MATRUSRI ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Osmania University, Hyderabad)
16-1-486, Saidabad, Hyderabad – 500059

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Certificate

This is to certify that this record is a Bonafide laboratory work carried out by Mr./Ms. _____bearing Hall. Ticket No. _____of **B.E (IV Semester) CSE** for the course **Software Engineering Lab (PC401CSU23)** prescribed under Osmania University within the department during the academic year **2024-2025.**

Signature of HOD **Signature of Faculty In-charge**

Signature of External Examiner

INDEX

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION

The Computer Science and Engineering Department aims to produce competent professionals with strong analytical skills, technical skills, research aptitude and ethical values.

MISSION

M1: To provide hands-on-experience and problem-solving skills by imparting quality education
M2: To conduct skill-development programmes in emerging technologies to serve the needs of industry, society and scientific community.
M3: To promote comprehensive education and professional development for effective teaching-learning processes.
M4: To impart project management skills with an attitude for life-long learning with ethical values.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

- To learn engineering knowledge and problem analysis skills to design and develop solutions for computer science and engineering problems.
- To address the feature engineering with the usage of modern IT and Software tools.
- To acquire and practice the profession with due consideration to environment issues in conformance with societal needs and ethical values.
- To manage projects in multidisciplinary environments as a member and as a leader with effective communications.
- To engage in life-long learning in the context of ever-changing technology.

PROGRAM OUTCOMES (POs)

Upon the completion of programme, the student will be able to

1. **Engineering knowledge:** Apply and integrate the knowledge of computing to computer science and engineering problems.
2. **Problem analysis:** Identify, formulate and analyse complex engineering problems using computer science and engineering knowledge.
3. **Design/development of solutions:** Design and develop components or processes to engineering problems as per specification with environmental consideration.
4. **Conduct investigations of complex problems:** Interpret and integrate information to provide solutions to real world problems.
5. **Modern tool usage:** Select and apply modern engineering and information technology tools for complex engineering problems.
6. **The engineer and society:** Assess and responsible for societal, health, safety, legal and cultural issues in professional practice.
7. **Environment and sustainability:** Understand the impact of computing solutions in the context of societal, environmental and economic development.
8. **Ethics:** Commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function as an individual, as a member or leader in multidisciplinary environment.
10. **Communication:** Acquire effective written and oral communication skills on technical and general aspects.
11. **Project management and finance:** Apply engineering and management principles to manage projects in multidisciplinary environments.
12. **Lifelong learning:** Identify the need of self-learning and life-long learning in the broad context of technological evolution.

PROGRAM SPECIFIC OUTCOMES (PSOs)

Upon the completion of programme, the student will be able to

1. **Efficient coding:** Familiar with open-ended programming environments to develop software application.
2. **Software deployment:** Apply the knowledge of Computer System Design, Principles of Algorithms and Computer Communications to manage projects in multidisciplinary environments.

SOFTWARE ENGINEERING LAB COURSE OBJECTIVES

- To understand the software engineering methodologies for project development.
- To gain knowledge about open-source tools for computer aided software engineering (CASE). To develop test plans and test cases to perform validation testing.

SOFTWARE ENGINEERING LAB COURSE OUTCOMES

- ✓ Analyse and design software requirements in efficient manner.
- ✓ Use open-source case tools to develop software.
- ✓ Implement the design, debug and test the code.

Abstract

Literature survey:

In recent years, fitness and wellness tracking systems have gained significant popularity due to the increasing awareness of health-related issues. Several research studies have demonstrated the effectiveness of digital platforms in improving health metrics. A study by Carter et al. (2013) titled *"Effectiveness of a Web-Based Weight Loss Program"* found that participants who used a digital tracking system for meals and exercise showed 22% more weight loss over 12 weeks compared to a control group. Similarly, Payne et al. (2015), in their research on physical activity interventions using web-based platforms, concluded that online tools significantly increased daily activity levels and reduced sedentary behaviour, especially when combined with reminder features. Furthermore, Strecher et al. (2008) found that systems offering personalized goal setting and reminders led to improved health outcomes in 68% of users over a 6-month period. Supporting these findings, Wang et al. (2014) emphasized that smartphone-based interventions greatly enhance long-term health management in individuals with chronic conditions. Additionally, Coughlin et al. (2016) noted the positive role of mobile health technology in promoting healthy behaviors, particularly among underrepresented populations. These studies collectively highlight the positive impact of multi-functional fitness platforms in supporting long-term lifestyle changes and improving overall well-being.

Project Contribution

Building upon these research outcomes, this project introduces a comprehensive Fitness Website that integrates the most impactful features highlighted in previous studies:

- **Personal details management** (name, email, age, height, weight, goal)
- **Meal tracking** with automatic calorie and protein counting
- **Activity logging** (e.g., walking, cycling, running)
- **Progress tracker** (weight history)
- **BMI calculator** (using weight and height)
- **Reminders** for water intake, sleep, and workout notifications
- **Exercises** for gym, home, and ground-based training
- **Wellness videos**, including yoga, meditation, and breathing exercises

Estimated Results

The expected outcome of this project includes:

- A fully functional and responsive fitness tracking platform
- Accurate tracking of nutritional and activity data
- High usability and effectiveness confirmed through user testing
- Increased user awareness and motivation via reminders and visual progress

By integrating diverse wellness tools into a single platform, this project aims to inspire consistent, healthy habits and promote a smarter approach to personal fitness.

Introduction

Overview

The modern lifestyle has led to a growing need for personalized health and wellness solutions. To address this, the fitness website developed serves as a comprehensive, user-friendly platform tailored to help individuals monitor, manage, and improve their physical health and mental well-being. Built with a focus on functionality, usability, and user engagement, the website integrates core features such as personal data management, meal and activity tracking, progress monitoring, reminders, and educational content.

This platform not only allows users to log their daily fitness activities and nutrition but also supports them with informative resources, and a personalized dashboard. The goal is to create an all-in-one digital fitness companion that adapts to the needs of users at different stages of their wellness journey—whether beginners, intermediates, or advanced fitness enthusiasts.

Key Features and Functionalities

1. Personal Details Management

Users begin their journey by entering personal information including name, email, gender, weight, height, date of birth, weight goal and activity level. All data is securely stored and editable at any time, ensuring flexibility and ongoing relevance as users progress.

2. Meal Tracking with Calorie and Protein Calculation

The meal tracker enables users to log their daily food intake by selecting meals and specifying gram quantities. The system automatically calculates the total calories and protein consumed based on a built-in food database, helping users maintain nutritional awareness aligned with their health objectives. This module promotes mindful eating and supports users in achieving goals like muscle gain, weight loss, or maintenance.

3. Activity Logging

To track physical movement, the activity logging feature allows users to input daily activities such as walking, cycling, and running. For each activity, users can log duration, and the system estimates calories burned. This offers an effective way to evaluate daily energy expenditure and balance it with nutritional intake.

4. Progress Tracker

Progress tracking is a vital motivator in any fitness program. Users can log their weight over time, and the platform presents this data in a visual format such as line charts. This historical view helps users see how their efforts translate into real results and motivates them to stay on track with their goals.

5. BMI Calculator

The Body Mass Index (BMI) calculator uses the height and weight data to provide users with an instant snapshot of their health status (Underweight, Normal, Overweight, or Obese). This is a useful tool to set realistic targets and understand where users stand at the beginning of their fitness journey.

6. Smart Reminders

To maintain consistency, the website includes customizable reminders for:

- **Water intake**: Encourages regular hydration.
- **Sleep**: Promotes adequate rest for recovery and mental health.
- **Workout**: Ensures that users stick to their exercise routines.

These reminders can be configured by the user based on their preferred timing and frequency.

7. Exercises

The site offers structured workout modules categorized into:

- **Gym-based workouts**
- **Home workouts**
- **Ground-based exercises (e.g., running drills, outdoor routines)**

Each module includes recommended exercises, number of sets/reps, and equipment guidance, if required. These allow users to follow a guided workout plan tailored to their environment and equipment availability.

8. Wellness Videos

To support mental wellness and provide holistic fitness education, the site includes a video streaming section with high-quality content such as:

- **Yoga sessions**
- **Meditation guides**
- **Breathing exercises**

These videos are categorized for ease of access and allow users to explore practices that enhance flexibility, mindfulness, and relaxation—key components of total health.

This fitness website represents a robust digital platform aimed at empowering users to take control of their fitness in a holistic and structured manner. By merging diet tracking, physical activity monitoring, educational content, and smart automation, the site acts as a digital fitness coach and wellness assistant.

The user-centric design ensures accessibility for all levels of fitness, and the integration of multiple wellness components—from BMI calculations to yoga videos—makes it a valuable tool for anyone looking to enhance their health, one step at a time.

Aim

The aim of this fitness website project is to develop a comprehensive, interactive platform that empowers users to take control of their physical and mental health. By combining personalized data management, activity tracking, nutritional monitoring, and educational content, the website serves as a virtual fitness assistant. It is designed to help users stay consistent with healthy habits, track their progress over time, and access reliable resources for both physical training and mental wellness.

The platform encourages users to set and pursue their fitness goals—whether weight loss, muscle gain, or maintaining overall health—by providing structured tools and reminders, ensuring that health and fitness become an integral part of their daily lives.

Scope

- This project involves the design and development of a fully functional fitness website with the following key modules:
- **User Profile Management:** Users can register, log in, and manage personal details such as name, age, height, weight, gender, and activity level.
- **Meal Tracker:** Allows users to log meals and food intake with automatic calorie and protein calculations based on the quantity selected.
- **Activity Logger:** Supports tracking of physical activities like walking, running, and cycling, with estimated calories burned.
- **Progress Tracker:** Users can log their weight history over time and view progress through visual charts or summaries.
- **BMI Calculator:** Instantly calculates the user's Body Mass Index using height and weight data, providing health classifications.
- **Reminders:** Offers customizable alerts for water intake, sleep routines, and workouts to help build consistent health habits.
- **Exercises:** Includes categorized routines for gym, home, and outdoor (ground-based) exercises with instructional guidance.
- **Wellness Videos:** Provides access to videos on yoga, meditation, and breathing exercises to promote holistic well-being.

Software Requirements:

- **Operating System:**
  - **Development:** Windows 10+ (used for coding and local testing)
  - **Production (if deployed):** Ubuntu Server (recommended)
- **Development Tools:**
  - **Visual Studio Code** (used for writing and editing code)
- **Backend Environment:**
  - **Node.js** (LTS version 18.x or 20.x)
  - **npm** (Node Package Manager)
  - **Express.js** (web framework)
  - **dotenv** (to manage environment variables)
  - **jsonwebtoken** (for user authentication)
  - **bcrypt** (for password encryption)
- **Database:**
  - **MySQL** (used to store user data, meals, activity logs, etc.)
  - **MySQL Workbench or phpMyAdmin** (for managing the database)
- **Optional Deployment Tools (if hosted online):**
  - **PM2** (for managing Node.js in production)
  - **Docker** (for containerized deployment)
  - **Nginx** (for reverse proxy and SSL)

Hardware Requirements

- **For Local Development:**
  - **Processor:** Intel Core i5 / Ryzen 5 or better
  - **RAM:** 8 GB minimum (16 GB recommended)
  - **Storage:** 256 GB SSD minimum (512 GB recommended)
  - **Network:** Stable internet connection
- **For Production Server (only if deploying):**
  - **Small Scale:** 2 vCPU, 4–8 GB RAM, 80–100 GB SSD
  - **Platform:** Cloud VM (e.g., AWS, DigitalOcean) with Ubuntu Server.

UNIFIED MODELLING LANGUAGE

INTRODUCTION

UML is a graphical notation used to visualize, specify, construct and document the artifact of software intensive. UML is appropriate for modelling systems ranging from Enterprise Information Systems to Distributed Web-based Application and even to Hard Real-time Embedded systems. UML effectively starts with forming a conceptual modelling of the language.

MODEL

- ✓ A model is a simplification of reality.
- ✓ A model provides the blueprints of system.
- ✓ A model may be structural, emphasizing the organization of the system, or it may be behavioural, emphasizing the dynamics of the system.
- ✓ We build models so that we can better understand the system we are developing.
- ✓ We build models of complex systems because we cannot comprehend such a system in its entirety**.**

Through modelling, we achieve four aims.

1. Models help us to visualize a system as it is or as we want it to be.
2. Models permit us to specify the structure or behaviour of a system.
3. Models give us a template that guides us in constructing as system.
4. Models document the decisions we have made

Principles of Modelling:

- The choice of what models to create has a profound influence on how a problem is attacked and how a solution is shaped
- Every model may be expressed at different levels of precision
- The best models are connected to reality
- No single model is sufficient. Every nontrivial system is best approached through a small set of nearly independent models

Applications of UML: UML is intended primarily for software intensive systems. It has been used effectively for such domains as

1. Enterprise Information Systems
2. Banking and Financial Services
3. Telecommunications
4. Transportation
5. Defence and Aerospace
6. Medical Electronics
7. Distributed Web-based Services

Basic Building Blocks of UML:

The building blocks of UML can be categorized as

1. Things
2. Relationships and
3. Diagrams

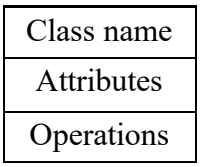
THINGS:

Things are the most important building blocks of UML.

Things can be a) Structural
b) Behavioural
c) Grouping
d) Annotational

a) Structural Things: These define the static part of the model. These represent physical and conceptual elements. Following are the structural things:

1. *Class*: It describes set of objects that share same attributes, operations, relationships and semantics.



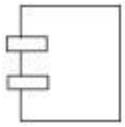
2. *Object*: It is a collection of operations that specifies the service of a class or a component.

3. *Collaboration*: It defines interaction between elements.

