

NUTRITION ASSISTANT APPLICATION

S.NO	REG.NO	NAME	DEPARTMENT	TEAM
1.	19CS073	NISHANTH C	CSE	Team Lead
2.	19CS085	PRANESH R N	CSE	Team Member 1
3.	19CS087	RANJITHA R	CSE	Team Member 2
4.	19CS008	BHARATHRAJ P	CSE	Team Member 3

DONE BY
TEAM ID: PNT2022TMID19500

1.INTRODUCTION:

A primary goal of the project is to provide you with information backed by nutritional science, and a variety of resources that use scientific evidence to optimize health and prevent disease. This text was designed to support, enrich, and expand the materials provided. The objective of this study is to identify dietary self-monitoring implementation strategies on a mobile application. Nutritional knowledge is essential for promoting good eating habits since it ensures that necessary nutrient requirements are met to avoid malnutrition.

2.PROJECT OVERVIEW:

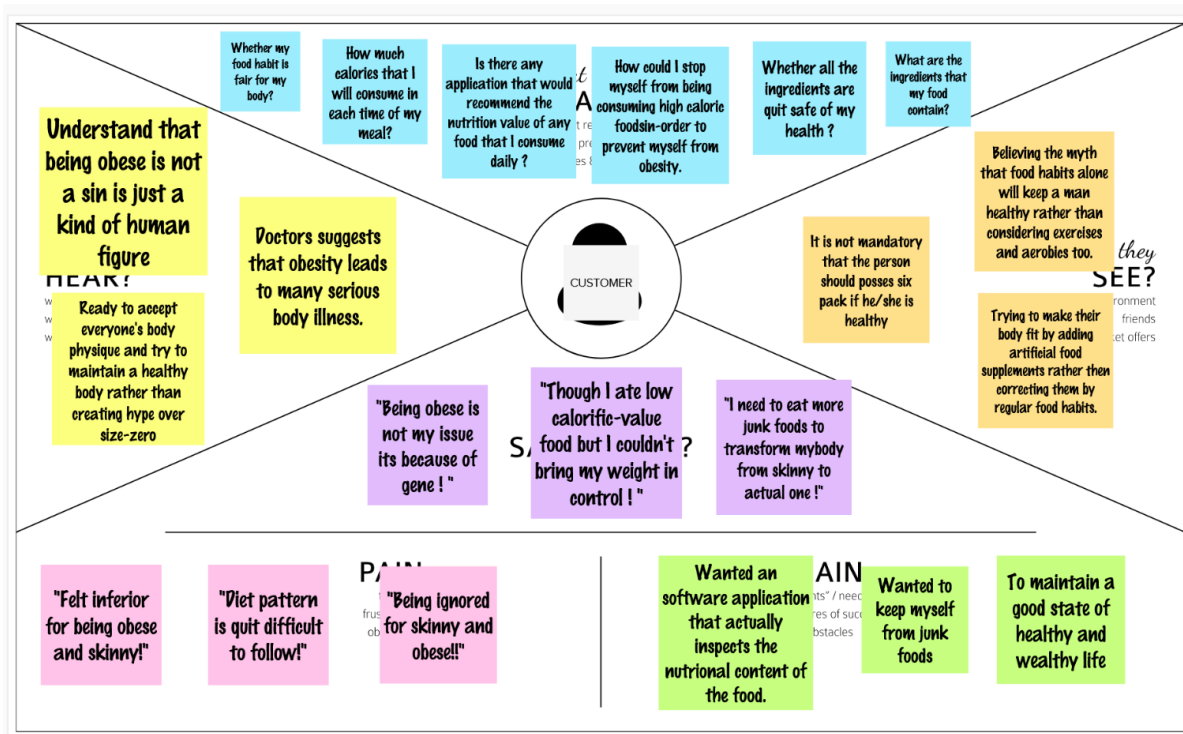
This project aims at building a web App that automatically estimates food attributes such as ingredients and nutritional value by classifying the input image of food. Our method employs **Clarifai's AI-Driven Food Detection Model** for accurate food identification and Food API's to give the nutritional value of the identified food.

3.PURPOSE:

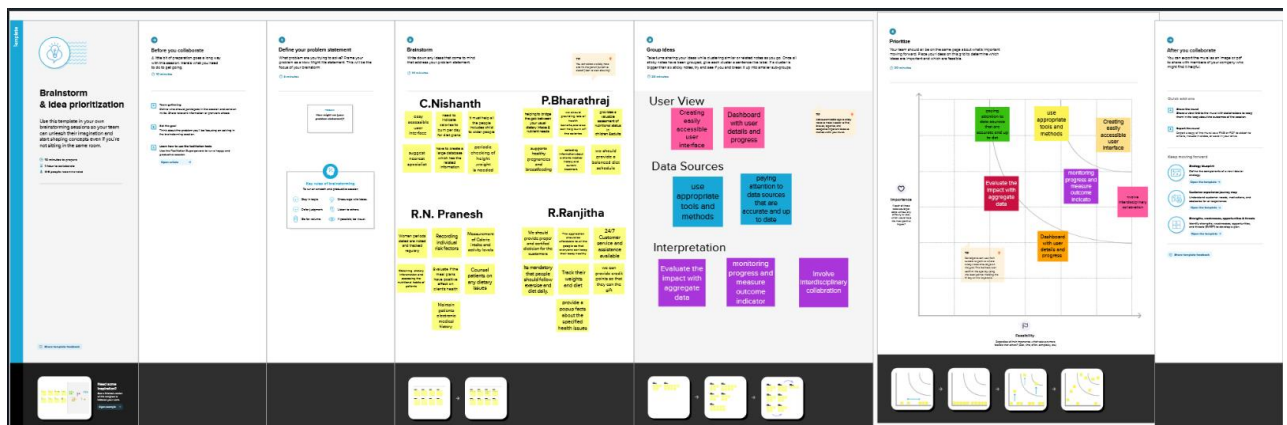
One of the most basic functions is to guide its users towards a healthy diet and assist them to achieve their health goals. So, once your user specifies the goal like desired weight goal, body type, food habits, and preferred food items, your app must suggest them with a proper diet accordingly. You can automatically calculate the nutritional information for any recipe, analyze recipe costs, visualize ingredient lists, find recipes for what's in your fridge, find recipes based on special diets, nutritional requirements, or favorite ingredients, classify recipes into types and cuisines, convert ingredient amounts, or even compute an entire meal plan.

