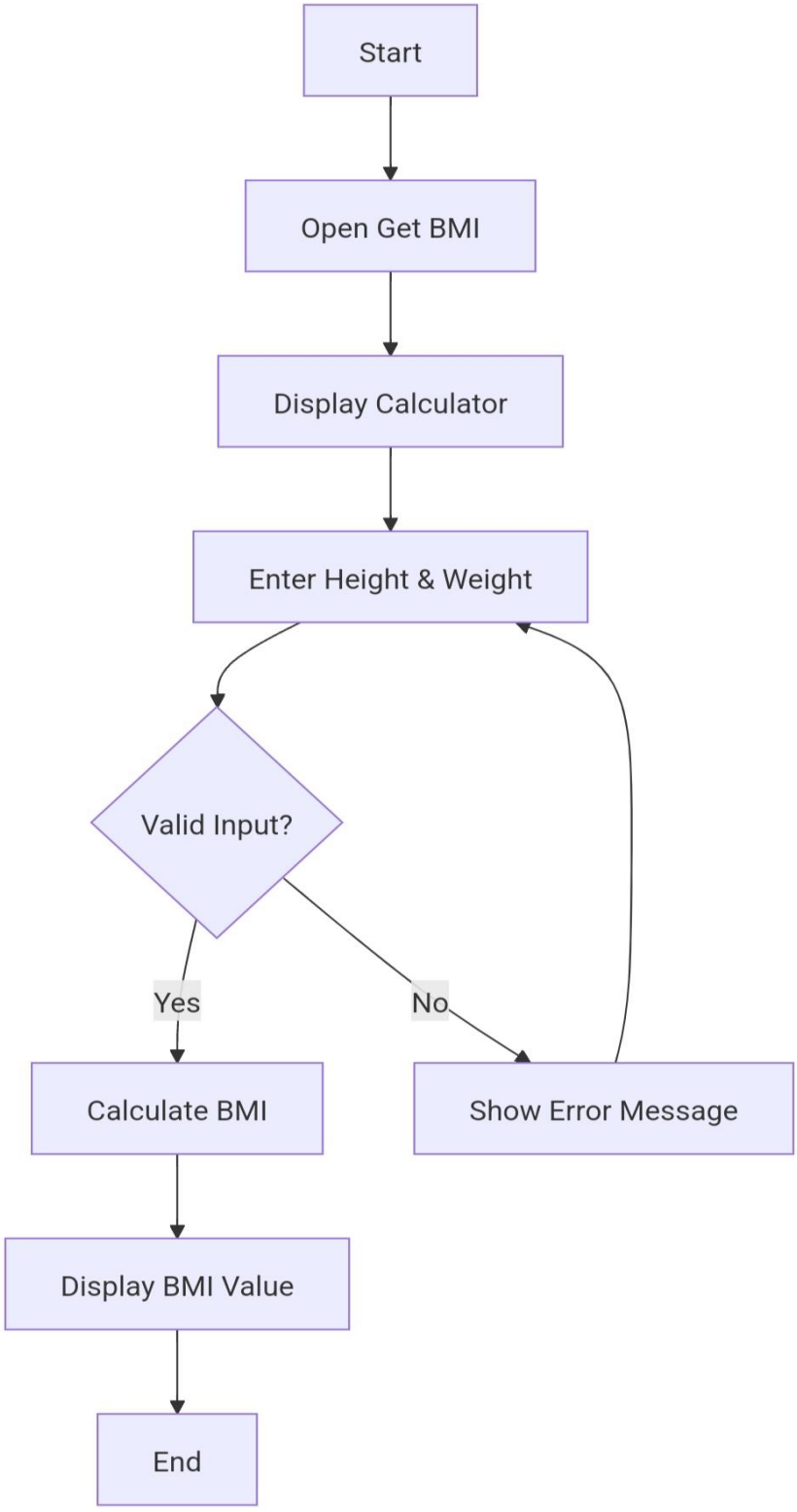

Figure 15 : Activity diagram for get BMI

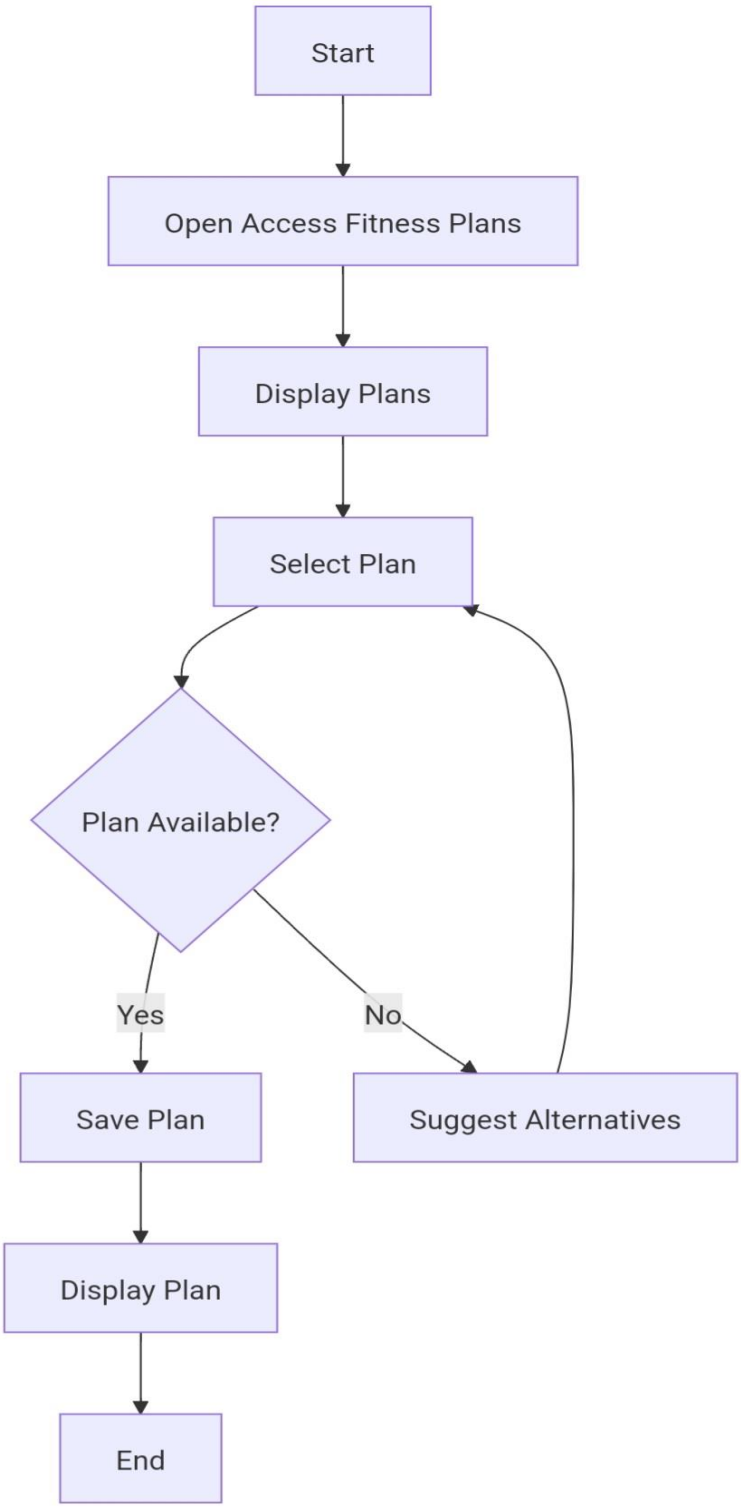

Figure 16 : Activity diagram for access fitness plans

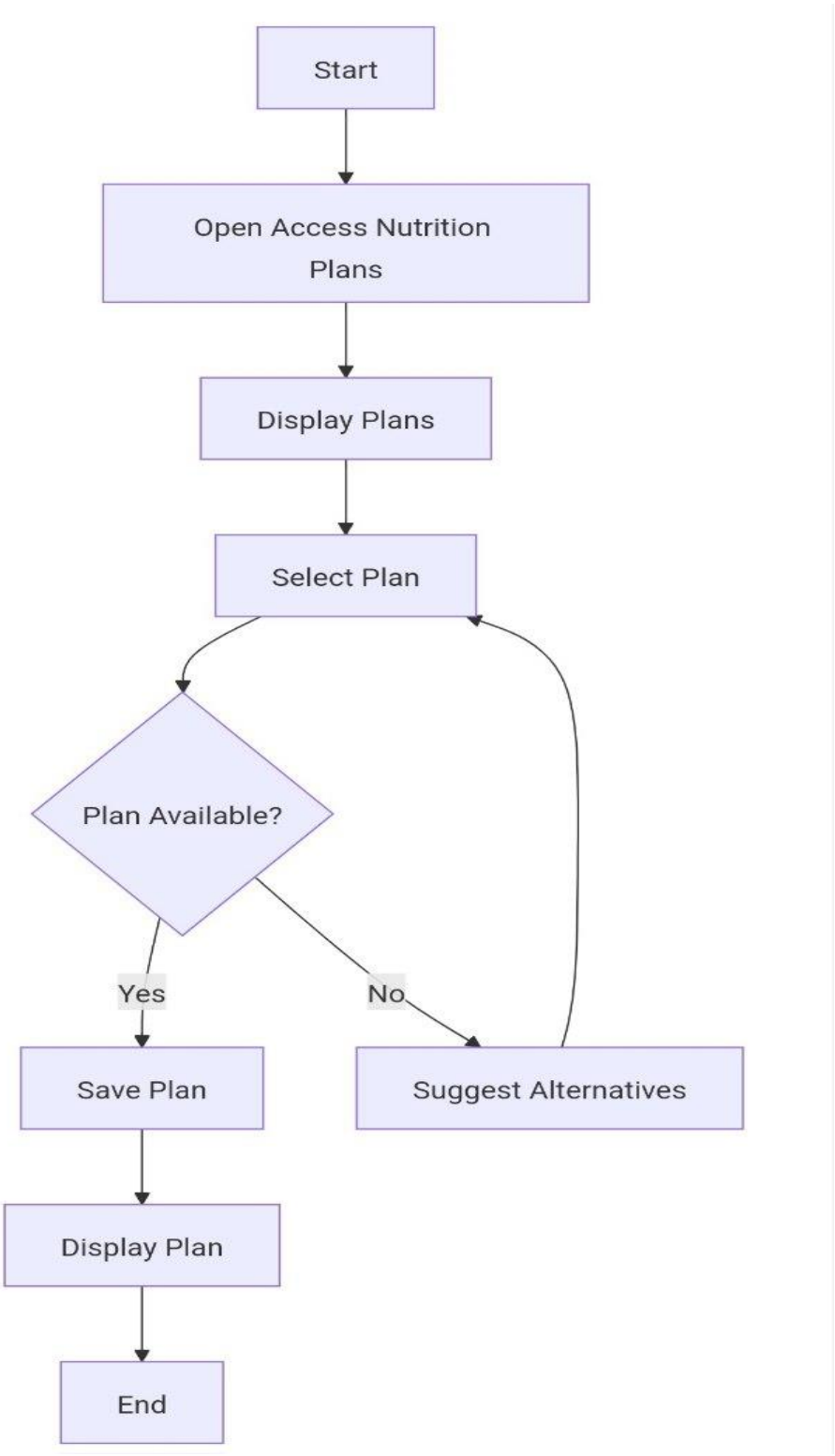

Figure 17 : Activity diagram for access nutrition plans

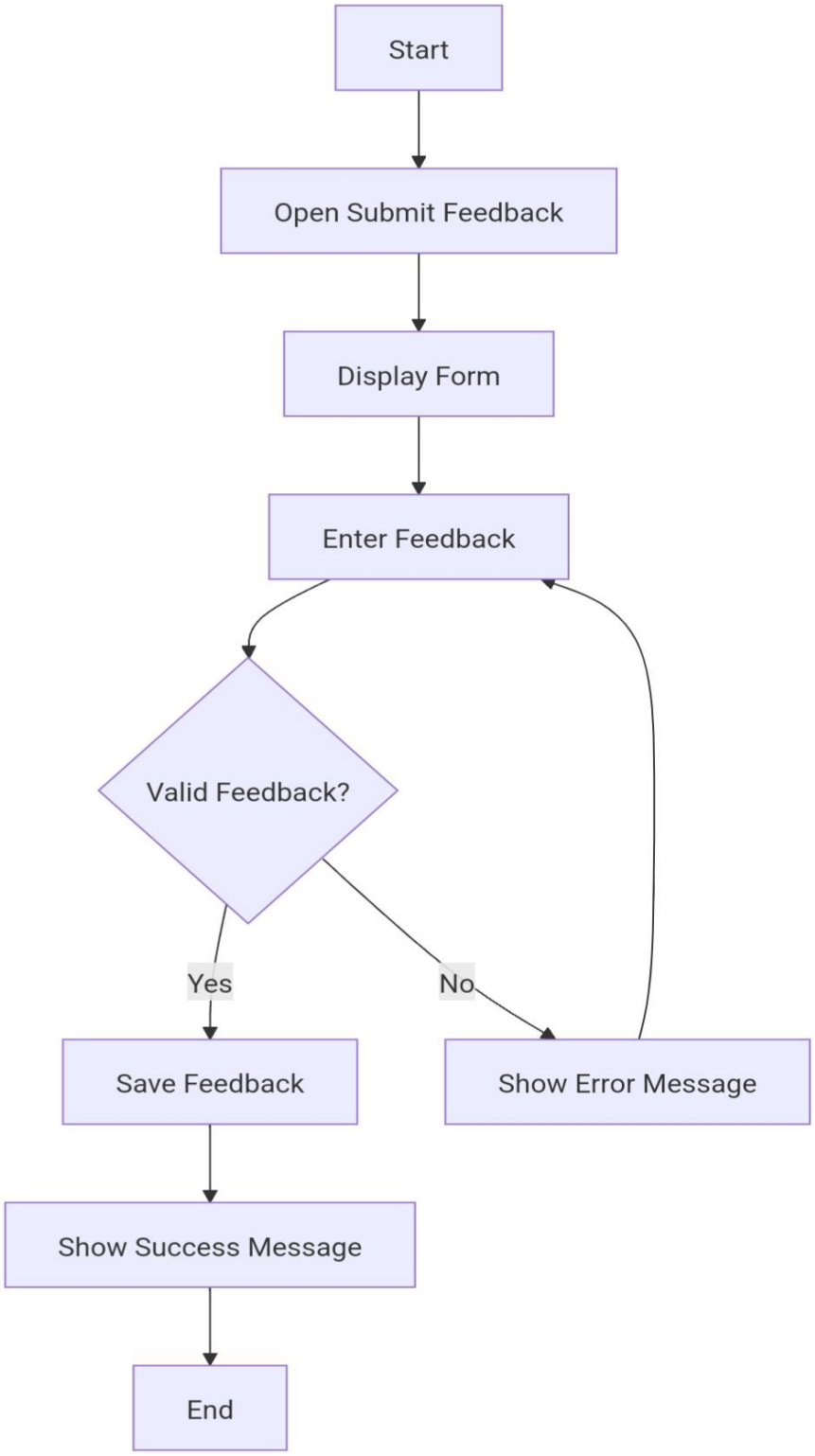

Figure 18 : Activity diagram for submit feedback

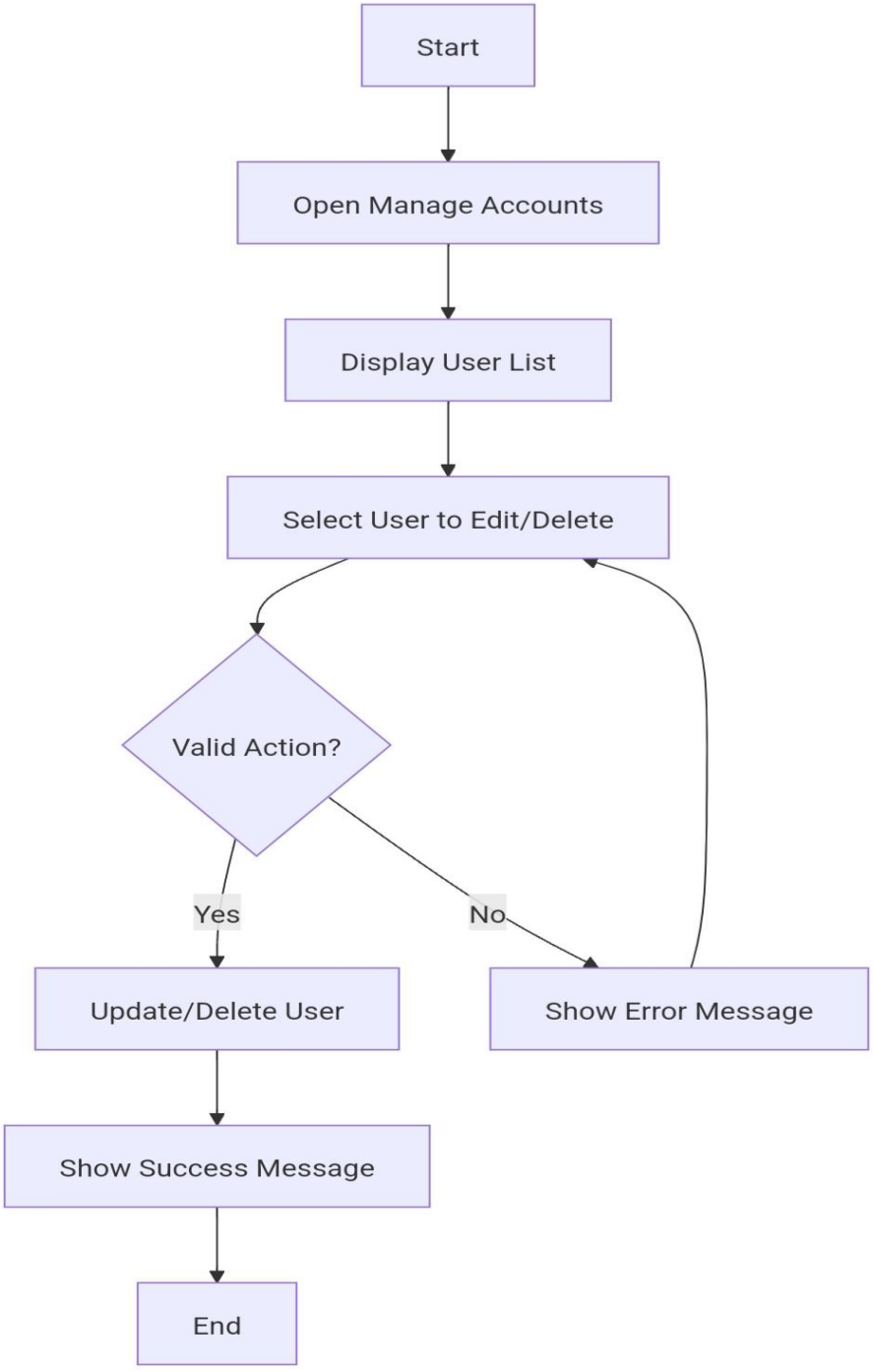

Figure 19 : Activity diagram for manage accounts