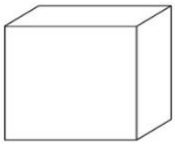
4. *Use case*: They are used to identify different use case components of a particular software project. It is used to model the operation.

5. *Actor*: The outside entity that communicates with a system. Typically, a person playing a role on an external device.

6. *Component*: It is a physical and replaceable part that confirms to and provides realization of set of interfaces.

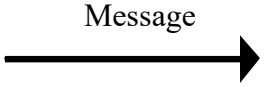


7. *Node*: A physical resource that exists in runtime and represents a computational resource.



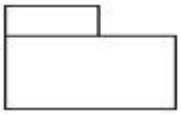
b) Behavioural Things: These consist of dynamic parts of the UML model. These are:

1. *Interaction*: It is defined as a behaviour that consists of a group of messages exchanged among elements to accomplish a specific task.



2. *State machine*: It is useful when the states of an object in its life cycle. It defines the sequence of states and object goes through in response to events.

c) Grouping Things: They can be defined as a mechanism to group elements of UML model together. There is only one grouping thing available i.e., Package. Package is used for gathering structural and behavioural things.



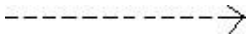
d) Annotational Things: They can be defined as a mechanism to capture remarks, description and comments of UML model elements. There is only one annotational thing available i.e., Note. Note is used to render comments, constraints and so on of a UML element.

RELATIONSHIPS:

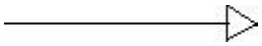
The relationship is another most important building block of UML. They show how elements are associated with each other, and their association describes the functionality of application.

There are 4 types of relationships. They are:

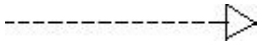
1. Dependency: It is a relationship between two things in which one element is dependent on another element. i.e., change in one element also affects another.



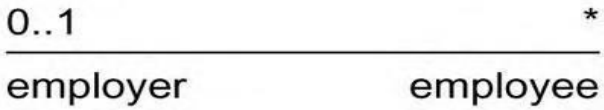
2. Generalization: It can be defined as a relationship which connects a specialized element with a generalized element. It basically describes inheritance relationship in the object. It is also called as
   'Is a' relationship.



3. Realization: It can be defined as a relationship in which two elements are connected. One element describes some responsibility which is not implemented, and the other one implements it. This relationship exists in case of interfaces.



4. Association: It is a set of links that connects elements of an UML mode



UML DIAGRAMS

There are 2 types of UML diagrams. They are:

Static Diagrams:

1. Class diagram
2. Object diagram
3. Component diagram
4. Deployment diagram

Dynamic Diagrams:

5. Use case diagram
6. Activity diagram
7. Sequence diagram
8. Collaboration diagram
9. State machine diagram

1. USE CASE DIAGRAM

A use case diagram describes a set of sequences in which each sequence indicates the relation with outside things. A use case involves the interaction of actor and system. There exist 3 types of relationships – Association, Dependency, Generalization. Use case diagrams can contain

- Actors – "things" outside the system
- Use cases – system boundaries identifying what the system should do.

Use case diagram can be used during analysis to capture the system requirements and to understand how the system should work. During the design phase, you can use use-case diagrams to specify the behaviour of the system as implemented.

Actor: An actor represents system users. They help delimit the system requirements and give a clearer picture of what the system should do. An actor is someone or something that interacts with or uses system.

- Provides input to and receives information from the system.
- Is external to the system and has no control over the use cases.

Customers/Actors are discovered by examining,

- Who directly uses the system
- Who is responsible for maintaining the system?
- External hardware used by the system.
- Other systems that need to interact with the system.

Use case: A use case is defined as a set of sequence of actions performed by an actor to achieve a specific result. Use case can be characterized as:

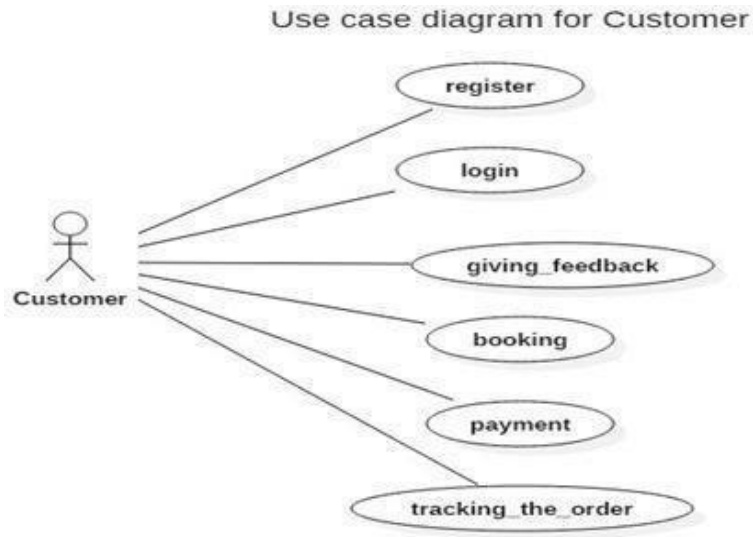
- A pattern of behaviour the system exhibits.
- A sequence of related transactions performed by an actor and the system.
- Delivering something of value to the actor.

Use cases provide a means to,

- Capture system requirements.
- Communicate with end users and domain experts. Test the system.

Note: Use cases often start with a "verb".

Example:



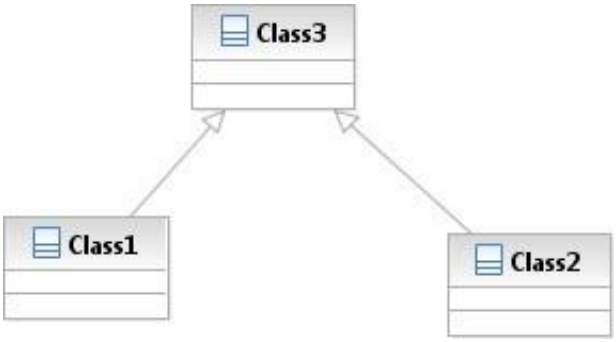
2. CLASS DIAGRAM

A class is a set of objects that share a common structure and common behaviour (the same attributes, operations and semantics). A class is an abstraction of real-world *things*. When these items exist in the real-world, they are instances of the class and are referred to as objects.

A class is represented using rectangle, which can be partitioned further.

Class name
Attributes
Operations

1. **Generalization Relationship for classes:** It shows that sub classes share the structure or behaviour defined in one or more super classes. Use a generalize relationship to show "is a" relationship. **Super class**



Subclass 1 **Subclass 2**

2. **Dependency Relationship:** The dependency is a relationship between two model elements in which change in one element will affect the other model element. Typically, in class diagrams, a dependency relationship indicates that the operations of the client invoke operation of the supplier.

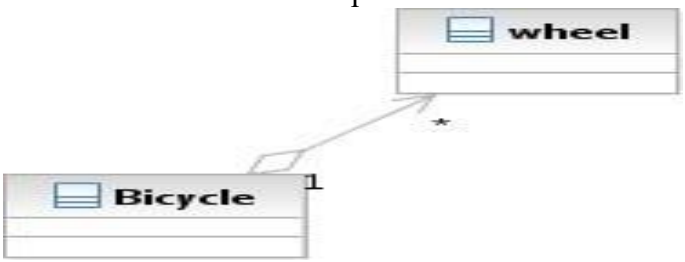
3. **Association Relationship:** An association provides a pathway of communications. The communication can be between use cases, actors, classes or interfaces. If two classes are usually considered independently, the relationship is an association. An association is an orthogonal or straight line.

Unidirectional **Bidirectional**

Multiplicity: Multiplicity specifies how many instances of one class may be associated with single instance of other class. When you apply a cardinality adornment to a class, you are indicating number of instances allowed for that class. A relationship, you are indicating number of links allowed between one instance of a class and the instances of another class.

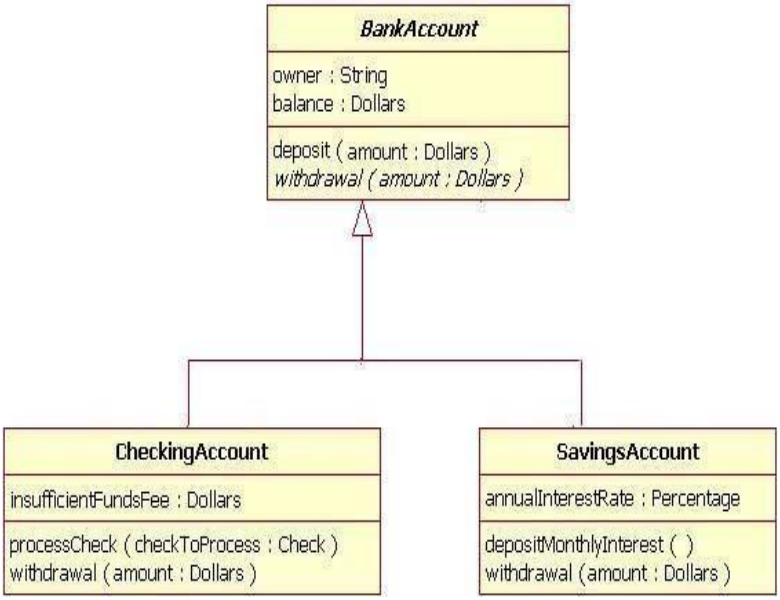
Valid Values:	**Value Description**
0..0	zero
0..1	zero or one
0.. n	zero or more
1..1	one
1.. n	one or more
n	unlimited number

Aggregate Relationship: Use the aggregate relationship to show a whole or part relationship between two classes. The diamond end represents the class which is whole.



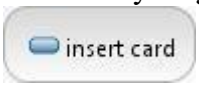
Interface: An interface specifies the externally visible operations of a class and/or component and has no implementation of its own. An interface specifies only a limited part of behaviour of class or a component. It is represented using a small circle.

Example:

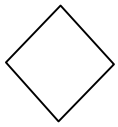


3. ACTIVITY DIAGRAM

An activity diagram is a special case of state diagram. An activity diagram is like a flow Machine showing the flow a control from one activity to another. An activity diagram is used to model dynamic aspects of the system. Activity diagram contains:



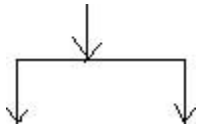
- ***Action state:*** These are atomic, executable computation which represents the execution of an action.
- ***Activity state:*** They can be decomposed. That is, their activity is represented by other activity diagrams.
- ***Branching:*** In branching, we have one incoming transition and two or more outgoing transitions.



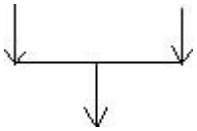
. Decision

- ***Forking:*** It is a process of splitting a single flow of control into multiple flow of controls.

Generally, a fork has a single incoming flow of control but multi outgoing flow of control.

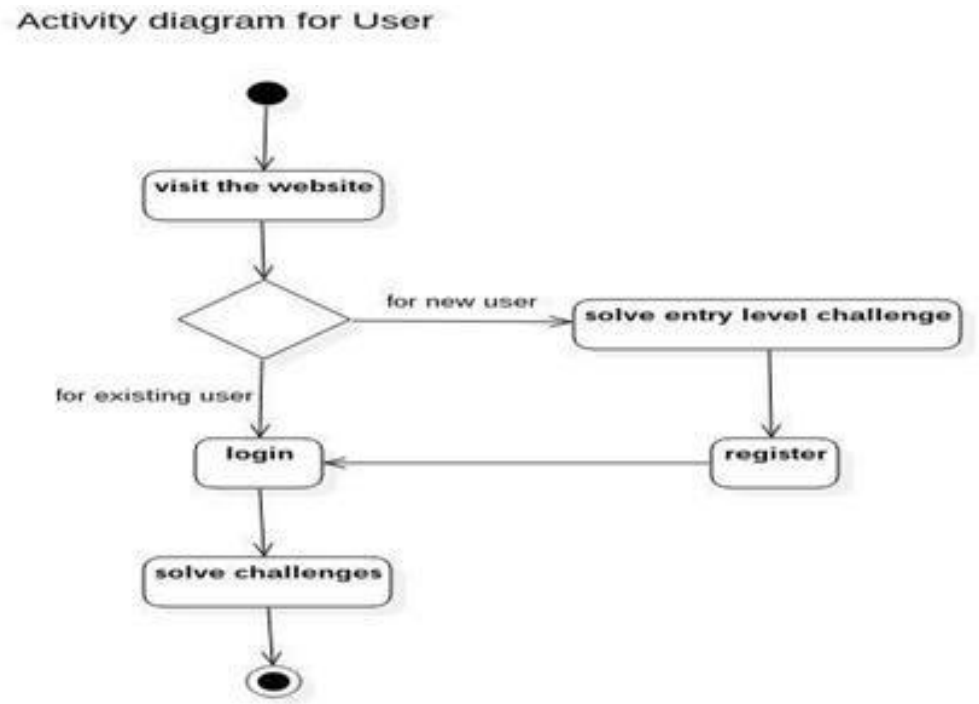


- ***Joining:*** It is exactly opposite of forking. It generally has multiple incoming flows of control but single outgoing flow of control.



Swim-lanes: They represent the columns in the activity diagram to group the related activities. These are represented in the form of partitioned region. Swim-lanes are helpful when modelling a business workflow because they can represent organizational units or role within business model. Swim-lanes are very similar to an object because they provide a way to tell who is performing a certain role.

Example:



INTERACTION DIAGRAMS

An interaction is an important sequence of interactions between objects. There are two types of interaction diagrams,

- Sequence Diagrams.
- Collaboration diagrams.

