NUTRIPLAN– AN AI ENABLED NUTRITION ANALYZER

A PROJECT REPORT

Submitted by

JANANI A (2116210701085)

JANANI PRIYA N (2116210701086)

KANIMOZHI (2116210701104)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY, CHENNAI

MAY 2024

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Thesis titled **"NUTRIPLAN-AN AI ENABLED NUTRITION ANALYSER**" is the bonafide work of "**JANANI A (2116210701085), JANANI PRIYA (2116210701086),**who carried out the work under my supervision. Certified furtherthat to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred onan earlier occasion on this or any other candidate.

SIGNATURE

Dr . K.Ananth M.E.,Ph.D.,

PROJECT COORDINATOR

Professor

Department of Computer Science and Engineering

Rajalakshmi Engineering College

Chennai - 602 105

Submitted to Project Viva-Voce Examination held on_____

Internal Examiner **External Examine**r

ABSTRACT

This project aims to create an AI-driven Fitness Analyzer using Python's flask framework. The primary objective is to develop an adaptive AI agent capable of autonomously learning and improving the UI for the fitness enthusiasts. Embark on your personalized fitness journey with our AI-enabled Fitness Analyzer, designed for enthusiasts striving towards their wellness goals. Utilizing Python with the Flask framework, our platform seamlessly integrates nutritional data analysis and personalized fitness recommendations, empowering users to regulate and achieve their overall fitness objectives. Through a single-page interface, users access comprehensive reports outlining recommended meals, portion sizes, nutrient breakdowns, and suggested exercise routines. Our intuitive design ensures accessibility and ease of navigation, allowing users to seamlessly integrate our recommendations into their daily lives. With AI-driven analysis, our Fitness Analyzer evolves alongside users, adapting to their progress and preferences over time. Whether a beginner or seasoned fitness enthusiast, our platform provides the guidance and motivation needed to unlock your full potential and achieve lasting fitness success.

ACKNOWLEDGMENT

First, we thank the almighty god for the successful completion of the project. Our sincere thanks to our chairman **Mr. S. Meganathan B.E., F.I.E.,** for his sincere endeavor in educating us in his premier institution. We would like to express our deep gratitude to our beloved Chairperson **Dr. Thangam Meganathan Ph.D.,** for her enthusiastic motivation which inspired us a lot in completing this project and Vice Chairman **Mr. Abhay Shankar Meganathan B.E., M.S.,** for providing us with the requisite infrastructure.

We also express our sincere gratitude to our college Principal, **Dr. S. N. Murugesan M.E., PhD.,** and **Dr. P. KUMAR M.E., PhD, Director computing and information science , and Head Of Department of Computer Science and Engineering** and our project coordinator **Dr. K.Ananthajothi M.E.,Ph.D.,** for her encouragement and guiding us throughout the project towards successful completion of this project and to our parents, friends, all faculty members and supporting staffs for their direct and indirect involvement in successful completion of the project for their encouragement and support.

JANANI A

JANANI PRIYA N

KANIMOZHI A

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In an era where prioritizing health and wellness is paramount, our platform emerges as a beacon of precision and efficacy. Built upon the robust foundation of Python with the Flask framework, our solution offers a meticulously crafted amalgamation of cutting-edge technology and scientific rigor. By adeptly parsing user-provided nutritional data and synthesizing it with advanced algorithms, we deliver bespoke fitness recommendations tailored to individual needs and aspirations. This singular fusion of technology and health science not only facilitates informed decision-making but also fosters a sense of empowerment among users, instilling confidence as they navigate their fitness odyssey. Bid adieu to one-size-fits-all approaches; embrace a paradigm shift towards personalized wellness strategies that evolve synchronously with your aspirations. Welcome to the vanguard of fitness innovation.

By adeptly synthesizing user-provided nutritional data with advanced algorithms, we offer tailored fitness recommendations of unparalleled precision. Within this digital domain, users embark on a transformative journey towards their wellness objectives, guided by meticulously curated insights designed to cater to their unique requirements and ambitions.

Our platform breaks free from generic prescriptions and offers individualized wellness strategies for users. This marks a new era of customized fitness technology that can help users achieve their optimal health and vitality. Embrace this paradigm shift towards tailored excellence and take the first step towards success in your fitness journey.

Liberated from generic approaches, our platform marks a shift towards personalized wellness strategies, poised to empower individuals on their paths to optimal health and vitality. Join us at the forefront of fitness innovation, where every step is a stride towards individualized excellence.

1.1 PROBLEM STATEMENT

The challenge at hand is to develop an AI-enabled fitness analyzer tailored for fitness enthusiasts aiming to regulate their overall fitness goals. In an increasingly health-conscious society, individuals encounter significant hurdles in navigating the myriad of information and recommendations available. The lack of personalized guidance often leads to confusion and frustration, hindering progress towards achieving optimal fitness levels. Leveraging the capabilities of Python with the Flask framework, the objective is to create a robust platform capable of seamlessly gathering nutritional details from users and generating tailored recommendations for diet and exercise. This project aims to address the pressing need for a comprehensive and user-friendly solution that empowers individuals to make informed decisions and successfully navigate their fitness journeys with efficacy and confidence.

1.2 SCOPE OF THE WORK

The AI-enabled fitness analyzer, designed to assist fitness enthusiasts in regulating their overall fitness goals, presents a multifaceted scope targeting both individuals seeking personalized guidance and educational institutions exploring AI integration in wellness management. By leveraging Python with the Flask framework to collect user nutritional data and provide tailored fitness recommendations, the project aims to engage a broad audience, encompassing both fitness enthusiasts and researchers interested in AI applications in health and wellness. Potential extensions, such as incorporating multiplayer functionality and customizable AI coaching, enhance the platform's scalability and appeal to diverse user demographics. Despite challenges related to computational resources and algorithm refinement, the adaptability and simplicity of the platform offer promising avenues for future enhancements, positioning it as a valuable tool in the realm of personalized fitness management.

1.3 AIM AND OBJECTIVES OF THE PROJECT

This project aims to develop an AI-driven Fitness Analyzer using Python's Flask framework, with the primary objective of creating an adaptive AI agent capable of autonomously learning and improving the user interface for fitness enthusiasts. Our goal is to provide personalized and seamless fitness guidance to users, empowering them to achieve their wellness goals effectively.

To fulfill this aim, the project delineates several key objectives. Firstly, we aim to establish a robust AI-driven platform that seamlessly integrates nutritional data analysis and personalized fitness recommendations. This involves utilizing Python with the Flask framework to develop a single-page interface where users can access comprehensive reports outlining recommended meals, portion sizes, nutrient breakdowns, and suggested exercise routines.

Additionally, our objective is to design an intuitive user interface that ensures accessibility and ease of navigation, allowing users to seamlessly integrate our recommendations into their daily lives. Through iterative learning mechanisms, our Fitness Analyzer will evolve alongside users, adapting to their progress and preferences over time.

1.4 RESOURCES

This project has been developed through widespread secondary research of accredited manuscripts, standard papers, business journals, white papers, analysts' information, and conference reviews. Significant resources are required to achieve an efficacious completion of this project.

The following prospectus details a list of resources that will play a primary role in the successful execution of our project:

- A properly functioning workstation (PC, laptop, net-books etc.) to carry out desired research and collect relevant content.
    - Unlimited internet access.
- Unrestricted access to the university lab in order to gather a variety of literature including academic resources (for e.g. publications,sciencedirect,journals etc.), technical manuscripts, etc.Visual Studio code is required to run the all the files and applications.