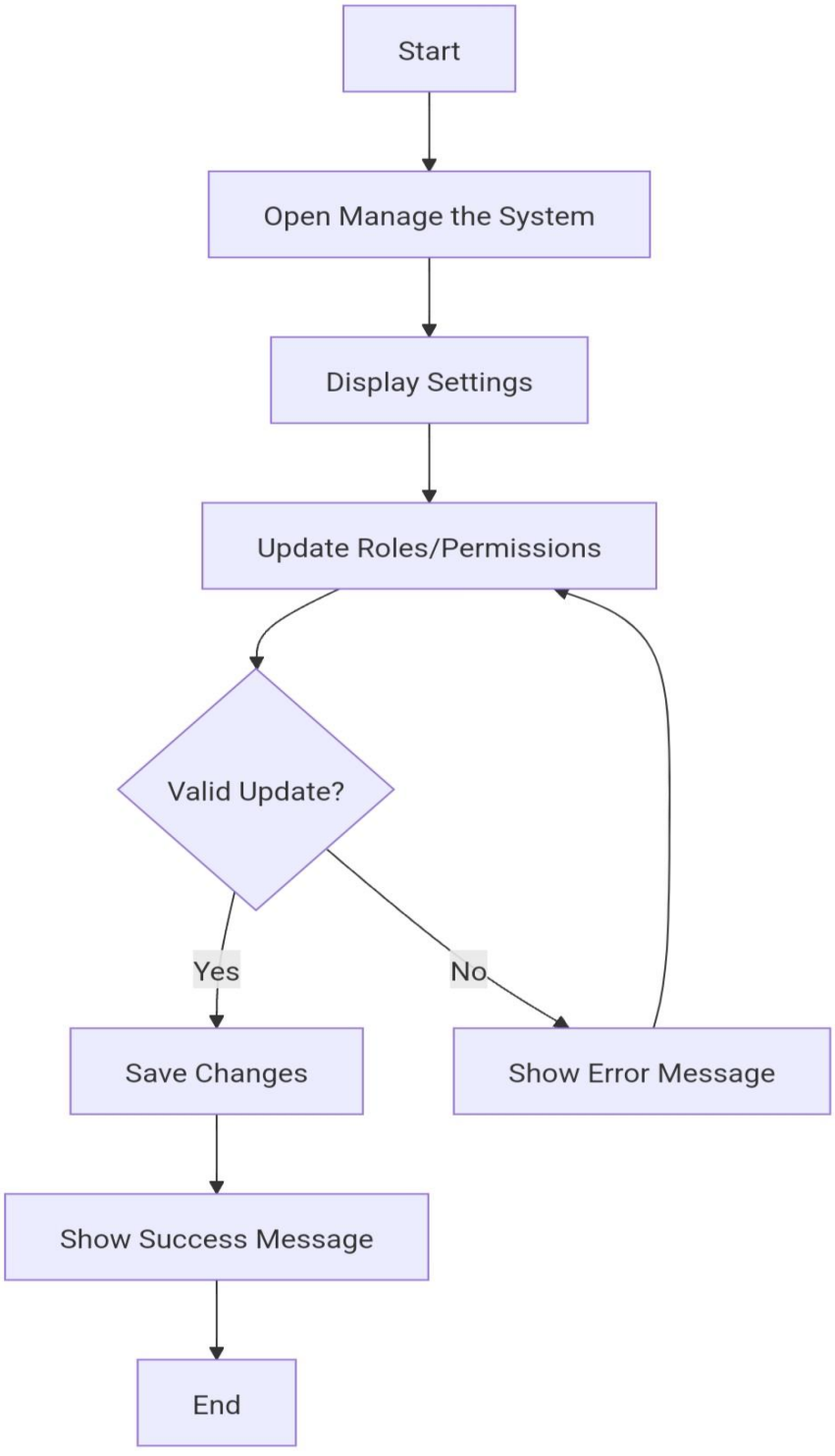

Figure 20 : Activity diagram for manage the system

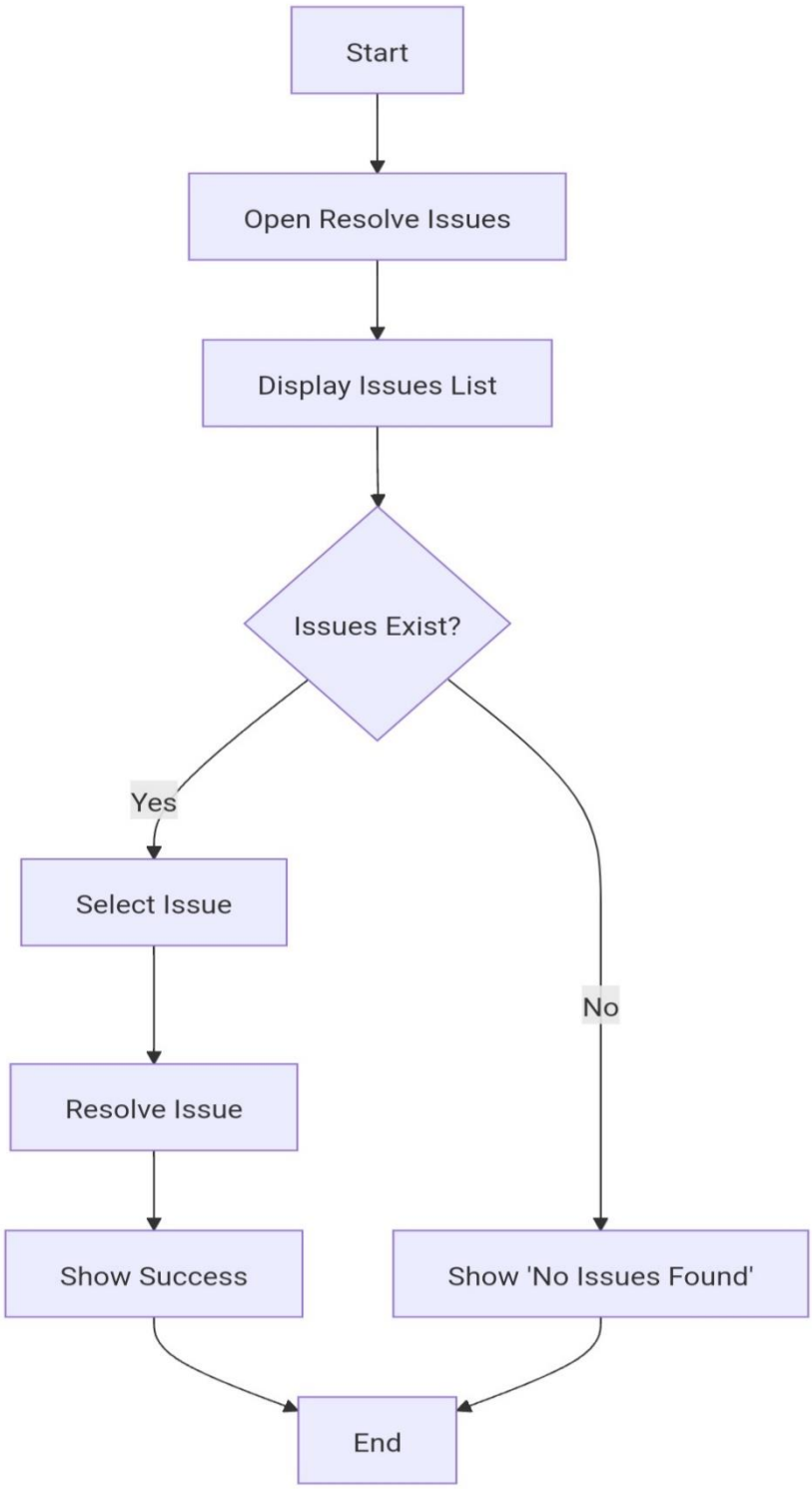

Figure 21 : Activity diagram for resolve issues

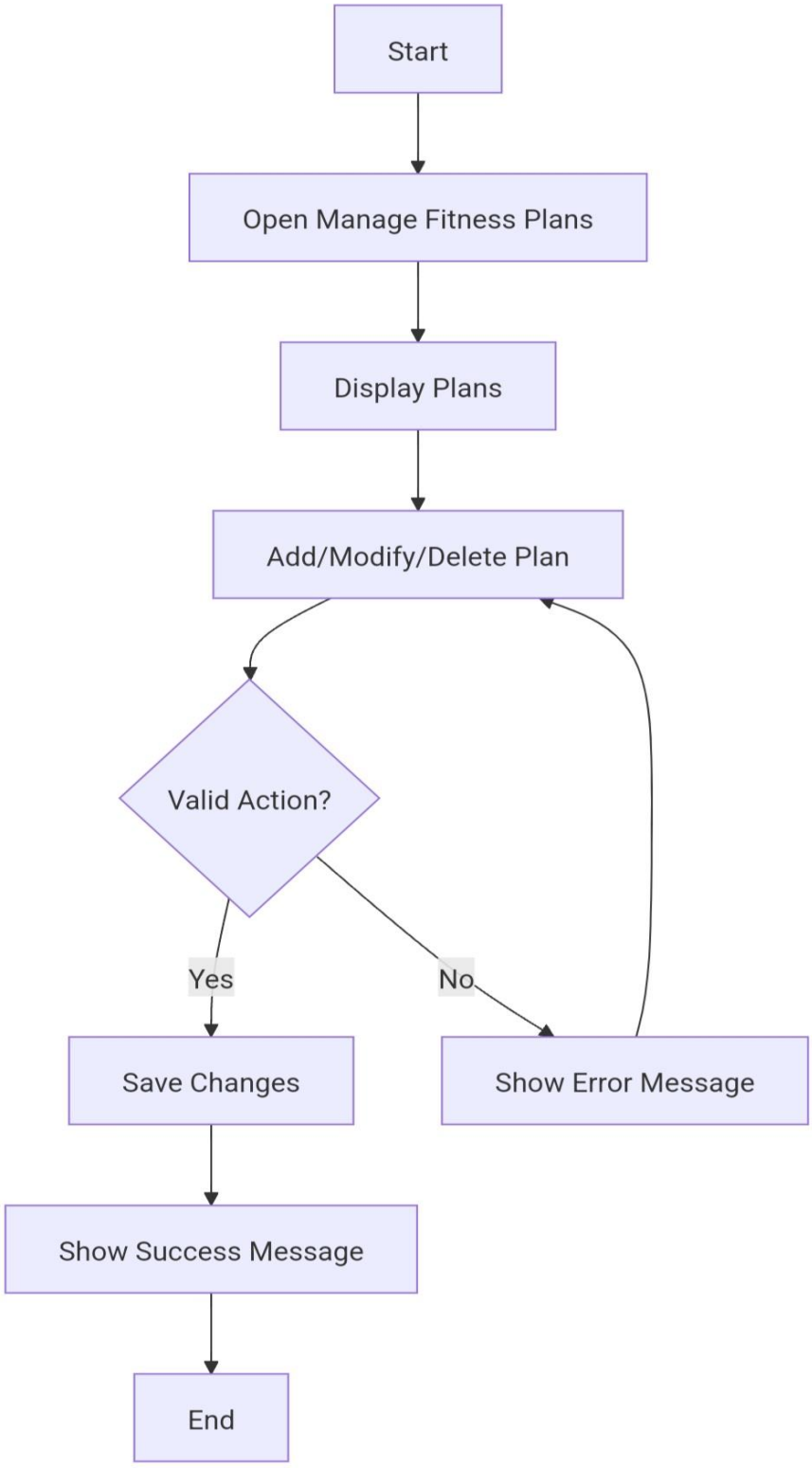

Figure 22 : Activity diagram for manage fitness plans

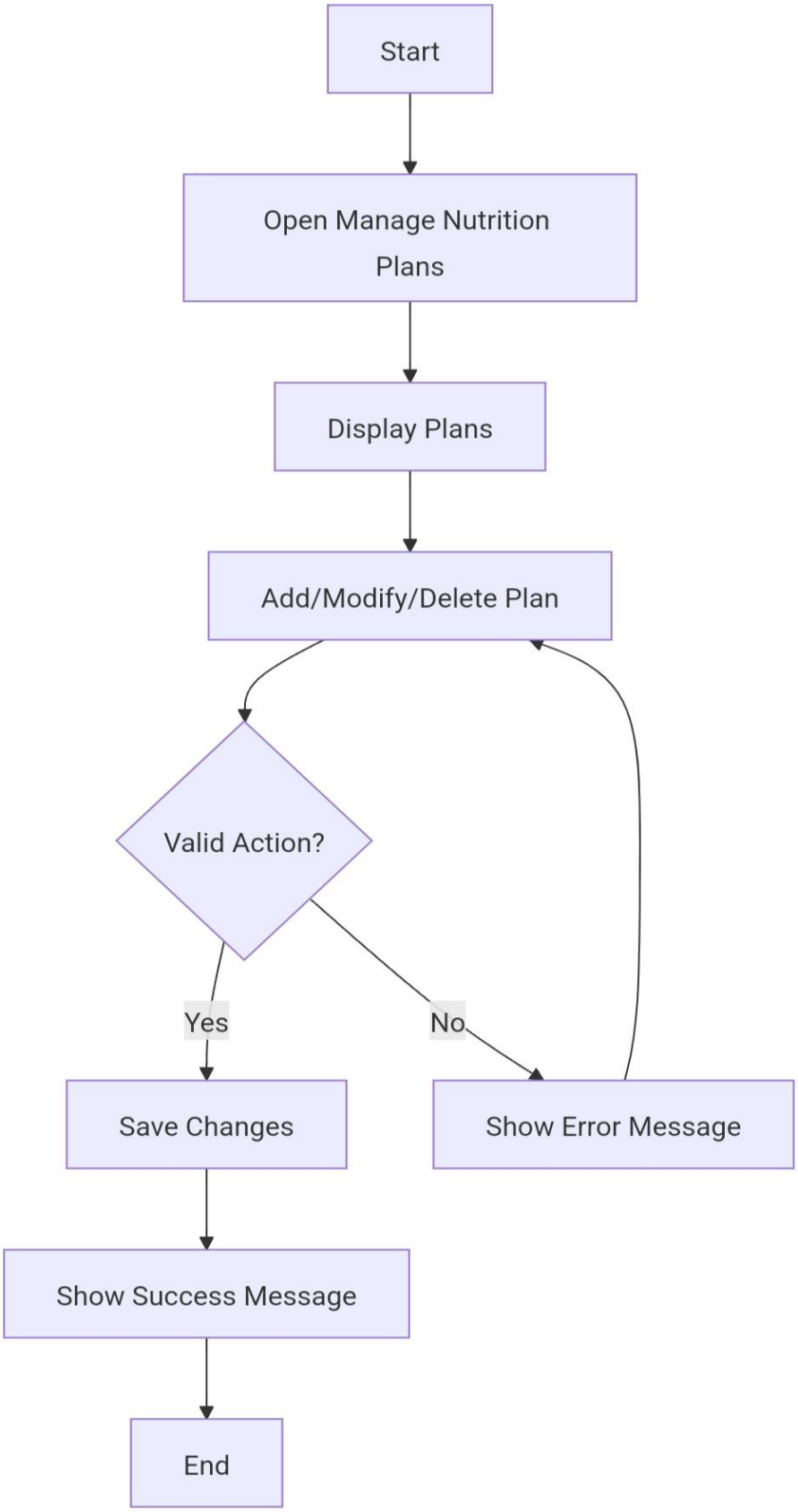

Figure 23 : Activity diagram for manage nutrition plans

2.26 State Chart Diagram

The state diagram used to show the sequence of states that an object goes through the