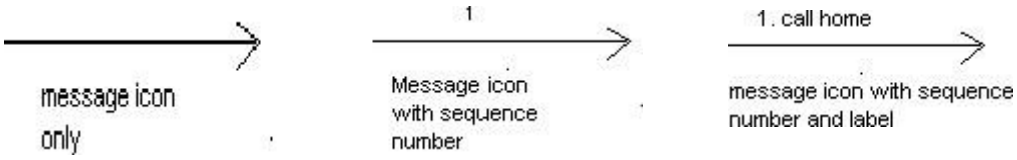
4. SEQUENCE DIAGRAM

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence, what happens first, what happens next.

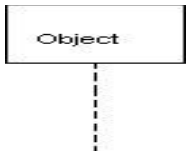
Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.
A sequence diagram has two dimensions: vertical placement represents time, and horizontal placement represents different objects.

Link: Objects interact through their links to other objects. A link is an instance of an association, analogous to an object being instance of a class. A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes.

Message icons: A message icon represents the communication between objects indicating that an action will follow. The message icon is a horizontal, solid arrow connecting two lifelines together. A message icon in a sequence diagram represents exactly one message.



Lifeline: Each object appearing on the sequence diagram contains a dashed vertical line, called lifeline, which represents the location of an object at a particular point in time. The lifeline also serves as a place for messages to start and stop and a place for the focus of control to reside.

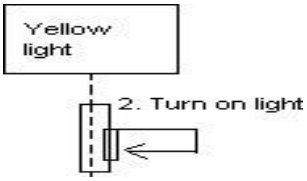


Lifeline

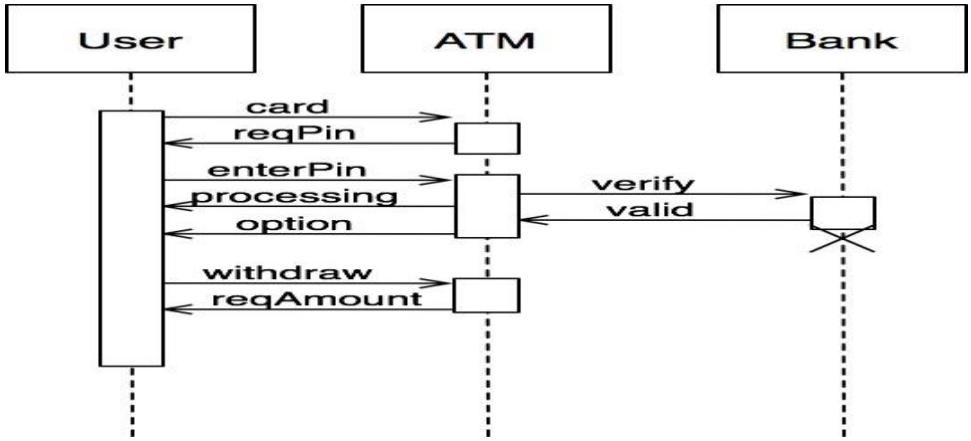
Message or Event: A message is a communication carried between two objects that trigger an event. A message is represented in collaboration and sequence diagrams by a message icon which usually indicates its synchronization. Synchronization types that are supported are:

1. Synchronous Call
2. Asynchronous Call
3. Asynchronous Signal
4. Create
5. Delete
6. Reply

Message to self: It is a tool that sends a message from one object back to the same object. The sender of the message is same as the receiver.



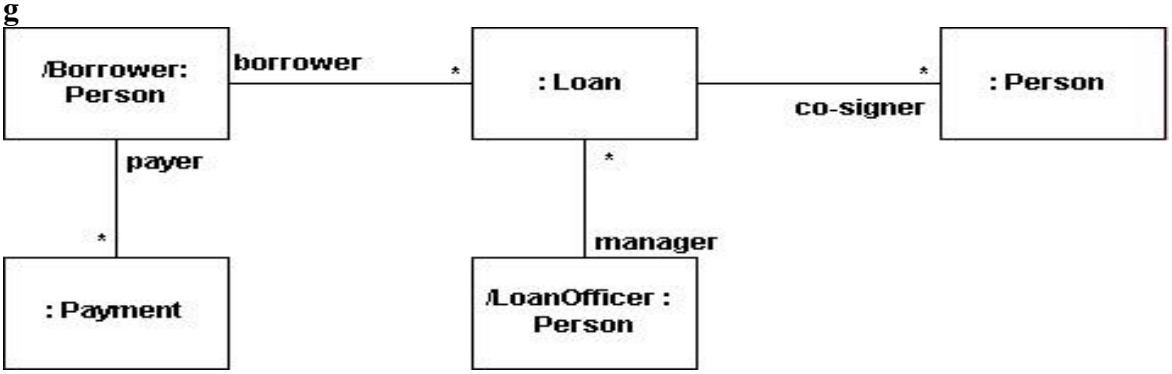
Example



5. COLLABORATION DIAGRAM

A collaboration diagram is an interaction diagram that shows the order of messages that implement an operation or a transaction. Collaboration diagrams show objects, their links, and their messages. They can also contain simple class instances and class utility instances. Each collaboration diagram provides a view of interactions or structural relationships that occur between objects and object-like entities in the current model. Collaboration diagrams contain icons representing objects.

E
g



Sequence and collaboration diagrams are semantically equivalent as both shows the interaction among objects. From one diagram we can generate another diagram. To generate a collaboration diagram from sequence diagram, right click on sequence diagram, select - Add Diagram - communication diagram. Similarly, a sequence diagram can be generated from collaboration diagram.

6. STATE MACHINE DIAGRAM

State Machine diagrams model the dynamic behaviour of individual classes or any other kind of object. They show the sequence of states that an object goes through the events that cause a transition from one state to another and the actions that result from a state change. A State Machine diagram is typically used to model the discrete stages of an object's lifetime. A State Machine diagram typically contains one start state and multiple end states.

State: A state represents a condition or situation during the life of an object during which it satisfies some condition or waits for an event. Each state represents a cumulative history of its behaviour. States can be shared between state machines. Transitions cannot be shared.

Naming: The name of the state must be unique to its enclosing class, within the state

Actions: Actions on states can occur at one of four times on entry
On exit, Do Activity, On event and Deferred.

Start state: A start state (also called an "initial state") explicitly shows the beginning of the execution of the state machine on the state Machine diagram or beginning of the workflow on an activity diagram. Normally, one outgoing transition can be placed from the start state. However, multiple transitions may be placed on start state, if at least one of them is labelled with a condition. No incoming transitions are allowed.
The start state icon is a small, filled circle that may contain the name (Begin process).

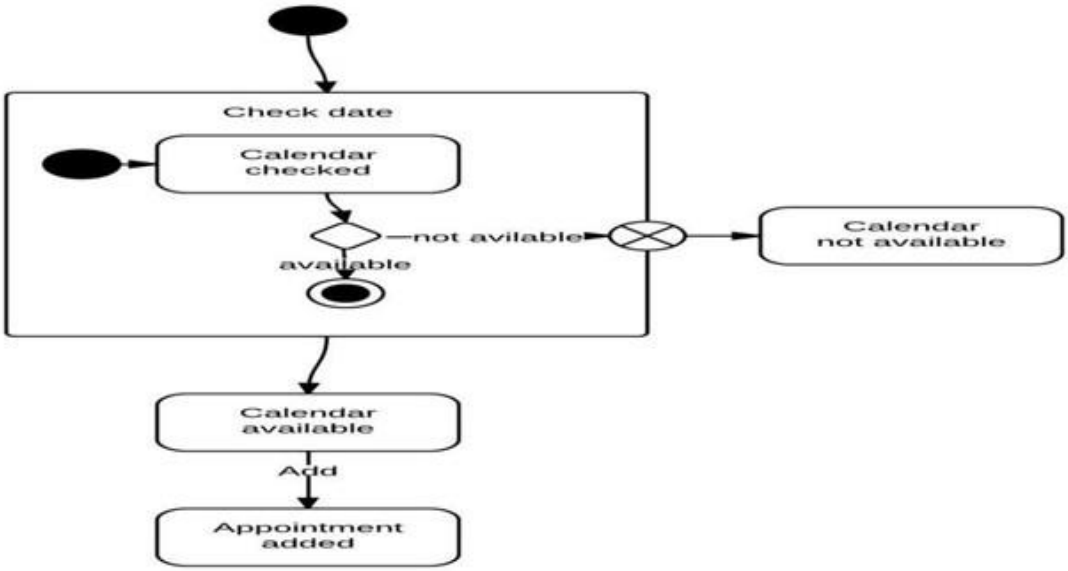
 Begin Process

End state: An end state represents a final or terminal state on an activity or State Machine diagram. Transitions can only occur into an end state. The end state icon is a filled circle inside a slightly larger unfilled circle that may contain the name (End process).

 End process

State transition: A state transition indicates that an action in the source state will perform certain specified actions and enter the destination state when a specified event occurs or when certain conditions are satisfied. A state transition is a relationship between two states, two activities or between an activity or a state. The icon for a state transition is a line with an arrowhead pointing toward the destination state or activity.
We can show one or more state transitions from a state as long as each transition is unique. Transitions originating from a state cannot have the same event, unless there are conditions on the event.

Example:



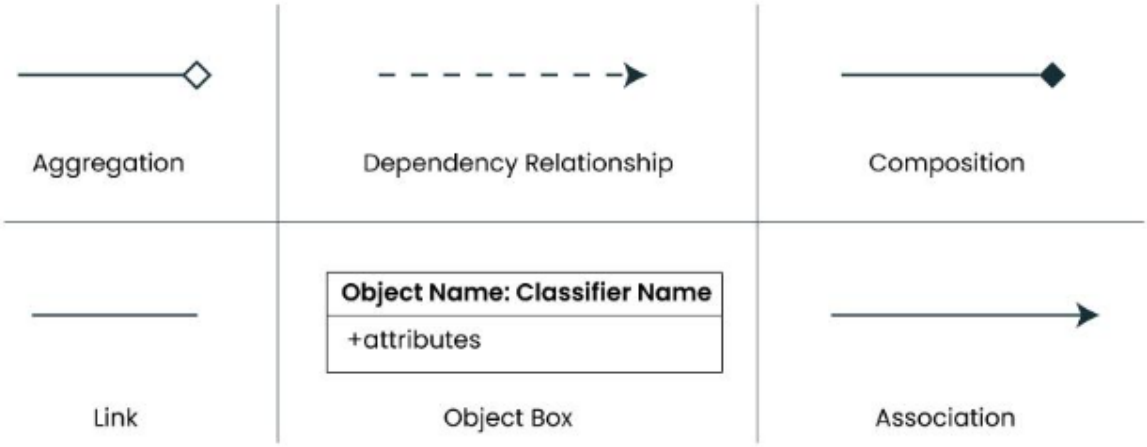
7. OBJECT DIAGRAM

An Object Diagram can be referred to as a screenshot of the instances in a system and the relationship that exists between them.

An object diagram in UML is useful because it provides a clear and visual representation of specific instances of classes and their relationships at a particular point in time, aiding in understanding and communicating the structure and interactions within a system.

In other words, "An object diagram in the Unified Modelling Language (UML), is a diagram that shows a complete or partial view of the structure of a modelled system at a specific time.

Notations:



1. Objects or Instance specifications: When we instantiate a classifier in a system, the object we create represents an entity which exists in the system. We can represent the changes in object over time by creating multiple instance specifications. We use a rectangle to represent an object in an object diagram.

2. Attributes and Values: Inside the object box, attributes of the object are listed along with their specific values.

3. Link: We use a link to represent a relationship between two objects. We represent the number of participants on the link for each, at the end of the link. The term link is used to specify a relationship between two instance specifications or objects. We use a solid line to represent a link between two objects.

4. Dependency Relationships: We use a dependency relationship to show when one element depends on another element. A dependency is used to depict the relationship between dependent and independent entities in the system.

- Any change in the definition or structure of one element may cause changes to the other.
- This is a unidirectional kind of relationship between two objects.
- Dependency relationships are of various types specified with keywords like Abstraction, Binding, Realization, Substitution and Usage are the types of dependency relationships used in UML.

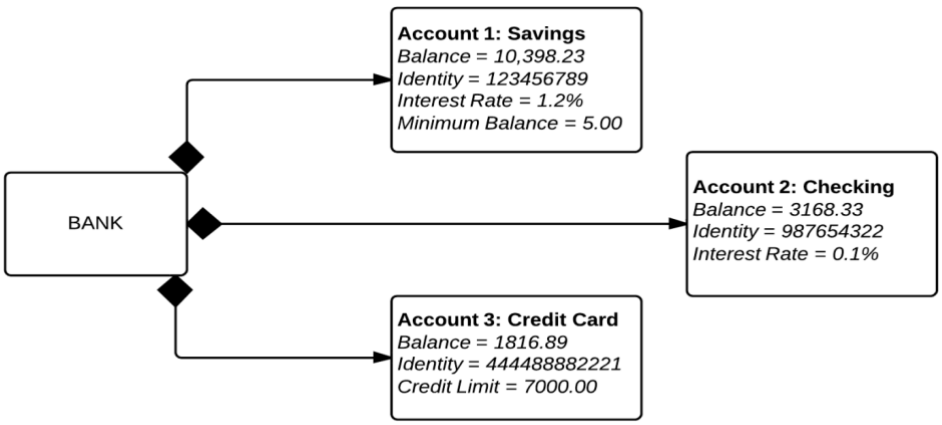
5. Association: Association is a reference relationship between two objects (or classes). An association line connects two object boxes, representing a relationship between instances of two classes. We use association when one object references members of the other object. Association can be uni-directional or bi-directional. We use an arrow to represent association.

6. Aggregation: Aggregation represents a "has a" relationship. We use a hollow diamond on the containing object with a line which joins it to the contained object.

- Aggregation is a specific form of association.
- It is a kind of parent-child relationship however it isn't inheritance.
- Aggregation occurs when the lifecycle of the contained objects does not strongly depend on the lifecycle of container objects.

