Project Report

NUTRITION ASSISTANT APPLICATION

S.NO	REG.NO	NAME	DEPARTMENT	TEAM
1.	952819104054	SIVA RANJANI R	CSE	Team Lead
2.	952819104045	SANTHANAMARIAMMAL A	CSE	Team Member1
3.	952819104050	SINDHU C	CSE	Team Member2
4.	952819104058	SUSEELA S	CSE	Team Member3

DONE BY

TEAM ID: PNT2022TMID50577

1. INTRODUCTION

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 metto avoid malnutrition.

Wellness and healthy lifestyles have become mainstream. Interest in fitness applications and revenue from them grow as fast as the number of people striving tobe fit.

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- DrivenFood Detection Model for** accurate food identification and Food API's to give the nutritional value ofthe identified food.

3. **PURPOSE**

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 typesand cuisines, convert ingredient amounts, or even compute an entire meal plan.

LITERATURE SURVEY

Study on manufacture inventories :

It is the most comprehensive study on manufacturers' inventories. They used the CMI data and the consolidated balance sheet data of public limited companies published by the RBI, in order to analyses each of the major components, like the raw materials, goods-in-process and finished goods, for 21 industries over the period ranging from 1946-62. The study was a time series one although there were some inter-industry cross-section analyses that were carried out in the analysis. The Accelerator represented by change in sales, bank finance and short-term interest rate was found to be an important determinant. The utilization of productive capacity and price anticipations was also found to be relevant in the study

Flask : https://www.youtube.com/embed/uxZuFm5tmhM

Send-Grid : https://sendgrid.com/

Rapid API : https://rapidapi.com/hub

Docker :https://www.youtube.com/embed/pTFZFxd4hOI

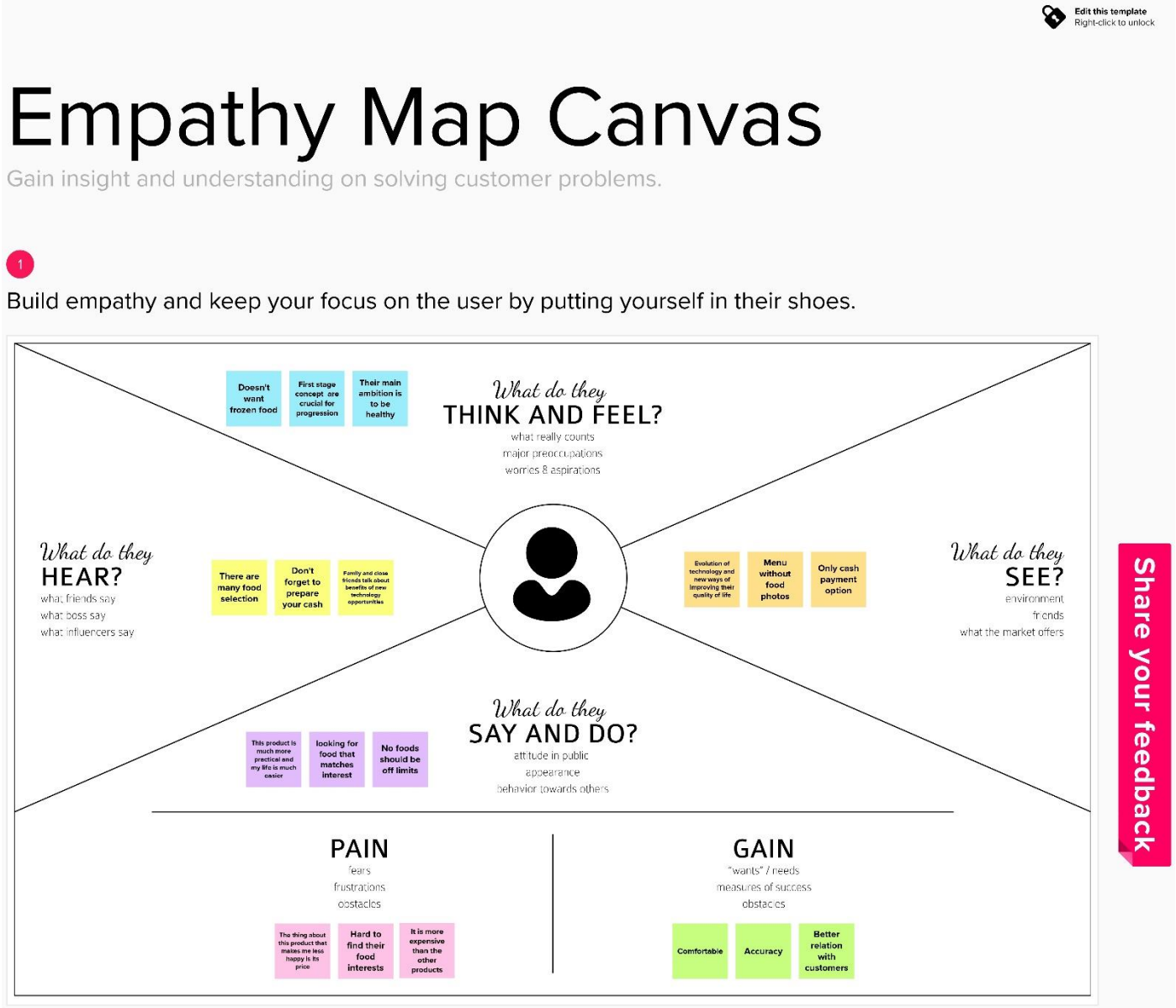
Kubernetes : https://www.youtube.com/embed/d6WC5n9G_sM