events that cause the transition from one state to the other and the actions that result from a state change or states of an object during its lifetime and these states are changed by events. Moreover, a state chart diagram is a view of a state machine that models the changing behavior of a state. State chart diagrams show the various states that an object goes through, as well as the events that cause a transition from one state to another. The common model elements that state chart diagrams contain are States, Start, end states and Transitions.

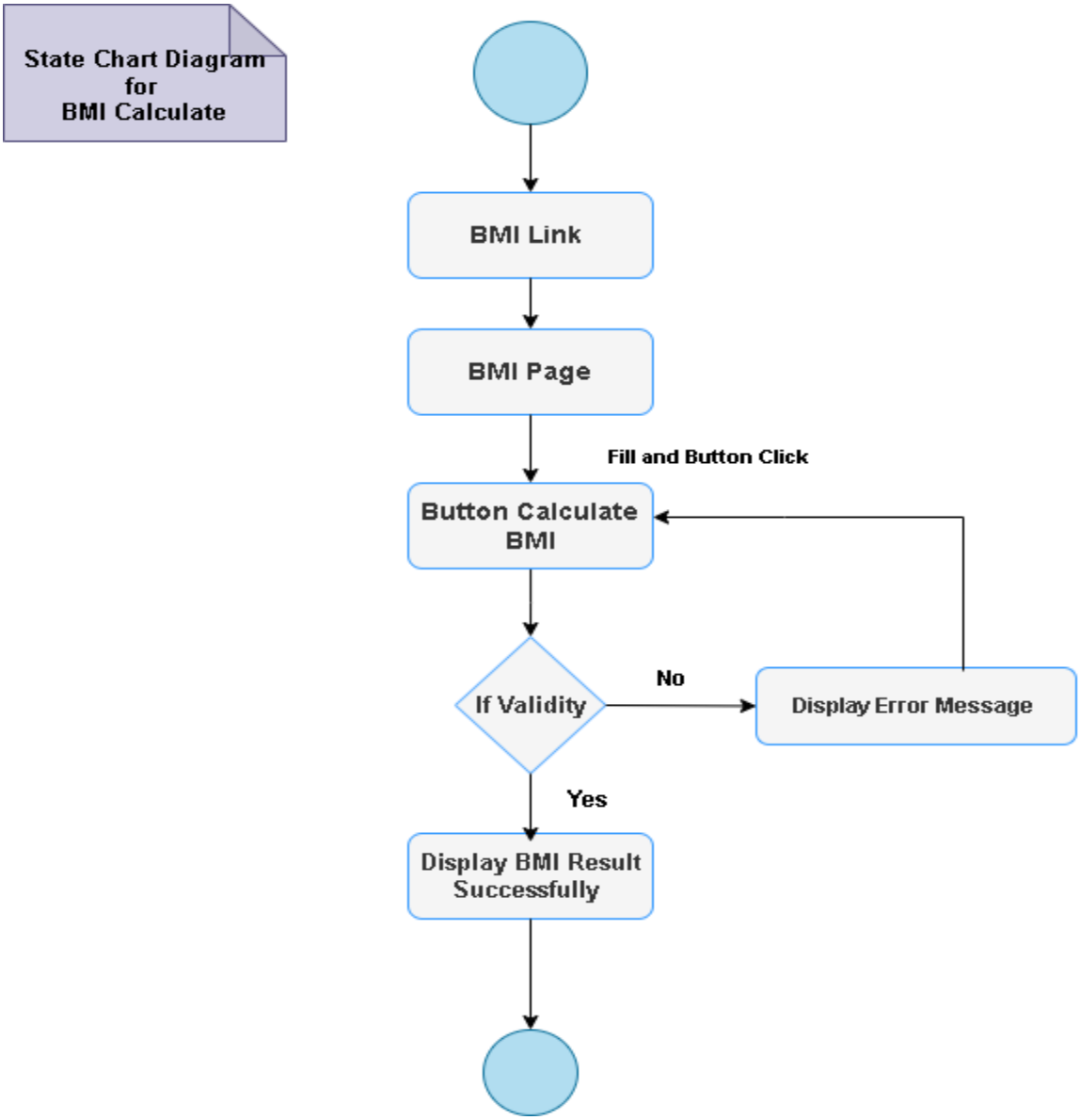

Figure 24 : State chart diagram for BMI Calculation

2.27 Object Model

An object model is a description of an object-oriented architecture, including the details of the object structure, and interfaces between objects. It identifies the attributes and functions of each class. In this section, we discuss the object model class diagram and data dictionary.