4. IDEATION PHASE:

4.1 EMPATHY MAP:



4.2 BRAINSTORMING:



4.3 LITERATURE SURVEY:

LITERATURE SURVEY

TITLE AND AUTHOR(S)	YEAR	TECHNIQUE (S)	FINDINGS	PROS AND CONS
Enhancing Cloud and healthy Food Nutrition Information Systems Practice- Paul, PK and Aithal, PS and Bhumali, A	2017	Cloud Computing, Mobile Computing	Among the common mass food information systems are not yet popularized as a domain and thus there are huge potentialities to work on this.	P: Regarding manpower development there are a lot of things are pending and possible to work with. Hence cloud will do an attention on skill and manpower development for sophisticated development of food information systems.
Mobile cloud based system recognizing nutrition and freshness of food image- Kumbhar, Diptee and Patil, Sarita	2017	Cloud Computing, Image Segmentation	Mobile cloud computing (MCC) has been introduced to be a potential paradigm for mobile health services to overcome the interoperability issues over distinctive information formats. In this, we propose a mobile cloud-based food calorie measurement framework.	P: Multiple Platform Support Cost-Efficient C: Connectivity and Performance Issues

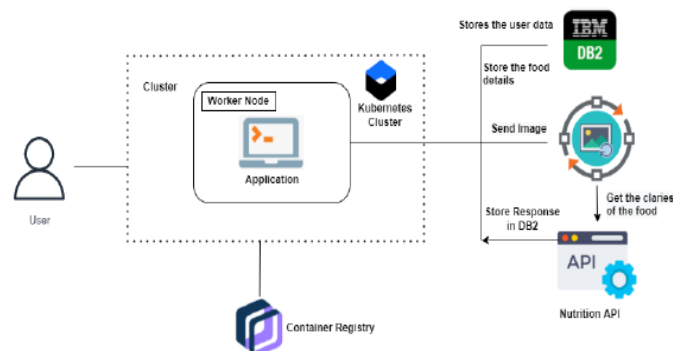
Predicting calorific value for mixed food using image processing- Kohila, R and Meenakumari, R	2017	Cloud Computing, Image Segmentation	The objective of this paper is to predict and to fix diet control for various diseases by measuring the calorific value to help the patients and nutritionists. The image captured through a mobile phone/tablet camera will provide information concerning the calorie rate of the food.	P: Increased security Reduced cost C: Limited control Lacks Support
Use of artificial intelligence in precision nutrition and fitness- de Moraes Lopes, Maria Helena Baena and Ferreira, Danton Diego and Ferreira, Ana Claudia Barbosa Honorio and da Silva, Giuliano Roberto and Caetano, Aletha Silva and Braz	2020	Artificial Intelligence, Nutritional surveillance	Among the available computational tools, artificial intelligence (AI) has gained more and more attention recently, since it is able to learn and model linear and nonlinear relationships between variables by constructing an input-output mapping such that hidden and extremely useful information for decision-making is revealed and interpreted.	P: A large amount of data is collected by these technologies C: AI is not yet widely used in the areas of nutrition and fitness

4.4 PROBLEM STATEMENTS

Software Required: Python, Flask , Docker System Required: 8GB RAM, Intel Core i3,OS-Windows/Linux/MAC ,Laptop or Desktop Due to the ignorance of healthy food habits, obesity rates are increasing at an alarming speed, and this is reflective of the risks to people's health. People need to control their daily calorie intake by eating healthier foods, which is the most basic method to avoid obesity. However, although food packaging comes with nutrition (and calorie) labels, it's still not very convenient for people to refer to App-based nutrient dashboard systems which can analyze real-time images of a meal and analyze it for nutritional content which can be very handy and improves the dietary habits, and therefore, helps in maintaining a healthy lifestyle. This project aims at building a web App that automatically estimates food attributes such as ingredients and nutritional value by classifying the input image of food. Our method employs Clarifai's AI-Driven Food Detection Model for accurate food identification and Food API's to give the nutritional value of the identified food. Work Flow of the Project:

- User interacts with the Web App to Load an image.
- The image is passed to the server application, which uses Clarifai's AI-Driven Food Detection Model Service to analyze the images and Nutrition API to provide nutritional information about the analyzed Image.
- Nutritional information of the analyzed image is returned to the app for display.

Technical Architecture:



5.REQUIREMNT ANALYSIS

5.1FUNCTIONAL REQUIREMENTS

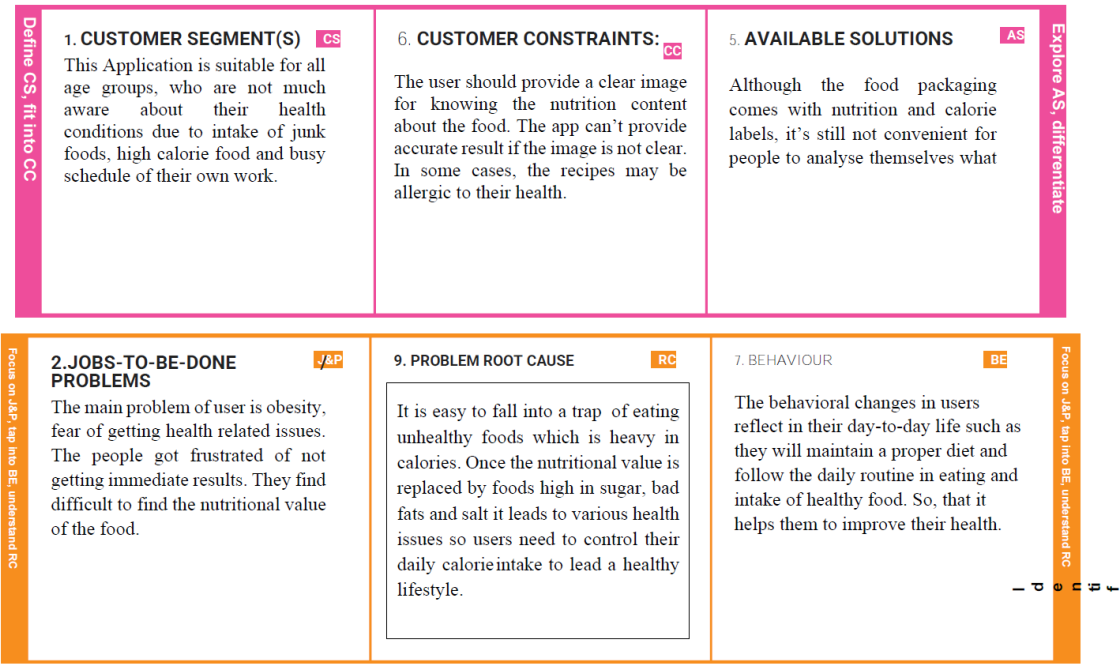
- User Registration
- User Confirmation
- Update Profile
- User Authentication
- Report

