

# **AI-Based Diet And Workout Plan Generator**

*An Industry Oriented Project Report Submitted  
In partial fulfillment of the requirement for the award of the degree of*

*Bachelor of Technology  
In  
Computer Science and Engineering -Artificial Intelligence and Machine  
Learning*

**By**

<b>V. Sai Venkat</b>	<b>-</b>	<b>22N31A66F9</b>
<b>SK. Imran</b>	<b>-</b>	<b>22N31A66G2</b>
<b>T. Sai Yathish</b>	<b>-</b>	<b>23N35A6618</b>

Under the Guidance of

***Mr. T. Hari Babu***  
*Assistant Professor*

**Computational Intelligence Department  
MRCET**



**DEPARTMENT OF COMPUTATIONAL INTELLIGENCE  
MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY**  
(Affiliated to JNTU, Hyderabad)  
**ACCREDITED by AICTE-NBA**  
**Maisammaguda, Dhulapally post, Secunderabad-500014.**  
**2022-2026**

## **DECLARATION**

I hereby declare that the project entitled “**AI-Based Diet And Workout Plan Generator**” submitted to **Malla Reddy College of Engineering and Technology**, affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of **Bachelor of Technology in Computer Science and Engineering- Artificial Intelligence and Machine Learning** is a result of original research work done by me.

It is further declared that the project report or any part there of has not been previously submitted to any University or Institute for the award of degree or diploma.

**V. Sai Venkat (22N31A66F9)**

**SK. Imran (22N31A66G2)**

**T. Sai Yathish (23N35A6618)**



# **MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY**

**(AUTONOMOUS INSTITUTION - UGC, GOVT. OF INDIA)**

**Affiliated to JNTUH; Approved by AICTE, NBA-Tier 1 & NAAC with A-GRADE | ISO 9001:2015**

## **CERTIFICATE**

This is to certify that this is the bonafide record of the project titled “**AI-Based Diet And Workout Plan Generator**” submitted by **V. Sai Venkat (22N31A66F9)**, **SH. Imran (22N31A66G2)**, **T. Sai Yathish (23N35A6618)** of B.Tech in the partial fulfillment of the requirements for the degree of **Bachelor of Technology in Computer Science and Engineering- Artificial Intelligence and Machine Learning**, Dept. of CI during the year 2023-2024. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

**Mr. T. Hari Babu**  
Assistant Professor

**INTERNAL GUIDE**

**Dr. D. Sujatha M.Tech, PhD**  
Professor & HOD

**HEAD OF THE DEPARTMENT**

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

We feel honored and privileged to place our warm salutation to our college Malla Reddy College of Engineering and technology (UGC-Autonomous), our Director ***Dr. VSK Reddy*** who gave us the opportunity to have experience in engineering and profound technical knowledge.

We are indebted to our Principal ***Dr. S. Srinivasa Rao*** for providing us with facilities to do our project and his constant encouragement and moral support which motivated us to move forward with the project.

We would like to express our gratitude to our Head of the Department ***Dr. D. Sujatha*** for encouraging us in every aspect of our system development and helping us realize our full potential.

We would like to thank our application development guide as well as our internal guide **Mr. T. Hari Babu (Assistant Professor)**, for his structured guidance and never-ending encouragement. We are extremely grateful for valuable suggestions and unflinching co-operation throughout application development work.

We would also like to thank all supporting staff of department of CI and all other departments who have been helpful directly or indirectly in making our application development a success.

We would like to thank our parents and friends who have helped us with their valuable suggestions and support has been very helpful in various phases of the completion of the application development.

By

**V. Sai Venkat (22N31A66F9)**

**SK. Imran (22N31A66G2)**

**T. Sai Yathish (23N35A6618)**

## **ABSTRACT**

“Health Is Wealth” the most known phrase we all are aware about. Nowadays, people tend to eat unhealthy food and their careless behaviour may cause severe diseases. For such cases, more efforts should be made to plan a diet but people are reluctant of consulting a dietitian. Artificial Intelligence is a technology that will help to make interaction between man and machine using natural language possible. It asks all the data from the user and processes it to provide the diet plan to the user. This will help common people to maintain their health in better way with proper guidance. Builds a balanced meal and workout plan in seconds with this AI-powered diet plan generator. Generate a budget friendly diet and workouts according to your need. Now a days it is observed that most of the aged people and even some of the youngsters are suffering from more than one health issue (such as diabetics and BP, thyroid and Gastric problems etc). For such cases, more efforts should be made to plan a diet but people are reluctant of consulting a dietitian (due to laziness or unawareness). It would be super simple and comfortable which analyses your health issues and tells you what to eat and what not to. It is also important how you process the food. In this we can have a complete picture of what all things should be eaten in your diet and what all to avoid. This can help you in many ways as it is having features with calories intake which healthy diet could make you decrease your problems. Hence making one aware and motivating to lead a healthier life with a proper personalized diet and workout plan.

# TABLE OF CONTENTS

<b>S.No.</b>	<b>Topic</b>	<b>Page No.</b>
1	<b>INTRODUCTION.....</b>	<b>1</b>
	1.1 Purpose.....	1
	1.2 Background of project.....	1
	1.3 Scope of project.....	2
	1.4 Project features.....	2
2	<b>SYSTEM REQUIREMENTS.....</b>	<b>3</b>
	2.1 Hardware Requirements.....	3
	2.2 Software Requirements.....	3
	2.3 Existing System.....	3
	2.4 Proposed System.....	5
3	<b>SYSTEM DESIGN.....</b>	<b>6</b>
	3.1 System Architecture.....	6
	3.2 UML Diagram.....	7
4	<b>IMPLEMENTATION.....</b>	<b>11</b>
	4.1 Source Code.....	11
	4.2 Output Screens.....	30
5	<b>CONCLUSION &amp; FUTURE SCOPE.....</b>	<b>33</b>
	5.1 Conclusion.....	33
	5.2 Future Scope.....	33
6	<b>BIBILOGRAPHY.....</b>	<b>34</b>

# CHAPTER 1

## INTRODUCTION

### **1.1 Purpose:**

The project on an AI-based diet and workout plan generator aims to revolutionize the way individuals approach their health and fitness goals. By leveraging artificial intelligence, this system analyzes user data such as dietary preferences, fitness levels, and goals to generate personalized diet plans and workout routines.

Through machine learning algorithms, the system continually refines its recommendations based on user feedback and progress, ensuring tailored and effective strategies. This innovative approach not only streamlines the process of creating personalized health plans but also empowers individuals to make informed decisions about their well-being, leading to improved adherence and long-term success in achieving fitness goals. Overall, the project represents a significant step towards harnessing the power of AI to promote healthier lifestyles and enhance overall well-being.

### **1.2 Background of project:**

The project on AI-based diet and workout plan generators arises from the increasing demand for personalized health and fitness solutions. Traditional diet and exercise plans often follow a one-size-fits-all approach, which fails to account for individual differences in metabolism, lifestyle, fitness goals, and dietary preferences. With advancements in artificial intelligence, there is now the potential to develop sophisticated systems that can tailor diet and workout plans to meet the specific needs of each user, thereby enhancing the effectiveness and sustainability of health and fitness programs.