7. Composition: Composition is a type of association where the child cannot exist independent of the other. We use a filled diamond on the containing object with a line which joins it to the contained object. Composition is also a special type of association. It is also a kind of parent child relationship but it is not inheritance. So whenever independent existence of the child is not possible we use a composition relationship.

Example:



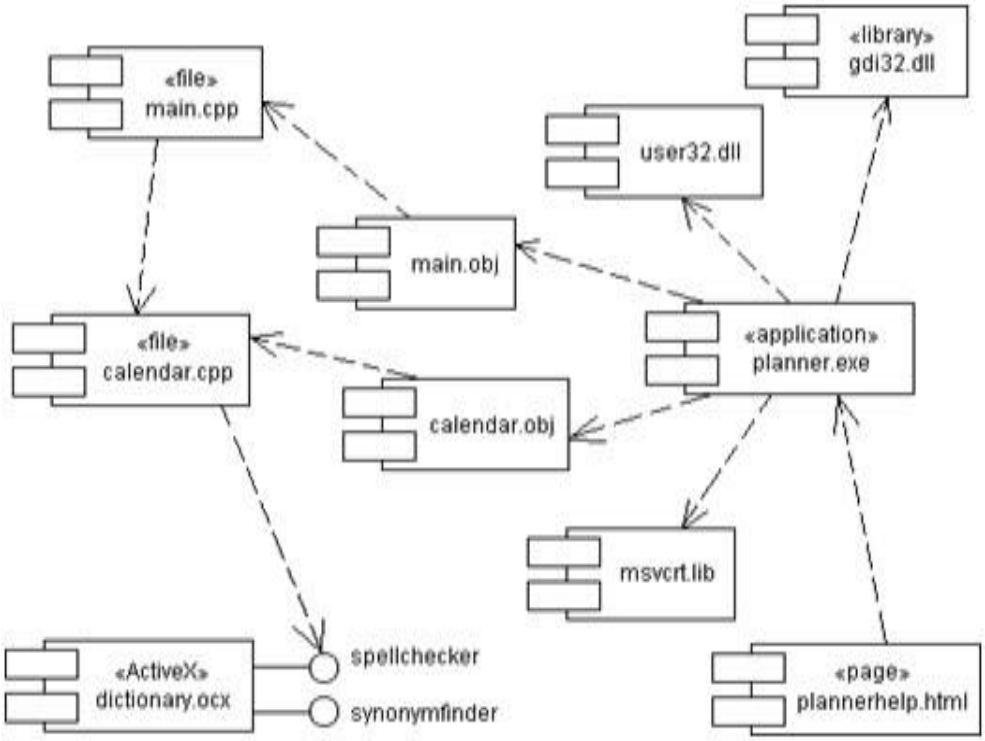
8. COMPONENT DIAGRAM

Component diagrams provide a physical view of the current model, illustrating how various parts of a system are structured and interact at a high level. A component diagram shows the organization and dependencies among software components, including source code components, binary code components, and executable components. These diagrams help developers and stakeholders understand how the software is divided into modular parts and how these parts communicate with one another. Additionally, component diagrams show the externally visible behaviour of each component by displaying the interfaces through which components interact with other parts of the system. This visual representation aids in system design, development, and maintenance by promoting component reuse, simplifying integration, and enhancing overall system understanding.

Component diagrams contain:

1. Component package
2. Components
3. Interfaces
4. Dependency relationship

Example:

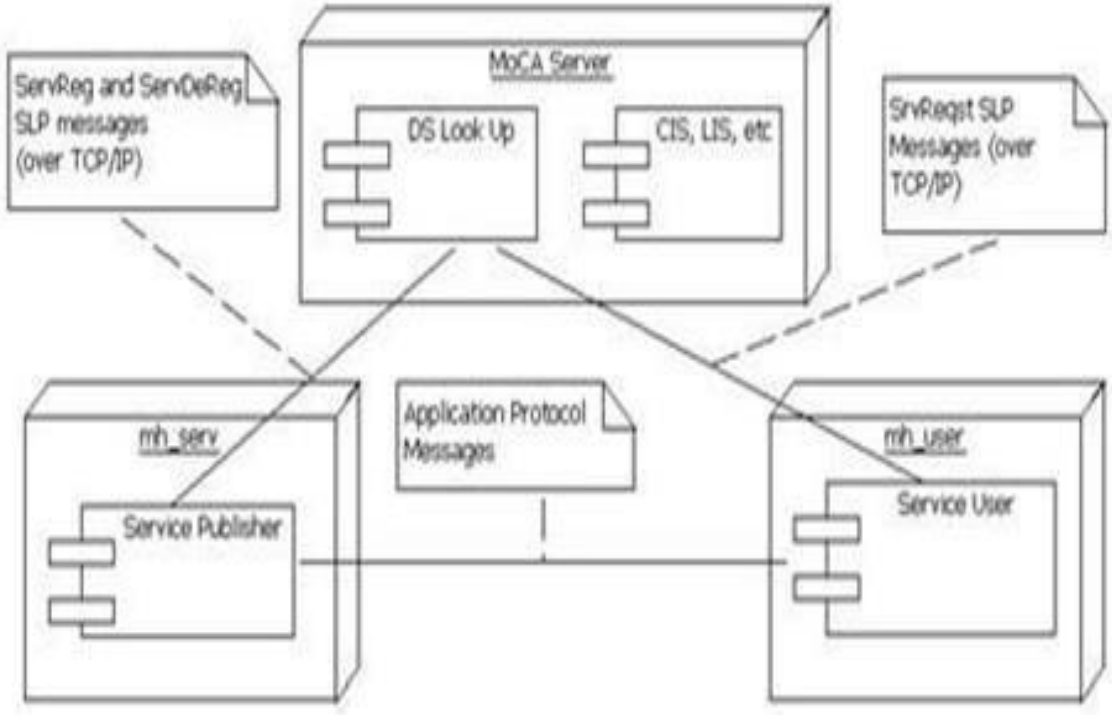


9. DEPLOYMENT DIAGRAM

A deployment diagram shows the physical layout of a system, including processors, devices, and the connections between them. It provides a static view of the runtime architecture, helping to visualize where and how software components are deployed across the hardware infrastructure. Each model typically contains a single deployment diagram that depicts the communication paths and relationships between nodes, ensuring a clear understanding of the system's distribution.

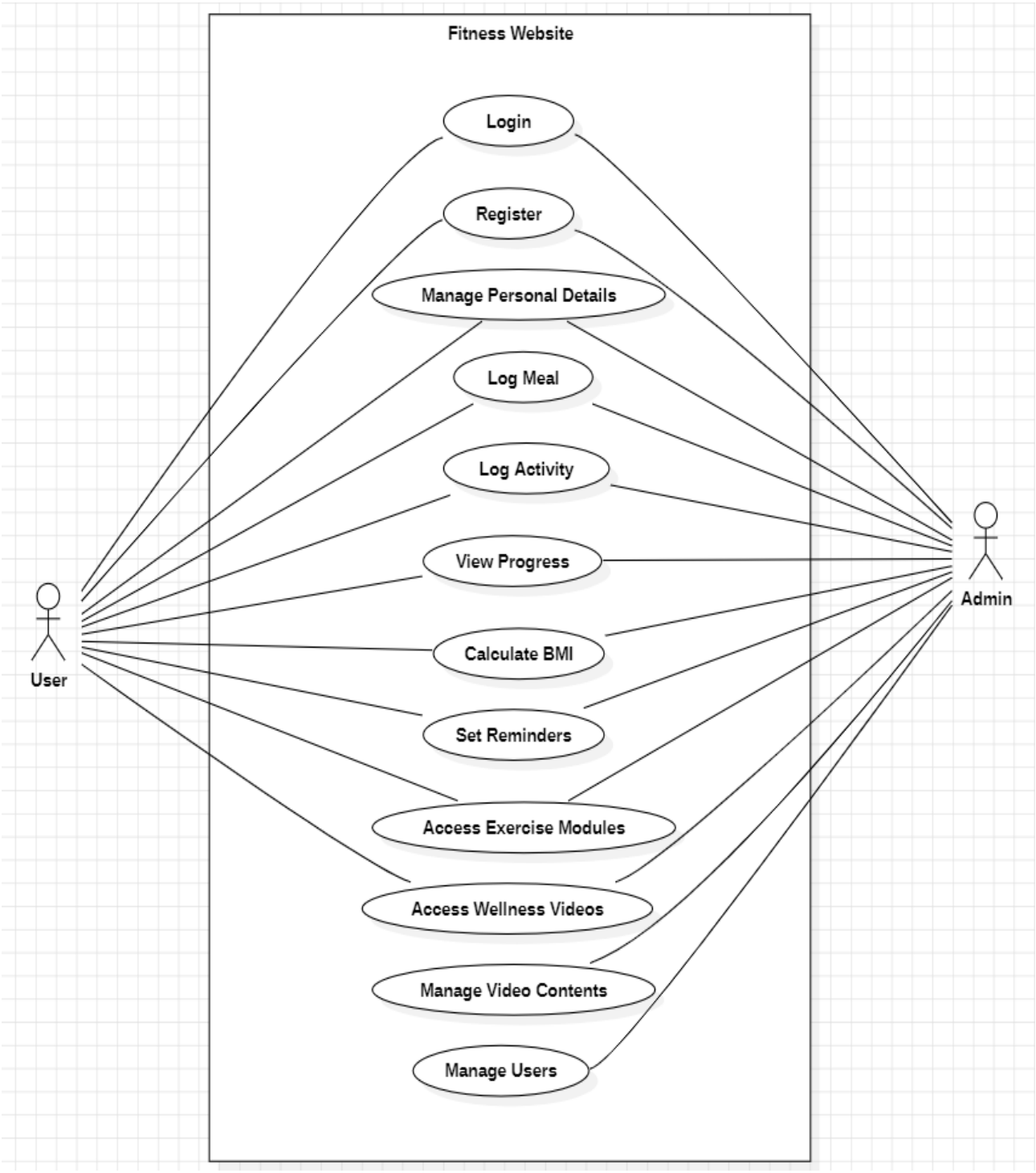
Node: A node represents a physical hardware component in the system that can execute software programs. This can include devices such as servers, workstations, mobile devices, or network hardware. Each node must have a unique name to distinguish it within the diagram, although there are no strict constraints on the naming convention, as nodes represent physical hardware rather than software elements. Nodes can also be connected to each other to indicate communication links, and they often host components or artifacts that are deployed and executed on them.

Example:

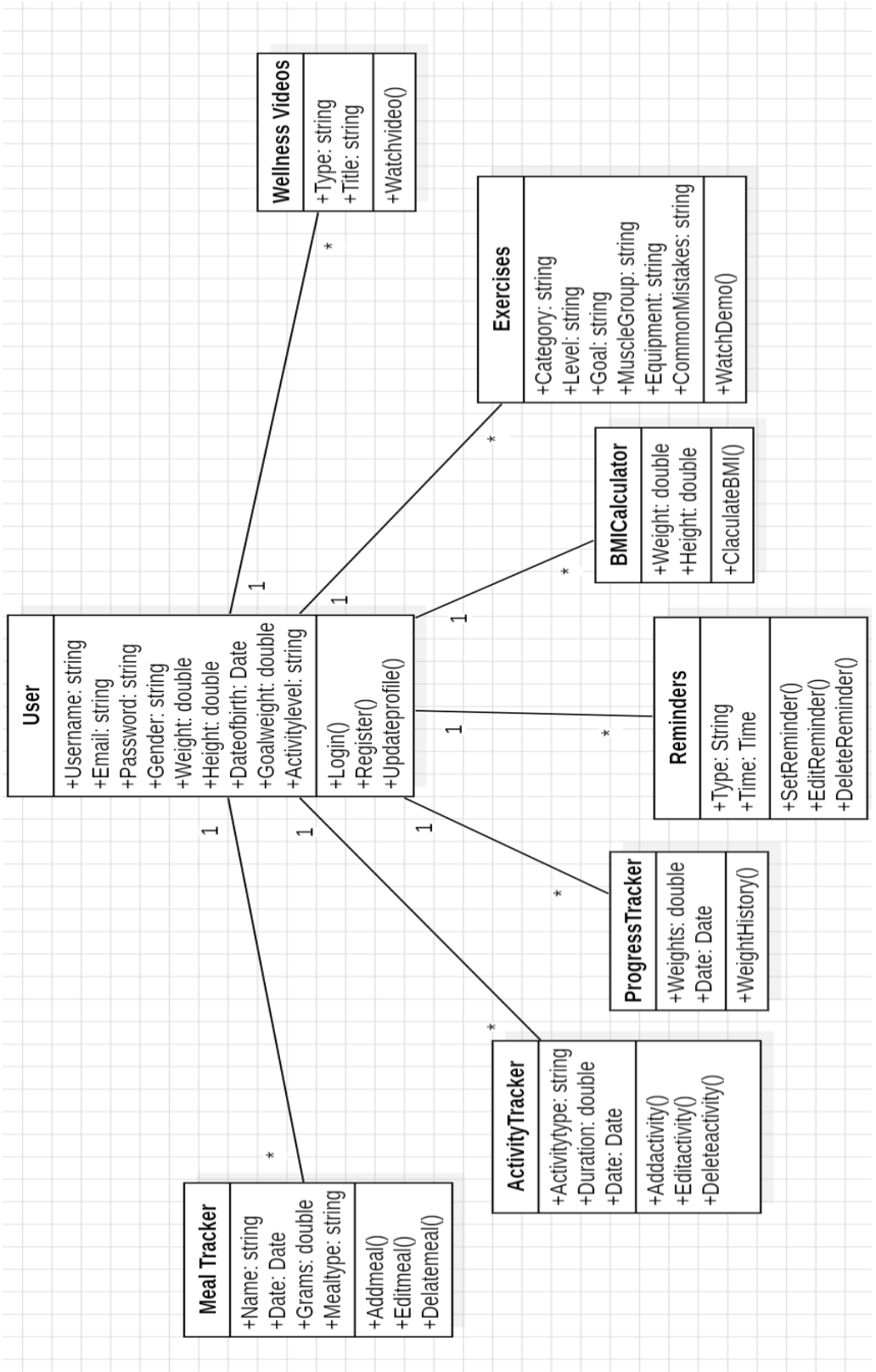


Architectural Patterns (UML Diagrams) for Fitness Website:

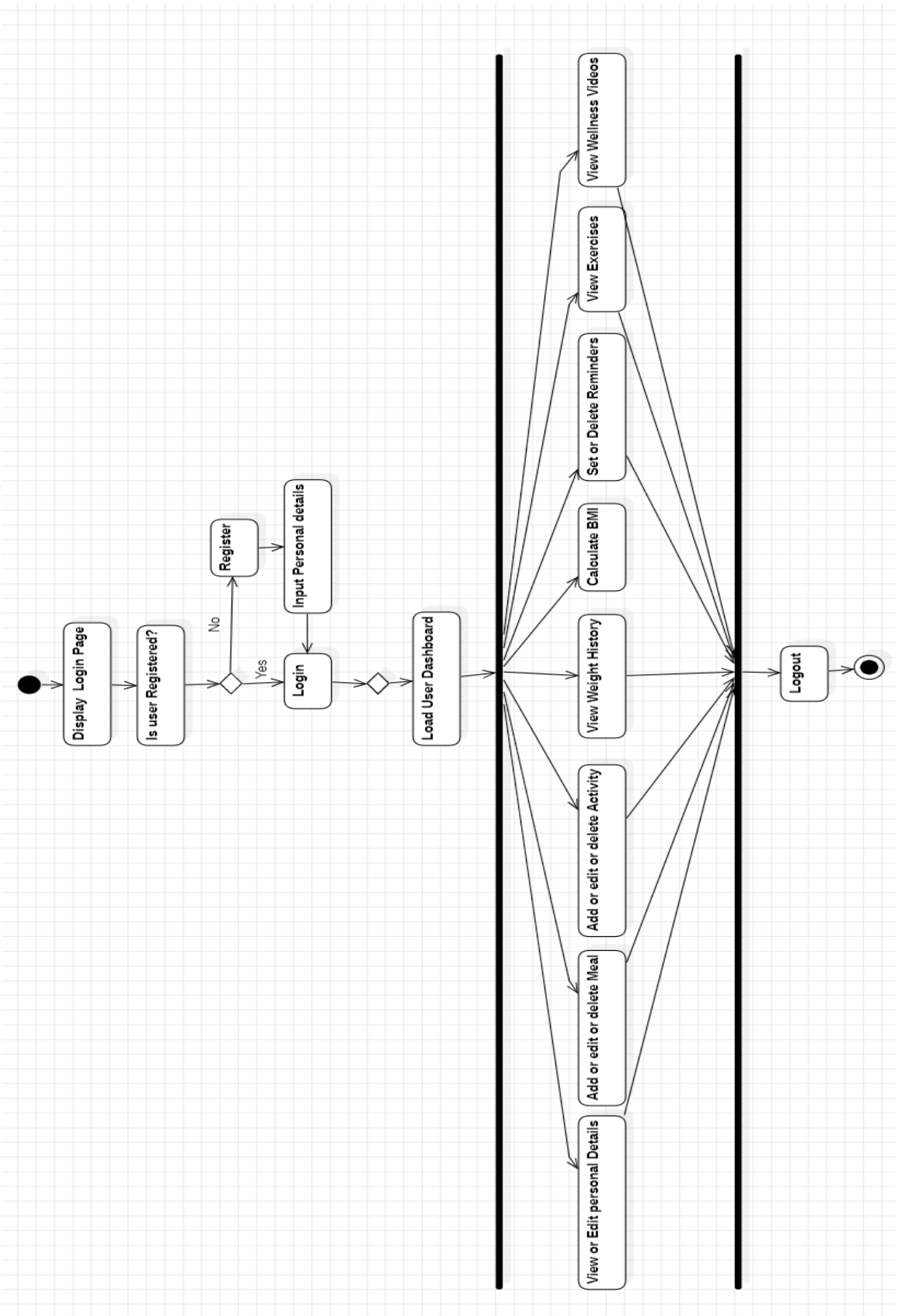
1.Usecase Diagram:



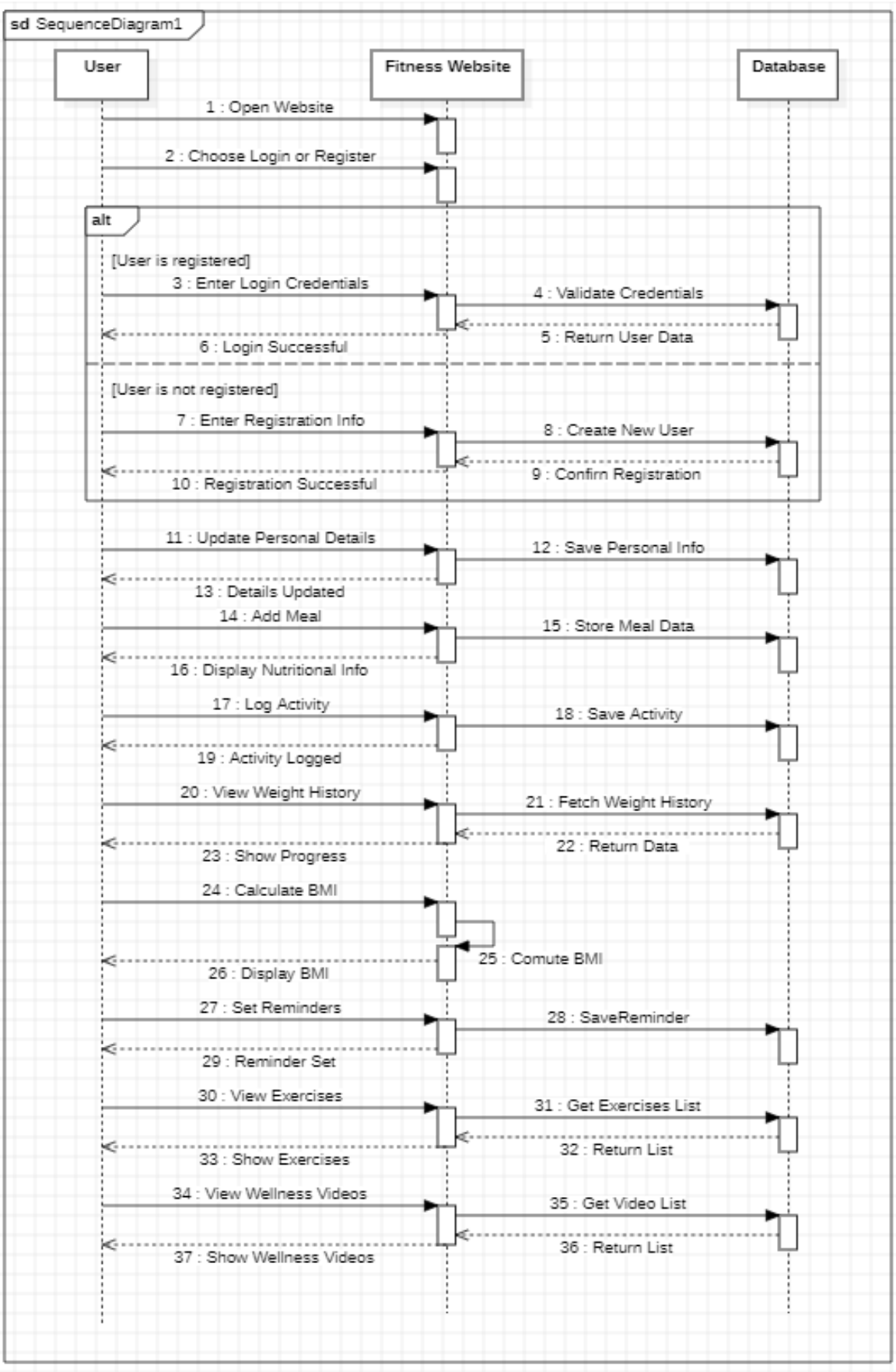
2. Class Diagram:



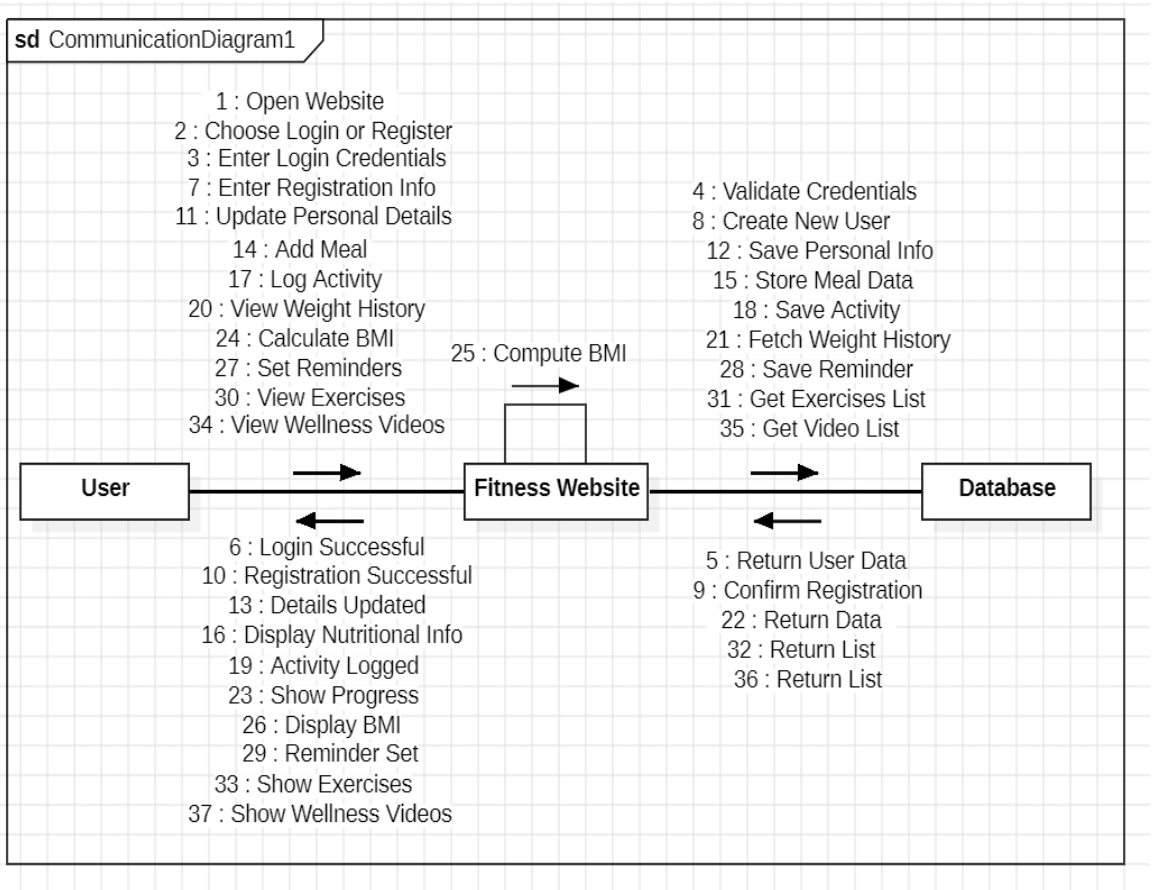
3. Activity Diagram:



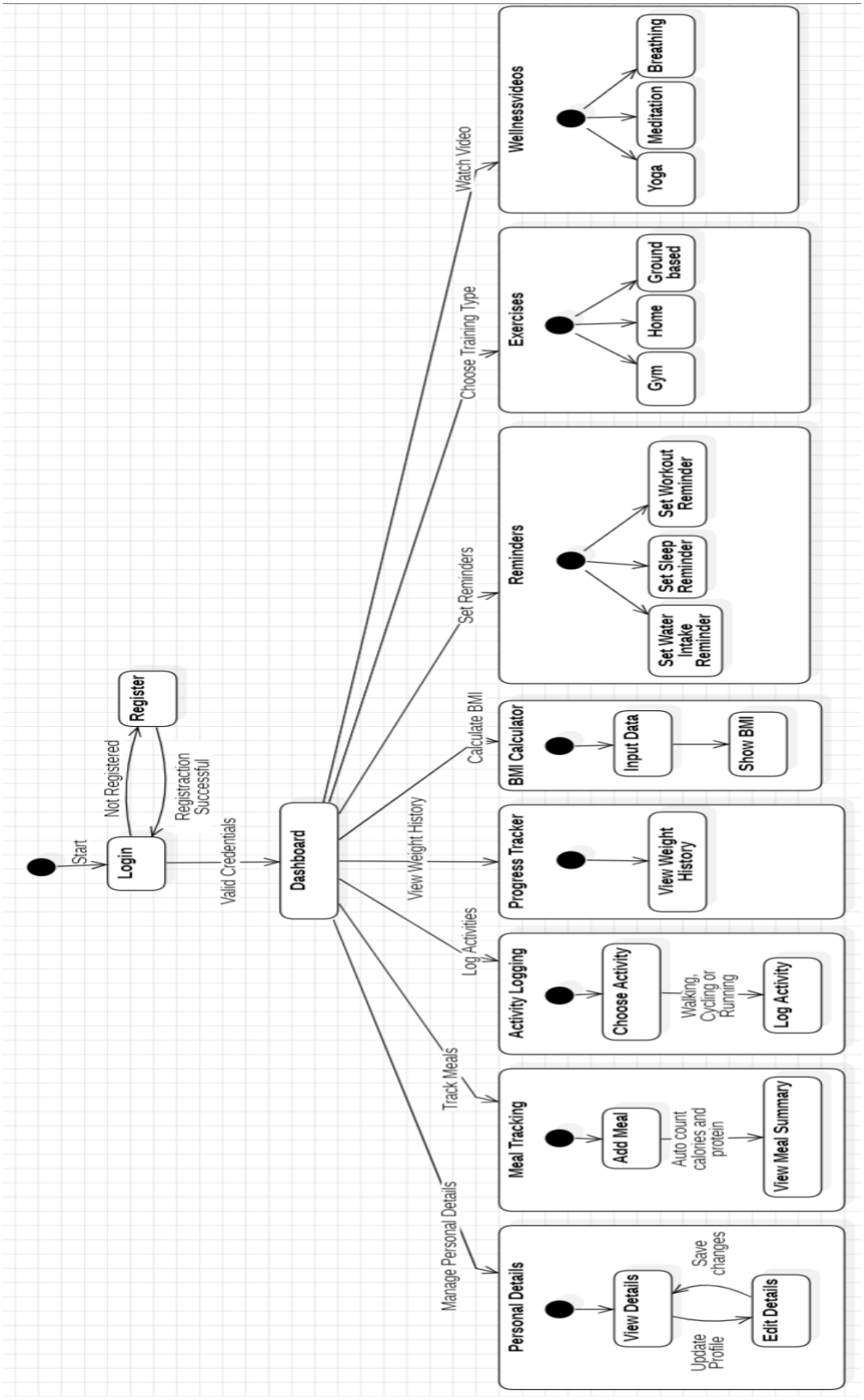
4. Sequence Diagram:



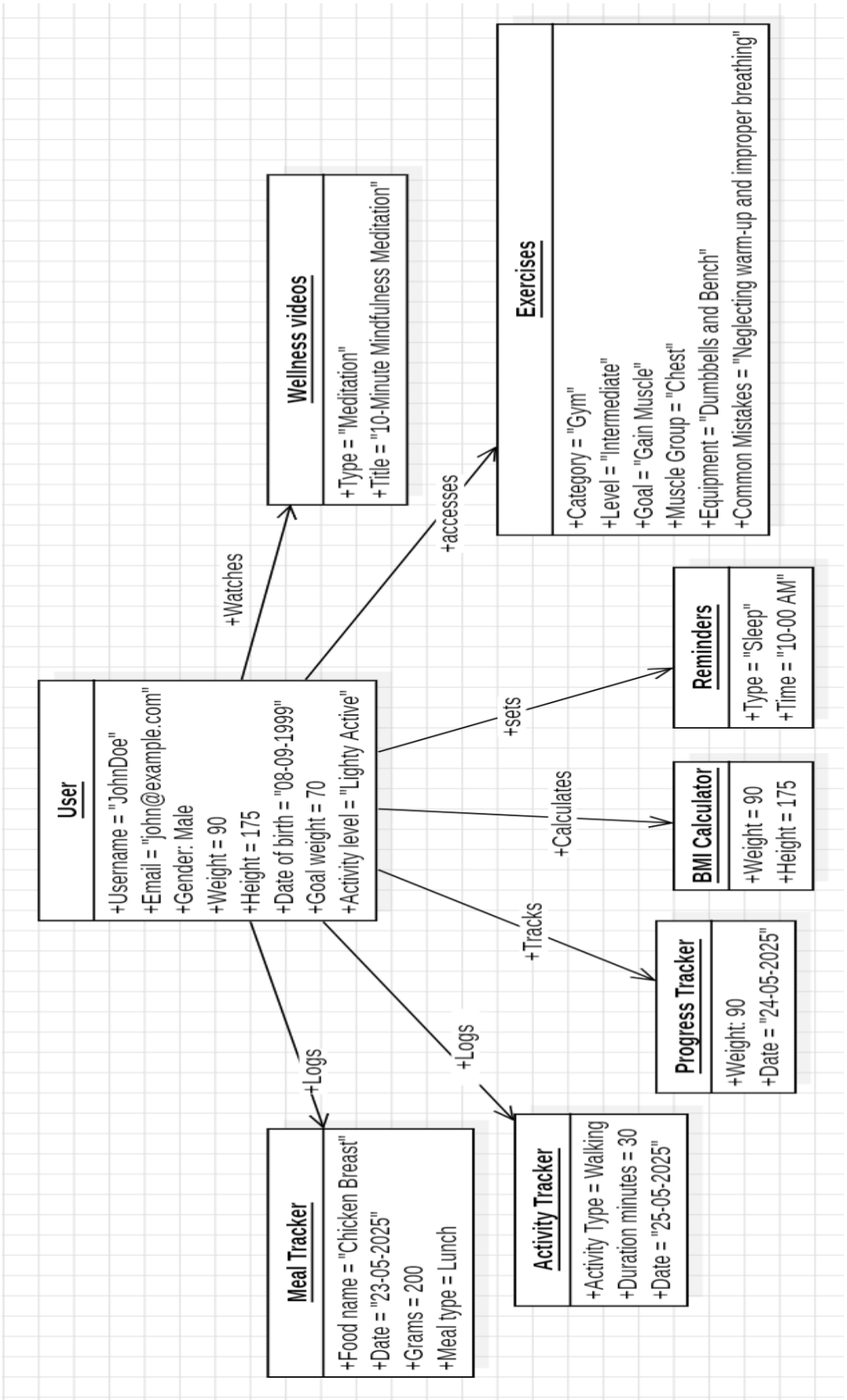
5.Collaboration Diagram:



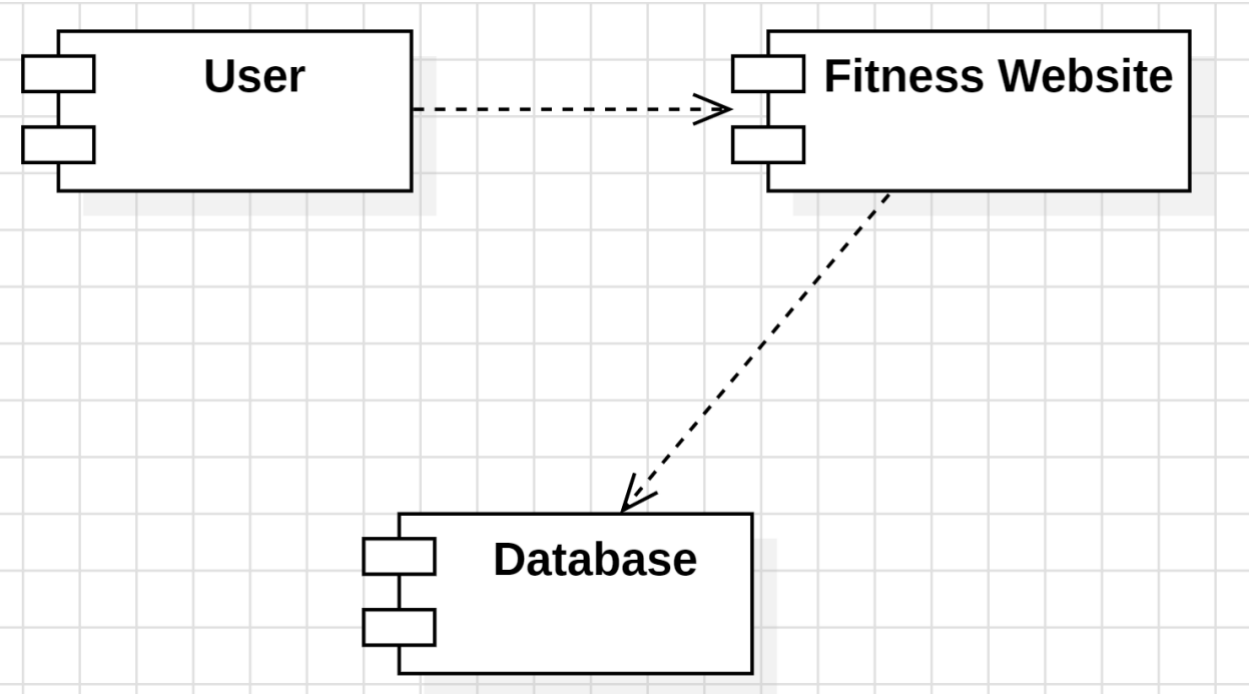
6. State Machine Diagram:



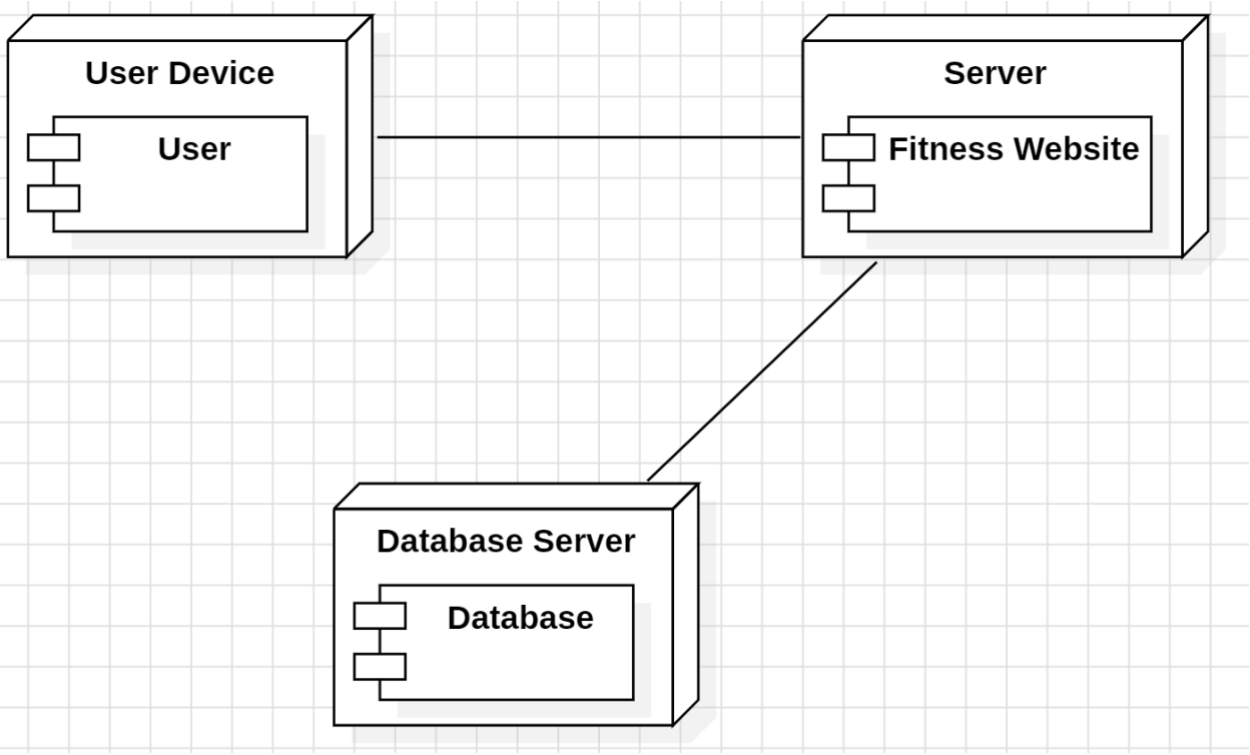
7. Object Diagram:



8. Component Diagram:



9. Deployment Diagram:



Coding

Introduction

The project consists of a frontend built with HTML, CSS, and JavaScript, and a backend developed using Node.js with Express.js that handles API requests and communicates with a MySQL database. The backend is run using the command node server.js.

The code implements all core features together, including user authentication, meal tracking with calorie and protein calculation, activity tracking, graphs, and reminders for water intake, sleep, and workouts.