1.5 MOTIVATION

The project's primary motivation stems from the exploration of evolutionary algorithms within the domain of fitness management, utilizing Python with the Flask framework. By employing advanced algorithms in the development of an AI fitness analyzer, the project aims to demonstrate the system's potential to learn, adapt, and refine its recommendations over time. This offers a compelling glimpse into how networks can autonomously enhance their fitness guidance capabilities without explicit programming. Leveraging the Flask framework, the project endeavors to provide a seamless user experience, allowing fitness enthusiasts to input their nutritional details effortlessly and receive personalized fitness recommendations promptly.

Furthermore, by integrating user-friendly interfaces and interactive features, the AI fitness analyzer aims to empower individuals to make informed decisions and take control of their health and wellness journey with confidence. Through collaborative efforts and meticulous development, the project aspires to contribute to the advancement of AI-driven solutions in promoting holistic well-being.

CHAPTER 2

2.1 LITRETURE SURVEY

(Zhou et al., 2012)This study examined how dietary vitamin C levels affect juvenile cobia's growth, immune responses, and health. Fish on the basal diet, lacking vitamin C, showed signs of deficiency and poorer growth compared to those on supplemented diets. Survival rates correlated with vitamin C levels, with higher rates in supplemented groups. Liver vitamin C levels rose with dietary intake, but liver health, indicated by TBARS, wasn't affected. Immune markers like lysozyme and SOD were higher in supplemented groups. Glucose, triglycerides, and hematologic parameters improved with vitamin C. Survival against Vibrio harveyi increased with vitamin C up to 96.6 mg/kg, plateauing after. These results highlight the crucial role of dietary vitamin C in cobia's growth, immunity, and survival.

(W. J. MAENG and R. L. BALDWIN et al., 2019)In a study conducted on a fistulated Jersey cow fed a purified diet with urea as the sole nitrogen source, ruminal fermentation dynamics were elucidated. With an estimated rumen volume of 59.8 liters, turnover time and rate of passage of rumen contents were calculated at 33.4 hours and 1.8 liters per hour, respectively. Glucose, starch, and cellulose exhibited distinct turnover and fermentation times, with glucose disappearing rapidly within an hour, while starch and cellulose displayed longer durations of 4.7 and 14.2 hours, respectively.

(Lucy et al., 1917)This study reviewed probiotics for dairy calves using data from 90 manuscripts (97 trials) retrieved from five electronic databases up to October 2018. Most studies were conducted in Asia between 2008 and 2018. Lactobacillus spp., Bacillus spp., and Saccharomyces spp. were the most evaluated genera. Probiotics were administered to calves via various methods depending on age, with a focus on preweaned calves. Despite consistent evaluation of calf performance, health outcomes varied widely across studies. Standardized health measurement guidelines are needed

for future research. Funding was provided by CDFA–AUS project.

This study investigates how conjugated linoleic acid (CLA) (Lin et al., 2024)affects blood lipids in obese rats. Administering CLA to rats on a high-fat diet led to reduced body weight and fat deposition. Metabolomic analysis highlighted changes in the arachidonic acid pathway, showing key biomarkers linked to blood lipid reduction. Further analysis revealed that CLA influences lipid metabolism through the ARA-Cox/Lox-PGE2-Pparγ pathway. This research provides valuable insights into treating obesity and dyslipidemia.

This study(Ferreira-Santos et al., 2020) aimed to assess the impact of a lycopene-supplemented diet on fructose-induced metabolic syndrome in male Wistar rats. Rats consuming a normal diet with 20% fructose (F) experienced increased blood pressure, cardiac hypertrophy, endothelial dysfunction, and metabolic disturbances. Those concurrently treated with 0.01% lycopene (FL) showed significant attenuation of hypertension, endothelial dysfunction, and cardiac hypertrophy. Lycopene treatment did not affect rats on a standard diet (C, L groups). Additionally, lycopene improved insulin resistance, dyslipidemia, liver enlargement, intraperitoneal fat accumulation, and oxidative stress. These findings suggest that lycopene may effectively mitigate the pathophysiological effects of fructose-induced metabolic syndrome.

Examining the influence of fish oil supplementation on postpartum dairy cows' adipose tissue, this research(Elis et al., 2016) investigated lipid profiles and gene expression. Holstein cows were divided into groups, one receiving long-chain n-3 polyunsaturated fatty acids (PUFA) and the other control PUFA post-calving. Despite consistent milk production and metabolic parameters, supplemented cows exhibited decreased early embryo mortality rates. Adipose tissue analysis revealed alterations in lipid composition, including increased long-chain PUFA from fish oil. Additionally, supplementation led to reduced triacylglycerolipids levels. Gene expression analysis suggested upregulation of genes involved in fatty acid metabolism and adipokine function, hinting at enhanced lipolysis and altered secretory functions in adipose tissue. These findings illuminate the potential reproductive and metabolic benefits of n-3 PUFA supplementation in postpartum dairy cows.

In this investigation(Grainger et al., 2009), the focus was on understanding how diet type and energy intake level influence the performance of cows undergoing extended lactations. Ninety-six Holstein-Friesian cows calving in July and August 2004 were randomly assigned to eight groups, each comprising 12 cows (including four primiparous cows). The four treatments varied in lactation length (300 or 670 days) and

diet (control, high, or full total mixed ration (TMR)). Results showed that increasing energy level in the diet resulted in similar milk yield and fat concentration but greater yields of milk fat and protein, as well as higher milk protein percentage. The full TMR group exhibited greater yields of milk and milk components compared to other groups. Milk solids ratio was not affected by increasing grain in pasture-based diets but decreased with the TMR diet. The study suggests that Holstein cows with a high proportion of Northern Hemisphere genes offered pasture-based diets could achieve favorable milk solids ratios and lower body weight gain over extended lactation periods.

Patient satisfaction(Chen et al., 2024) is crucial in healthcare, and integrating artificial intelligence (AI) into medical processes is a growing trend. Traditional Chinese Medicine (TCM) examinations are renowned for their non-invasive nature and positive impact on patient satisfaction. This study explores the development and effectiveness of an intelligent physical examination system that merges TCM with Western medicine. By integrating expert input and comparing patient satisfaction levels between the intelligent and traditional examination methods, the study evaluates the system's efficacy in enhancing patient satisfaction, particularly among those with chronic diseases. Results indicate significant improvements in satisfaction levels with the intelligent system, underscoring its potential to optimize healthcare experiences.

In this investigation, the focus (Aldian et al., 2023)was on unraveling the physiological mechanisms behind the enhancement of physiological stress tolerance in ruminants fed diverse forage diets. Six crossbred Shiba wethers were subjected to different forage regimens, and markers of oxidative stress were quantified alongside serum metabolomics. Results indicated improved intake and digestibility in the group fed a mixture of Sudan grass, timothy grass, and alfalfa hay (SDN-TMT-ALF). Total antioxidant capacity (TAC) was higher in the groups receiving mixed forage diets, while superoxide dismutase (SOD) levels remained consistent. Serum metabolite analysis revealed alterations in metabolites related to linoleic acid (LA) metabolism, steroid biosynthesis, and phenolic compounds. Pathway analysis suggested upregulation of glycolysis metabolism and LA metabolism, correlating with increased TAC levels and altered serum metabolites. Overall, the findings shed light on the intricate mechanisms influencing physiological stress tolerance in ruminants fed diverse forage diets.