5.2NON FUNCTIONAL REQUIREMENTS


- Usability
- Security
- Reliability
- Performance
- Availability
- Scalability

6.PROJECT DESIGNS:

6.1PROBLEM SOLUTION FIT:



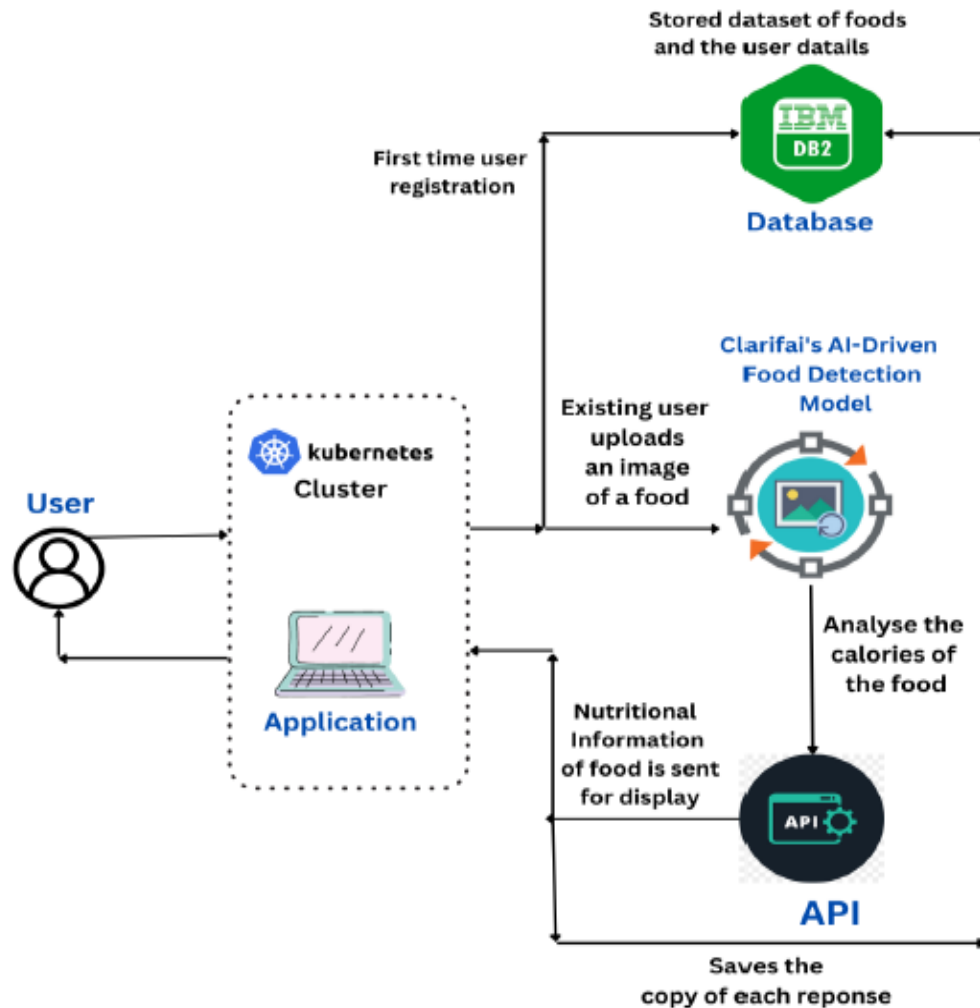
6.2 PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Due to the ignorance of healthy food habits, obesity rates are increasing at an alarming speed, and this is reflective of the risks to people's health. People need to control their daily calorie intake by eating healthier foods, which is the most basic method to avoid obesity. However, although food packaging comes with nutrition (and calorie) labels, it's still not very convenient for people to refer to App-based nutrient dashboard systems which can analyze real-time images of a meal and analyze it for nutritional content which can be very handy and improves the dietary habits, and therefore, helps in maintaining a healthy lifestyle.
2.	Idea / Solution description	This project aims at building a web App that automatically estimates food attributes such as ingredients and nutritional value by classifying the input image of food. Our method employs Clarifai's AI-Driven Food Detection Model for accurate food identification and Food API's to give the nutritional value of the identified food.
3.	Novelty / Uniqueness	<ul style="list-style-type: none"> • High accuracy in collecting the nutrition of each ingredient of the food. • Clear cut view of food's nutritional view. • Complex food can also be analysed.
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"> • Will get an idea whether his/her food is right choice for them. • Avoid conditions like Obese and Skinny body. • Helps to maintain healthy body.
5.	Business Model (Revenue Model)	 <pre> graph TD A([Nutrition Assistant Application]) --- B[Displays the nutritive value of ingredients present in the food.] A --- C[Prevents the user from becoming obese and skinny.] A --- D[Avoid overconsumption of food and High caloric valued food.] A --- E[Helps to maintain a healthy body with healthy digestive system.] A --- F[Shows the details inspection of each ingredients of the food so that the user can analyse whether it is suited to him or not.] </pre>
6.	Scalability of the Solution	Ability to know the nutritional value of each food we eat, Since it will be used by everyone who keep on considering their health in mind.