1) Problem Statement Definition

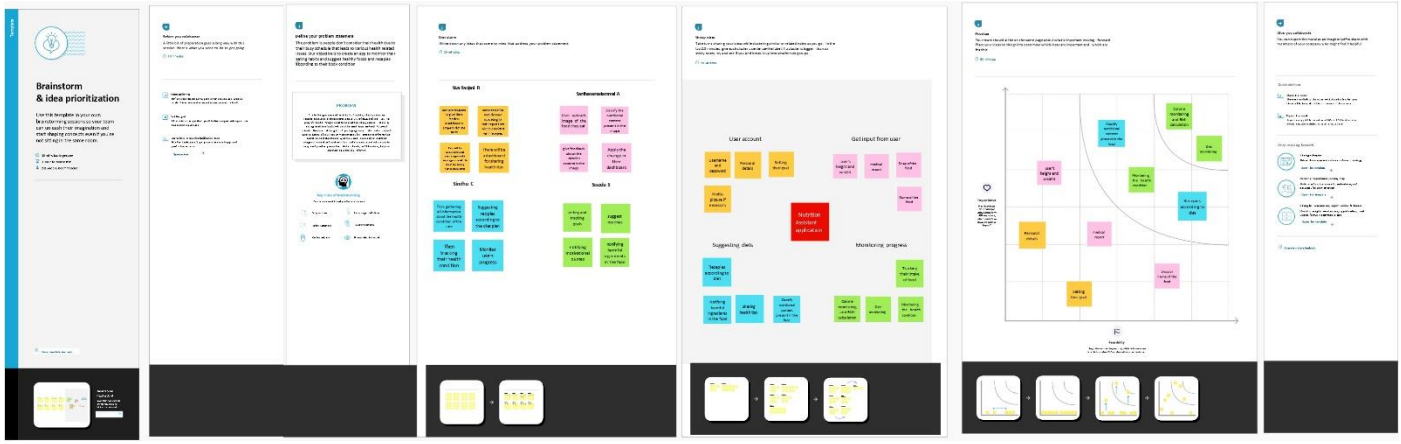
App-based nutrient dashboard systems which can analyze real-time images of a meal and analyze it for nutritional content which can be very handy and improves the dietary habits, and therefore, helps in maintaining a healthy lifestyle.

IDEATION & PROPOSED SOLUTION

1) Empathy Map Canvas



2) Ideation & Brainstorming



3) Proposed Solution

Sl. No.	Parameter	Description
1.	Problem Statement (Problem to besolved)	• Now a days peoples are not eating healthyfoods with respect to their health condition. If it happens continuously means, it will lead to obesity and any otherhealth problems. • To avoid that the system will detect andrecognize the food and evaluating the nutrient values present in the food.
2.	Idea / Solution description	• To store the food and details of thenutrients present in it. • Then scan the real time food and retrievethe corresponding food's nutrient values.
3.	Novelty / Uniqueness	• Clustering the peoples based on their BMIvalue.
4.	Social Impact / Customer Satisfaction	• The application which gives awareness among the people about the obesity andvarious health problems.
5.	Business Model (Revenue Model)	• In market, this application gives a benefitacross the people by health wise and economical wise.
6.	Scalability of the Solution	• Providing regular updates • Efficient goal tracking assistance • The additional features such that sleep tracking, mensuration tracking can be done.

4) Problem Solution fit

Problem-Solution fit canvas 2.0

Purpose/vision NUTRITION ASSISTANT APPLICATION

TEAM ID :PNT2022TMID50577

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?

All kind of people who want to maintain their nutrition

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions?

1. Spending power, budget, no cash, network connection, available devices.

2.Users will not be able to use the application without registering.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

1. If the users forget their password they can create a new password by using email verification

Explore AS, differentiate

Focus on J&P, tap into

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

1. People have many problems in maintaining their nutrition in day to day life.

2. It include raising the level of nutrition for people without the knowledge for maintaining the nutrition.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

1. It is challenging for people to manage their diet flow day to day.

2. A variety of medical problems can affect your appetite, illness, medicines or surgery can cause these problems.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

1. When its come to dieting some people may not have proper guidance to maintain their diet

2. This problem can be overcome by this application users can view their nutrition flow and eat or drink accordingly.

Focus on J&P, tap int C

Identify strong TR & EM

3. TRIGGERS TR

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

1. Maintaining the nutrition problem is a major problem among people

2. Once their realize their health condition and how much can make necessary adjustment and manage their health better

4. EMOTIONS: BEFORE / AFTER EM

How do customers feel when they face a problem or a job and afterwards?
i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

Before : Unhealth, imbalanced nutrition

After : Healthy diet, balanced nutrition

10. YOUR SOLUTION SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

A variety of medical problems can affect your appetite. Your illness, medicines or surgery can cause these problems. Many people become frustrated when they know they need to eat to get well but they aren't hungry, or when they gain weight because they are fatigued and unable to exercise. Each of the following sections describes a nutritional problem and suggests possible solutions. .

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

What kind of actions do customers take online? Extract online channels from #7

People can check the amount of nutrition they need to take on daily basics.

8.2 OFFLINE

What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

With the knowledge of nutrition plan from the application people can eat and drink accordingly.

Extract online & offline CH of BE

AMALTAMA

REQUIREMENT ANALYSIS

1) Functional requirement

- User Registration
- User login
- User Request
- Server Response
- User activity
- User ->ServerServer ->User