2.28 Class Diagram

Class diagrams are fundamental to the object modeling process and model the static structure of a system. Depending on the complexity of a system, you can use a single class diagram to model an entire system, or you can use several class diagrams to model the components of a system. Class diagrams are the blueprints of your system or subsystem. You can use class diagrams to model the objects that make up the system, to display the relationships between the objects, and to describe what those objects do and the services that they provide. You can use class diagrams to visualize, specify, and document structural features in your models.

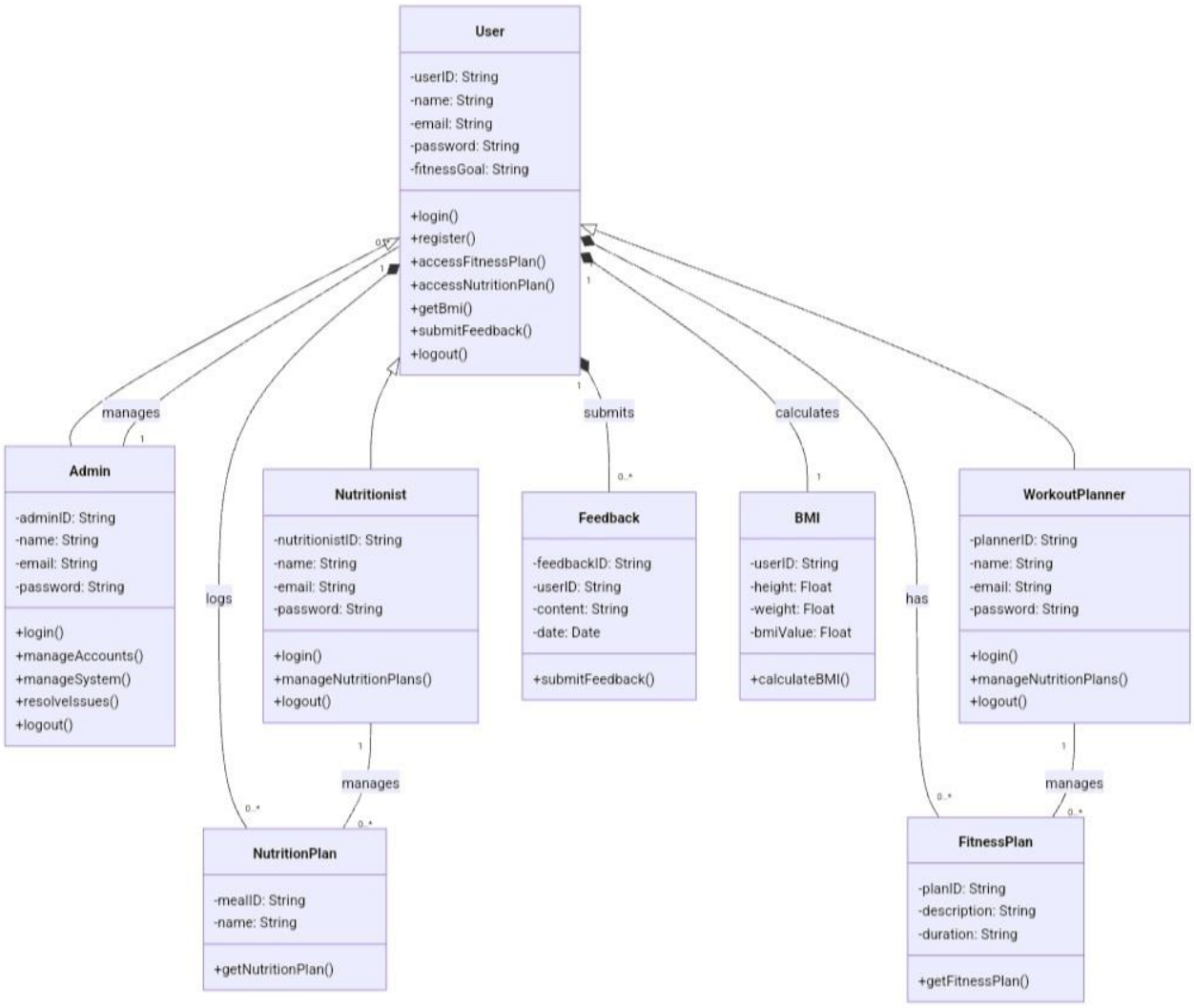

Figure 25 : Class diagram of the system

2.29 Data Dictionary

A set of information describing the contents, format and structure of a data base and the relationship between elements used to control access to other data, origin A repository of information about the layout of a database, a flat file, and a class, and any mappings among the three.

Some of our data dictionary is Described below:-

Table 13 : User table data dictionary

Attribute	**Data Type**	**Constraints**	**Description**
userID	Integer	PRIMARY KEY, NOT NULL	Unique ID for each user
name	String(50)	NOT NULL	Full name of the user
email	String(100)	UNIQUE, NOT NULL	User's email address
password	String(255)	NOT NULL	Encrypted password for user

Table 14: Get BMI Table data dictionary

Attribute	**Data Type**	**Constraints**	**Description**
userID	Integer	FOREIGN KEY	Reference to User table
height	Float	NOT NULL	Height of the user (in meters)
weight	Float	NOT NULL	Weight of the user (in kilograms)
bmiResult	Float	AUTO-CALCULATED	Result of BMI calculation

Table 15: Access fitness Plan Table data dictionary

Attribute	**Data Type**	**Constraints**	**Description**
planID	Integer	PRIMARY KEY, NOT NULL	Unique ID for each workout plan
userID	Integer	FOREIGN KEY	Reference to User table
planType	String(30)	NOT NULL	Type of workout plan (e.g., Cardio)
duration	String(20)	NOT NULL	Duration of the workout plan
description	String(255)	NOT NULL	Description of the workout

Table 16: Access nutrition Plan Table data dictionary

Attribute	**Data Type**	**Constraints**	**Description**
MealID	Integer	PRIMARY KEY, NOT NULL	Unique ID for nutrition plan
Name	String	FOREIGN KEY	Name of the nutrition plan

2.30 User Interface Design

Everything stems from knowing our users, including understanding their goals, skills, preferences, and tendencies. Once we know about our user, make sure to consider the following when designing our interface:

- Keep the interface simple.
- Create consistency and use common UI elements.
- Be purposeful in page layout.
- Strategically use color and texture.
- Make sure that the system communicates what's happening. Always inform your users of location, actions, changes in state, or errors.

Figure 26 : Login page UI

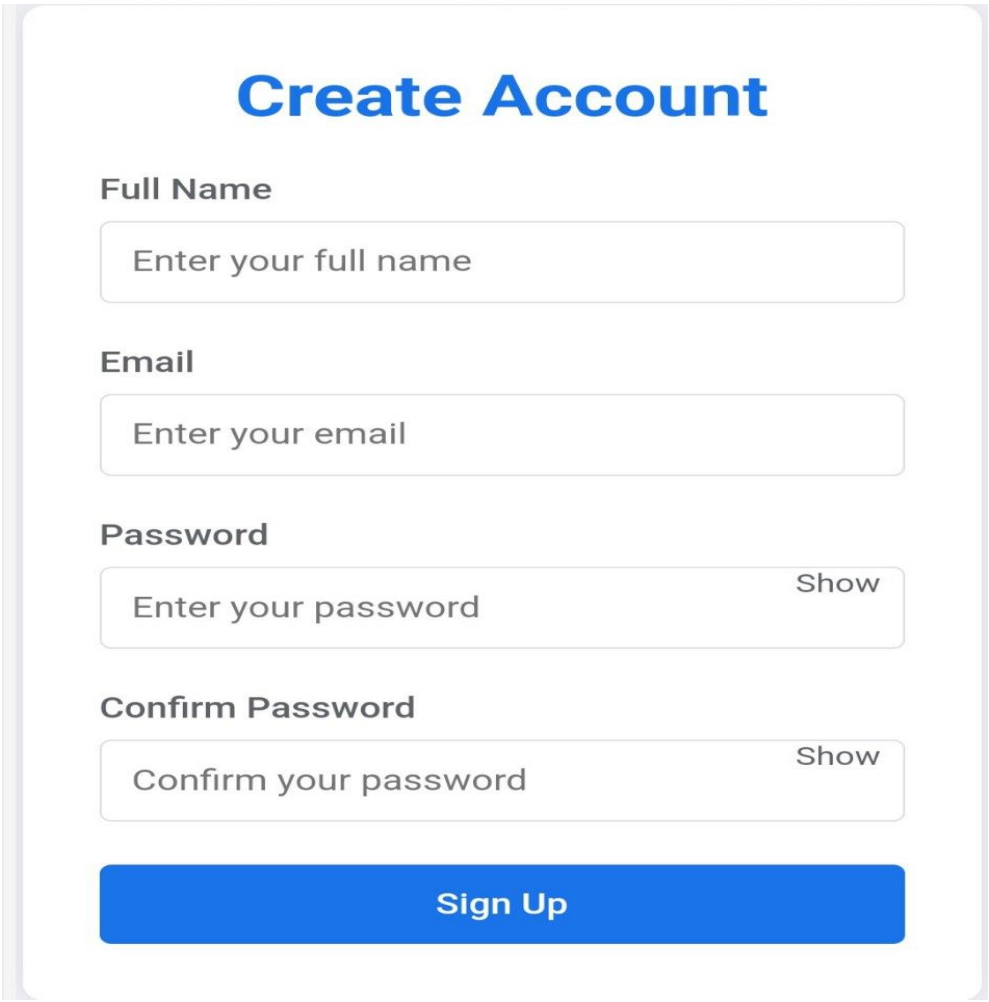

Figure 27 : Registration page UI

CHAPTER THREE

3. SYSTEM DESIGN

The system Design phase is a process of describing, organizing, and structuring system components at the architectural design level and detailed design level. Build a system Design converts functional models from analysis into models that helps to represent the solution for the problem (system design model). System design is the first part to get into the solution domain in software development.