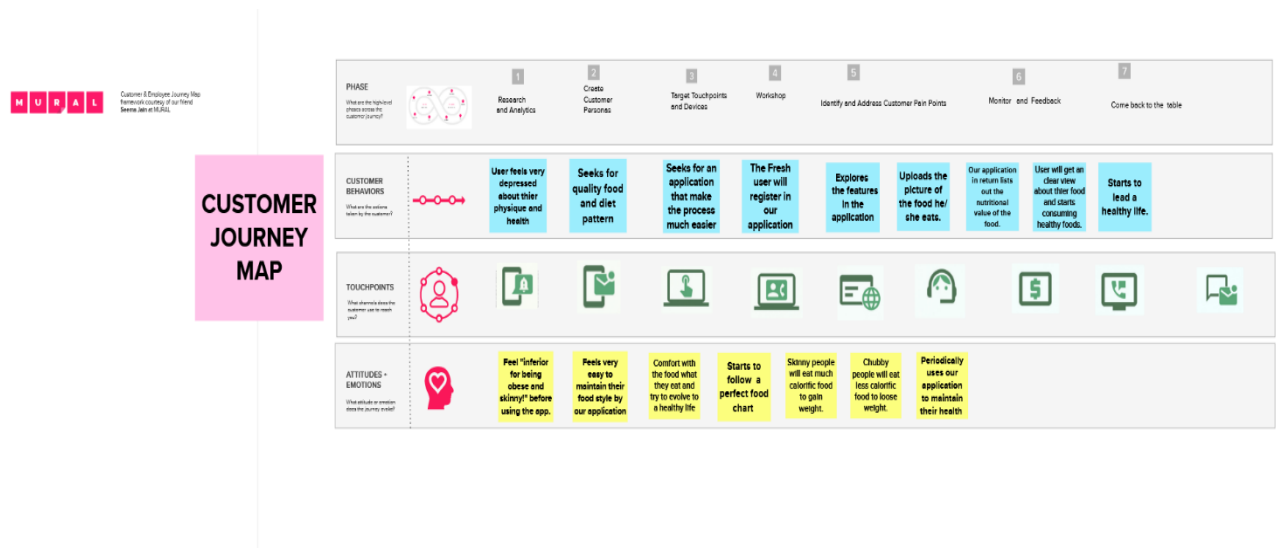
6.3 SOLUTION ARCHITECTURE:

Due to the ignorance of healthy food habits, obesity rates are increasing at an alarming speed, and this is reflective of the risks to people's health. People need to control their daily calorie intake by eating healthier foods, which is the most basic method to avoid obesity. However, although food packaging comes with nutrition (and calorie) labels, it's still not very convenient for people to refer to App-based nutrient dashboard systems which can analyze real-time images of a meal and analyze it for nutritional content which can be very handy and improves the dietary habits, and therefore, helps in maintaining a healthy lifestyle.

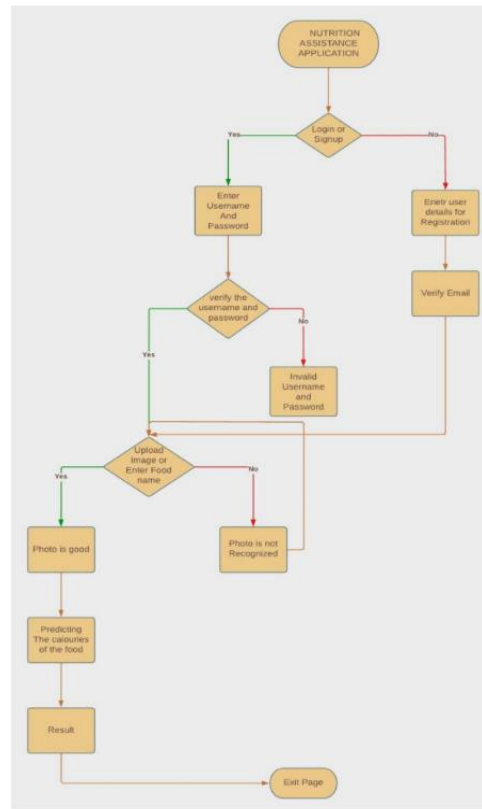
SOLUTION ARCHITECTURE DIAGRAM



6.4 CUSTOMER JOURNEY:



6.5 DATA FLOW DIAGRAM:



6.6 SOLUTION REQUIRMENTS

Project description:

This project is aimed at developing a desktop-based application named Nutrition Assistant Application for estimating food attributes such as ingredients and nutritional value by classifying the input images of food. The Nutrition Assistant Application refers to the system and processes to help the user to analyze the intake of food with the involvement of a Technology system. This system can be used to store the details of the user's health, calculating the BMI, Classifying the food image to know the nutritional value, update the status of their health condition based on the information provided, and generate health reports weekly or monthly based. This project is categorizing the individual health condition of the user. The Nutrition Assistant Application is important to control their daily calorie intake by eating healthier foods, which is the most basic method to avoid obesity. Without proper diet control, and this is reflective of the risks to people's health. A good Nutrition Assistant Application will alert the users when it is time to avoid. This project aims at building a web App that automatically estimates food attributes such as ingredients and nutritional value by classifying the input image of food.

Scope:

Maintains good health: The application can help in guiding them on how to remain healthy and how to take good nutrition. The application will help them without personally going to the doctor. Promote better nutrition in the community by educating about better diet and nutrition.

Functional limitation: The user to be specific can't access the web or admin module, whereas the administrator has all the rights to modify and manage the contents such as news, tips, etc

Improve Usability: In the part of user's just the internet connection is enough in order to access the news, updates and other contents provided by the admin regarding their health condition.

Health conscious: This will provide convenience to persons/users who wants to learn about nutrition and other related health topics by just using the Nutrition Assistant Application

Purpose:

The users continue to demand to know the nutritionalvalue that is in their food. The users learn about the effect of different foods on human health. Evidently, the ultimate aim of this application is to provide the ways in which one can lead a healthy life by maintaining his/her diet. The user can access the nutritional information by taking a photo of the food, uploading a photo from the gallery, or by entering manually. Nutrition is more than just obtaining nutrients and calories from food. It's more than just eating the healthy stuff. It's more than just following the most recent fad diet. Nutrition, the food we eat and the way we eat it, is an integral part of life. Nutrition is an experience. It evokes memories, helps us celebrate good times, and is there for us in times of grief. I believe the purpose of nutrition is to nourish the body and soul. The Nutrition Assistant Application helps the users to eat nutritional rich food which yield to lead a healthy life.

Functional Requirements:

Following are the functional requirements of the proposed solution.