### 1.3 Scope of project:

The scope of the AI-based diet and workout plan generator project involves developing a comprehensive and personalized health management tool that leverages artificial intelligence to create tailored diet and exercise regimens. This system will analyze individual user data, including physical characteristics, fitness goals, dietary preferences, health conditions, and lifestyle habits, to generate customized meal plans and workout routines. The AI will continuously adapt and refine these plans based on user feedback and progress, offering dynamic adjustments to optimize results. The project aims to make healthy living more accessible and effective by providing users with scientifically backed, data-driven recommendations that cater to their unique needs and objectives.

### 1.4 Project Features:

The system features are as follows:

**1.Personalized Recommendations:** Tailored diet and workout plans based on individual goals, body type, fitness level, and dietary preferences.

**2.Nutritional Analysis:** Detailed breakdown of nutritional information for meals, including calorie count, macronutrient distribution, and micronutrient content.

**3.Meal Planning:** Weekly or monthly meal plans with recipes and portion sizes for breakfast, lunch, dinner, and snacks.

**4.Exercise Routine:** Customized workout routines with exercises, sets, reps, and rest periods based on fitness goals (e.g., weight loss, muscle gain, endurance).

**5.Progress Tracking:** Tools to monitor progress, including weight, body measurements, workout performance, and adherence to the diet plan.



# CHAPTER 2

## SYSTEM REQUIREMENTS

### 2.1 Hardware Requirements:

- HDD 512 GB
- RAM 4 GB
- Processor i5
- Keyboard
- Mouse

### 2.2 Software requirements:

#### Frontend:

- HTML5
- CSS3
- JavaScript ES6
- Python 3.8

#### Backend:

- MySQL 8.0

#### Operating System:

- Windows 11

### 2.3 Existing System:

Existing system focuses on one particular issue/requirement such that weight loss/weight gain. Calculates the number of calories that must be taken by the person. Suggests a diet plan for that.

Ex:HealthifyMe, Fitbit

Here HealthifyMe & Fitbit gives us the calories count of the food the client consume and help us to take the balanced food and FitBit gives the the count of the client steps and count of the calories decreased.

### **2.3.1 Drawbacks of existing system:**

**1.Lack of personalization:** Some systems may not take into account individual preferences, dietary restrictions, or fitness levels, resulting in generic plans that may not be suitable or sustainable for users.

**2.Limited data inputs:** Many systems rely on limited data inputs such as age, gender, and weight, without considering other important factors like medical history, activity levels, or lifestyle habits.

**3.Over-reliance on algorithms:** While algorithms play a crucial role in generating plans, an over-reliance on them without human oversight may lead to inaccuracies or unsafe recommendations.

**4.Static recommendations:** Some systems provide static plans without considering changes in user goals, progress, or external factors, resulting in outdated or ineffective recommendations over time.

**5.Lack of behavior change support:** Successful diet and workout plans require more than just recommendations; they also need support for behavior change, motivation, and accountability, which may be lacking in some systems.

**6.Ethical considerations:** There are ethical concerns regarding data privacy, biases in algorithmic recommendations, and potential harm caused by promoting unrealistic body standards or unhealthy behaviors.

**7.Limited feedback mechanisms:** Many systems lack effective feedback mechanisms for users to provide input on the effectiveness of recommendations or to report issues, hindering improvements and adjustments to the system.

## **2.4 Proposed System:**

Provides dietary assistance to the users and helps the users to control their hunger issues and maintain their diet properly. Any age group can use this application. Through BMI this application is used. BMI is calculated with the help of user's height, weight, gender and their daily activity. This application helps the user to have a good and healthy diet with protein ingredients.

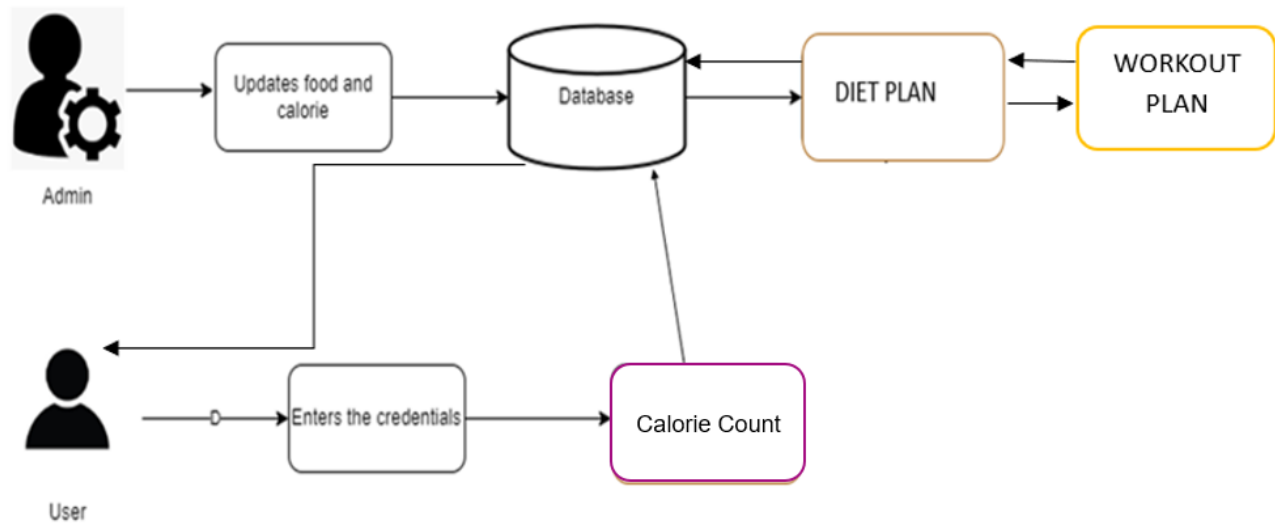
It's crucial to note that any AI-based system should be designed with user safety and well-being in mind, and users with specific health concerns should consult healthcare professionals before making significant changes to their diet or exercise routine.

AI-based diet and workout plan generator involves a combination of machine learning, nutrition science, and exercise physiology. Such a system would need to take into account individual preferences, goals, health conditions, and other relevant factors.

# CHAPTER 3

## SYSTEM DESIGN

### 3.1 System Architecture



**Fig 3.1 System Architecture of AI-Based Diet And Workout Plan Generator**

## 3.2 UML Diagrams

### 3.2.1 Use case diagram

Use Case during requirement elicitation and analysis to represent the functionality of the system. Use case describes a function by the system that yields a visible result for an actor. The identification of actors and use cases result in the definitions of the boundary of the system i.e., differentiating the tasks accomplished by the system and the tasks accomplished by its environment.



**Fig 3.2 Use Case Diagram of AI-Based Diet And Workout Plan Generator**

### 3.2.2 Class Diagram

Class diagrams model class structure and contents using design elements such as classes, packages and objects. Class diagram describe the different perspective when designing a system-conceptual, specification and implementation. Classes are composed of three things: name, attributes, and operations. Class diagram also display relationships such as containment, inheritance, association etc. The association relationship is most common relationship in a class diagram. The association shows the relationship between instances of classes.