3.1 Design Goals

Design goals describe the qualities of the system that developers should optimize. The design part is very important so as to make the implementation very easy. The design goals are derived from non-functional requirements that mean the non-functional requirement is the description of the feature characteristics and attribute of the system as well as any constraints that may limit the boundary of the proposed solution.

3.2 Proposed Software Architecture

This chapter outlines the software architecture of the Ethiopian Fitness and Nutrition Web App by providing a structured design. It includes high-level views, subsystem decomposition, mapping with hardware, persistent data management, security mechanisms, and subsystem services. Each section explains the assignment of functionality and relationships among system components. Three-tier architectures consist of three components distributed in 3 layers: client, application server and database.

- The client layer contains UI (User Interface) part. This layer takes input and gives output to the user.
- The middle tier (web/application server) is between the database layer and the presentation layer. A web server is a program that runs on a network server (computer) to respond to HTTP requests. The most commonly used web servers are Internet Information Server (IIS) and Apache. HTTP is used to transfer data across the Internet. It is the standard protocol for moving data across the internet.

- The database layer will be more secure and the client will not have direct access to the database.

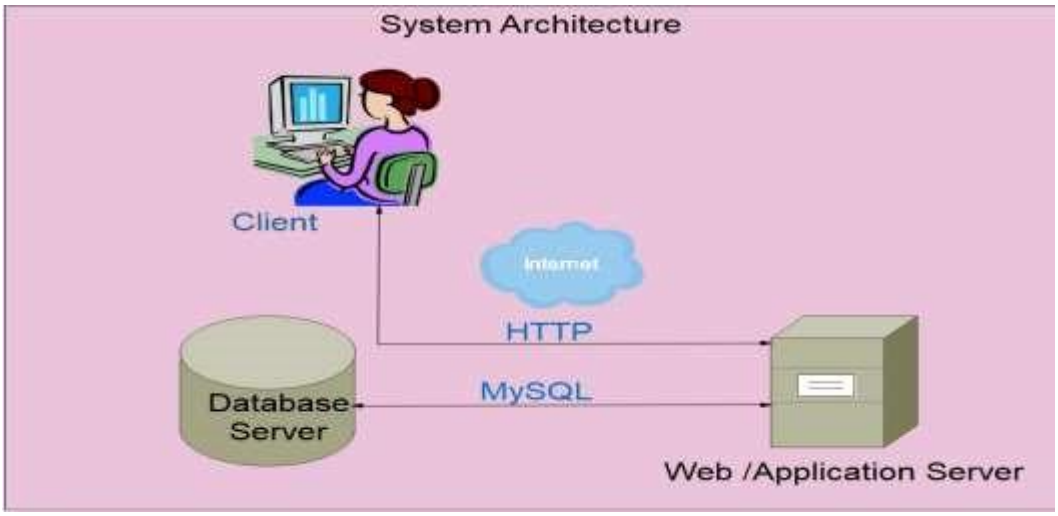

Figure 28 : system architecture

3.3 Overview

This section provides a bird's-eye view of the architecture and describes how functionalities are divided into subsystems. The architecture follows the multi-tier architecture model, separating the presentation layer, business logic layer, and data layer to ensure scalability, maintainability, and flexibility.

Key Components:

- **Presentation Layer** - User Interface (UI) for interacting with the system.
- **Application Logic Layer** - Manages business rules, workflows, and process execution.
- **Database Layer** - Manages persistent data storage and retrieval.

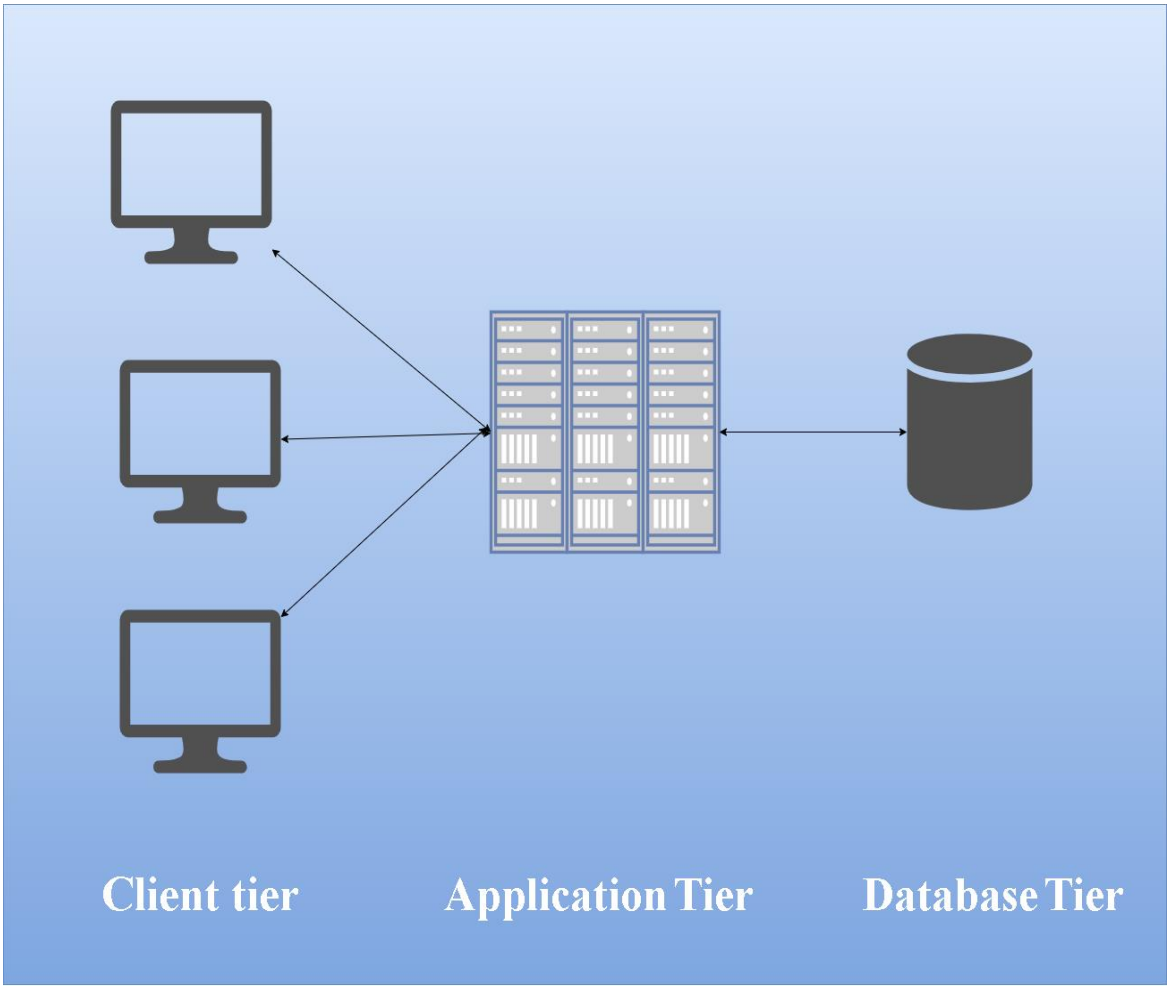

Figure 29 : General proposed architecture

High-Level View:

- Users access the system via web browsers or mobile devices.
- Requests are handled by the application server, which processes business logic.
- Data is retrieved or stored in the database server using structured queries.

3.4 Subsystem Decomposition

Subsystem decomposition is the activity of identifying subsystems, their services, and their relationships to each other. Decomposition is the process of breaking complex entities (processes, technology, business problems, business needs) into smaller sub parts, and then breaking those

smaller parts down even more, until the complex entity has been broken down into more discreet components with a more understandable.

User Management Subsystem:

- Handles user registration, login.

Authentication management Subsystem:

- Provides secure login and user authentication.

BMI Management Subsystem:

- Calculates the body mass index (BMI) based on user input (weight and height) .

Workout plan Management Subsystem:

- Provides workout routines.

Nutrition plan Management Subsystem:

- Recommends meal plans.
- Allows experts to create diet plans.

Food database Management Subsystem:

- Stores nutritional information about Ethiopian food items.