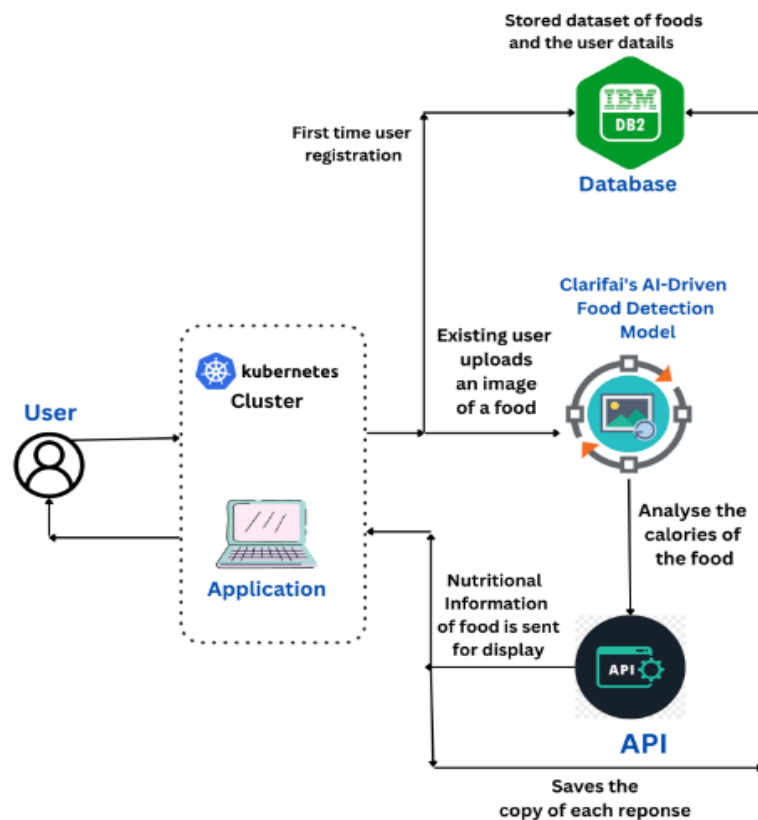
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	<ul style="list-style-type: none">➤ Registration through Form➤ Registration through Gmail➤ Registration through Facebook
FR-2	User Confirmation	<ul style="list-style-type: none">➤ Confirmation via Email➤ Confirmation via OTP
FR-3	User Login	<ul style="list-style-type: none">➤ Login with Username➤ Login with Password
FR-4	User Profile Update	<ul style="list-style-type: none">➤ Update User's Name➤ Update Portrait Photograph➤ Update Date of Birth
FR-5	Uploading Food image	<ul style="list-style-type: none">➤ Upload from Gallery➤ Capture using Camera
FR-6	Enter Food name	<ul style="list-style-type: none">➤ Type the name of the food
FR-7	Result	<ul style="list-style-type: none">➤ Download Result➤ Share Result through Social media
FR-8	Ratings and Reviews	<ul style="list-style-type: none">➤ Share the experiences➤ Provide Feedback

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	➤ Accessible through INTERNET
NFR-2	Security	➤ Secure through unique Username and Password
NFR-3	Reliability	➤ Accurate result ➤ User friendly
NFR-4	Performance	➤ Using Standard algorithm to get faster and accurate results ➤ Clarifai's AI-Driven Food Detection Model is used.
NFR-5	Availability	➤ Available for 24/7 ➤ Deep Learning of the food image and predict the results with using the given dataset.
NFR-6	Scalability	➤ It can be accessed by a greater number of users at the same time without any compromise in the performance.

6.7 TECHNOLOGY ARCHITECTURE



Guidelines:

1. To use the app the user must register / login.
2. After successful registration/login, the user can upload the meal image.
3. Using Clarifai AI- Driven API the name of the meal will be identified.
4. The identified name will be sent to Nutrition API using Flask.
5. Using Nutrition API, the nutritional value of the meal will be obtained and displayed in the UI using Flask.
6. The diet history will be added to the database to track their daily calorie intake.

Table 1: Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The user can able to see the UI via mobile application.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	The application will have the login/sign up page where the user can login into the home page of the application.	Python
3.	Application Logic-2	The user uploads an image from the gallery or takes a picture by using camera. It will be scanned by the model.	Clarifai AI-Driven Food Detection Model
4.	Application Logic-3	The output will be a list contains the nutritional value of the food.	API
5.	Database	Relational database containing the collection of nutrition of many foods.	MySQL.
6.	Cloud Database	Data about the users (Login credentials) are stored.	IBM DB2
7.	File Storage	File storage requirements	IBM Block Storage / Local Filesystem
8.	Artificial Intelligence Model	To analyse/examine the nutrients of the food.	Clarifai AI-Driven Food Detection Model
9.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration.	Local, Cloud Foundry, Kubernetes, etc.

Table 2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask ,a web framework in Python is used in the implementation of Nutrition Application System.	Python-Flask
2.	Security Implementations	This application uses Container Registry in IBM cloud so that the user details are kept as more secure and confidential. User have to confirm the login while logging in to avoid any misuse of the credentials	Container Registry, Kubernetes Cluster
3.	Scalable Architecture	The Nutrition Assistant Application is more useful for the people who wanted to maintain a good healthy diet pattern. Such that our application will examine the nutritional value of the food and exposes it.	Container Registry, Kubernetes Cluster
4.	Availability	Docker helps to improve the network management so that the application can be accessed at anytime	Docker , Kubernetes Cluster
5.	Performance	The performance of this application is high and efficient as the network traffic can be easily managed.	Docker , Kubernetes Cluster

6.8 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Gmail Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can register & access the dashboard with entering email & password	High	Sprint-1
	Dashboard	USN-6	As a user, I can find the Nutritional value of the food	I can view the nutritional value of the food	High	Sprint-1
Customer Care Executive	Alerts and notifications	USN-7	At first the customer can ask their small queries in chatbot. If the customer is not satisfied with chatbot, customer can contact the Customer care Executive	To solve the queries and problems of the users	High	Sprint-1
Administrator	Manage the hardware and software requirements	USN-8	As a Administrator, I can change or update the application	I will analyse the progress of the application and do necessary update as per the customer reviews	High	Sprint-1