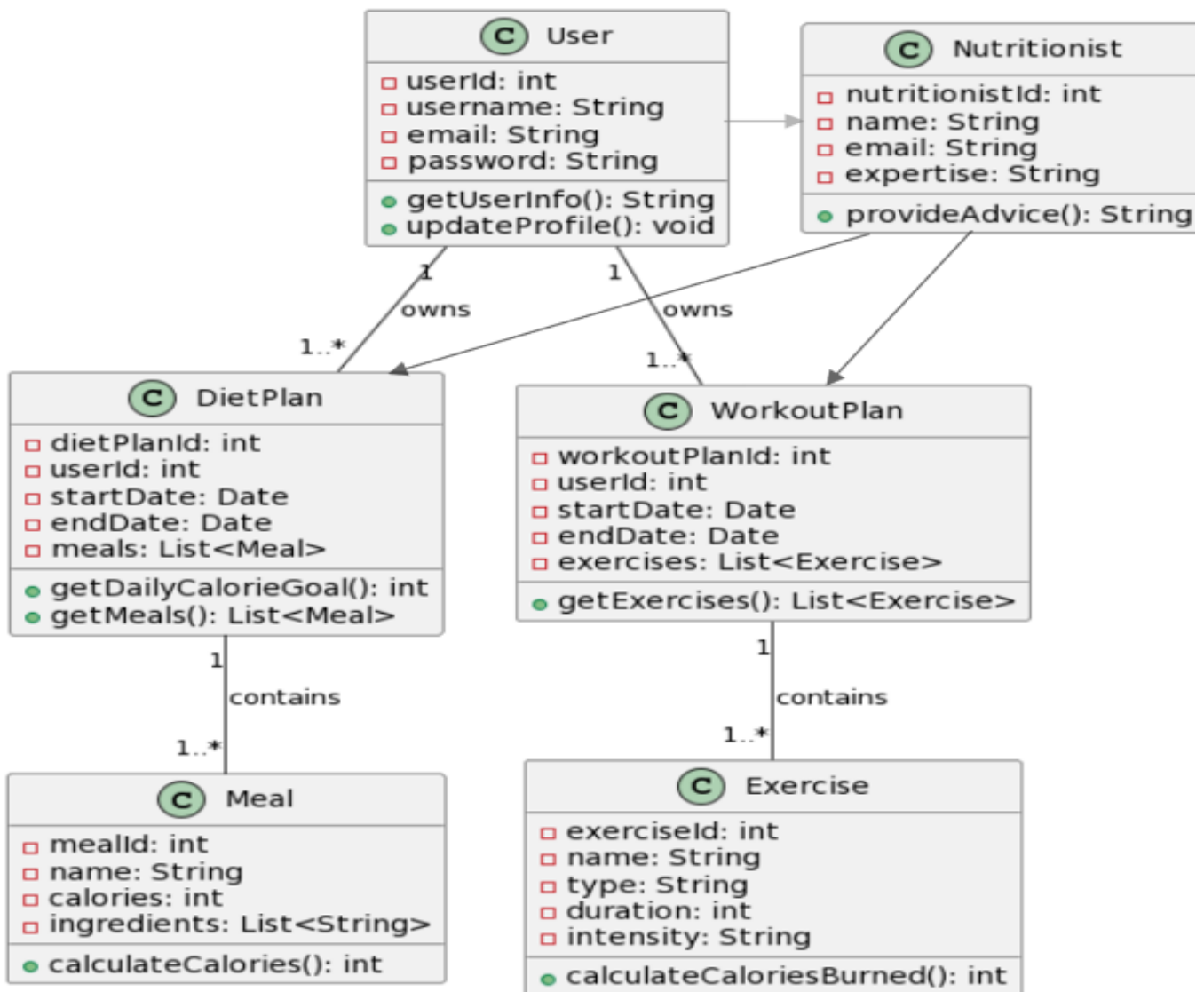
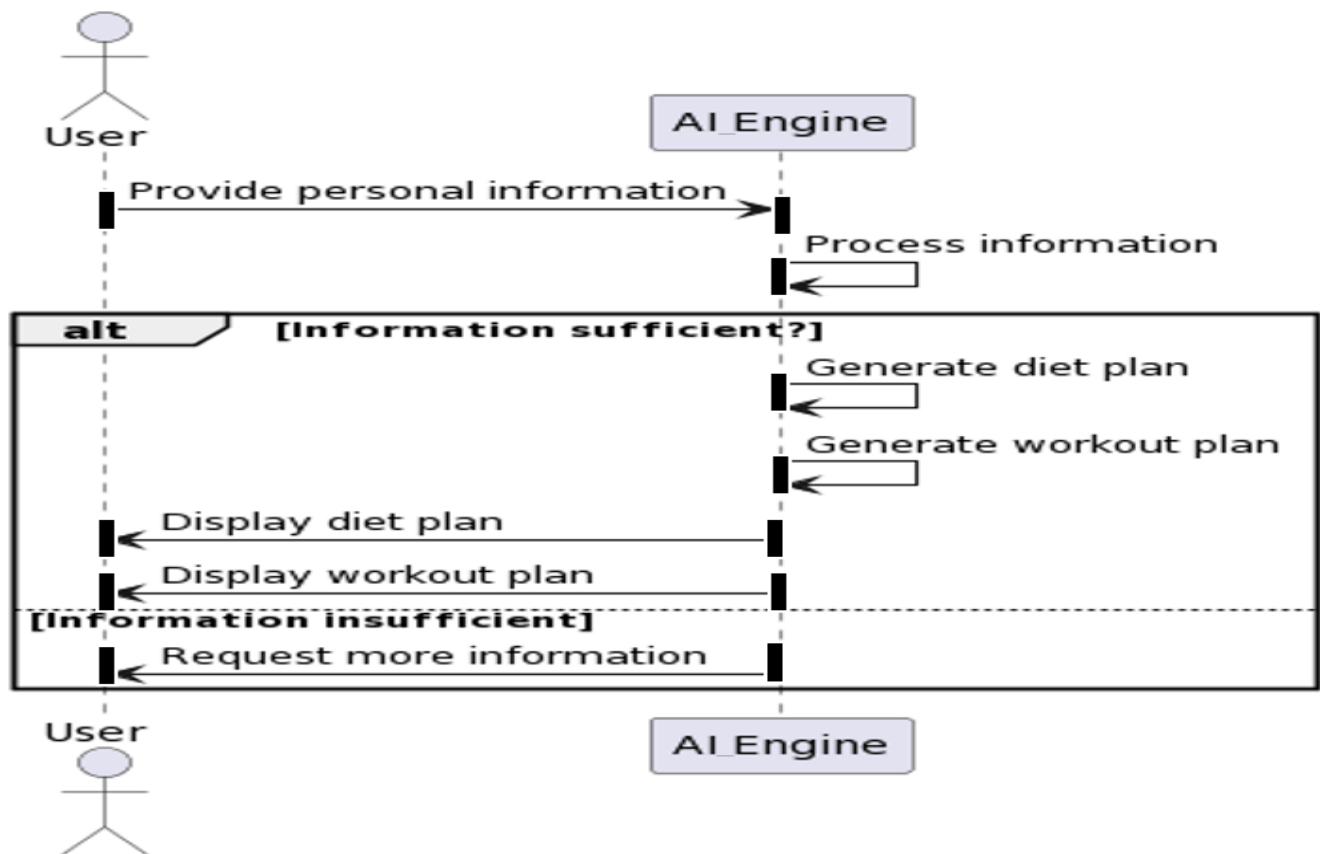


Fig 3.3 Class Diagram of AI-Based Diet And Workout Plan Generator

### 3.2.3 Sequence Diagram

Sequence diagram displays the time sequence of the objects participating in the interaction. This consists of the vertical dimension (time) and horizontal dimension (different objects).