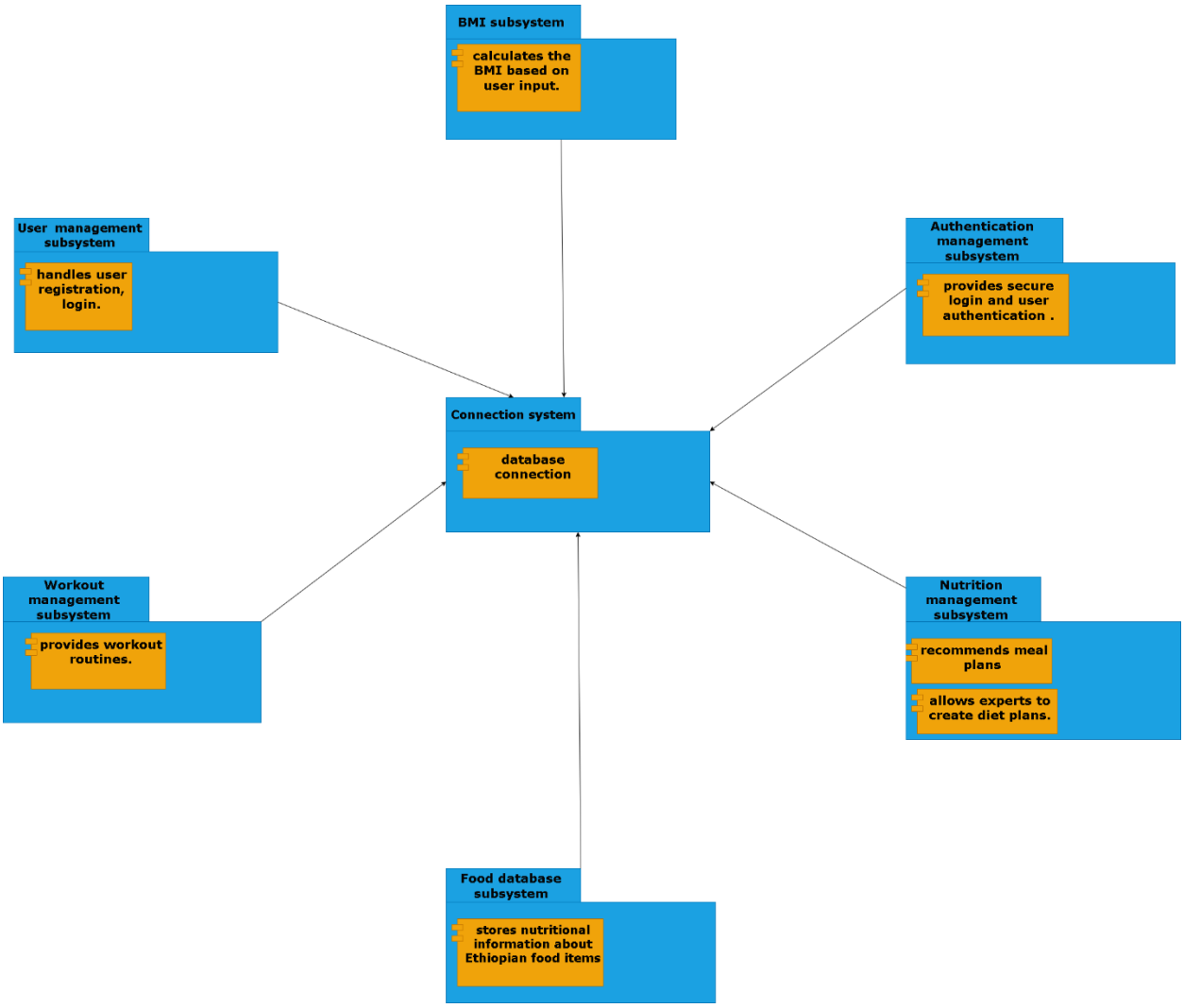

Figure 30: subsystem decomposition

3.5 Hardware/Software Mapping

This section focuses on mapping software subsystems to hardware resources for deployment. By using Deployment diagrams our system is designed by visualizing the topology of the physical components of the system, where the software components are deployed. Deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships. Deployment diagram represents the deployment view of a system. It is related to the component diagram because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware used

to deploy the application. An efficient deployment diagram is very important as it controls the following parameters.

- Performance
- Scalability
- Maintainability
- Portability

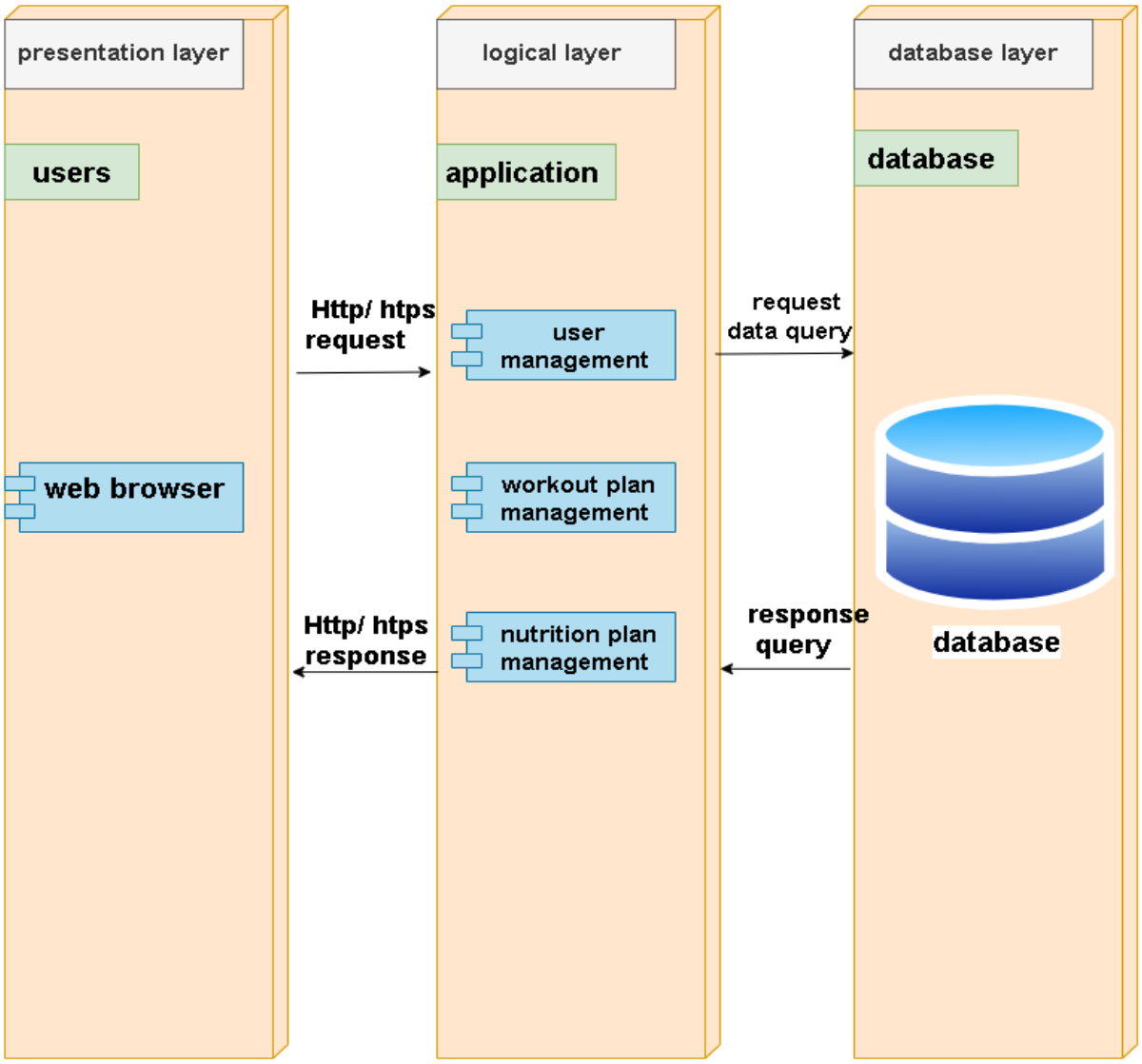

Figure 31: deployment diagram

3.6 Persistent Data Management

The ability to persistently store, retrieve, and destroy objects and data is encompassed by persistence data. Different tables have been employed as objects in the current database system, and each object is connected to the others. This schema makes it possible to choose, search, delete, and update data in the database.

The following figure indicates the persistent data management of the system.

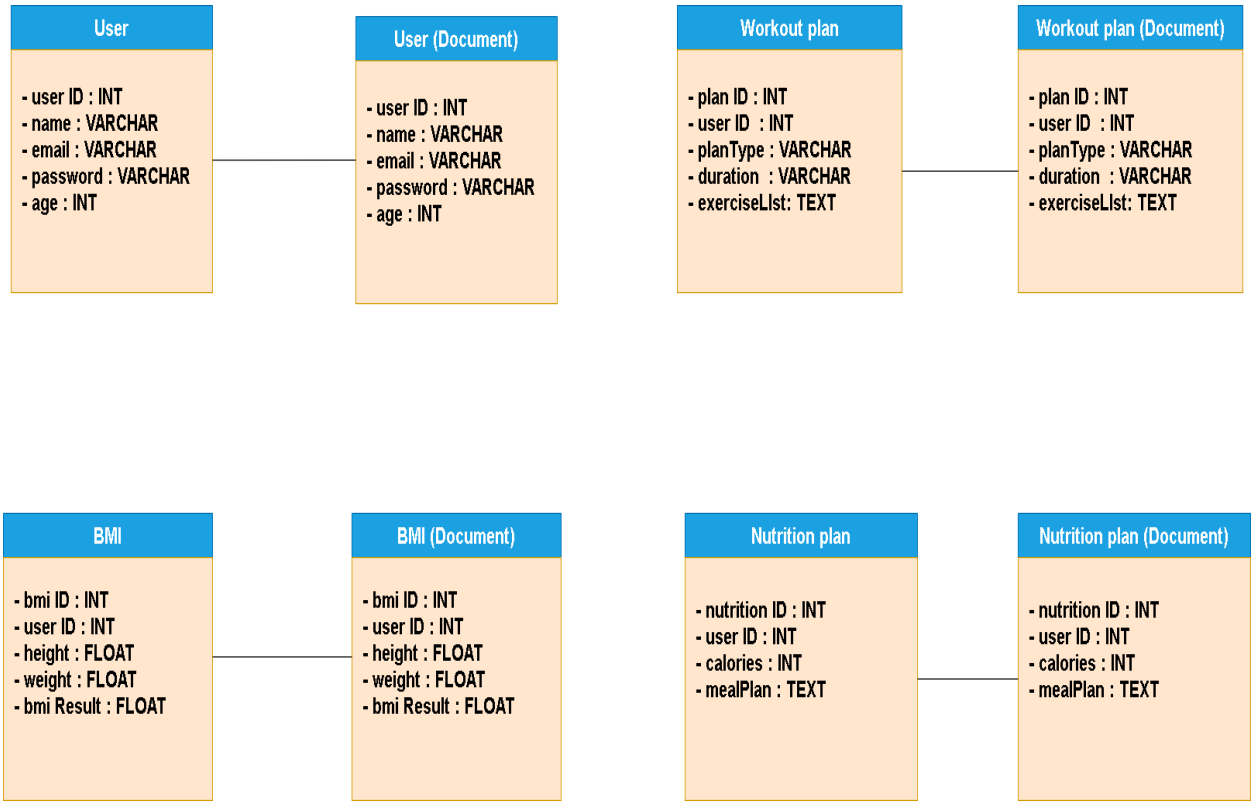

Figure 32: Persistent data management

Database Design and Schema:

Relational Database Model: Structured tables to store user details, plans, schedules, and logs.