The nutrient composition of milk from Cambodian mothers,(Whitfield et al., 2020) especially fat-soluble and water-soluble vitamins, is poorly understood despite low thiamin content. This study aimed to analyze these nutrients in Cambodian mothers' milk (n = 68) and explore their interrelationships. Thiamin fortification increased thiamin levels but decreased γ-tocopherol, with <40% of samples meeting Adequate Intake (AI) values for all vitamins. Macronutrients and vitamins exhibited correlations, yet thiamin pyrophosphate (TPP) was associated with other water-soluble vitamins

rather than B-1 vitamins. Strong correlations were observed among fat-soluble vitamins, with TPP, riboflavin (B-2), and nicotinamide (B-3) associating with nearly all fat-soluble vitamins. Further investigation into influential factors affecting nutrient composition is warranted.

2.2 PROPOSED SYSTEM

The proposed nutrition analyzer app is designed to offer users a holistic solution for managing their diet and health goals. It will start by allowing users to create a profile with details such as weight, height, age, gender, and any existing medical conditions. Users can also set their weight goals, whether it's to gain, lose, or maintain weight, along with the desired timeframe.Once the profile is set up, the app will provide personalized diet recommendations tailored to the user's profile and goals. This will include suggestions for daily calorie intake, macronutrient distribution (carbs, proteins, fats), and meal plans. The app will take into account any dietary restrictions or preferences that the user may have.

Users will be able to track their daily calorie intake and compare it against the recommended intake based on their goals. The app will also offer a variety of healthy recipes based on the user's preferences and goals, along with nutritional information for each recipe.A comprehensive food database will be available for users to search for foods and view their nutritional values. They can add these foods to their daily intake, which will help in planning their meals for the week. The app will generate a grocery list based on the planned meals, making it easier for users to shop for the ingredients they need.Progress tracking is a key feature of the app, allowing users to track their progress towards their weight goals. Visualizations such as graphs will show changes in weight over time, providing motivation and a sense of achievement.Integration with the Olive app will enhance the user experience by providing additional recipe suggestions and allowing users to view recipes seamlessly. This integration will leverage Olive's features to enrich the app's content and functionality.

2.3 FLASK FRAMEWORK

The algorithm developed to process user data and generate personalized fitness plans is seamlessly integrated into the backend of the Flask web application. This integration ensures that the algorithm is readily accessible and can be invoked based on user interactions with the frontend interface. Flask provides a mechanism to define API endpoints that map specific URLs to functions within the application. These endpoints serve as the interface through which the frontend communicates with the backend. For example, when a user submits their fitness goals and dietary preferences through a form on the website, the data is sent to a designated API endpoint in the Flask application for processing.Upon receiving a request from the frontend, Flask routes the request to the appropriate function or view within the application. This function then utilizes the integrated algorithm to process the user data, generate fitness recommendations, and formulate a response.After processing the user input using the algorithm, the Flask application generates a response, typically in the form of JSON data or HTML content. This response may include personalized fitness plans, dietary recommendations, exercise routines, progress reports, or other relevant information based on the user's input. The frontend of the web application, developed using HTML, CSS, and JavaScript, communicates with the Flask backend through asynchronous HTTP requests. This enables dynamic interaction with the application without requiring the entire page to reload, enhancing user experience and responsiveness.

2.4 Machine Learning Algorithm

At the heart of the fitness analyzer lies the development of advanced algorithms designed to process user data and generate personalized fitness plans. Leveraging Python libraries and machine learning techniques, such as regression and classification algorithms, the system analyzes the wealth of user inputs to discern underlying correlations, patterns, and dependencies. By identifying key relationships between demographic factors, health indicators, dietary habits, and fitness goals, the algorithm can derive actionable insights to inform optimal recommendations. This involves sophisticated data processing techniques to prioritize factors influencing fitness outcomes and adjust recommendations dynamically based on user feedback and progress tracking. Through iterative refinement and continuous learning, the algorithm evolves to deliver increasingly accurate and relevant fitness guidance tailored to each user's unique circumstances.

CHAPTER 3

SYSTEM DESIGN

3.1 GENERAL

In this section, we would like to show how the general outline of how all the components end up working when organized and arranged together. It is further represented in the form of a flow chart below.

3.2 SYSTEM ARCHITECTURE DIAGRAM

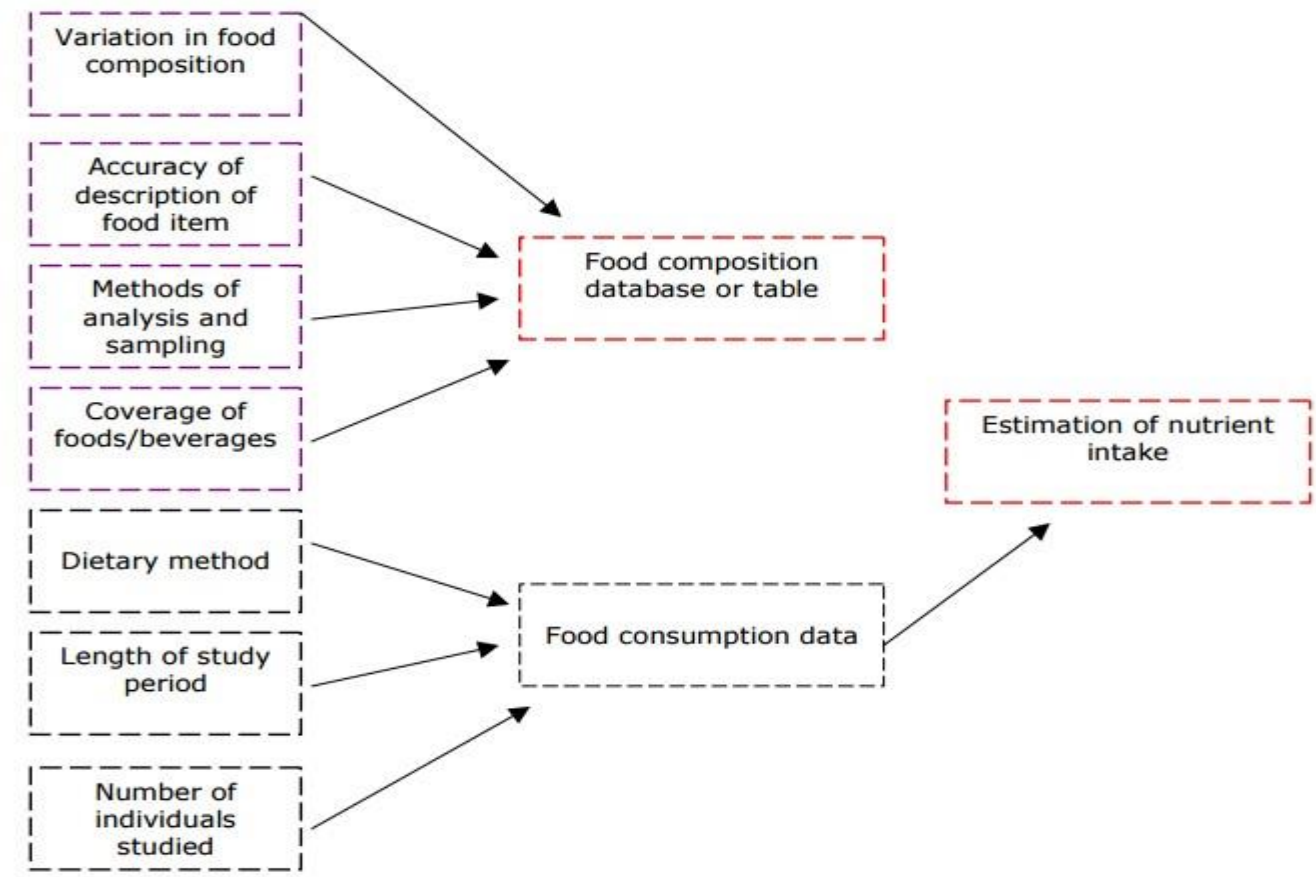


Fig 3.1: System Architecture

3.3 DEVELOPMENTAL ENVIRONMENT

3.3.1 HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the system's implementation. It should therefore be a complete and consistent specification of the entire system. It is generally used by software engineers as the starting point for the system design.