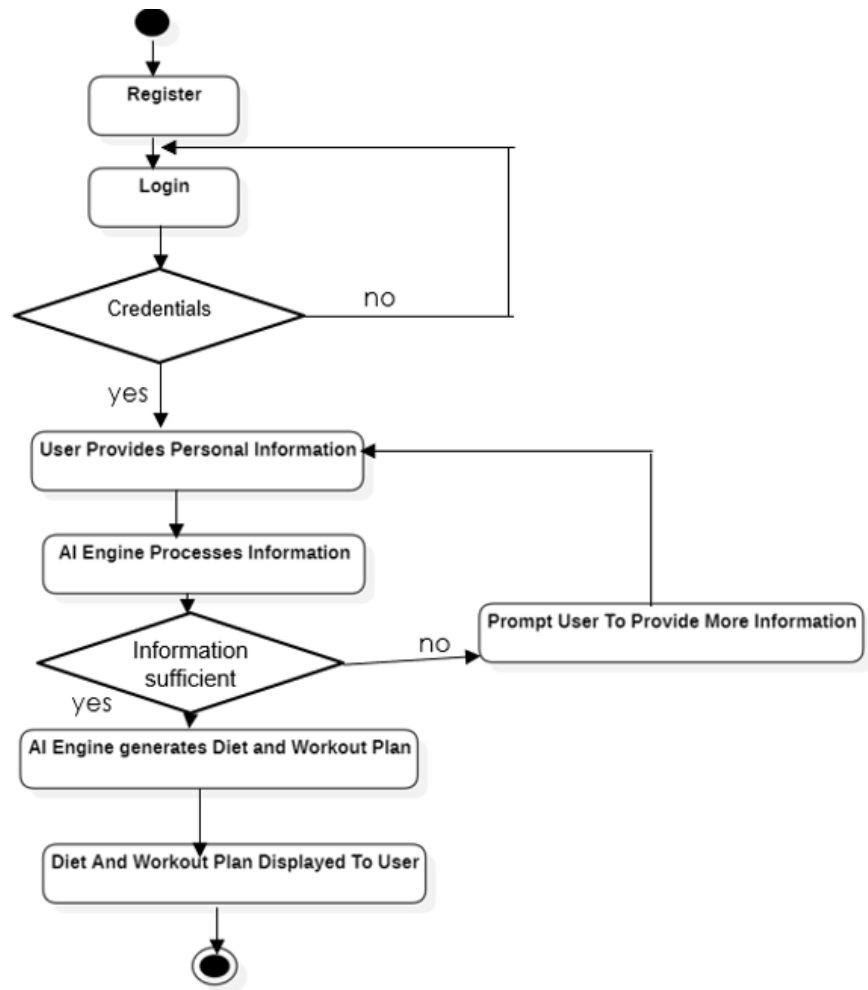
Objects: An object can be thought of as an entity that exists at a specified time and has a definite value, as well as a holder of identity. A sequence diagram depicts item interactions in chronological order. It illustrates the scenario's objects and classes, as well as the sequence of messages sent between them in order to carry out the scenario's functionality. In the Logical View of the system under development, sequence diagrams are often related with use case realizations. Event diagrams and event scenarios are other names for sequence diagrams. A sequence diagram depicts multiple processes or things that exist simultaneously as parallel vertical lines (lifelines), and the messages passed between them as horizontal arrows, in the order in which they occur. This enables for the graphical specification of simple runtime scenarios.



**Fig 3.4 Sequence Diagram of AI-Based Diet And Workout Plan Generator**

### 3.2.4 Activity Diagram

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions



**Fig 3.5 Activity Diagram of AI-Based Diet And Workout Plan Generator**



# CHAPTER 4

## IMPLEMENTATION

### 4.1: Code

```
"use client";

import type { FC } from "react";
import { account } from "@pages/api/appwriteConfig";
import { useState, useEffect } from "react";
import alanBtn from "@alan-ai/alan-sdk-web";
import { v4 as uuidv4 } from "uuid";
import { useRouter } from "next/navigation";
import Link from "next/link";

interface pageProps {}

const SignUpPage: FC<pageProps> = ({}) => {
  //AlanAI
  const [name, setName] = useState("");
  const ALAN_Key = `${process.env.NEXT_PUBLIC_ALAN_KEY!}`;
  useEffect(() => {
    // if (typeof window !== "undefined")
    //   alanBtn({
    //     key: ALAN_Key,
    //     onCommand: (commandData) => {
    //       //@ts-ignore
    //       if(commandData.command === 'name'){
    //         //@ts-ignore
    //         setName(commandData.data)
    //       }
    //     },
    //   });
  }, []);
  //Appwrite
  const navigate = useRouter();
  const [user, setUser] = useState({
```

```

    name: "",
    email: "",
    password: "",
  });

  //Signup
  const signupUser = async (e: { preventDefault: () => void }) => {
    e.preventDefault();

    const promise = account.create(
      uuidv4(),
      user.email,
      user.password,
      user.name
    );

    promise.then(
      function (response: any) {
        console.log(response);
        navigate.push("/login");
        alert("Account Created Successfully"); //success
      },
      function (error: any) {
        console.log(error);
        alert(error); // Failure
      }
    );
  };

  return (
    <>
    <div className="min-h-full flex flex-col font-product justify-center py-12 sm:px-6 lg:px-8">
      <div className="text-center text-violet-600 text-2xl font-bold">
        Sign up
      </div>
      <div className="mt-8 sm:mx-auto sm:w-full sm:max-w-md">
        <div className="bg-white py-8 px-4 shadow-md sm:rounded-lg sm:px-10">
          <form className="space-y-6" action="#" method="POST">
            <div>

```

```

<label
  htmlFor="name"
  className="block text-sm font-medium text-gray-700"
>
  Name
</label>
<div className="mt-1">
  <input
    id="name"
    name="name"
    type="text"
    autoComplete="name"
    value={ name }
    required
    className="appearance-none block w-full px-3 py-2 border border-gray-
300 rounded-md shadow-sm placeholder-gray-400 focus:outline-none focus:ring-
indigo-500 focus:border-indigo-500 sm:text-sm"
    onChange={(e) => {
      setUser({
        ...user,
        name: e.target.value,
      });
    }}
  />
</div>
</div>
<div>
  <label
    htmlFor="email"
    className="block text-sm font-medium text-gray-700"
  >
    Email address
  </label>
  <div className="mt-1">
    <input
      id="email"
      name="email"
      type="email"

```

```

        className="appearance-none block w-full px-3 py-2 border border-gray-
300 rounded-md shadow-sm placeholder-gray-400 focus:outline-none focus:ring-
indigo-500 focus:border-indigo-500 sm:text-sm"
        onChange={(e) => {
            setUser({
                ...user,
                email: e.target.value,
            });
        }}
    />
</div>
</div>

<div>
    <label
        htmlFor="password"
        className="block text-sm font-medium text-gray-700"
    >
        Password
    </label>
    <div className="mt-1">
        <input
            id="password"
            name="password"
            type="password"
            autoComplete="current-password"
            required
            className="appearance-none block w-full px-3 py-2 border border-gray-
300 rounded-md shadow-sm placeholder-gray-400 focus:outline-none focus:ring-
indigo-500 focus:border-indigo-500 sm:text-sm"
            onChange={(e) => {
                setUser({
                    ...user,
                    password: e.target.value,
                });
            }}
        />
    </div>
</div>