7.PROJECT PLANNING AND SCHEDULING

7.1SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration (Mobile User)	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Nishanth.C
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Bharathraj.P
Sprint-1		USN-3	As a user, I can register for the application through Facebook	2	Medium	Pranesh RN
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Ranjitha R
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	2	High	Nishanth C
Sprint-2	Registration (Web User)	USN-6	As a user, I can register for the web page by entering my email, password, and confirming my password	1	High	Pranesh RN
Sprint-2		USN-7	As a user, Once I have registered I will receive Confirmation in email.	3	High	Ranjitha R
Sprint-2		USN-8	As a user I can register for the Web page Through mail.	2	Medium	Nishanth C
Sprint-2	Login (Mobile User)	USN-9	As a user , I can log into the web page by Entering email/username & password.	1	High	Bharathraj.P
Sprint-2	Login (Web User)	USN-10	As a user , I can log into the web page by Entering email/username & password.	1	High	Pranesh RN

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Dashboard (Mobile User)	USN-11	As a user, I can find the Nutritional value of the food by logging into the application.	2	High	Pranesh RN
Sprint-3	Dashboard (Web User)	USN-12	As a user, I can find the Nutritional value of the food by logging into the Web Page.	1		Ranjitha R
Sprint-3	Customer Care Executive	USN-13	At first the customer can ask their small queries in chatbot. If the customer is not satisfied with chatbot, customer can contact the Customer care Executive	2	High	Nishanth C
Sprint-4	Manage the hardware and software requirements	USN-14	As a Administrator, I can change or update the application	2	High	Bharathraj.P

7.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	9	6 Days	24 Oct 2022	29 Oct 2022	9	29 Oct 2022
Sprint-2	8	6 Days	31 Oct 2022	05 Nov 2022	8	05 Nov 2022
Sprint-3	5	6 Days	07 Nov 2022	12 Nov 2022	5	12 Nov 2022
Sprint-4	2	6 Days	14 Nov 2022	19 Nov 2022	2	19 Nov 2022

8.CODING & SOLUTIONING:

8.1FEATURE 1:

The user can upload any food image Nutrients present in the uploaded image will be displayed

FOOD SERVICES		Dashboard	About Us	Food Services	Daily Tracker	Personal Diary	Logout
FOOD ITEM		Ingredients					
ESTIMATE	PERCENTAGE						
Your entered food is samosa	--	Nutrition		Nutrition Value			
		Protein		11.8 G			
		Total lipid (fat)		7.06 G			
		Carbohydrate, by difference		9.41 G			
		Energy		153 KCAL			
		Sugars, total including NLEA		2.35 G			
		Fiber, total dietary		2.4 G			
		Calcium, Ca		24.0 MG			
		Iron, Fe		1.27 MG			
		Sodium, Na		553 MG			
		Vitamin A, IU		353 IU			
		Vitamin C, total ascorbic		2.8 MG			

8.2 FEATURE 2:

```
1 from flask import Flask,render_template,request,redirect,url_for ,session
2 import ibm_db
3 import re
4 import os
5 import math
6 import random
7 import smtplib
8 import requests
9 app=Flask(__name__,template_folder='templates',static_folder='static')
10 app.secret_key='a'
11 conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=3
12 print("successfully connected")
13 @app.route('/')
14 def home():
15     return render_template('index.html')
16
17
18 @app.route('/login',methods=['GET','POST'])
19 def login():
20     global userid
21     msg=''
22
23     if request.method=='POST':
24         username=request.form.get('username',False)
25         password=request.form.get('password',False)
26         sql='SELECT * FROM USER WHERE username=? AND password=?'
27         stmt=ibm_db.prepare(conn,sql)
28         ibm_db.bind_param(stmt,1,username)
29         ibm_db.bind_param(stmt,2,password)
30         ibm_db.execute(stmt)
31         account=ibm_db.fetch_assoc(stmt)
32         print(account)
33         if account:
34             session['logged in']=True
35             session['id']=account['USERNAME']
36             userid=account['USERNAME']
37             session['username']=account['USERNAME']
```

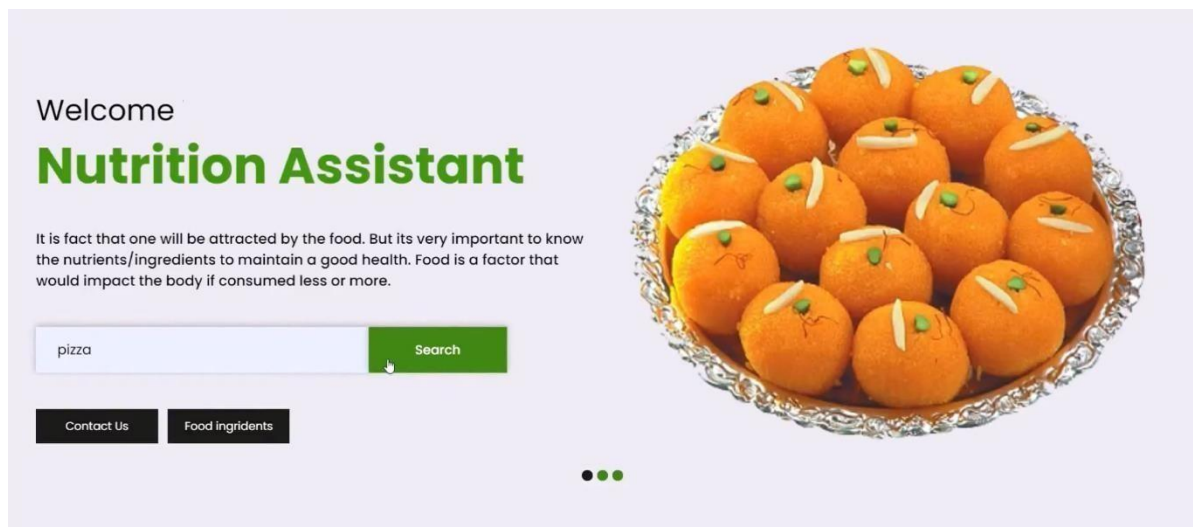

9.TESTING:

9.1TEST CASES:

1. Our code was tested on various food to check whether it gives the correct output
2. To satisfy the customer's expectations we tested it fully.

9.2USER ACCEPTANCE TESTING:

Our project was tested by an end user to verify that it's working correctly.



!!!!!!! scroll down !!!!!!!!!