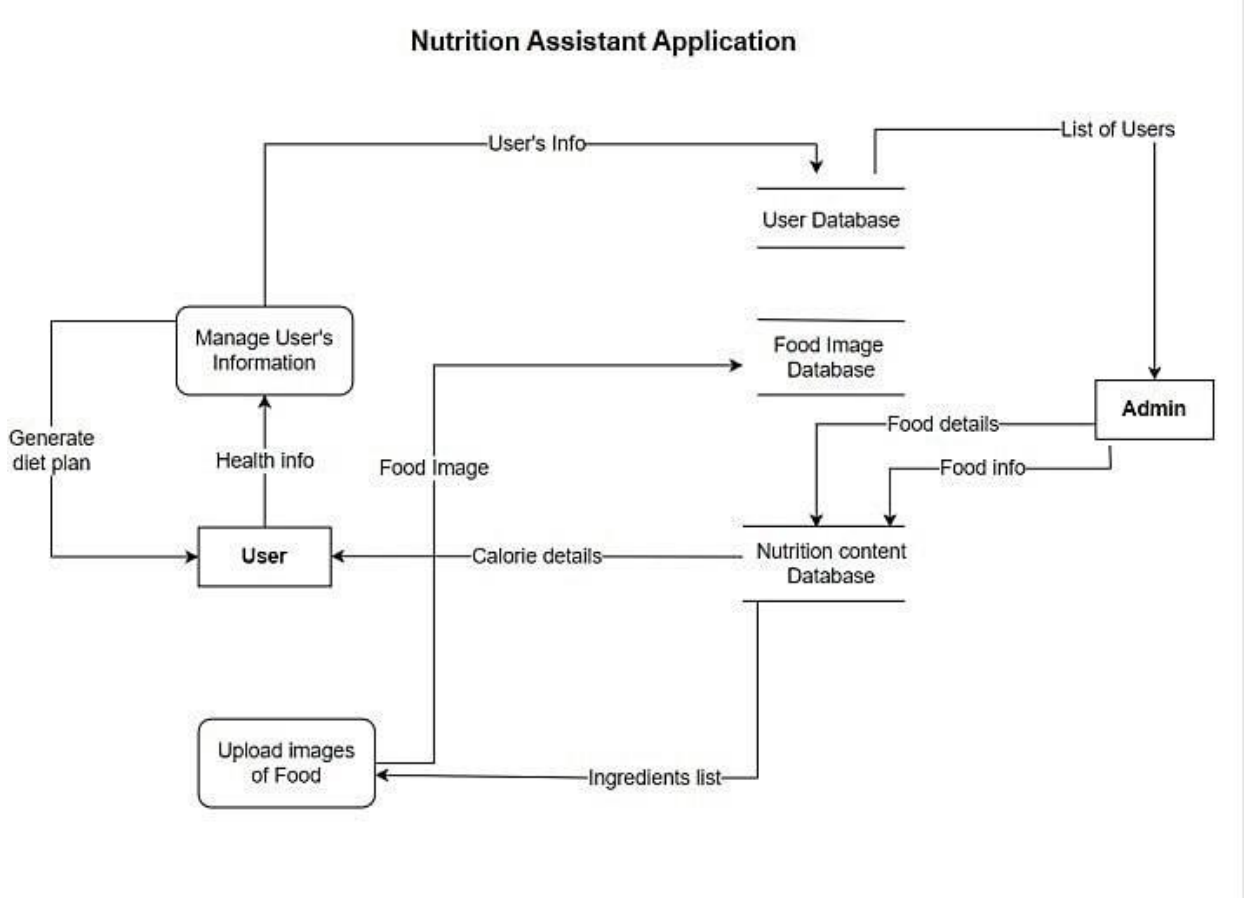
  (User interaction with the application)

2) Non-Functional requirements

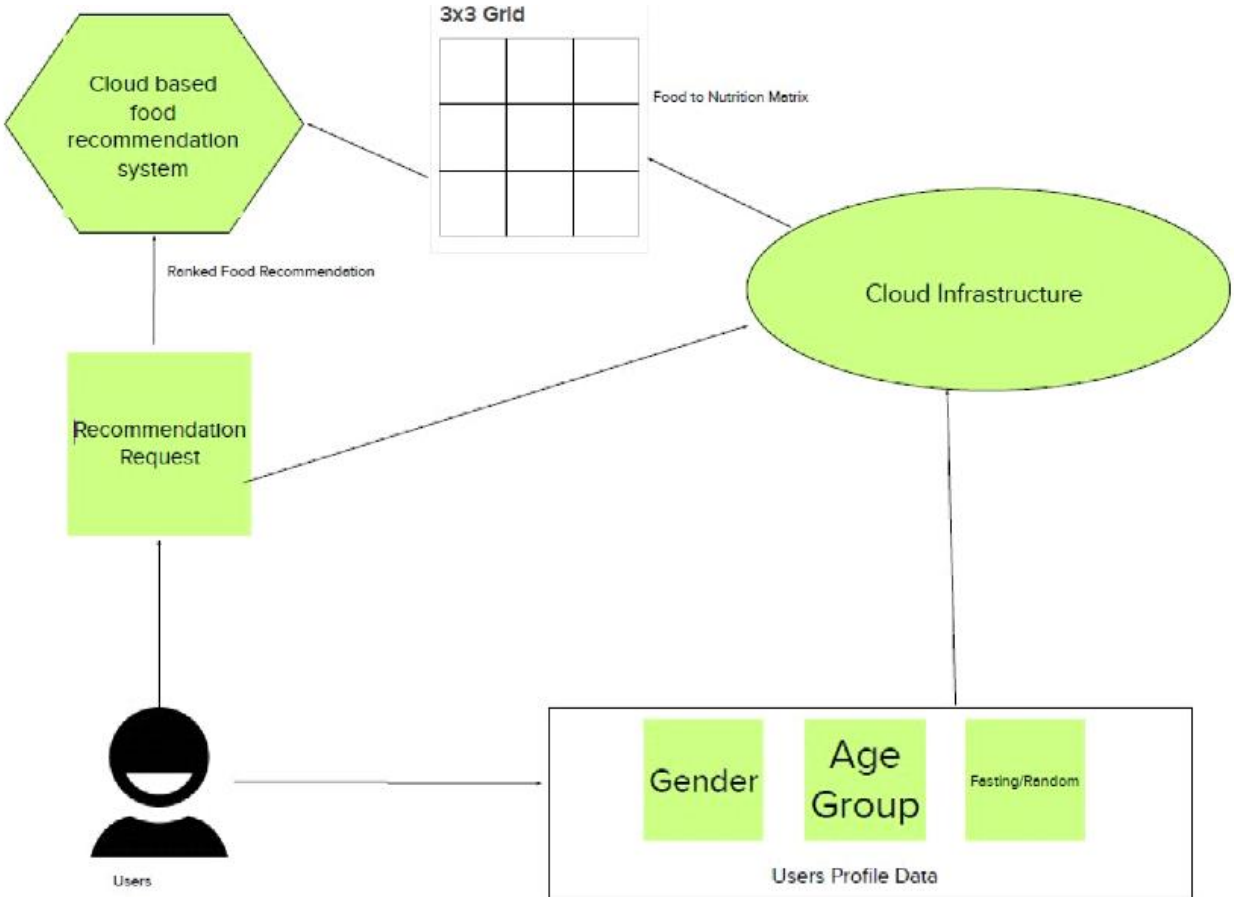
- Usability
- Security
- Reliability
- Performance
- Availability
- Scalability

PROJECT DESIGN

1) Data Flow Diagrams



2) Solution & Technical Architecture



3) User Stories

- As a user, I can register for the application by entering my email, password, andConfirming my password
- As a user, I will receive confirmation email once I have registered for theapplication
- As a user, I can log into the application by entering email & password
- As a user, I can fill the details.
- As a user,I will search the food items.
- As a user, I can scan the food an get the nutrition details and recipe for relatedscanned food.