```

```

<div>
  <button
    type="submit"
    className="w-full flex justify-center py-2 px-4 border border-transparent
rounded-md shadow-sm text-lg font-medium text-white bg-violet-600 hover:bg-violet-
700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500"
    onClick={ signupUser }
  >
    Sign up
  </button>
</div>
</form>

```

```

<div className="mt-6">
  <div className="form-content-wrapper ml-6 mx-6 my-6 text-sm text-center
font-semibold font-manrope text-violet-600">Already have an account, Click here to
<Link className="" href="/login"><button className="underline"> Login </button>
</Link></div>

```

```

  <div className="relative">
    <div className="absolute inset-0 flex items-center">
      <div className="w-full border-t border-gray-300" />
    </div>
    <div className="relative flex justify-center text-sm">
      <span className="px-2 bg-white text-gray-500">
        Or continue with
      </span>
    </div>
  </div>
</div>

```

```

<div className="mt-6 grid grid-cols-3 gap-3">
  <div>
    <a
      href="/"
      className="w-full inline-flex justify-center py-2 px-4 border border-gray-
300 rounded-md shadow-sm bg-white text-sm font-medium text-gray-500 hover:bg-
gray-50"
    >
      <span className="sr-only">Sign in with Facebook</span>
      <svg
        className="w-5 h-5"

```

```

        aria-hidden="true"
        fill="currentColor"
        viewBox="0 0 20 20"
      >
        <path
          fillRule="evenodd"
          d="M20 10c0-5.523-4.477-10-10-10S0 4.477 0 10c0 4.991 3.657 9.128
8.438 9.878v-6.987h-2.54V10h2.54V7.797c0-2.506 1.492-3.89 3.777-3.89 1.094 0
2.238.195 2.238.195v2.46h-1.26c-1.243 0-1.63.771-1.63 1.562V10h2.773l-.443 2.89h-
2.33v6.988C16.343 19.128 20 14.991 20 10z"
          clipRule="evenodd"
        />
      </svg>
    </a>
  </div>

  <div>
    <a
      href="/"
      className="w-full inline-flex justify-center py-2 px-4 border border-gray-
300 rounded-md shadow-sm bg-white text-sm font-medium text-gray-500 hover:bg-
gray-50"
    >
      <span className="sr-only">Sign in with Twitter</span>
      <svg
        className="w-5 h-5"
        aria-hidden="true"
        fill="currentColor"
        viewBox="0 0 20 20"
      >
        <path d="M6.29 18.251c7.547 0 11.675-6.253 11.675-11.675 0-.178 0-
.355-.012-.53A8.348 8.348 0 0020 3.92a8.19 8.19 0 01-2.357.646 4.118 4.118 0
001.804-2.27 8.224 8.224 0 01-2.605.996 4.107 4.107 0 00-6.993 3.743 11.65 11.65 0
01-8.457-4.287 4.106 4.106 0 001.27 5.477A4.073 4.073 0 018 7.713v.052a4.105
4.105 0 003.292 4.022 4.095 4.095 0 01-1.853.07 4.108 4.108 0 003.834 2.85A8.233
8.233 0 010 16.407a11.616 11.616 0 006.29 1.84" />
      </svg>
    </a>
  </div>

```

```
</div>
<a
  href="/"
  className="w-full inline-flex justify-center py-2 px-4 border border-gray-
300 rounded-md shadow-sm bg-white text-sm font-medium text-gray-500 hover:bg-
gray-50"
>
  <span className="sr-only">Sign in with GitHub</span>
  <svg
    className="w-5 h-5"
    aria-hidden="true"
    fill="currentColor"
    viewBox="0 0 20 20"
  >
    <path
      fillRule="evenodd"
      d="M10 0C4.477 0 0 4.477 0 10c0 5.523 4.477 10 10 10c5.523 0 10-4.477 10-10
9.504 5.092 .682-.217 .682-.483 0-.237-.008-.868-.013-1.703-.278-2.782-.605-3.369-1.343-
3.369-1.343-.454-1.158-1.111-1.466-1.111-1.466-.908-.621-.608-.608-1.003-.07
1.531 1.032 1.531 1.032 .892 1.53 2.341 1.088 2.918 .32 .092-.647 .35-1.088 .636-1.338-
2.22-.253-4.555-1.113-4.555-4.951 0-1.093 .39-1.988 1.029-2.688 .103-.253-.446-
1.272 .098-2.65 0 0 .84-.27 2.75 1.026 A9.564 9.564 0 0 10 4.844 c.85 .004 1.705 .115
2.504 .337 1.909-1.296 2.747-1.027 2.747-1.027 .546 1.379 .203 2.398 .1 2.651 .647
1.028 1.595 1.028 2.688 0 3.848-2.339 4.695-4.566 4.942 .359 .31 1.678 .921 .678 1.856 0
1.338-.012 2.419-.012 2.747 0 .268 .18 5.688 .482 A10.019 10.019 0 0 0 20 10.017 C20
4.484 15.522 0 10 0z"
      clipRule="evenodd"
    />
  </svg>
</a>
</div>
</div>
</div>
</div>
</div>
</div>
```

```

export default SignUpPage;

"use client";

import type { FC } from "react";
import React, { useState } from "react";
import { account } from "@/pages/api/appwriteConfig";
import { useRouter } from "next/navigation";

interface pageProps {}

const LoginPage: FC<pageProps> = ({}) => {
  const navigate = useRouter();
  const [user, setUser] = useState({
    email: "",
    password: "",
  });

  const loginUser = async (e: { preventDefault: () => void }) => {
    e.preventDefault();
    try {
      await account.createEmailSession(user.email, user.password);
      navigate.push("/form");
    } catch (error) {
      console.log(error);
      alert(error);
    }
  };

  return (
    <div className="min-h-full flex flex-col font-product justify-center py-24 sm:px-6 lg:px-8">
      <div className="text-center text-violet-600 text-2xl font-bold">
        Login
      </div>
      <div className="mt-8 sm:mx-auto sm:w-full sm:max-w-md">
        <div className="bg-white py-8 px-4 shadow-md sm:rounded-lg sm:px-10">
          <form className="space-y-6" action="#" method="POST">
            <div>
              <label

```



```

      htmlFor="email"
      className="block text-sm font-medium text-gray-700"
    >
      Email address
    </label>
    <div className="mt-1">
      <input
        id="email"
        name="email"
        type="email"
        className="appearance-none block w-full px-3 py-2 border border-gray-
300 rounded-md shadow-sm placeholder-gray-400 focus:outline-none focus:ring-
indigo-500 focus:border-indigo-500 sm:text-sm"
        onChange={ (e) => {
          setUser({
            ...user,
            email: e.target.value,
          });
        }}
      />
    </div>
  </div>

  <div>
    <label
      htmlFor="password"
      className="block text-sm font-medium text-gray-700"
    >
      Password
    </label>
    <div className="mt-1">
      <input
        id="password"
        name="password"
        type="password"
        autoComplete="current-password"
        required
        className="appearance-none block w-full px-3 py-2 border border-gray-
300 rounded-md shadow-sm placeholder-gray-400 focus:outline-none focus:ring-
indigo-500 focus:border-indigo-500 sm:text-sm"