The entered food is PIZZA

Ingredients

Nutrition	Nutrition Value
Protein	13.0 G
Total lipid (fat)	21.7 G
Carbohydrate, by difference	17.4 G
Energy	319 KCAL
Sugars, total including NLEA	2.17 G
Fiber, total dietary	0.4 G
Calcium, Ca	167 MG

10.RESULT:

10.1PERFORMANCE METRICS:

The proposed procedure was implemented and tested set of images. The training database consists of various images of food items. Once a food is recognized the equivalent **Nutrition** is shown on the screen.

11.ADVANTAGES:

- It provides a maintained strategy of healthy eating habits.
- It delivers information on the nutritional value of foods and how balanced and healthy eating habits are important for us.
- It limits the amount of unnecessary food such as fat that people consume a lot.

12.CONCLUSION:

In conclusion, many people have become aware of their health. Moreover, they are also informed how to live a healthy lifestyle. Most of the research related to these themes aims to identify changes in healthy lifestyle behavior with web applications that are considered effective in dietary self-monitoring.

13.DESRIPTION:

Nutrition assistants help dieticians with providing proper nutrition at healthcare facilities. They determine patients' nutritional needs, assess risk factors, and plan meals and menus. They also ensure proper sterilization of plates and utensils.

14.APPENDIX:

Source Code:

```
import os, re, string, random, time, datetime, requests, sendgrid, random, flask
import ibm_db
from sendgrid.helpers.mail import *
from flask import Flask, request, render_template, flash, redirect, url_for, session
from werkzeug.utils import secure_filename
from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel
from clarifai_grpc.grpc.api import service_pb2, resources_pb2, service_pb2_grpc
from clarifai_grpc.grpc.api.status import status_code_pb2

#####

UPLOAD_FOLDER = 'static/uploads'
ALLOWED_EXTENSIONS = set(['png', 'jpg', 'jpeg'])
SENDGRID_API_KEY = "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"

app = Flask(__name__)
app.secret_key = "bimbilikibilapi"
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32304;Security=SSL;PROTOCOL=TCPIP;UID=pzt20234;PWD=r7CB0AmR1QtOHfR4;","", "")
#;SSLServerCertificate=DigiCertGlobalRootCA.crt

YOUR_CLARIFAI_API_KEY = "af4bc9886c744e998ee0e20f104b1518"
YOUR_APPLICATION_ID = "test"
SAMPLE_URL = "https://res.cloudinary.com/swiggy/image/upload/f_auto,q_auto,fl_lossy/nxmlubuz0b1qixa29gov"
metadata = (("authorization", f"Key {YOUR_CLARIFAI_API_KEY}"),)
channel = ClarifaiChannel.get_grpc_channel()
stub = service_pb2_grpc.V2Stub(channel)

RAPIDAPI_KEY = "74e62205b6msha6b4e69e0088de5p12c619jsn1ed9cc5e0727"
```

```

def allowed_file(filename):
    return '.' in filename and \
        filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

def sendMail(to, title, text):
    sg = sendgrid.SendGridAPIClient(api_key=SENDGRID_API_KEY)
    from_email = Email("nsnandhaa1@gmail.com")
    to_email = To(to)
    subject = title
    content = Content("text/plain", text)
    mail = Mail(from_email, to_email, subject, content)
    response = sg.client.mail.send.post(request_body=mail.get())
    print(response.status_code)
    print(response.body)
    print(response.headers)

@app.route("/forgot-pw", methods=["GET", "POST"])
def forgotpw():
    if flask.request.method == "POST":
        data = flask.request.form
        username=data['username']
        code = ''.join(random.choices(string.ascii_letters, k=6))

        sql= "SELECT * FROM users WHERE username=?"
        stmt=ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account=ibm_db.fetch_assoc(stmt)
        print(account)
        session['userid'] = account['USERID']

        insert_sql = "INSERT INTO VERIFY VALUES(?,?)"
        prep_stmt=ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prepare_stmt, 1, account['USERID'])
        ibm_db.bind_param(prepare_stmt, 2, code)
        ibm_db.execute(prepare_stmt)