Technology Stack

- **Frontend:** HTML, CSS, JavaScript (served via VS Code Live Server)
- **Backend:** Node.js with Express.js
- **Database:** MySQL
- **Libraries:** body-parser, Chart.js (for graphs)

Setup and Running Instructions

1. **Download Required Software**
    a. **Node.js** and **npm**
    b. **MySQL**
    c. **Visual Studio Code**
2. **Backend**
    a. Navigate to backend/ folder in terminal
    b. Run npm install to install dependencies
    c. Start backend server with node server.js
    d. Backend listens on port 3000
3. **Frontend**
    a. Open frontend/ folder in VS Code
    b. Right-click index.html and select **Go Live.**
    c. The site will open on a local port (usually 5500)

Features Implemented

- User Registration and Login (with validation and MySQL storage)
- Meal Tracker with dropdown food selection, gram input, and automatic calorie/protein calculation
- Activity Tracker that logs exercises and displays daily calories burned charts
- Reminders for water intake, workouts, and sleep with customizable intervals

Project Files:

index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Fitness Tracker - Login / Register</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <div class="auth-container card">
    <h1 class="text-center">Pulsecare Vita</h1>
    <h2 id="authTitle" class="text-center">Login</h2>
    <form id="loginForm">
      <div class="input-group">
        <label for="loginEmail">Email:</label>
        <input type="email" id="loginEmail" required>
      </div>
      <div class="input-group">
        <label for="loginPassword">Password:</label>
        <input type="password" id="loginPassword" required>
      </div>
      <button type="submit" class="primary-button full-width">Login</button>
    </form>
    <form id="registerForm" class="hidden">
      <div class="input-group">
        <label for="registerUsername">Username:</label>
        <input type="text" id="registerUsername" required>
      </div>
      <div class="input-group">
        <label for="registerEmail">Email:</label>
        <input type="email" id="registerEmail" required>
      </div>
      <div class="input-group">
        <label for="registerPassword">Password:</label>
        <input type="password" id="registerPassword" required>
      </div>
      <div class="input-group">
        <label for="registerGender">Gender:</label>
        <select id="registerGender" required>
          <option value="">Select Gender</option>
          <option value="Male">Male</option>
          <option value="Female">Female</option>
```

```
                <option value="Other">Other</option>
            </select>
        </div>
        <div class="input-group">
            <label for="registerWeight">Current Weight (kg):</label>
            <input type="number" id="registerWeight" step="0.1" required>
        </div>
        <div class="input-group">
            <label for="registerHeight">Height (cm):</label>
            <input type="number" id="registerHeight" step="0.1" required>
        </div>
        <div class="input-group">
            <label for="registerDob">Date of Birth:</label>
            <input type="date" id="registerDob" required>
        </div>
        <button type="submit" class="primary-button full-width">Register</button>
    </form>
    <p id="authMessage" class="message hidden"></p>
    <div class="auth-toggle">
        <p>
            <span id="toggleText">Don't have an account?</span>
            <a href="#" id="toggleAuthMode">Register</a>
        </p>
    </div>
</div>
<script src="js/api.js" type="module"></script>
<script src="js/auth.js" type="module"></script>
</body>
</html>
```

dashboard.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home - Fitness Tracker</title> <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/dashboard.css"> <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">
</head>
<body>
    <div class="sidebar" id="sidebar">
        <div class="logo">
            <h2>Pulsecase Vita</h2>
```

```
    </div>
    <nav class="sidebar-nav">
      <ul>
        <li><a href="dashboard.html" class="active"><i class="fas fa-home"></i>
Home</a></li>
        <li><a href="profile.html"><i class="fas fa-user"></i> Profile</a></li>
        <li><a href="meal_tracker.html"><i class="fas fa-utensils"></i> Meal
Tracker</a></li>
        <li><a href="activity_tracker.html"><i class="fas fa-running"></i> Activity
Tracker</a></li>
        <li><a href="progress_tracker.html"><i class="fas fa-chart-line"></i> Progress
Tracker</a></li>
        <li><a href="bmi_calculator.html"><i class="fas fa-calculator"></i> BMI
Calculator</a></li>
        <li><a href="reminders.html"><i class="fas fa-bell"></i> Reminders</a></li>
        <li><a href="exercises.html"><i class="fas fa-dumbbell"></i> Exercises</a></li>
        <li><a href="wellness_videos.html"><i class="fas fa-heartbeat"></i> Wellness
Videos</a></li>
        <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i>
Logout</a></li> </ul> </nav>
  </div>
  <div class="main-content">
    <header>
      <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-
bars"></i></button>
      <h1>Home</h1> <span class="welcome-message" id="welcomeMessage">Welcome,
!</span> </header>
    <main>
      <section class="card dashboard-summary">
        <h2>Today's Summary</h2>
        <div class="summary-cards">
          <div class="summary-card">
            <h3>Calories Consumed</h3>
            <p class="summary-value" id="caloriesConsumed">0.0</p>
            <p class="summary-unit">kcal</p>
          </div>
          <div class="summary-card">
            <h3>Calories Burnt</h3>
            <p class="summary-value" id="caloriesBurnt">0.0</p>
            <p class="summary-unit">kcal</p>
          </div>
           <div class="summary-card net-calories">
            <h3>Net Calories</h3>
            <p class="summary-value" id="netCalories">0.0</p>
```

```
                <p class="summary-unit">kcal</p>
              </div>
               <div class="summary-card estimated-calories">
                <h3>Estimated Daily Needs</h3>
                <p class="summary-value" id="estimatedDailyCalories">--</p>
                <p class="summary-unit">kcal</p>
              </div>
            </div>
          </section>
          <section class="card quick-links-section">
            <h2>Quick Links</h2>
            <div class="quick-links">
              <a href="meal_tracker.html" class="quick-link-button primary-button"><i
class="fas fa-utensils"></i> Log Meal</a>
              <a href="activity_tracker.html" class="quick-link-button primary-button"><i
class="fas fa-running"></i> Log Activity</a>
              <a href="progress_tracker.html" class="quick-link-button secondary-button"><i
class="fas fa-chart-line"></i> View Progress</a>
              <a href="profile.html" class="quick-link-button secondary-button"><i class="fas
fa-user"></i> Edit Profile</a>
            </div>
          </section>
           <section class="card your-stats-section">
            <h2>Your Key Stats</h2>
            <div class="stats-grid">
              <div class="stat-item">
                <strong>Current Weight:</strong> <span id="statWeight">--</span> kg
              </div>
               <div class="stat-item">
                <strong>Height:</strong> <span id="statHeight">--</span> cm
              </div>
               <div class="stat-item">
                <strong>Age:</strong> <span id="statAge">--</span> years
              </div>
               <div class="stat-item">
                <strong>Weight Goal:</strong> <span id="statWeightGoal">--</span> kg
              </div>
               <div class="stat-item">
                <strong>Activity Level:</strong> <span id="statActivityLevel">--</span>
              </div> </div> </section> </main>
  </div>
  <script src="js/dashboard_common.js" type="module"></script>
  <script src="js/dashboard.js" type="module"></script> </body> </html>
```

proflie.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>User Profile - Fitness Tracker</title>
   <link rel="stylesheet" href="css/style.css">
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">
</head><body>
   <div class="sidebar" id="sidebar">
      <div class="logo">
         <h2>Pulsecase Vita</h2>
      </div>
      <nav class="sidebar-nav">
         <ul>
            <li><a href="dashboard.html" ><i class="fas fa-home"></i> Home</a></li>
            <li><a href="profile.html"class="active"><i class="fas fa-user"></i>
Profile</a></li>
            <li><a href="meal_tracker.html"><i class="fas fa-utensils"></i> Meal
Tracker</a></li>
            <li><a href="activity_tracker.html"><i class="fas fa-running"></i> Activity
Tracker</a></li>
            <li><a href="progress_tracker.html"><i class="fas fa-chart-line"></i> Progress
Tracker</a></li>
            <li><a href="bmi_calculator.html"><i class="fas fa-calculator"></i> BMI
Calculator</a></li>
            <li><a href="reminders.html"><i class="fas fa-bell"></i> Reminders</a></li>
            <li><a href="exercises.html"><i class="fas fa-dumbbell"></i> Exercises</a></li>
            <li><a href="wellness_videos.html"><i class="fas fa-heartbeat"></i> Wellness
Videos</a></li>
            <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i>
Logout</a></li>
         </ul>
      </nav>
   </div>
   <div class="main-content">
      <header>
         <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-
bars"></i></button>
         <h1>Your Profile</h1>
      </header> <main>
         <section class="card">
```

```
        <h2>Profile Information</h2>
        <form id="profileForm">
          <div class="input-group">
            <label for="username">Username:</label>
            <input type="text" id="username" readonly>
          </div>
          <div class="input-group">
            <label for="email">Email:</label>
            <input type="email" id="email" readonly> </div>
          <div class="input-group">
            <label for="gender">Gender:</label>
            <select id="gender">
              <option value="Male">Male</option>
              <option value="Female">Female</option>
              <option value="Other">Other</option>    </select>
  </div>
          <div class="input-group">
            <label for="currentWeight">Current Weight (kg):</label>
            <input type="number" id="currentWeight" step="0.1" min="1">
          </div>
          <div class="input-group">
            <label for="height">Height (cm):</label>
            <input type="number" id="height" step="0.1" min="1">
          </div>
          <div class="input-group">
            <label for="dob">Date of Birth:</label>
            <input type="date" id="dob">
          </div>
          <div class="input-group">
            <label for="weightGoal">Weight Goal (kg):</label>
            <input type="number" id="weightGoal" step="0.1" min="1">
          </div>
          <div class="input-group">
            <label for="activityLevel">Activity Level:</label>
            <select id="activityLevel">
              <option value="Sedentary">Sedentary (little to no exercise)</option>
              <option value="Lightly Active">Lightly Active (light exercise/sports 1-3
days/week)</option>
              <option value="Moderately Active">Moderately Active (moderate
exercise/sports 3-5 days/week)</option>
              <option value="Very Active">Very Active (hard exercise/sports 6-7 days a
week)</option>
              <option value="Extra Active">Extra Active (very hard exercise/physical
job)</option>
```

```
                </select>
              </div>
              <button type="submit" class="primary-button">Update Profile</button>
              <p id="profileMessage" class="message hidden"></p>
            </form>
          </section>
        </main>
    </div>
    <script src="js/dashboard_common.js" type="module"></script>
    <script src="js/profile.js" type="module"></script>
</body>
</html>
```

meal_tracker.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Meal Tracker - Fitness Tracker</title>
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/meal_tracker.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">
</head>
<body>
    <div class="sidebar" id="sidebar">
        <div class="logo">
            <h2>Pulsecase Vita</h2>
        </div>
        <nav class="sidebar-nav">
            <ul>
                <li><a href="dashboard.html" ><i class="fas fa-home"></i> Home</a></li>
                <li><a href="profile.html"><i class="fas fa-user"></i> Profile</a></li>
                <li><a href="meal_tracker.html"class="active"><i class="fas fa-utensils"></i>
Meal Tracker</a></li>
                <li><a href="activity_tracker.html"><i class="fas fa-running"></i> Activity
Tracker</a></li>
                <li><a href="progress_tracker.html"><i class="fas fa-chart-line"></i> Progress
Tracker</a></li>
                <li><a href="bmi_calculator.html"><i class="fas fa-calculator"></i> BMI
Calculator</a></li>
                <li><a href="reminders.html"><i class="fas fa-bell"></i> Reminders</a></li>
                <li><a href="exercises.html"><i class="fas fa-dumbbell"></i> Exercises</a></li>
```

```
            <li><a href="wellness_videos.html"><i class="fas fa-heartbeat"></i> Wellness
Videos</a></li>
            <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i>
Logout</a></li>
          </ul>
        </nav>
      </div>
      <div class="main-content">
        <header>
          <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-
bars"></i></button>
          <h1>Meal Tracker</h1>
        </header>
        <main>
          <section class="date-navigation card">
            <button id="prevDayBtn" class="secondary-button">&laquo; Prev Day</button>
            <input type="date" id="mealDateInput">
            <button id="nextDayBtn" class="secondary-button">Next Day &raquo;</button>
          </section>
          <section class="add-meal-section card">
            <h2>Log a New Meal</h2>
            <div class="input-group">
              <label for="foodSearchInput">Search Food:</label>
              <input type="text" id="foodSearchInput" placeholder="e.g., Chicken Breast">
              <span id="selectedFoodName">None selected</span>
              <input type="hidden" id="selectedFoodId">
            </div>
            <div id="foodSearchResults" class="food-search-results-container">
              </div>
            <p id="foodSearchMessage" class="message hidden"></p>
            <div class="input-group">
              <label for="gramsInput">Grams:</label>
              <input type="number" id="gramsInput" placeholder="e.g., 150" min="1"
step="0.1">

            </div>
            <div class="input-group">
              <label for="mealTypeSelect">Meal Type:</label>
              <select id="mealTypeSelect">
                <option value="Breakfast">Breakfast</option>
                <option value="Lunch">Lunch</option>
                <option value="Dinner">Dinner</option>
                <option value="Snack">Snack</option>
              </select>
```

```
            </div>
            <button id="addMealBtn" class="primary-button">Add Meal</button>
             <button id="cancelMealEditBtn" class="secondary-button" style="display:
none;">Cancel Edit</button>
            <p id="mealAddMessage" class="message hidden"></p>

        </section>
        <section class="daily-summary-section card">
            <h2>Daily Meal Summary</h2>
            <div class="summary-totals">
                <p>Total Calories: <span id="totalCalories">0</span> kcal</p>
                <p>Total Protein: <span id="totalProtein">0</span> g</p>
                <p>Total Carbs: <span id="totalCarbs">0</span> g</p>
                <p>Total Fat: <span id="totalFat">0</span> g</p>
            </div>
            <div id="mealLog" class="meal-log">
                </div>
        </section>
    </main>

  </div>
  <script src="js/dashboard_common.js" type="module"></script>
  <script src="js/meal_tracker.js" type="module"></script>
</body>
</html>
```

activity_tracker.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Activity Tracker - Fitness Tracker</title>
  <link rel="stylesheet" href="css/style.css">
  <link rel="stylesheet" href="css/activity_tracker.css">
 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0
beta3/css/all.min.css">
</head>

<body>
  <div class="sidebar" id="sidebar">
    <div class="logo">
      <h2>Pulsecase Vita</h2>
    </div>
```

```
    <nav class="sidebar-nav">
        <ul>
          <li><a href="dashboard.html" ><i class="fas fa-home"></i> Home</a></li>
           <li><a href="profile.html"><i class="fas fa-user"></i> Profile</a></li>
           <li><a href="meal_tracker.html"><i class="fas fa-utensils"></i> Meal Tracker</a></li>
           <li><a href="activity_tracker.html"class="active"><i class="fas fa-running"></i> Activity Tracker</a></li>
           <li><a href="progress_tracker.html"><i class="fas fa-chart-line"></i> Progress Tracker</a></li>
           <li><a href="bmi_calculator.html"><i class="fas fa-calculator"></i> BMI Calculator</a></li>
           <li><a href="reminders.html"><i class="fas fa-bell"></i> Reminders</a></li>
           <li><a href="exercises.html"><i class="fas fa-dumbbell"></i> Exercises</a></li>
           <li><a href="wellness_videos.html"><i class="fas fa-heartbeat"></i> Wellness Videos</a></li>
           <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i> Logout</a></li>
        </ul>
      </nav>
    </div>