```

```

        onChange={ (e) => {
            setUser({
                ...user,
                password: e.target.value,
            });
        }}
    />
</div>
</div>

<div>
    <button
        type="submit"
        className="w-full flex justify-center py-2 px-4 border border-transparent
rounded-md shadow-sm text-lg font-medium text-white bg-violet-600 hover:bg-violet-
700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500"
        onClick={loginUser}
    >
        Log In
    </button>
</div>
</form>

<div className="mt-6">
    <div className="relative">
        <div className="absolute inset-0 flex items-center">
            <div className="w-full border-t border-gray-300" />
        </div>
        <div className="relative flex justify-center text-sm">
            <span className="px-2 bg-white text-gray-500">
                Or continue with
            </span>
        </div>
    </div>
</div>

<div className="mt-6 grid grid-cols-3 gap-3">
    <div>
        <a
            href="/"

```

```
className="w-full inline-flex justify-center py-2 px-4 border border-gray-300 rounded-md shadow-sm bg-white text-sm font-medium text-gray-500 hover:bg-gray-50"
```

```
>
```

```
<span className="sr-only">Sign in with Facebook</span>
```

```
<svg
```

```
  className="w-5 h-5"
```

```
  aria-hidden="true"
```

```
  fill="currentColor"
```

```
  viewBox="0 0 20 20"
```

```
>
```

```
<path
```

```
  fillRule="evenodd"
```

```
  d="M20 10c0-5.523-4.477-10-10-10S0 4.477 0 10c0 4.991 3.657 9.128 8.438 9.878v-6.987h-2.54V10h2.54V7.797c0-2.506 1.492-3.89 3.777-3.89 1.094 0 2.238.195 2.238.195v2.46h-1.26c-1.243 0-1.63.771-1.63 1.562V10h2.773l-.443 2.89h-2.33v6.988C16.343 19.128 20 14.991 20 10z"
```

```
  clipRule="evenodd"
```

```
/>
```

```
</svg>
```

```
</a>
```

```
</div>
```

```
<div>
```

```
<a
```

```
  href="/"
```

```
  className="w-full inline-flex justify-center py-2 px-4 border border-gray-300 rounded-md shadow-sm bg-white text-sm font-medium text-gray-500 hover:bg-gray-50"
```

```
>
```

```
<span className="sr-only">Sign in with Twitter</span>
```

```
<svg
```

```
  className="w-5 h-5"
```

```
  aria-hidden="true"
```

```
  fill="currentColor"
```

```
  viewBox="0 0 20 20"
```

```
>
```

```
<path d="M6.29 18.251c7.547 0 11.675-6.253 11.675-11.675 0-.178 0-.355-.012-.53A8.348 8.348 0 0020 3.92a8.19 8.19 0 01-2.357.646 4.118 4.118 0 001.804-2.27 8.224 8.224 0 01-2.605.996 4.107 4.107 0 00-6.993 3.743 11.65 11.65 0
```

01-8.457-4.287 4.106 4.106 0 001.27 5.477A4.073 4.073 0 01.8 7.713v.052a4.105  
4.105 0 003.292 4.022 4.095 4.095 0 01-1.853.07 4.108 4.108 0 003.834 2.85A8.233  
8.233 0 010 16.407a11.616 11.616 0 006.29 1.84" />

</svg>

</a>

</div>

<div>

<a

href="/"

className="w-full inline-flex justify-center py-2 px-4 border border-gray-  
300 rounded-md shadow-sm bg-white text-sm font-medium text-gray-500 hover:bg-  
gray-50"

>

<span className="sr-only">Sign in with GitHub</span>

<svg

className="w-5 h-5"

aria-hidden="true"

fill="currentColor"

viewBox="0 0 20 20"

>

<path

fillRule="evenodd"

d="M10 0C4.477 0 0 4.484 0 10.017c0 4.425 2.865 8.18 6.839  
9.5045.092.682-.217.682-.483 0-.237-.008-.868-.013-1.703-2.782.605-3.369-1.343-  
3.369-1.343-.454-1.158-1.11-1.466-1.11-1.466-.908-.62.069-.608 1.003.07  
1.531 1.032 1.531 1.032.892 1.53 2.341 1.088 2.91.832.092-.647.35-1.088.636-1.338-  
2.22-.253-4.555-1.113-4.555-4.951 0-1.093.39-1.988 1.029-2.688-.103-.253-.446-  
1.272.098-2.65 0 0 .84-.27 2.75 1.026A9.564 9.564 0 0110 4.844c.85.004 1.705.115  
2.504.337 1.909-1.296 2.747-1.027 2.747-1.027.546 1.379.203 2.398.1 2.651.64.7  
1.028 1.595 1.028 2.688 0 3.848-2.339 4.695-4.566 4.942.359.31.678.921.678 1.856 0  
1.338-.012 2.419-.012 2.747 0 .268.18.58.688.482A10.019 10.019 0 0020 10.017C20  
4.484 15.522 0 10 0z"

clipRule="evenodd"

/>

</svg>

</a>

</div>

</div>

</div>

```

        </div>
      </div>
    </div>
  );
};

export default LoginPage;

import React, { ChangeEvent } from "react";
import useFormOneStore from "@store/formStore";
import Button from "@mui/material/Button";
import ButtonGroup from "@mui/material/ButtonGroup";
import FormControl from "@mui/material/FormControl";
import InputLabel from "@mui/material/InputLabel";
import InputAdornment from "@mui/material/InputAdornment";
import Input from "@mui/material/Input";
import { MenuItem, Select, SelectChangeEvent } from "@mui/material";

const StepForm4 = () => {
  const state = useFormOneStore();

  const handleExerciseType = (type: string) => {
    state.setExerciseType(type);
  };

  const handleExerciseExperience = (exp: string) => {
    state.setExerciseExperience(exp);
  };

  const handleFoodPreference = (food: string) => {
    state.setFoodPreference(food);
  };

  const handleDietType = (diet: string) => {
    state.setDietType(diet);
  };

  const handleDurationChange = (e: any) => {
    state.setTimeDuration(e.target.value);
  };

```

```

return (
  <div>
    {state.selectedPlan === "exercise" ? (
      <div className="font-product">
        <div className="flex flex-col justify-center items-center mt-12">
          <h1 className="font-bold text-3xl">Type of exercise</h1>
          <ButtonGroup
            size="large"
            aria-label="large button group"
            className="mt-8 gap-2 max-sm:flex-col"
          >
            <button
              className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
                state.exerciseType === "homeworkout" &&
                "bg-violet-500 text-white"
              }`}
              onClick={() => handleExerciseType("homeworkout")}
            >
              Home workout
            </button>
            <button
              className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
                state.exerciseType === "calisthenics" &&
                "bg-violet-500 text-white"
              }`}
              onClick={() => handleExerciseType("calisthenics")}
            >
              Calisthenics
            </button>
            <button
              className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
                state.exerciseType === "weightlifting" &&
                "bg-violet-500 text-white"
              }`}
              onClick={() => handleExerciseType("weightlifting")}
            >