Table 3.1 Hardware Requirements

COMPONENTS	SPECIFICATION
PROCESSOR	Intel Core i3
RAM	8 GB RAM
GPU	NVIDIA GeForce GTX 1650
MONITOR	14" COLOR
HARD DISK	512 GB
PROCESSOR SPEED	MINIMUM 1.1 GHz

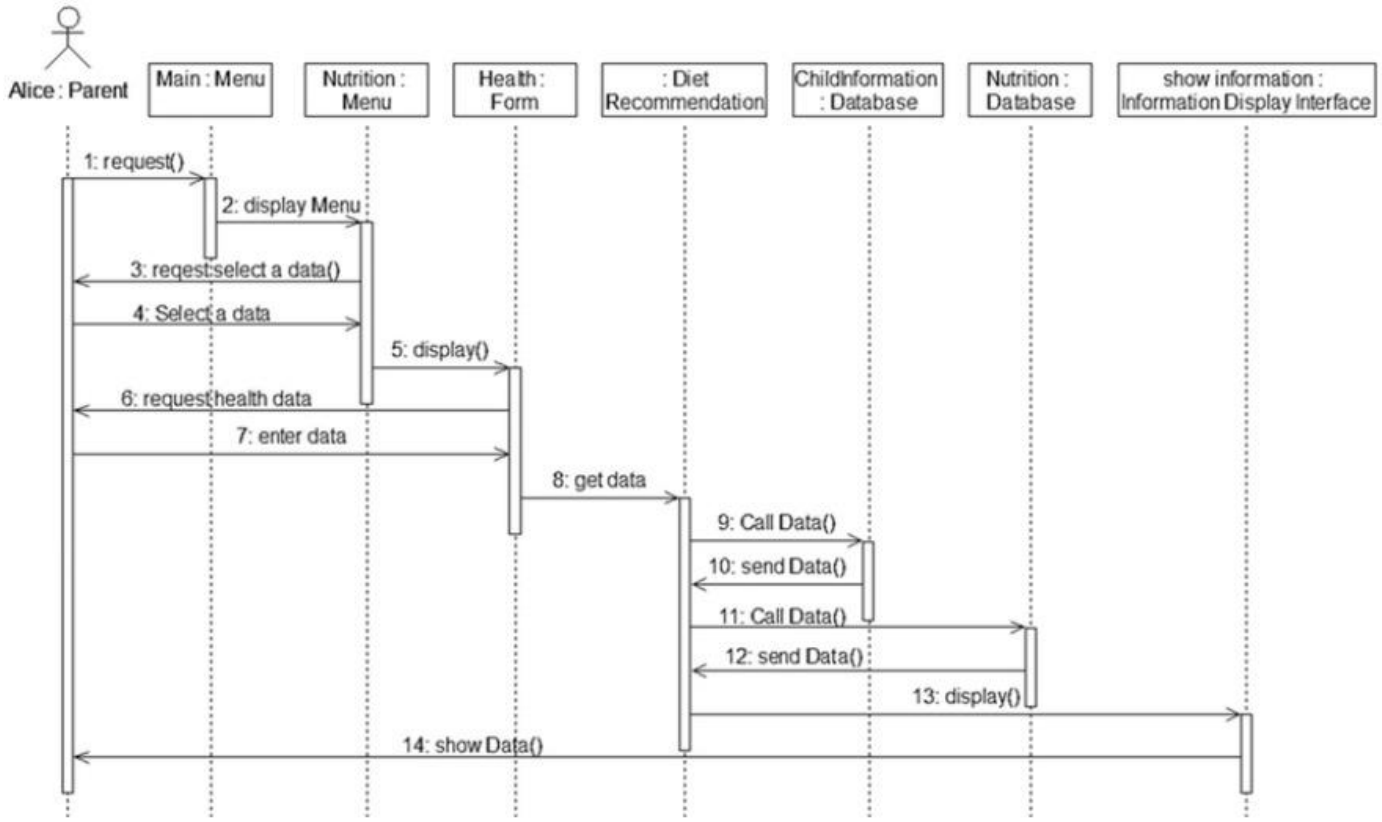
3.3.2 SOFTWARE REQUIREMENTS

The software requirements document is the specifications of the system. It should include both a definition and a specification of requirements. It is aset of what the system should rather be doing than focus on how it should be done. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating the cost, planning team activities, performing tasks, tracking the team, and tracking the team's progressthroughout the development activity.

Python IDLE, Visual Studio code and **chrome** would all be required.

3.4 DESIGN OF THE ENTIRE SYSTEM

3.4.1 SEQUENCE DIAGRAM



The sequence diagram illustrates the interaction between the user and the fitness analyzer system where the user selects fitness goal and inputs nutritional details.Next the system processes data and generates personalized fitness recommendations. At last the recommendations displayed to the user for diet and exercise.

CHAPTER 4

PROJECT DESCRIPTION

4.1 METHODOLODGY

The first phase involves gathering comprehensive data from users to inform personalized fitness recommendations. Utilizing the Flask framework in Python, the system collects user inputs encompassing demographic information, health status, fitness goals, dietary habits, and exercise preferences. Through intuitive interfaces, users provide details on their nutritional intake, physical activity levels, and any existing medical conditions, ensuring a holistic understanding of their fitness profile.The core of the fitness analyzer lies in the development of sophisticated algorithms capable of processing user data and generating tailored fitness plans. Leveraging Python libraries and machine learning techniques, including regression and classification algorithms, the system analyzes user inputs to identify correlations, patterns, and dependencies. This enables the algorithm to derive insights regarding optimal dietary recommendations, exercise regimens, and lifestyle modifications aligned with individual fitness objectives.

Integration of the algorithm within the Flask framework facilitates seamless communication between the user interface and backend processing. Python's Flask framework enables the development of web-based applications, allowing users to access the fitness analyzer through browsers across various devices. The system architecture ensures scalability, responsiveness, and real-time data processing, enhancing user experience and engagement.

Validation and testing constitute critical phases to ensure the accuracy, reliability, and effectiveness of the fitness analyzer. Rigorous testing protocols, including unit tests, integration tests, and user acceptance testing, validate the functionality and performance of the system across diverse scenarios. Additionally, user feedback and iterative refinement further enhance the system's usability, ensuring alignment with user expectations and fitness goals.

The methodology outlined above delineates a systematic approach to developing an AI-enabled fitness analyzer. By integrating data collection, algorithm development, system integration, and validation, the project aims to deliver a robust and user-centric fitness tool capable of empowering individuals to achieve their fitness aspirations effectively.

4.2 MODULE DESCRIPTION

The development of the AI-driven Fitness Analyzer encompasses several key modules, each playing a crucial role in achieving the overarching goal of providing personalized fitness guidance to users.

4.2.1 User Interface Module:

This module focuses on creating an intuitive and user-friendly interface for the Fitness Analyzer. It includes the design and implementation of a single-page interface using Python's Flask framework, allowing users to access comprehensive reports, recommended meals, portion sizes, nutrient breakdowns, and exercise routines.

4.2.2 Nutritional Data Analysis Module:

This module is responsible for integrating nutritional data analysis capabilities into the platform. It involves utilizing Python libraries to analyze user input and provide personalized fitness recommendations based on nutritional data, dietary preferences, and wellness goals.