```

```
sendMail(account['EMAIL'], "Verification Code", code)
flash("We have sent a code to your registered email. please check spam folder also.")
return redirect(url_for("confirmMail"))
flash("We will send you a confirmation code to your registered email")
return render_template("forgot-pw.html")
```

```
@app.route("/confirm-mail", methods=["GET", "POST"])
```

```
def confirmMail():
```

```
    session['LoggedIn'] = False
```

```
    if flask.request.method == "POST":
```

```
        data = flask.request.form
```

```
        usercode=data['code']
```

```
        sql= "SELECT * FROM verify WHERE userid=?"
```

```
        stmt=ibm_db.prepare(conn,sql)
```

```
        ibm_db.bind_param(stmt,1,session['userid'])
```

```
        ibm_db.execute(stmt)
```

```
        verify=ibm_db.fetch_assoc(stmt)
```

```
        print(verify)
```

```
        dbcode = verify['CODE']
```

```
        if usercode == dbcode:
```

```
            session['LoggedIn'] = True
```

```
            delete_sql = "DELETE FROM verify WHERE CODE=?"
```

```
            prep_stmt=ibm_db.prepare(conn, delete_sql)
```

```
            ibm_db.bind_param(prepare_stmt, 1, dbcode)
```

```
            ibm_db.execute(prepare_stmt)
```

```
            flash("Email verified. Enter new password")
```

```
            return redirect(url_for("changepw"))
```

```
        else:
```

```
            flash("Error")
```

```
            return render_template("confirm-mail")
```

```
            return render_template("confirm-mail.html")
```

```
@app.route("/change-pw", methods=["GET", "POST"])
```

```
def changepw():
```

```
    if flask.request.method == "POST" and session['LoggedIn']:
```

```
data = flask.request.form
password=data['pw']
sql = "UPDATE users SET PASSWORD=? WHERE USERID=?"
prep_stmt=ibm_db.prepare(conn, sql)
print(password, session['userid'])
ibm_db.bind_param(prepare_stmt, 1, password)
ibm_db.bind_param(prepare_stmt, 2, session['userid'])
ibm_db.execute(prepare_stmt)
flash("Password changed.")
return redirect(url_for("login"))
else:
flash("verification error")
redirect(url_for("confirmMail"))
return render_template("change-pw.html")
```

```
@app.route("/register", methods=["GET", "POST"])
def reg():
if flask.request.method == "POST":

data = flask.request.form
email=data['email']
username=data['username']
password=data['pw']

sql= "SELECT * FROM users WHERE username=?"
stmt=ibm_db.prepare(conn,sql)
ibm_db.bind_param(stmt,1,username)
ibm_db.execute(stmt)
account=ibm_db.fetch_assoc(stmt)
print(account)
if account:
flash("Account already exists!")
elif not re.match(r'^@]+@[^@]+\.[^@]+' , email):
flash("invalid email address")
elif not re.match(r'[A-Za-z0-9]+' , username):
flash("name must contain only characters and numbers")
```

```
else:
insert_sql = "INSERT INTO users VALUES(?,?,?,?)"
prep_stmt=ibm_db.prepare(conn, insert_sql)
ibm_db.bind_param(prepare_stmt, 1, username)
ibm_db.bind_param(prepare_stmt, 2, email)
ibm_db.bind_param(prepare_stmt, 3, password)
ibm_db.bind_param(prepare_stmt, 4, ".join(random.choices(string.ascii_letters, k=16)))
ibm_db.execute(prepare_stmt)
flash("logged in")

return redirect(url_for("dashboard"))
return render_template("reg.html")

@app.route("/login", methods=["GET", "POST"])
def login():
if flask.request.method == "POST":

data = flask.request.form
username=data['username']
password=data['pw']

sql = "SELECT * FROM users WHERE username=? AND password=?"
stmt = ibm_db.prepare(conn,sql)
ibm_db.bind_param(stmt, 1, username)
ibm_db.bind_param(stmt, 2, password)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
print(account)
if account:
session['LoggedIn'] = True
session['userid'] = account['USERID']
session['username'] = account['USERNAME']
userid = account['USERID']
flash("logged in")
return redirect(url_for("dashboard"))
else:
flash("error")
```

```

return render_template("login.html")

@app.route("/dashboard", methods=["GET", "POST"])
def dashboard():
    global request
    if flask.request.method == "POST" and session['LoggedIn']:
        if 'file' not in flask.request.files:
            flash('No file part')
            return redirect(flask.request.url)
        file = flask.request.files['file']
        if file.filename == "":
            flash('No image selected')
            return redirect(flask.request.url)
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            flash('Image successfully uploaded')

            with open(os.path.join(app.config['UPLOAD_FOLDER'], filename), "rb") as f:
                file_bytes = f.read()

            request = service_pb2.PostModelOutputsRequest(
                model_id="food-item-v1-recognition",
                user_app_id=resources_pb2.UserAppIDSet(app_id=YOUR_APPLICATION_ID),
                inputs=[
                    resources_pb2.Input(
                        data=resources_pb2.Data(image=resources_pb2.Image(
                            base64=file_bytes
                        ))
                    )
                ],
            )
            response = stub.PostModelOutputs(request, metadata=metadata)

            if response.status.code != status_code_pb2.SUCCESS:
                print(response)

```



```
raise Exception(f"Request failed, status code: {response.status}")

foodname = response.outputs[0].data.concepts[0].name

ingredients = "
for concept in response.outputs[0].data.concepts:
ingredients += f"{concept.name}: {round(concept.value, 2)}, "

nutritionValues = "
# nutritionApiUrl = "https://spoonacular-recipe-food-nutrition-v1.p.rapidapi.com/recipes/guessNutrition"
# querystring = {"title":foodname}
# headers = {
# "X-RapidAPI-Key": RAPIDAPI_KEY,
# "X-RapidAPI-Host": "spoonacular-recipe-food-nutrition-v1.p.rapidapi.com"
# }
# response = requests.request("GET", nutritionApiUrl, headers=headers, params=querystring)
# nutritions = response.text
nutritions = {
"recipesUsed": 10,
"calories": {
"value": 470,
"unit": "calories",
"confidenceRange95Percent": {
"min": 408.93,
"max": 582.22
},
"standardDeviation": 139.8
},
"fat": {
"value": 17,
"unit": "g",
"confidenceRange95Percent": {
"min": 12.81,
"max": 21.36
},
"standardDeviation": 6.9
},
```

```

"protein": {
"value": 15,
"unit": "g",
"confidenceRange95Percent": {
"min": 9.06,
"max": 29.78
},
"standardDeviation": 16.71
},
"carbs": {
"value": 65,
"unit": "g",
"confidenceRange95Percent": {
"min": 57.05,
"max": 77.9
},
"standardDeviation": 16.81
}
}

nutritions.pop('recipesUsed')
for i in nutritions:
nutritionValues += f"{i}: {nutritions[i]['value']} {nutritions[i]['unit']}, "

sql = "INSERT INTO foods VALUES(?,?,?,?)"
stmt=ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt, 1, session['userid'])
ibm_db.bind_param(stmt, 2, datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
ibm_db.bind_param(stmt, 3, foodname)
ibm_db.bind_param(stmt, 4, ingredients)
ibm_db.bind_param(stmt, 5, nutritionValues)
ibm_db.execute(stmt)

# os.remove(os.path.join(app.config['UPLOAD_FOLDER'], filename))
return render_template("dashboard.html",
filename = filename,
username = session['username'],

```

```
foodname = foodname,
ingredients = ingredients,
nutritionValues = nutritionValues,
)
else:
flash('Allowed image formats - png, jpg, jpeg')
return redirect(flask.request.url)

elif session['LoggedIn']:
return render_template("dashboard.html", username=session['username'])
else:
return redirect(url_for("login"))

@app.route('/logout', methods=["GET", "POST"])
def logout():
session.pop('LoggenIn', None)
session.pop('userid', None)
session.pop('username', None)
return render_template("index.html")

@app.route('/display/<filename>', methods=["GET", "POST"])
def display(filename):
print(filename)
return redirect(url_for('static', filename='uploads/' + filename), code=301)

@app.route('/app', methods=["GET", "POST"])
def other():
return render_template("index.html")

@app.route('/', methods=["GET", "POST"])
def index():
return render_template("index.html")

if __name__ == "__main__":
app.run(host='0.0.0.0', port=5000)
```

GitHub:

<https://github.com/IBM-EPBL/IBM-Project-24703-1659947740/tree/main/Project%20development%20phase>

PROJECT DEMONSTRATION LINK:

<https://drive.google.com/file/d/1PV6eGKAPdFSGv4>