```

```

        Weight lifting
      </button>
      <button
        className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
          state.exerciseType === "yoga" && "bg-violet-500 text-white"
        }`}
        onClick={() => handleExerciseType("yoga")}
      >
        Yoga
      </button>
    </ButtonGroup>
  </div>
  <div className="flex flex-col justify-center items-center mt-12 mb-24">
    <h1 className="text-3xl font-bold">Experience</h1>
    <ButtonGroup
      size="large"
      aria-label="large button group"
      className="mt-8 gap-2 max-sm:flex-col"
    >
      <button
        className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
          state.exerciseExperience === "beginner" &&
            "bg-violet-500 text-white"
        }`}
        onClick={() => handleExerciseExperience("beginner")}
      >
        Beginner
      </button>
      <button
        className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
          state.exerciseExperience === "Intermediate" &&
            "bg-violet-500 text-white"
        }`}
        onClick={() => handleExerciseExperience("Intermediate")}
      >
        Intermediate
      </button>

```

```

      <button
        className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
          state.exerciseExperience === "Expert" &&
            "bg-violet-500 text-white"
        }`}
        onClick={() => handleExerciseExperience("Expert")}
      >
        Expert
      </button>
    </ButtonGroup>
  </div>
</div>
):(
  <div className="font-product flex flex-col justify-center items-center mt-12 max-
sm:mt-8">
    <div className="flex items-center flex-col">
      <h1 className="text-3xl pb-4 font-bold max-sm:text-center max-sm:w-[12ch]">
        What do you prefer in eating?
      </h1>
      <ButtonGroup
        size="large"
        aria-label="large button group"
        className="gap-2 max-sm:flex-col"
      >
        <button
          className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
            state.foodPreference === "veg" && "bg-violet-500 text-white"
          }`}
          onClick={() => handleFoodPreference("veg")}
        >
          Veg
        </button>
        <button
          className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
            state.foodPreference === "non-veg" &&
              "bg-violet-500 text-white"

```



```

    } `}
    onClick={() => handleFoodPreference("non-veg")}
  >
    Non-veg
  </button>
  <button
    className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
      state.foodPreference === "vegandnonveg" &&
      "bg-violet-500 text-white"
    } `}
    onClick={() => handleFoodPreference("vegandnonveg")}
  >
    Both
  </button>
  <button
    className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
      state.foodPreference === "vegan" && "bg-violet-500 text-white"
    } `}
    onClick={() => handleFoodPreference("vegan")}
  >
    Vegan
  </button>
</ButtonGroup>
</div>

<div>
  <h1 className="text-3xl mt-8 font-bold">What's your budget?</h1>
  <FormControl fullWidth sx={{ m: 1 }} variant="standard">
    <InputLabel htmlFor="standard-adornment-amount">
      Amount
    </InputLabel>
    <Input
      className="w-[15em]"
      id="standard-adornment-amount"
      startAdornment={
        <InputAdornment position="start">₹</InputAdornment>
      }
    />
  </div>

```

```
</FormControl>
</div>
```

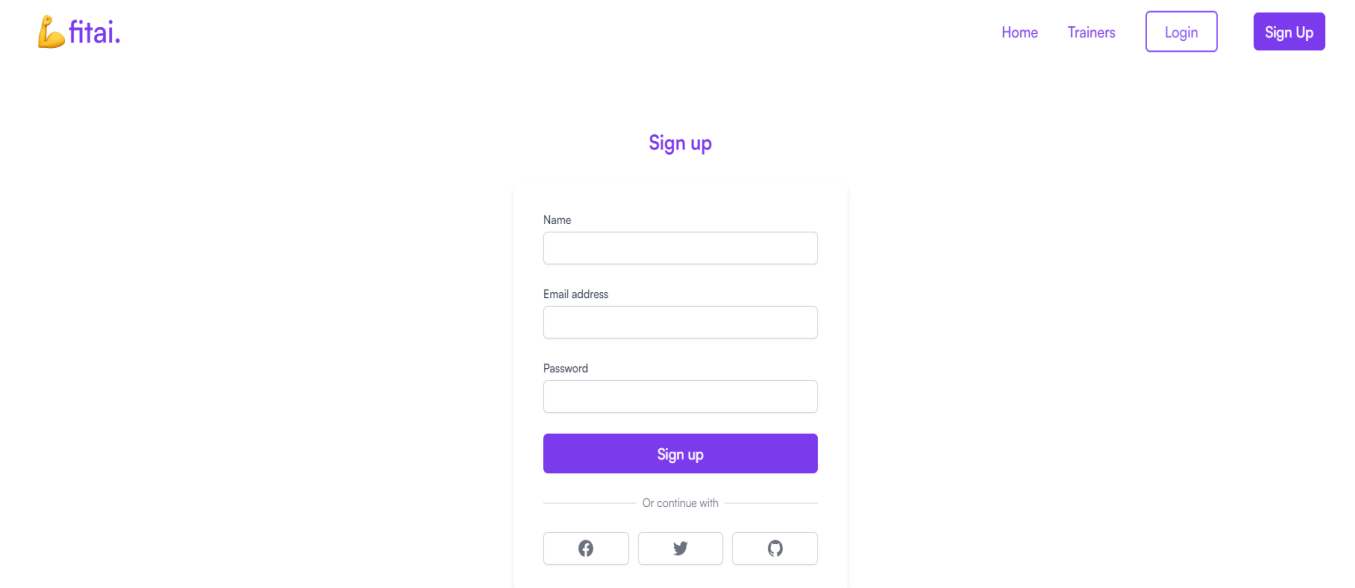
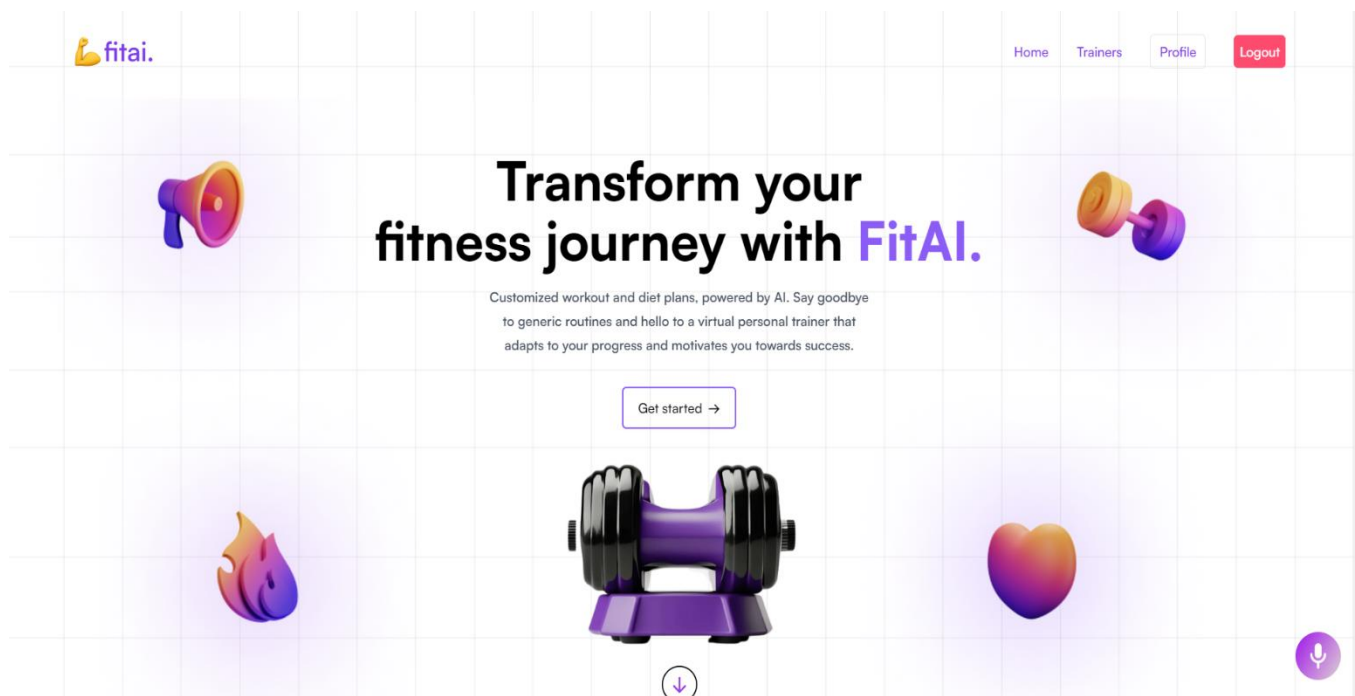
```
<div className="flex items-center flex-col pb-20">
  <h1 className="text-3xl mt-8 px-8 font-bold">Type of Diet</h1>
  <ButtonGroup
    size="large"
    aria-label="large button group"
    className="gap-2 mt-2 product-font max-sm:flex-col "
  >
    <button
      className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
        state.dietType === "homediet" && "bg-violet-500 text-white"
      } `}
      onClick={() => handleDietType("homediet")}
    >
      Home{ " " }
    </button>
    <button
      className={` bg-white hover:scale-105 transition-all border-2 border-black
font-medium py-2 px-4 rounded-md ${
        state.dietType === "hosteldiet" && "bg-violet-500 text-white"
      } `}
      onClick={() => handleDietType("hosteldiet")}
    >
      Hostel Diet
    </button>
  </ButtonGroup>
</div>
</div>
)}
```

```
<div className="mt-10 flex justify-center max-sm:-mt-10 max-sm:pb-10">
  <FormControl sx={{ minWidth: "200px" }}>
    <InputLabel id="demo-simple-select-label">Time Duration</InputLabel>
    <Select
      labelId="demo-simple-select-label"
      id="demo-simple-select"
      value={state.timeDuration}
    >
```

```
    label="Time Duration"
    onChange={handleDurationChange}
    defaultValue={ "Three Month" }
  >
    <MenuItem value={ "One Month" }>One Months</MenuItem>
    <MenuItem value={ "Three Month" }>Three Months</MenuItem>
    <MenuItem value={ "Six Month" }>Six Months</MenuItem>
  </Select>
</FormControl>
</div>
</div>
);
};

export default StepForm4;
```