4.2.3 AI Agent Module:

The AI Agent module is at the core of the Fitness Analyzer, housing the adaptive AI agent responsible for autonomously learning and improving the user interface. This module leverages machine learning algorithms to analyze user interactions and feedback, continuously enhancing the user experience over time.

4.2.4 Evolutionary Learning Module:

This module focuses on implementing evolutionary learning mechanisms to ensure that the AI agent evolves alongside users. It involves training the AI agent through iterative processes, adjusting parameters based on user feedback, and refining its recommendations to better align with user preferences and progress.

4.3 SOURCE CODE

SOURCE CODE:

```
import os
import json
from flask import Flask
from flask import render_template
from flask import request
from flask import jsonify


app = Flask(__name__)

@app.route("/")
def hello():
    message = "Hello World!"
    return render_template('index.html', message=message)

@app.route("/index.html")
def index():
    message = "Welcome to index.html!"
    return render_template('index.html', message=message)

@app.route("/history.html")
def history():
    message = "Welcome to history.html!"
    return render_template('history.html', message=message)

@app.route("/track.html")
def track():
    message = "Welcome to track.html!"
    return render_template('track.html', message=message)

@app.route("/preference.html")
def preference():
    message = "Welcome to preference.html!"
    return render_template('preference.html', message=message)
```

```
@app.route("/Recipes.html")
def Recipes():
    message = "Welcome to Recipes.html!"
    return render_template('Recipes.html', message=message)

@app.route("/food-database", methods = ['GET', 'POST'])
def foodDatabase():
    if request.method == 'GET':
        json_file = os.path.join(app.root_path, 'food_data', 'localFoods.json')

        with open(json_file) as file:
            data = json.load(file)
            return jsonify(data)

    elif request.method == 'POST':
        json_file = os.path.join(app.root_path, 'food_data', 'localFoods.json')
        if not os.path.isfile(json_file):
            with open(json_file, 'w') as file:
                json.dump([], file)

        stored_data = None
        received_data = None

        with open(json_file, 'r') as file:
            stored_data = json.load(file)  # in python list


        received_data = request.form.to_dict()  # in python dict
        received_data["tags"] = json.loads(received_data["tags"])


        if not any(temp["name"] == received_data["name"] for temp in stored_data):
          stored_data.append(received_data.copy())


        json_stored_data = json.dumps(stored_data)
        with open(json_file, 'w') as file:
            file.seek(0)
            file.write(json_stored_data)
            file.truncate()

        return json_stored_data
```

```
@app.route("/food-log", methods = ['GET', 'POST', 'DELETE'])
def foodLog():
    user = request.args.get('user', default = 'test', type = str)
    if request.method == 'GET':
        with open('user_data/{}_log.txt'.format(user)) as file:
            data = json.load(file)
            return jsonify(data)

    elif request.method == 'POST' or request.method == 'DELETE':
        json_file = os.path.join(app.root_path, 'user_data', '{}_log.txt'.format(user))
        if not os.path.isfile(json_file):
            with open(json_file, 'w') as file:
                try:
                    json.dump({"user": user}, file)
                except:
                    print("User doesn't exist and couldn't create user profile!")

        with open('user_data/{}_log.txt'.format(user), 'r+') as file:
            try:
                stored_data = json.load(file)
                received_data = request.form.to_dict()
                processFoodData(received_data, stored_data, request.method)
                json_data = json.dumps(stored_data)
                file.seek(0)
                file.write(json_data)
                file.truncate()
                return json_data
            except Exception as e:
                print("Couldn't write to user data!:", e)
                return "ERROR"

def processFoodData(received_data, stored_data, method):
    #Make sure required fields are present
    if method == 'POST' and not all(keys in received_data for keys in ["name", "meal", "date",
"user", "serving", "calories", "carbohydrates", "proteins", "fats"]):
        raise Exception("not all required data was present in received_data")

    elif method == 'DELETE' and not all(keys in received_data for keys in ["name", "meal",
"date", "user"]):
        raise Exception("not all required data was present in received_data")
```

```
    if received_data["name"] != "":

        if "tags" in received_data:
            received_data["tags"] = json.loads(received_data["tags"])
        remove = ["meal", "date", "user"]
        food_temp = {x: received_data[x] for x in received_data if x not in remove}
        if received_data["date"] in stored_data:
            if received_data["meal"] in stored_data[received_data["date"]]:
                if method == 'POST':
                    stored_data[received_data["date"]][received_data["meal"]].append(food_temp)
                    print("Received:", received_data)
                elif method == 'DELETE':

         for index in range(len(stored_data[received_data["date"]][received_data["meal"]])):
                        if(stored_data[received_data["date"]][received_data["meal"]][index]["name"]
== received_data["name"]):
                            print("Deleted", received_data["name"], "at", index )
                            stored_data[received_data["date"]][received_data["meal"]].pop(index)
                            break
            else:
                stored_data[received_data["date"]] = {"Breakfast": [], "Lunch": [], "Dinner": []}
                stored_data[received_data["date"]][received_data["meal"]].append(food_temp)
        else:
            stored_data[received_data["date"]] = {"Breakfast": [], "Lunch": [], "Dinner": []}
            stored_data[received_data["date"]][received_data["meal"]].append(food_temp)



@app.route("/food-pref", methods = ['GET', 'POST', 'DELETE'])
def foodPref():
    user = request.args.get('user', default = 'test', type = str)
    if request.method == 'GET':
        json_file = os.path.join(app.root_path, 'user_data', '{}_pref.txt'.format(user))
        if not os.path.isfile(json_file):
            with open(json_file, 'w') as file:
                json.dump({"user": "test", "plan": None}, file)

        with open('user_data/{}_pref.txt'.format(user)) as file:
            data = json.load(file)
            return jsonify(data)
```

```
    elif request.method == 'POST' or request.method == 'DELETE':
        json_file = os.path.join(app.root_path, 'user_data', '{}_pref.txt'.format(user))

        if not os.path.isfile(json_file):
            with open(json_file, 'w') as file:
                json.dump({"user": "test", "plan": None}, file)

        with open('user_data/{}_pref.txt'.format(user), 'r+') as file:
            stored_data = json.load(file)
            received_data = request.form.to_dict()
            print(received_data)
            stored_data["plan"] = received_data.copy()
            json_data = json.dumps(stored_data)
            file.seek(0)
            file.write(json_data)
            file.truncate()
            return json_data


@app.route("/food-dislike", methods = ['GET', 'POST', 'DELETE'])
def foodDislike():
    user = request.args.get('user', default = 'test', type = str)
    if request.method == 'GET':
        json_file = os.path.join(app.root_path, 'user_data', '{}_dislike.txt'.format(user))
        if not os.path.isfile(json_file):
            with open(json_file, 'w') as file:
                json.dump({"user": "test", "dislike": []}, file)

        with open('user_data/{}_dislike.txt'.format(user)) as file:
            data = json.load(file)
            return jsonify(data)

    elif request.method == 'POST' or request.method == 'DELETE':
        json_file = os.path.join(app.root_path, 'user_data', '{}_dislike.txt'.format(user))
        if not os.path.isfile(json_file):

            with open(json_file, 'w') as file:
                json.dump({"user": "test", "dislike": []}, file)

        with open('user_data/{}_dislike.txt'.format(user), 'r+') as file:
            stored_data = json.load(file)
            received_data = request.form.to_dict()
```

```
            print(received_data)
            stored_data["dislike"].append(received_data["dislike"])
            json_data = json.dumps(stored_data)
            file.seek(0)
            file.write(json_data)
            file.truncate()
            return json_data



@app.route("/food-tag-query", methods = ['POST'])
def foodTagQuery():
    if request.method == 'POST':
        json_file = os.path.join(app.root_path, 'food_data', 'localFoods.json')

        with open(json_file, 'r') as file:
            stored_data = json.load(file)
            received_data = request.form.to_dict()
            print(received_data)
            target_nutrient = received_data["nutrient"]
            target_condition = received_data["condition"]
            foods_qualified = []

            for food_dict in stored_data:
                if target_nutrient == "proteins" and target_condition == "high":
                    if ("High Proteins" in food_dict["tags"]):
                        foods_qualified.append(food_dict.copy())
                        print(foods_qualified)
                elif target_nutrient == "proteins" and target_condition == "low":
                    if ("Low Proteins" in food_dict["tags"]):
                        foods_qualified.append(food_dict.copy())
                        print(foods_qualified)

                elif target_nutrient == "carbohydrates" and target_condition == "high":
                    if ("High Carbohydrates" in food_dict["tags"]):
                        foods_qualified.append(food_dict.copy())
                        print(foods_qualified)
                elif target_nutrient == "carbohydrates" and target_condition == "low":
                    if ("Low Carbohydrates" in food_dict["tags"]):
                        foods_qualified.append(food_dict.copy())
                        print(foods_qualified)
```

```
            elif target_nutrient == "fats" and target_condition == "high":
                if ("High Fats" in food_dict["tags"]):
                    foods_qualified.append(food_dict.copy())
                    print(foods_qualified)
            elif target_nutrient == "fats" and target_condition == "low":
                if ("Low Fats" in food_dict["tags"]):
                    foods_qualified.append(food_dict.copy())
                    print(foods_qualified)

        json_foods_qualified = json.dumps({"selected_foods": foods_qualified})
        print(json_foods_qualified)
        return json_foods_qualified


if __name__ == "__main__":
    app.debug = True

    app.run(host='0.0.0.0', port=8888)
```