PROJECT PLANNING & SCHEDULING

1) Product Backlog, Sprint Schedule, and Estimation

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	SIVA RANJANI R SANTHANAMARI AMMAL A SINDHU C SUSEELA S
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	SIVA RANJANI R SANTHANAMARI AMMAL A SINDHU C SUSEELA S
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password	1	High	SIVA RANJANI R SANTHANAMARI AMMAL A SINDHU C SUSEELA S
Sprint-2	User details	USN-4	As a user , I can fill the Details.	2	High	SIVA RANJANI R SANTHANAMARI AMMAL A SINDHU C SUSEELA S
Sprint-3	Push notification	USN-5	As a user, I will search the food items.	2	Medium	SIVA RANJANI R SANTHANAMARI AMMAL A SINDHU C SUSEELA S
Sprint-4	Shown the nutrition details and Recipe for	USN-6	As a user, I can scan the food an get the nutrition details and recipe for related scanned	1	High	SIVA RANJANI R SANTHANAMARI AMMAL A SINDHU C SUSEELA S
Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
	scanned food		food.			SIVA RANJANI R SANTHANAMARI AMMAL A SINDHU C SUSEELA S

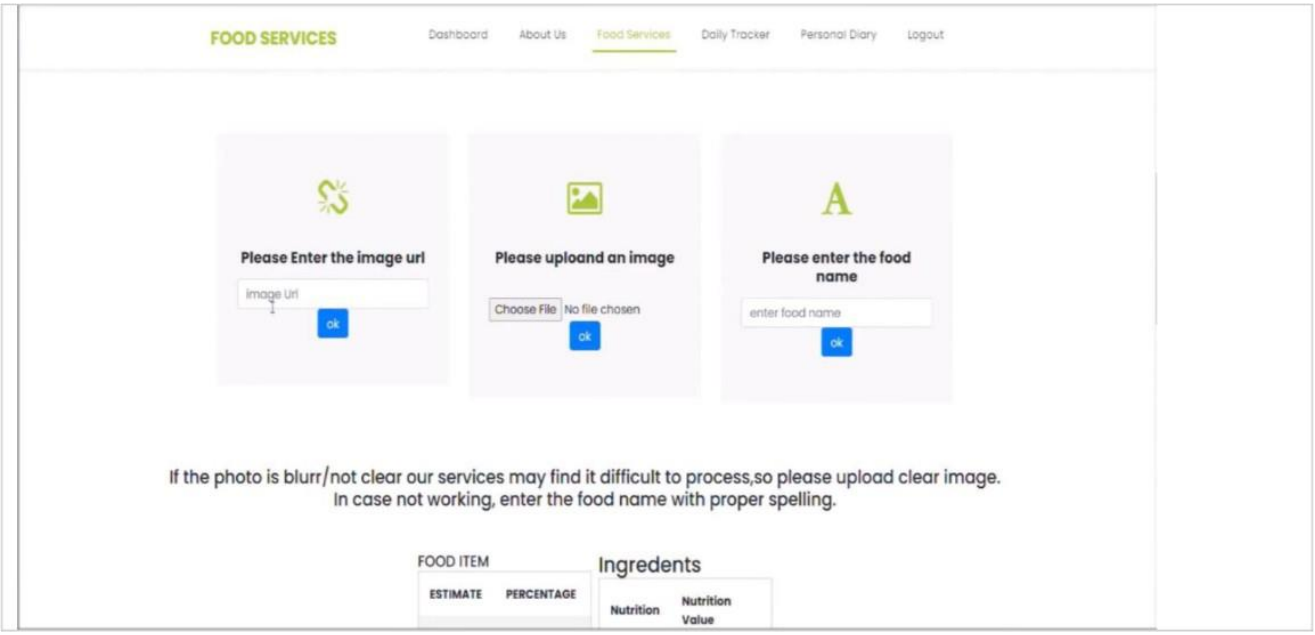
2) Project Tracker, Velocity & Burndown Chart:

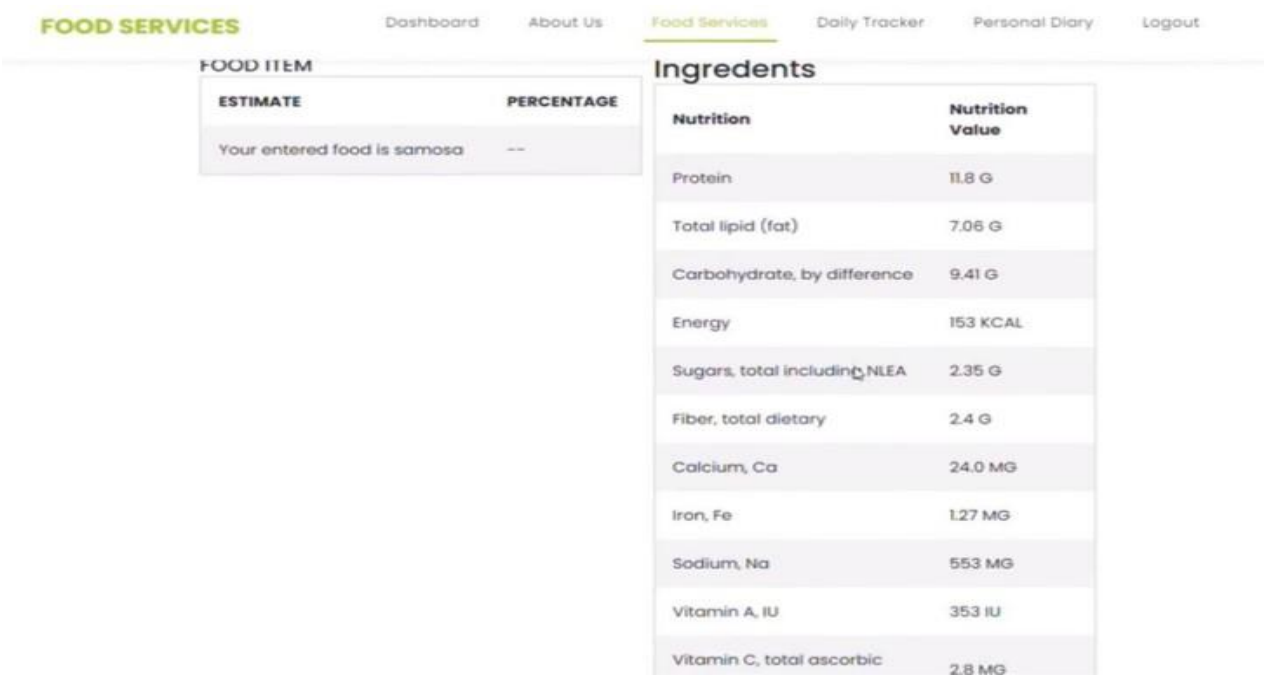
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

CODING & SOLUTIONING

1) Feature 1

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





2) Feature 2

```python
from flask import Flask,render_template,request,redirect,url_for ,session
import ibm_db
import re
import os
import math
import random
import smtplib
import requests
app=Flask(__name__,template_folder='templates',static_folder='static')
app.secret_key='a'
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=3
print("successfully connected")
@app.route('/')
def home():
    return render_template('index.html')


@app.route('/login',methods=['GET','POST'])
def login():
    global userid
    msg=''

    if request.method=='POST':
        username=request.form.get('username',False)
        password=request.form.get('password',False)
        sql='SELECT * FROM USER 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['Logged in']=True
            session['id']=account['USERNAME']
            userid=account['USERNAME']
            session['username']=account['USERNAME']
```