## 4.2: Output Screens:




## Login


Email address


Password

Log In

Or continue with







## Track Progress



Lorem ipsum dolor sit, amet consectetur adipiscing elit. Laboriosam soluta doloremque amet corporis fugit unde, possimus laudantium similique delectus nostrum rem blanditis error mollitia praesentium, asperiores odit deserunt! Voluptatum officiis, dolores repudiandae voluptates, libero vel nulla dolor soluta officia laboriosam nemo similique, aliquid facere quod fugiat minima deleniti eius quidem iste maiores vitae amet! Quis officia, doloribus perferendis asperiores velit officiis ratione illum est excepturi recusandae, autem molestiae aliquam. Ut, odit aperiam! Quas ea aut deserunt sint, consequatur magnam praesentium id, ipsa modi consectetur, atque velit porro et quia? Cumque officiis iste sit ullam, non minus aliquid. Veritatis, deleniti atque?

## Saved Plans

Day 1 - Cardio:

- Warmup (5 minutes)
- 10 minutes of jogging
- 7 minutes of high intensity interval training (HIIT)
- 5 minutes of cool down

Day 2 - Strength Training:

- Warmup (5 minutes)
- 10 minutes of body weight exercises (push-ups, squats, lunges, burpees, etc)
- 10 minutes of weighted exercises (deadlifts, chest presses, shoulders presses, bicep curls, etc)
- 5 minutes of cool down

Day 3 - Rest and Stretch:

- 30 minutes of stretching

Day 4 - Cardio:

- Warmup (5 minutes)
- 10 minutes of jogging
- 7 minutes of HIIT
- 5 minutes of cool down

### 4.3: Testing

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation.

# **CHAPTER 5**

## **CONCLUSION & FUTURE SCOPE**

### **5.1: Conclusion**

The designed system will be useful for common people to maintain their health by taking proper diet suggested by AI Dietitian and workout application. The cost of a personal dietitian will be reduced. It will save time as it will be an excellent feature of use anytime anywhere.

The user will get a clear idea on what kind of food he/she should include in their diet. For people who doesn't have enough knowledge about planning a diet and workout properly.

But have the determination to follow it, for them this app would be very useful as it directly provides the diet plan, the user need not go through what to eat and what not to and filter them out manually.

### **5.2: Future Scope**

The future scope for an AI-based diet and workout plan generator is promising. With advancements in machine learning algorithms and data analytics, such a system could become highly personalized, considering factors like genetics, medical history, and even real-time biometric data from wearables. Integrating AI with augmented reality (AR) or virtual reality (VR) technologies could provide immersive experiences, guiding users through workouts in real-time while adapting to their progress and preferences. Additionally, incorporating natural language processing (NLP) could enhance user interaction, allowing for easier customization and feedback. Furthermore, as society becomes increasingly health-conscious, there's potential for partnerships with healthcare providers or insurance companies to offer tailored wellness programs, leveraging AI to optimize health outcomes and reduce healthcare costs.

## **CHAPTER 6**

### **BIBILOGRAPHY**

- [1] Sharma, A., Madaan, V., & Petty, F. D. (2006). Exercise for mental health. Primary care companion to the Journal of clinical psychiatry, 8(2), 106.
- [2] Agarwal, N., & Mishra, A. (2016). Effect of yoga on physical and psychological outcomes in cancer patients and survivors: a systematic review and meta-analysis. Supportive Care in Cancer, 24(12), 5203-5211.
- [3] Indian Council of Medical Research. (2010). Nutrient Requirements and Recommended Dietary Allowances for Indians. National Institute of Nutrition.
- [4] Krishnan, S., & Coogan, P. F. (2017). Vegetarian diets and cardiovascular risk factors in black members of the Adventist Health Study-2. Public health nutrition, 20(6), 893-903.
- [5] Ramachandran, A., Snehalatha, C., & Krishnaswamy, C. V. (2014). Incidence of diabetes in India. In Textbook of diabetes mellitus (pp. 1281-1296). Jaypee Brothers Medical Publishers.
- [6] Anjana, R. M., Pradeepa, R., Das, A. K., Deepa, M., Bhansali, A., Joshi, S. R., ... & Mohan, V. (2017). Physical activity and inactivity patterns in India—results from the ICMR-INDIAB study (Phase-1) [ICMR-INDIAB-5]. International Journal of Behavioral Nutrition and Physical Activity, 14(1), 1-11.

•