CHAPTER 5

RESULTS AND DISCUSSIONS

5.1 OUTPUT

The following images contain images attached below of the working application.

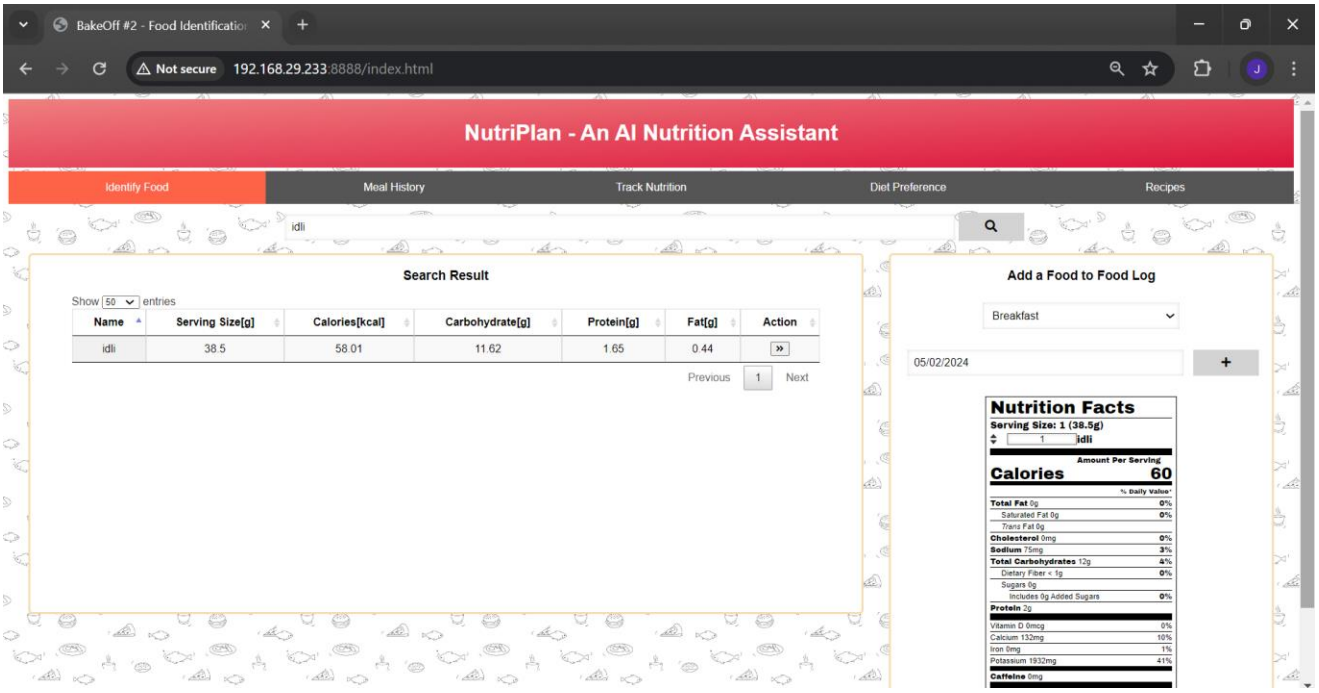


Fig 5.1: index page

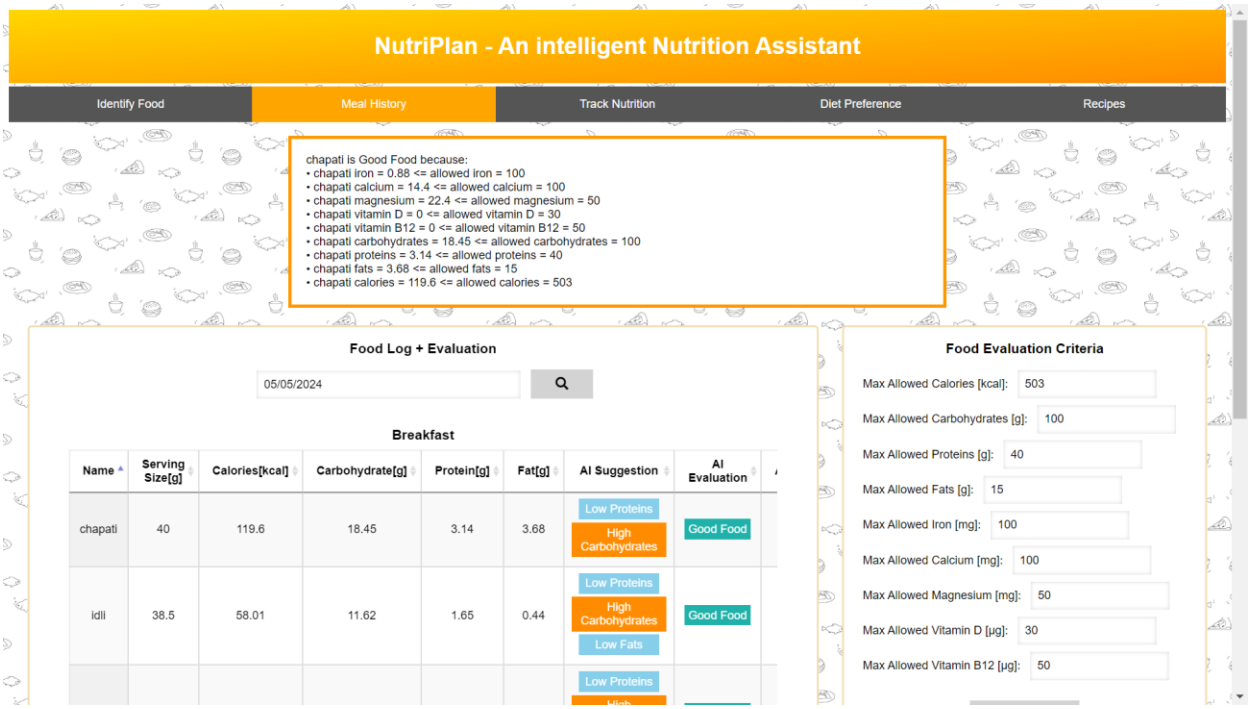


Fig 5.2 MEAL HISTORY

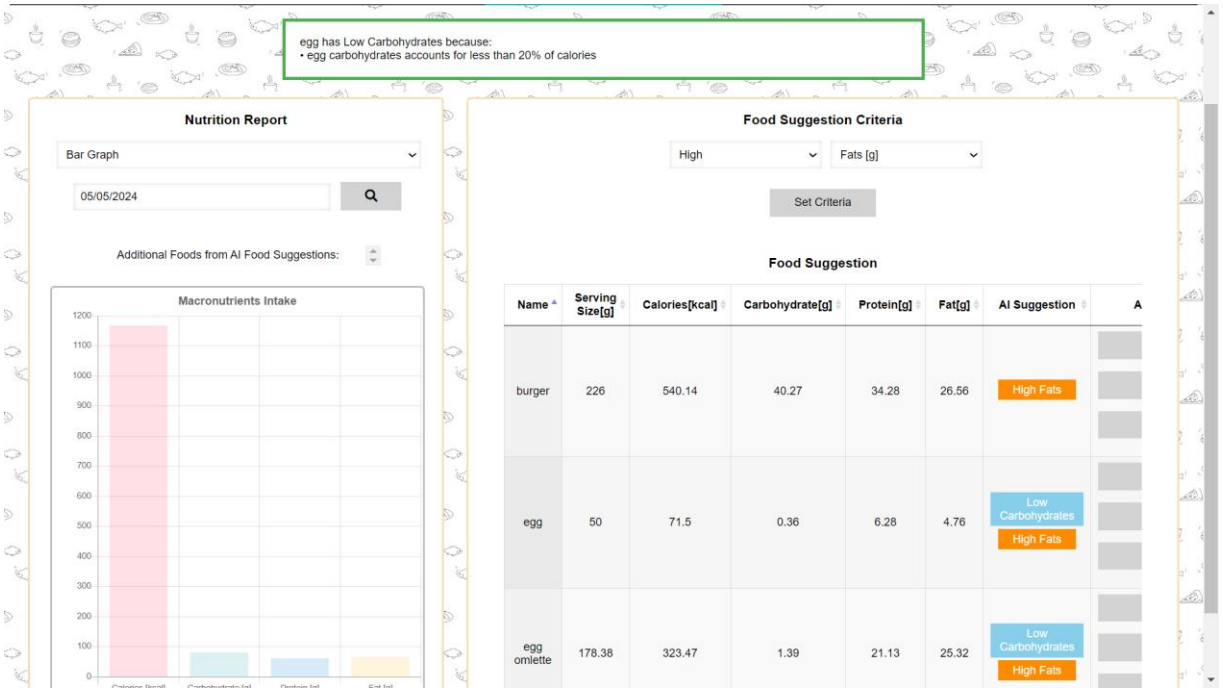


Fig 5.3 TRACK NUTRITION

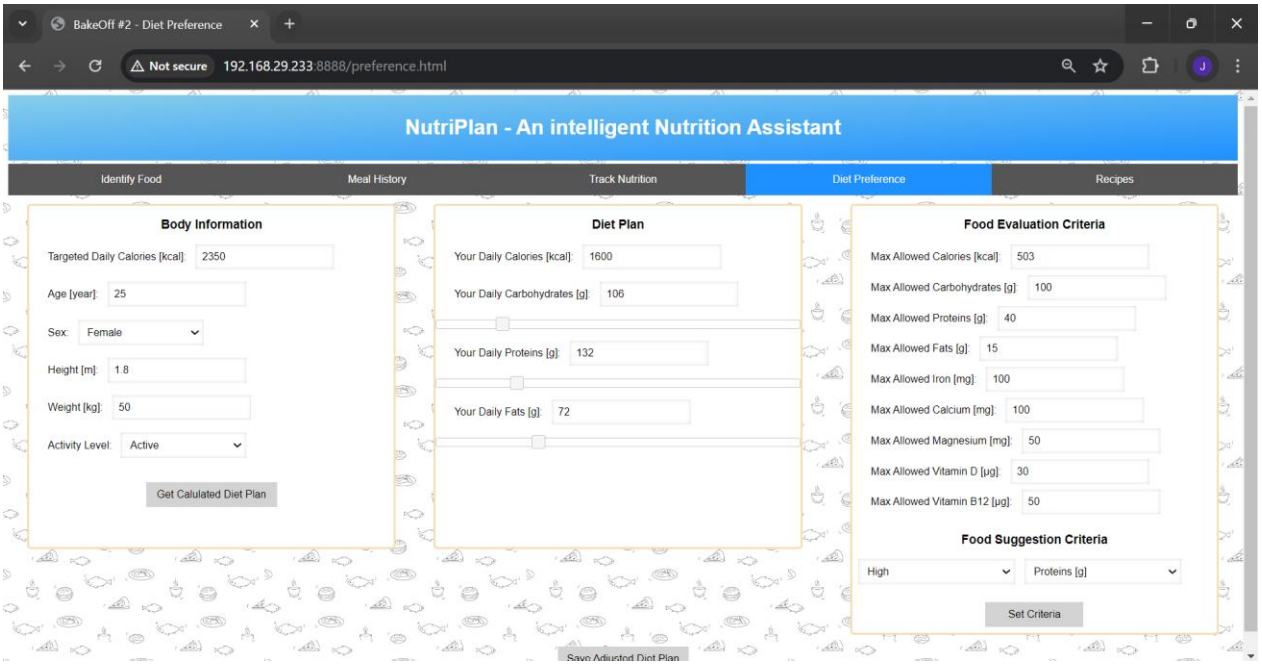


Fig 5.4 DIET PREFERENCES

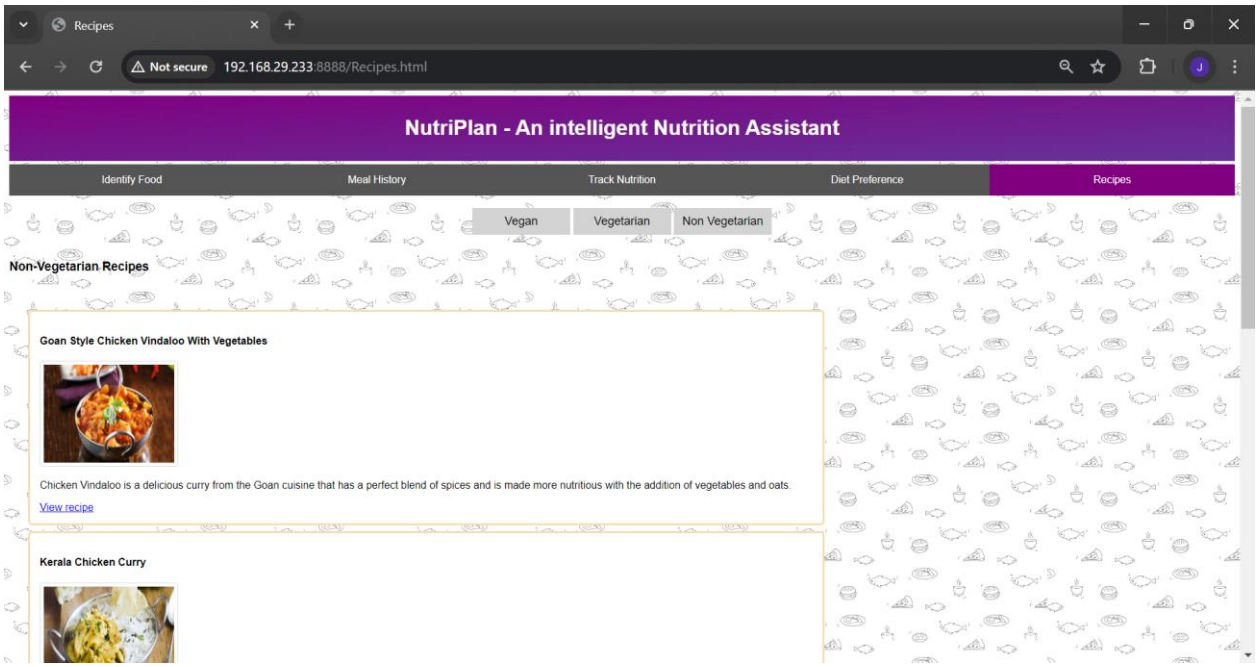


Fig 5.5 RECIPE

5.2 RESULT

The development of an AI-driven Fitness Analyzer using Python's Flask framework represents a significant advancement in personalized fitness technology. This project focuses on creating an adaptive AI agent capable of autonomously learning and enhancing the user interface for fitness enthusiasts. Our AI-enabled Fitness Analyzer is designed to accompany individuals on their unique wellness journeys, leveraging Python and Flask to seamlessly integrate nutritional data analysis and personalized fitness recommendations. Through a user-friendly single-page interface, our platform offers comprehensive reports featuring recommended meals, portion sizes, nutrient breakdowns, and suggested exercise routines. Our intuitive design ensures accessibility and ease of navigation, enabling users to effortlessly incorporate our recommendations into their daily routines. One of the key strengths of our Fitness Analyzer is its AI-driven analysis, allowing it to evolve alongside users by adapting to their progress and preferences over time. Whether users are beginners or seasoned fitness enthusiasts, our platform provides the guidance and motivation needed to unlock their full potential and achieve lasting success in their fitness endeavors.

In summary, this project exemplifies the synergy between artificial intelligence and fitness, offering a powerful tool to support individuals in reaching their fitness goals and optimizing their overall well-being. Through continuous refinement and adaptation, our AI-driven Fitness Analyzer sets the stage for further advancements in personalized fitness technology.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

In conclusion, the development of an AI-driven Fitness Analyzer utilizing Python's Flask framework represents a significant step forward in personalized fitness technology. By seamlessly integrating nutritional data analysis and personalized fitness recommendations, this platform empowers users to take control of their wellness journey. Through a user-friendly interface, comprehensive reports outline tailored meal plans, nutrient breakdowns, and exercise routines, making it easy for individuals to incorporate healthier habits into their daily lives.