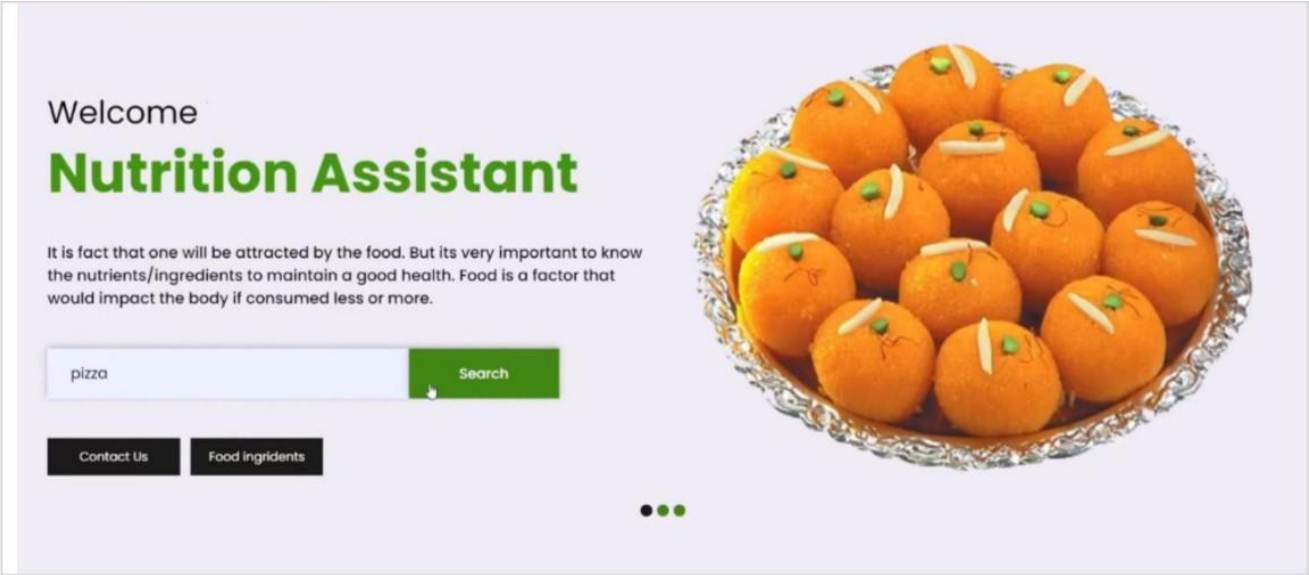
TESTING

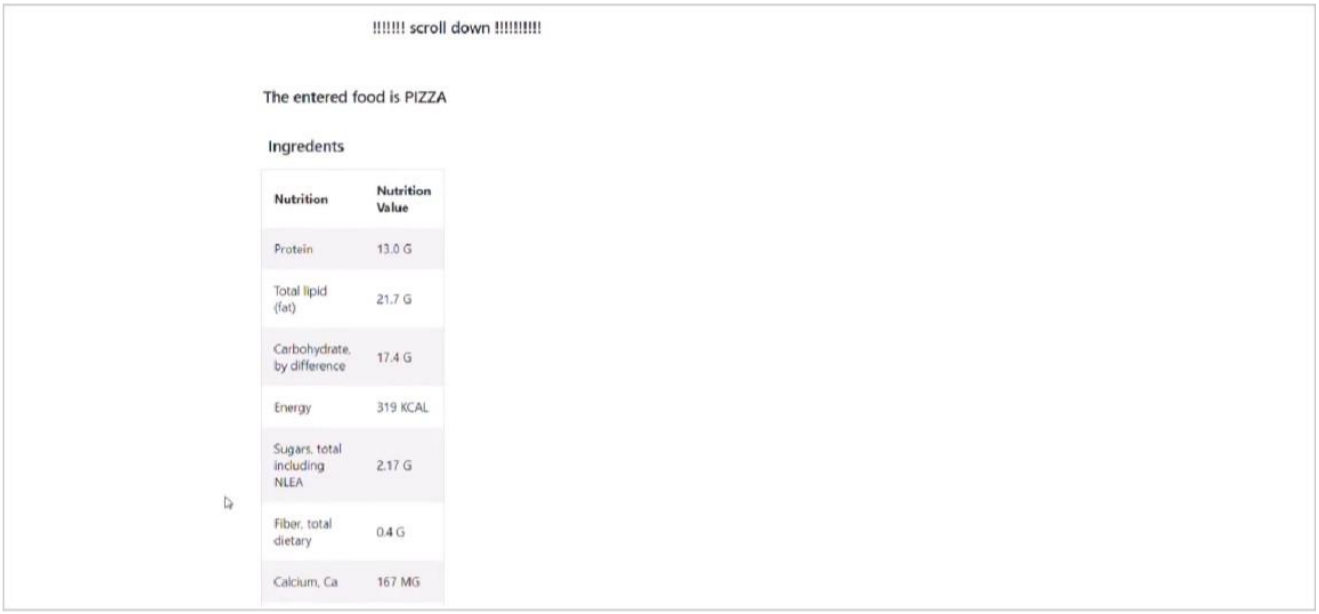
1) Test Cases

i. Our code was tested on various food to check whether it gives the correctoutput
ii. To satisfy the customer's expectations we tested it fully.

2) User Acceptance Testing

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



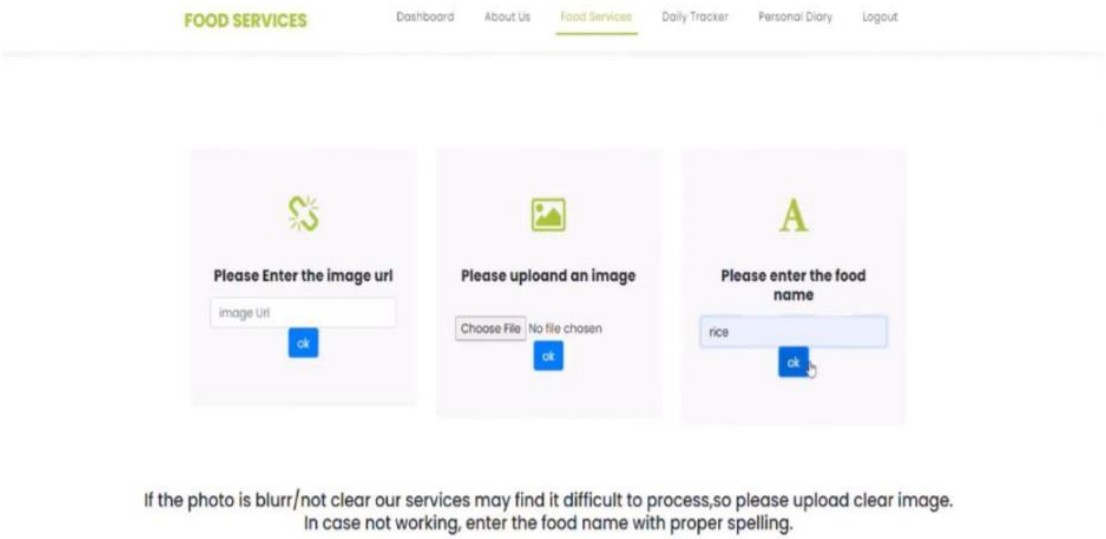


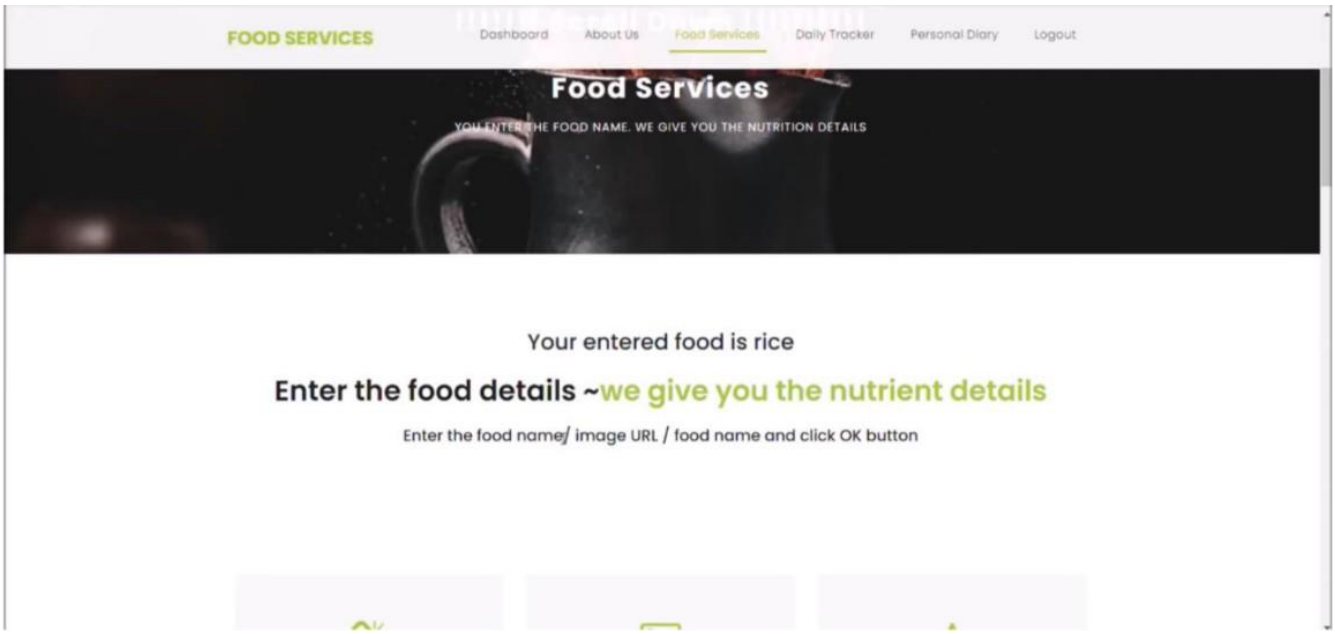
Nutrition	Nutrition Value
Protein	13.0 G
Total lipid (fat)	21.7 G
Carbohydrate, by difference	17.4 G
Energy	319 KCAL
Sugars, total including NLEA	2.17 G
Fiber, total dietary	0.4 G
Calcium, Ca	167 MG