    <div class="main-content">
      <header>
        <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-bars"></i></button>

        <h1>Activity Tracker</h1>
      </header>
      <main>
        <section class="date-navigation card">
          <button id="prevActivityDayBtn" class="secondary-button">&laquo; Prev Day</button>
          <input type="date" id="activityDateInput">
          <button id="nextActivityDayBtn" class="secondary-button">Next Day &raquo;</button>
        </section>
        <section class="add-activity-section card">
          <h2>Log a New Activity</h2>
          <div class="input-group">
            <label for="activityTypeSelect">Activity Type:</label>
            <select id="activityTypeSelect">
              </select>
          </div>
```

```
            <div class="input-group">
                <label for="durationMinutesInput">Duration (minutes):</label>
                <input type="number" id="durationMinutesInput" placeholder="e.g., 30"
min="1" step="1">
            </div>
            <div class="input-group">
                <p>Estimated Calories Burnt: <span id="estimatedCalories">0.0</span>
kcal</p>
            </div>
            <button id="addActivityBtn" class="primary-button">Add Activity</button>
             <button id="cancelActivityEditBtn" class="secondary-button" style="display:
none;">Cancel Edit</button>
            <p id="activityAddMessage" class="message hidden"></p>
        </section>
        <section class="daily-summary-section card">

            <h2>Daily Activity Summary</h2>
            <div class="summary-totals">
                <p>Total Calories Burnt: <span id="totalDailyCaloriesBurnt">0</span>
kcal</p>
            </div>
            <div id="activityLog" class="activity-log">
                </div>
        </section>
    </main>
  </div>
  <script src="js/dashboard_common.js" type="module"></script>
  <script src="js/activity_tracker.js" type="module"></script>
</body>
</html>
```

progress_tracker.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Progress Tracker - Fitness Tracker</title>
  <link rel="stylesheet" href="css/style.css">
  <link rel="stylesheet" href="css/progress_tracker.css"> <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">
  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
```

```
  <div class="sidebar" id="sidebar">
    <div class="logo">
      <h2>Pulsecase Vita</h2>
    </div>
    <nav class="sidebar-nav">
      <ul>
        <li><a href="dashboard.html" ><i class="fas fa-home"></i> Home</a></li>
        <li><a href="profile.html"><i class="fas fa-user"></i> Profile</a></li>
        <li><a href="meal_tracker.html"><i class="fas fa-utensils"></i> Meal
Tracker</a></li>
        <li><a href="activity_tracker.html"><i class="fas fa-running"></i> Activity
Tracker</a></li>
        <li><a href="progress_tracker.html"class="active"><i class="fas fa-chart-
line"></i> Progress Tracker</a></li>
        <li><a href="bmi_calculator.html"><i class="fas fa-calculator"></i> BMI
Calculator</a></li>
        <li><a href="reminders.html"><i class="fas fa-bell"></i> Reminders</a></li>
        <li><a href="exercises.html"><i class="fas fa-dumbbell"></i> Exercises</a></li>
        <li><a href="wellness_videos.html"><i class="fas fa-heartbeat"></i> Wellness
Videos</a></li>
        <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i>
Logout</a></li>
      </ul>
    </nav>
  </div>
  <div class="main-content">
    <header>
      <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-
bars"></i></button>
      <h1>Progress Tracker</h1>
    </header>
    <main>
      <section class="card">
        <h2>Weight History</h2>
        <div class="chart-container">
          <canvas id="weightHistoryChart"></canvas>
        </div>
         <p id="progressMessage" class="message hidden"></p>
      </section>
    </main>
  </div>
  <script src="js/dashboard_common.js" type="module"></script>
  <script src="js/progress_tracker.js" type="module"></script> </body></html>
```

bmi_calculator.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>BMI Calculator - Fitness Tracker</title>
  <link rel="stylesheet" href="css/style.css">
  <link rel="stylesheet" href="css/bmi_calculator.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">
</head>
<body>
  <div class="sidebar" id="sidebar">
    <div class="logo">
      <h2>Pulsecase Vita</h2>
    </div>
    <nav class="sidebar-nav">
      <ul>
        <li><a href="dashboard.html" ><i class="fas fa-home"></i> Home</a></li>
        <li><a href="profile.html"><i class="fas fa-user"></i> Profile</a></li>
        <li><a href="meal_tracker.html"><i class="fas fa-utensils"></i> Meal Tracker</a></li>
        <li><a href="activity_tracker.html"><i class="fas fa-running"></i> Activity Tracker</a></li>
        <li><a href="progress_tracker.html"><i class="fas fa-chart-line"></i> Progress Tracker</a></li>
        <li><a href="bmi_calculator.html"class="active"><i class="fas fa-calculator"></i> BMI Calculator</a></li>
        <li><a href="reminders.html"><i class="fas fa-bell"></i> Reminders</a></li>
        <li><a href="exercises.html"><i class="fas fa-dumbbell"></i> Exercises</a></li>
        <li><a href="wellness_videos.html"><i class="fas fa-heartbeat"></i> Wellness Videos</a></li>
        <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i> Logout</a></li>
      </ul>
    </nav>
  </div>
  <div class="main-content">
    <header>
      <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-bars"></i></button>
      <h1>BMI Calculator</h1>
    </header>
```

```
      <main>
        <section class="card">
          <h2>Calculate Your BMI</h2>
          <p class="info-message">Enter your current weight and height to calculate your Body Mass Index.</p>
          <div class="bmi-calculator-form">
            <div class="input-group">
              <label for="weightInput">Weight (kg):</label>
              <input type="number" id="weightInput" step="0.1" min="1">
            </div>
            <div class="input-group">
              <label for="heightInput">Height (cm):</label>
              <input type="number" id="heightInput" step="0.1" min="1">
            </div>
            <button id="calculateBmiBtn" class="primary-button">Calculate BMI</button>
             <p id="bmiMessage" class="message hidden"></p>
          </div>

          <div class="bmi-result-section hidden" id="bmiResultSection">
            <h3>Your BMI: <span id="bmiValue">--</span></h3>
            <p>Category: <span id="bmiCategory">--</span></p>
            <div class="bmi-categories">
              <h4>BMI Categories:</h4>
              <ul>
                <li>Underweight: < 18.5</li>
                <li>Normal weight: 18.5 – 24.9</li>
                <li>Overweight: 25 – 29.9</li>
                <li>Obesity: > 30</li>
              </ul>
               <p class="disclaimer">BMI is a screening tool and not diagnostic of the body fatness or health of an individual.</p>
            </div>
          </div>
        </section>
      </main>
    </div>
    <script src="js/dashboard_common.js" type="module"></script>
    <script src="js/bmi_calculator.js" type="module"></script>
</body
```

reminders.html

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Reminders - Fitness Tracker</title>
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/reminders.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">
</head>
<body>
    <div class="sidebar" id="sidebar">
        <div class="logo">
            <h2>Pulsecase Vita</h2>
        </div>
        <nav class="sidebar-nav">
            <ul>
                <li><a href="dashboard.html" ><i class="fas fa-home"></i> Home</a></li>
                <li><a href="profile.html"><i class="fas fa-user"></i> Profile</a></li>
                <li><a href="meal_tracker.html"><i class="fas fa-utensils"></i> Meal
Tracker</a></li>
                <li><a href="activity_tracker.html"><i class="fas fa-running"></i> Activity
Tracker</a></li>
                <li><a href="progress_tracker.html"><i class="fas fa-chart-line"></i> Progress
Tracker</a></li>
                <li><a href="bmi_calculator.html"><i class="fas fa-calculator"></i> BMI
Calculator</a></li>
                <li><a href="reminders.html"class="active"><i class="fas fa-bell"></i>
Reminders</a></li>
                <li><a href="exercises.html"><i class="fas fa-dumbbell"></i> Exercises</a></li>
                <li><a href="wellness_videos.html"><i class="fas fa-heartbeat"></i> Wellness
Videos</a></li>
                <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i>
Logout</a></li>
            </ul>
        </nav>
    </div>
    <div class="main-content">
        <header>
            <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-
bars"></i></button>
            <h1>Reminders</h1>
        </header>
        <main>
            <section class="card">
                <h2>Set Your Reminders</h2>
```

```
            <div class="reminder-setting" id="waterReminderSetting">
              <h3>Water Intake Reminder</h3>
              <div class="input-group">
                <label for="waterFrequency">Remind me every (minutes):</label>
                <input type="number" id="waterFrequency" min="0.1" step="0.1">
              </div>
              <button class="primary-button set-reminder-btn" data-type="water">Set Water
Reminder</button>
               <p id="waterMessage" class="message hidden"></p>
            </div>

            <div class="reminder-setting" id="sleepReminderSetting">
              <h3>Sleep Time Reminder</h3>
              <div class="input-group">
                <label for="sleepTime">Remind me at:</label>
                <input type="time" id="sleepTime">
              </div>
              <button class="primary-button set-reminder-btn" data-type="sleep">Set Sleep
Reminder</button>
               <p id="sleepMessage" class="message hidden"></p>
            </div>
            <div class="reminder-setting" id="workoutReminderSetting">
              <h3>Workout Reminder</h3>
              <div class="input-group">
                <label for="workoutTime">Remind me at:</label>
                <input type="time" id="workoutTime">
              </div>
              <button class="primary-button set-reminder-btn" data-type="workout">Set
Workout Reminder</button>
               <p id="workoutMessage" class="message hidden"></p>
            </div>
          </section>
           <section class="card">
            <h2>Active Reminders</h2>
            <div id="activeRemindersList">
              <p class="info-message">Loading active reminders...</p>
              </div>
          </section>
        </main>
    </div>
    <script src="js/dashboard_common.js" type="module"></script>
    <script src="js/reminders.js" type="module"></script>
</body>
</html>
```

exercises.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Exercises - Fitness Tracker</title>
   <link rel="stylesheet" href="css/style.css">
   <link rel="stylesheet" href="css/exercises.css">
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">
</head>
<body>
   <div class="sidebar" id="sidebar">
      <div class="logo">
         <h2>Pulsecase Vita</h2>
      </div>
      <nav class="sidebar-nav">
         <ul>
            <li><a href="dashboard.html" ><i class="fas fa-home"></i> Home</a></li>
            <li><a href="profile.html"><i class="fas fa-user"></i> Profile</a></li>
            <li><a href="meal_tracker.html"><i class="fas fa-utensils"></i> Meal
Tracker</a></li>
            <li><a href="activity_tracker.html"><i class="fas fa-running"></i> Activity
Tracker</a></li>
            <li><a href="progress_tracker.html"><i class="fas fa-chart-line"></i> Progress
Tracker</a></li>
            <li><a href="bmi_calculator.html"><i class="fas fa-calculator"></i> BMI
Calculator</a></li>
            <li><a href="reminders.html"><i class="fas fa-bell"></i> Reminders</a></li>
            <li><a href="exercises.html"class="active"><i class="fas fa-dumbbell"></i>
Exercises</a></li>
            <li><a href="wellness_videos.html"><i class="fas fa-heartbeat"></i> Wellness
Videos</a></li>
            <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i>
Logout</a></li>
         </ul>
      </nav>
   </div>
   <div class="main-content">
      <header>
         <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-
bars"></i></button>
```

```
        <h1>Exercises</h1>
      </header>
      <main>
        <section class="card">
          <h2>Browse Exercises</h2>
          <div id="exercisesList" class="exercises-list">
            <p class="info-message">Loading exercises...</p>
            </div>
           <p id="exercisesMessage" class="message hidden"></p>
        </section>
      </main>
    </div>
    <script src="js/dashboard_common.js" type="module"></script>
    <script src="js/exercises.js" type="module"></script>
</body>
</html>
```

wellness_videos.html

```
 <!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Wellness Videos - Fitness Tracker</title>
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/wellness_videos.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">

</head>
<body>
    <div class="sidebar" id="sidebar">
        <div class="logo">
            <h2>Pulsecase Vita</h2>
        </div>
        <nav class="sidebar-nav">

            <ul>
                <li><a href="dashboard.html" ><i class="fas fa-home"></i> Home</a></li>
                <li><a href="profile.html"><i class="fas fa-user"></i> Profile</a></li>
                <li><a href="meal_tracker.html"><i class="fas fa-utensils"></i> Meal
Tracker</a></li>
                <li><a href="activity_tracker.html"><i class="fas fa-running"></i> Activity
Tracker</a>
```

```
</li>
            <li><a href="progress_tracker.html"><i class="fas fa-chart-line"></i> Progress
Tracker</a>
</li>
            <li><a href="bmi_calculator.html"><i class="fas fa-calculator"></i> BMI
Calculator</a>
</li>
            <li><a href="reminders.html"><i class="fas fa-bell"></i> Reminders</a>
          </li>
            <li><a href="exercises.html"><i class="fas fa-dumbbell"></i> Exercises</a>
         </li>
            <li><a href="wellness_videos.html"class="active"><i class="fas fa-
heartbeat"></i> Wellness Videos</a>
       </li>
            <li><a href="#" id="logoutButton"><i class="fas fa-sign-out-alt"></i>
Logout</a></li>
          </ul>
       </nav>
    </div>
    <div class="main-content">
       <header>
          <button class="sidebar-toggle" id="sidebarToggle"><i class="fas fa-
bars"></i></button>
       <h1>Wellness Videos</h1>
       </header>
       <main>
          <section class="card">
             <h2>Browse Videos</h2>
             <div id="wellnessVideosList" class="wellness-videos-list">
                <p class="info-message">Loading videos...</p> </div>
              <p id="wellnessVideosMessage" class="message hidden">
          </p>
          </section></main>
     </div>
    <script src="js/dashboard_common.js" type="module">
</script>
    <script src="js/wellness_videos.js" type="module">
</script>
</body>
</html>
```