The adaptive nature of the AI agent ensures that the platform evolves alongside users, continually learning and improving based on their progress and preferences. Whether someone is just starting their fitness journey or is a seasoned enthusiast, the Fitness Analyzer provides the guidance and motivation needed to achieve lasting success. Overall, this project embodies the intersection of technology and wellness, offering a powerful tool to support individuals in reaching their fitness goals and optimizing their overall well-being.

6.2 FUTURE ENHANCEMENT

A potential future enhancement for a AI-enabled nutrition analyer “NUTRIPLAN” could involve incorporating more advanced AI techniques or expanding the features to create a more user friendly and immersive experiance. Here's an idea for a future enhancement.

Dynamic Difficulty Adjustment:

Implement a system that dynamically adjusts the application difficulty based on the user demands. This enhancement could involve:

1.Advanced Machine Learning Algorithms: Incorporate more sophisticated machine learning algorithms to enhance the AI's ability to analyze user data and provide even more personalized recommendations.

2.Integration with Wearable Devices: Allow users to sync their wearable fitness trackers, such as smartwatches or fitness bands, to provide real-time data input. This integration could provide more accurate information for the AI to analyze and tailor recommendations accordingly.

3.Expanded Nutritional Database: Continuously update and expand the nutritional database to include a wider range of foods and ingredients, as well as their nutrient profiles. This ensures that the AI can provide more accurate and comprehensive meal recommendations tailored to individual dietary preferences and restrictions.

4.Voice Assistant Integration: Enable voice assistant integration (e.g., Siri, Google Assistant) to provide hands-free access to the Fitness Analyzer, allowing users to get recommendations and track their progress using voice commands.

REFERENCES

1. Aldian, D., Harisa, L. D., Mitsuishi, H., Tian, K., Iwasawa, A., & Yayota, M. (2023). Diverse forage improves lipid metabolism and antioxidant capacity in goats, as revealed by metabolomics. *Animal*, *17*(10). https://doi.org/10.1016/j.animal.2023.100981