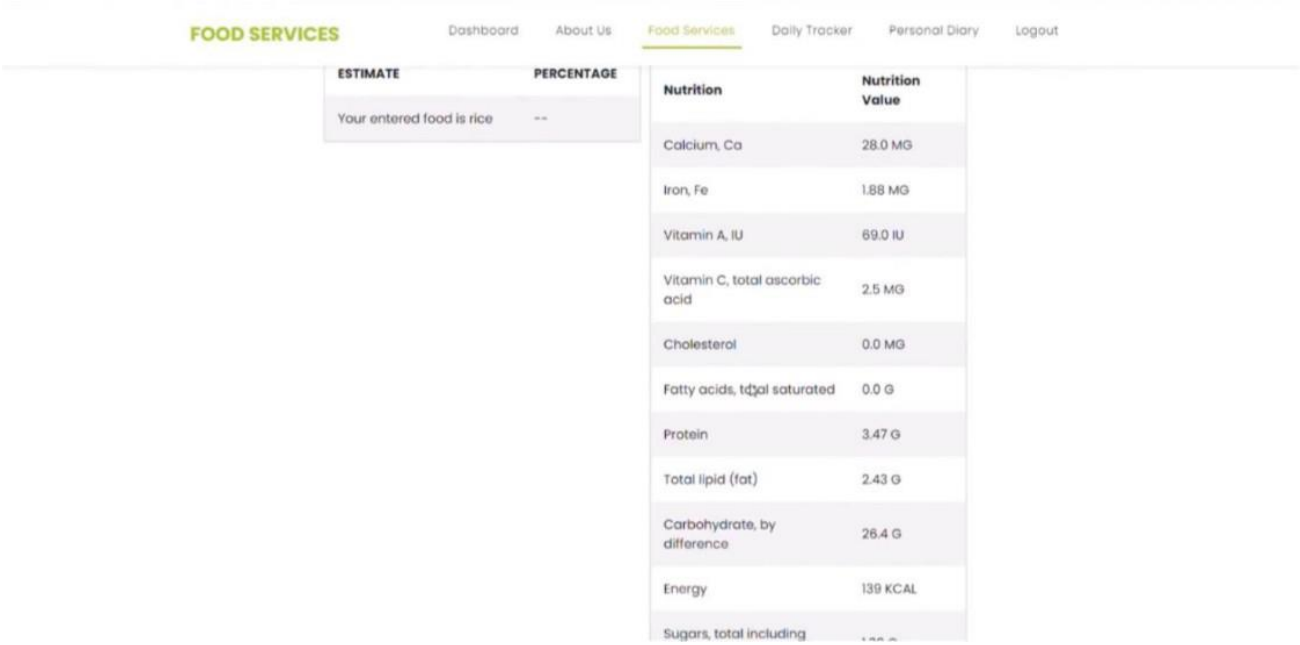
RESULTS

Performance 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** in shown on the screen







ADVANTAGES

- It provides a maintained strategy of healthy eating habits.

- It delivers information on the nutritional value of foods and howbalanced and healthy eating habits are important for us.

- It limits the amount of unnecessary food such as fat that peopleconsumea lot.

CONCLUSION

In conclusion, many people have become aware of their health. Moreover, they arealso 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.

FUTURE SCOPE

Nutrition assistants help dietitians 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.