Normalization: Optimized database structure to avoid redundancy.

SQL Queries: For data insertion, update, and retrieval.

Backup Strategies: Ensures disaster recovery and data safety.

Indexes: Improves data retrieval performance.

Data Management Tools:

SQL Queries: For data insertion, update, and retrieval.

Backup Strategies: Ensures disaster recovery and data safety.

3.7 Access Control And Security

This section describes user privileges and security mechanisms to protect the system from unauthorized access.

User Privileges Table:

Table 17 : user previlage table

User Role	Access Privileges
User (Client)	View content, register, login, access fitness plans.
Admin	Manage users, add/update/delete content.
Nutritionist	Provide meal plans.
Workout planner	Provide workout plans.

Security Mechanisms:

- **Authentication:** Username/Password validation and multi-factor authentication.
- **Encryption:** Use HTTPS for data transmission and encrypt sensitive data.
- **Session Management:** Session timeouts and automatic logout for idle users.

Detailed Class Diagram

This detailed class diagram shows classes, attributes, methods (operations), data types, visibility ((public (+), private (-), protected (#)) of the attributes and methods), inheritances, associations, aggregation, composition, dependencies, and multiplicities. Below we use the detailed class UML diagram to show how the aforementioned concepts are organized and designed for a better understanding of the system's detailed class diagram.

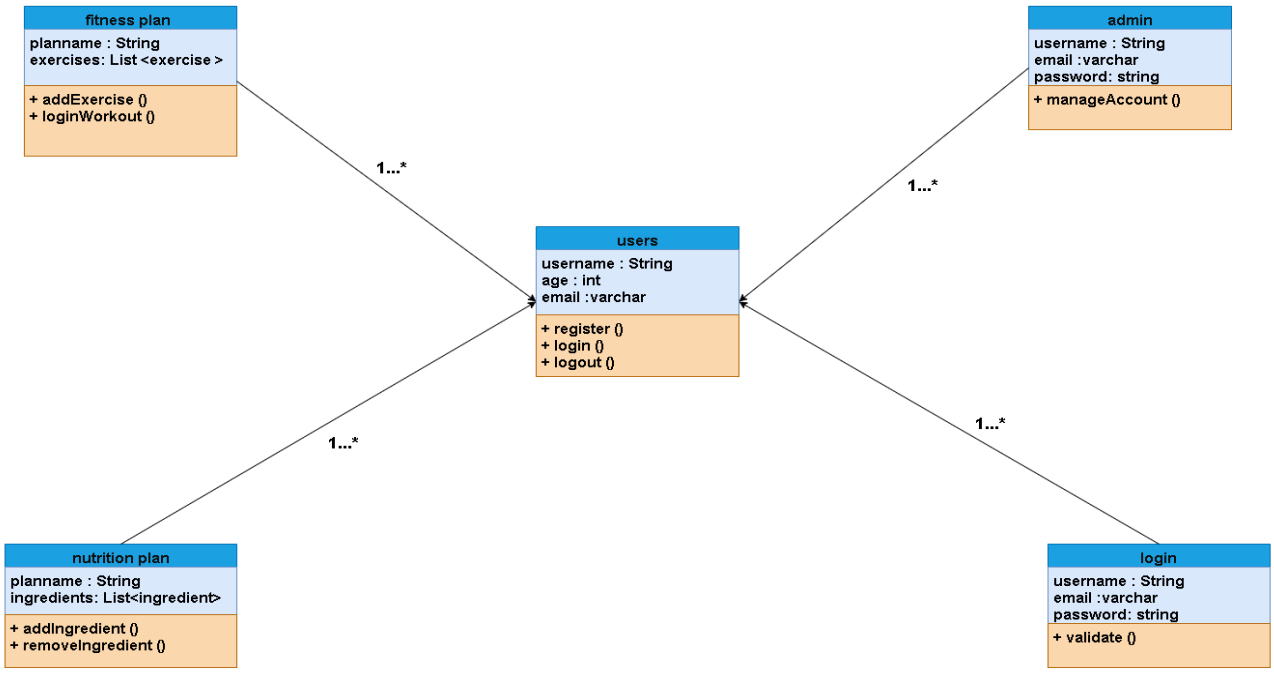

Figure 33: detailed class diagram

packages

A package diagram is a type of structural diagram that demonstrates how model pieces are arranged and organized within the system. It can display a system's numerous perspectives as well as its structure and connections between modules or subsystems. We divided the huge system into smaller modules and used a package diagram to manage high-level system components.

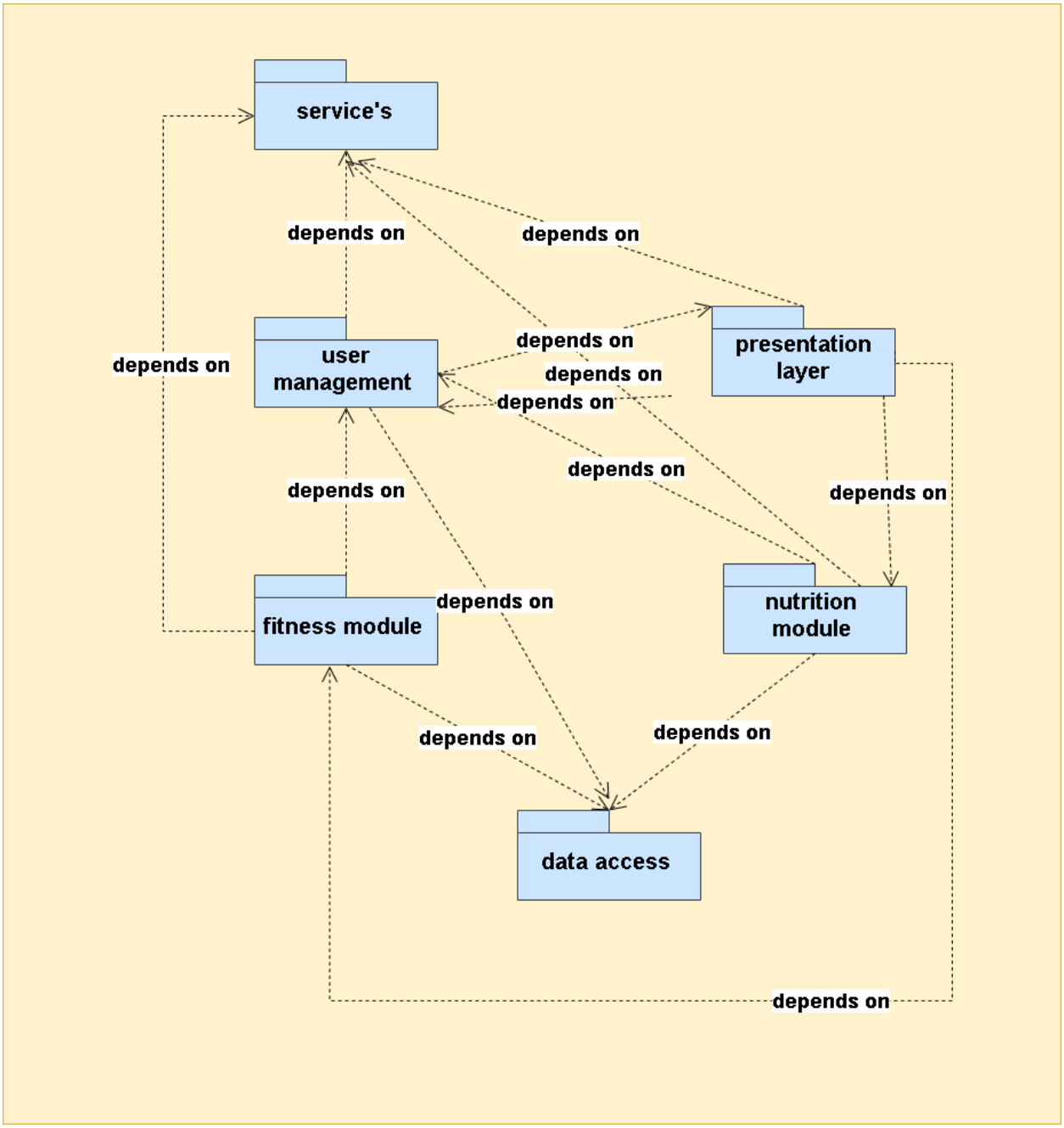

Figure 34: Package diagram

REFERENCES

Tools Used

1. Draw.io

Description: a free open-source diagramming tool that allows users to create, edit and collaborate on diagrams.

Website: draw.io

2. Gpt Models by Open Ai

Description: GPT-4 is a language model developed by OpenAI that generates human-like text based on the input it receives. It can assist with a variety of tasks, including writing, summarization, and information retrieval.

Website: OpenAI

3. **Figma**

Description: is a collaborative design tool that allows users to create, share, and test desighns for digital products.

Website : Figma.com

Informational Sources

1. Ethiopian Health and Nutrition Research Institute (EHNRI).
2. The Ethiopian Health And Nutrition Research Institute And Food And Agricutlute Organization Of The United Nations (Fao) 1995 -1997.