2. Chen, X., Duan, R., Shen, Y., & Jiang, H. (2024). Design and evaluation of an intelligent physical examination system in improving the satisfaction of patients with chronic disease. *Heliyon*, *10*(1). https://doi.org/10.1016/j.heliyon.2023.e23906
3. Elis, S., Desmarchais, A., Freret, S., Maillard, V., Labas, V., Cognié, J., Briant, E., Hivelin, C., Dupont, J., & Uzbekova, S. (2016). Effect of a long-chain n-3 polyunsaturated fatty acid–enriched diet on adipose tissue lipid profiles and gene expression in Holstein dairy cows. *Journal of Dairy Science*, *99*(12), 10109–10127. https://doi.org/10.3168/jds.2016-11052
4. Ferreira-Santos, P., Aparicio, R., Carrón, R., Montero, M. J., & Sevilla, M. Á. (2020). Lycopene-supplemented diet ameliorates metabolic syndrome induced by fructose in rats. *Journal of Functional Foods*, *73*. https://doi.org/10.1016/j.jff.2020.104098
5. Grainger, C., Auldist, M. J., O'Brien, G., Macmillan, K. L., & Culley, C. (2009). Effect of type of diet and energy intake on milk production of Holstein-Friesian cows with extended lactations. *Journal of Dairy Science*, *92*(4), 1479–1492. https://doi.org/10.3168/jds.2008-1530
6. Lin, D., Fu, X., Li, B., Huo, Y., Xie, M., Li, T., Zhu, P., Li, G., & Huang, F. (2024). Integrating untargeted and oxylipins-targeted metabolomics to reveal the anti-obesity and hypolipidemic mechanism of conjugated linoleic acid in high-fat diet rats. *Journal of Functional Foods*, *116*. https://doi.org/10.1016/j.jff.2024.106182
7. Lucy, M. C., Pollock Adam, S. L., Dahl, P. G., McFadden, T. T., President Schmidt, P. K., Pollock, S., Editor Louise Adam Mandy Eastin-Allen Sharon Frick Christine Horger Ron Keller Lisa Krohn Theresa Lawrence Shauna Miller, M., Stelwagen, K., & McKillip, J. (1917). Ste 100, Champaign, IL 61820 Phone 217 and Editorial Offi ce. *Customer Services Offi Ce*, 3251.
8. Whitfield, K. C., Shahab-Ferdows, S., Kroeun, H., Sophonneary, P., Green, T. J., Allen, L. H., & Hampel, D. (2020). Macro- and Micronutrients in Milk from Healthy Cambodian Mothers: Status and Interrelations. *Journal of Nutrition*, *150*(6), 1461–1469. https://doi.org/10.1093/jn/nxaa070
9. Zhou, Q., Wang, L., Wang, H., Xie, F., & Wang, T. (2012). Effect of dietary vitamin C on the growth performance and innate immunity of juvenile cobia (Rachycentron canadum). *Fish and Shellfish Immunology*, *32*(6), 969–975. https://doi.org/10.1016/j.fsi.2012.01.024