APPENDIX

1) Source Code

```
from flask import Flask,render_template,request,redirect,url_for ,session
import ibm_db
import re
import os
import math
import random
import smtplib
import requests
app=Flask(__name__,template_folder='templates',static_folder='static')
app.secret_key='a'
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=3
print("successfully connected")
@app.route('/')
def home():
    return render_template('index.html')


@app.route('/login',methods=['GET','POST'])
def login():
    global userid
    msg=''

    if request.method=='POST':
        username=request.form.get('username',False)
        password=request.form.get('password',False)
        sql='SELECT * FROM USER 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['Logged in']=True
            session['id']=account['USERNAME']
            userid=account['USERNAME']
            session['username']=account['USERNAME']
            msg='Logged in successfully'
```

```
            msg="You have successfully registered"
            return render_template('verify.html',msg=msg)
    elif request.method=="POST":
        msg="Please fill out the form"
    return render_template('register.html',msg=msg)

@app.route('/welcome')
def welcome():
    return render_template('welcome.html')

@app.route('/verify')
def verify():
        email=request.args.get('email', None)
        server=smtplib.SMTP('smtp.gmail.com',587)
        server.starttls()
        password="nsgeuedwbzptosyp"
        server.login(email,password)
        otp=''.join([str(random.randint(0,9))for i in range(4)])
        msg=' YOUR OTP IS'+str(otp)
        server.sendmail(email,email,msg)
        server.quit()
        if request.method=='POST':
            verify=request.method['code']
            if verify==otp:
                return render_template('login.html')
        return render_template('verify.html')

@app.route('/frgpwd', methods=['GET','POST'])
def frgpwd():
    msg =" "
    print(request.form)
    username1=request.form.get("uname", False)
    oldpassword=request.form.get("oldpassword", False)
    newpassword=request.form.get("newpassword", False)
    sql='SELECT * FROM USER WHERE username=?'
    stmt=ibm_db.prepare(conn,sql)
    ibm_db.bind_param(stmt,1,username1)
```

```python
            return render_template('dash.html')
        else:
            msg='Incorrect username/password'
    return render_template('login.html',msg=msg)

@app.route('/register',methods=['GET','POST'])
def register():
    msg=''
    if request.method =='POST':
        username=request.form['username']
        email=request.form['email']
        password=request.form['password']
        Firstname=request.form['firstname']
        lastname=request.form['lastname']
        #phoneno=request.form['phoneno']
        sql='SELECT * FROM USER WHERE username=?'
        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:
            msg="Account already exist!"
        elif not re.match(r'[^@]+@[^@]+\.[^@]+',email):
            msg="Invalid email address"
        elif not re.match(r'[A-Za-z0-9]+',username):
            msg="name must contain character and numbers"

        else:
            insert_sql='INSERT INTO USER values(?,?,?,?,?)'
            prep_stmt=ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt,1,username)
            ibm_db.bind_param(prep_stmt,2,email)
            ibm_db.bind_param(prep_stmt,3,password)
            ibm_db.bind_param(prep_stmt,4,Firstname)
            ibm_db.bind_param(prep_stmt,5,lastname)
            ibm_db.execute(prep_stmt)
```

```python
        chgpwd_sql='UPDATE USER SET password = ? WHERE username = ?'
        prep_stmt=ibm_db.prepare(conn, chgpwd_sql)
        ibm_db.bind_param(prep_stmt,1,newpassword)
        ibm_db.bind_param(prep_stmt,2,username1)
        ibm_db.execute(prep_stmt)
        msg="You have successfully changed password"
        return render_template('forgot password.html',msg=msg)
    return render_template('forgot password.html',msg=msg)

url = "https://low-carb-recipes.p.rapidapi.com"

headers = {
  "x-rapidapi-key": "ad933ea36amsh6b0a83e514b1a58p14bc9ejsne745a5851a1b",
  "x-rapidapi-host":  "low-carb-recipes.p.rapidapi.com"
  }

searchForRecipes = "/search"
getRecipe="/recipes/"
getImage="/images/2807982c-986a-4def-9e3a-153a3066af7a.jpeg"
getRandomRecipe="/random"

@app.route('/login/dash')
def dashboard():
    return render_template('dash.html')

@app.route('/login/dash/viewprofile')
def viewprofile():
    username=session['id']
    sql='SELECT * FROM USER 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:
        return  render_template('viewprofile.html')
    else:
```

```python
@app.route('/login/dash/viewprofile/personinfo',methods=['GET','POST'])
def per_info():
    msg=''
    if request.method =='POST':
        Name=request.form['Name']
        gender=request.form['gender']
        tar_weight=request.form['Target Weight']
        Age=request.form['Age']
        Height=request.form['Height']
        Weight=request.form['Weight']
        email=request.form['email']
        location=request.form['location']
        phoneno=request.form['phoneno']
        sql='SELECT * FROM USER WHERE username=?'
        stmt=ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,Name)
        ibm_db.execute(stmt)
        account=ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            insert_sql='INSERT INTO USER values(?,?,?,?,?,?,?)'
            prep_stmt=ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt,1,Name)
            ibm_db.bind_param(prep_stmt,2,gender)
            ibm_db.bind_param(prep_stmt,3,Age)
            ibm_db.bind_param(prep_stmt,4,Height)
            ibm_db.bind_param(prep_stmt,5,Weight)
            ibm_db.bind_param(prep_stmt,7,location)
            ibm_db.execute(prep_stmt)
            msg="Your details are successfully stored"
            return render_template('viewprofile.html',msg=msg)
    elif request.method=="POST":
        msg="Please fill out the form"
    return render_template('personal info.html',msg=msg)
```

```python
        if account:
            insert_sql='INSERT INTO USER values(?,?,?)'
            prep_stmt=ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt,1,Name)
            ibm_db.bind_param(prep_stmt,2,email)
            ibm_db.bind_param(prep_stmt,3,Feedback)
            ibm_db.execute(prep_stmt)
            msg="Your Feedback has been stored"
            return render_template('ratings.html',msg=msg)
    elif request.method=="POST":
        msg="Please fill out the form"
    return render_template('ratings.html',msg=msg)

@app.route('/dash/view recipe')
def search_page():
  #session ['item']=request.form.get("Ingridients", False)
  return render_template('search.html')
@app.route('/recipes')
def get_recipes():
  #food=session['item']
  if (str(request.args['ingridients']).strip() != ""):
      print(request.args['ingridients'])
      # If there is a list of ingridients -> list
      querystring = {"name":request.args['ingridients'],"tags":request.args['tag'],"includeIngredients":request.args['included'],"exclude
      response = requests.request("GET", url + searchForRecipes, headers=headers, params=querystring)
      data=response.json()
      return render_template('recipes.html', recipes=data)
  else:
      # Random recipes
      response = requests.request("GET", url+ getRandomRecipe  , headers=headers)
      data=response.json()
      return render_template('recipes.html', recipes=data)

@app.route('/recipe')
def get_recipe():
  recipe_id = request.args['id']
  recipe_info_endpoint = "/recipes/{0}".format(recipe_id)
```

```
        data=response.json()
        return render_template('recipes.html', recipes=data)

@app.route('/recipe')
def get_recipe():
  recipe_id = request.args['id']
  recipe_info_endpoint = "/recipes/{0}".format(recipe_id)
  print(recipe_info_endpoint)
  recipe_info = requests.request("GET", url + recipe_info_endpoint, headers=headers)
  data=recipe_info.json()
  return render_template('recipe.html', recipe=data)


@app.route('/logout')
def logout():
    session.pop('loggedin',None)
    session.pop('id',None)
    session('username',None)
    return render_template("index.html")

if __name__=="__main__":
    app.run(debug=True ,host='0.0.0.0',use_reloader=False)
```

2) GitHub

https://github.com/IBM-EPBL/IBM-Project-47955-1660803563

3) Project Demo Link

https://drive.google.com/file/d/1fBzJgK8U861eC8IkAxpqvuwj2hcm2YB5/view?usp=share_link