Project report

IBM
Nutrition Assistant Application

Project done by,

TEAM ID:PNT2022TMID45649

ALJAJITH S

MALATHI T

PRATHEENA G

SYEDSAFIULLAH S

BARATHI S

TABEL OF CONTENTS

1. INTRODUCTION

1.1 Overview

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.

1.2 Purpose

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 Al-Driven Food Detection Model for accurate food identification and Food API's to give the nutritional value of the identified food.

2. LITERATURE SURVEY

2.1 Existing problem

In this busy world people can't track the food they consume and it is difficult to find the nutrients of all the food they consume. Over consumption or under nutrition can lead to serious health issues. These may be calcium/ iron/vitamin deficiencies or the over consumption of carbohydrates and sugar that causes obesity and diabetes. Which may further lead to serious health issues. There is urgent action required to maintain a balanced diet in order to have a good immunity.

2.2 References

1. Diptee Kumbhar, Sarita Patil " **Mobile cloud based system recognizing nutrition and freshness of food image,"** 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS),2017,pp.709-714, DOI:10.1109/ICECDS.2017.8389528

2. Ktenris N DiFilippo, Wen-Hao Huang, Karen M. Chapman-Novakofski," **The use of mobile apps to improve nutrition outcomes",** 2015 jul:21, pp-243-53,DOI: 10.1177/1357633X15572203

3. Jitao Yang, **"Personalized Nutrition Solution Based on Nutrigenomics",** 2019 19th International Conference on Computational Science and Its Applications (ICCSA),2019, pp. 73-103 ,DOI:10.1109/ICCSA.2019.00006

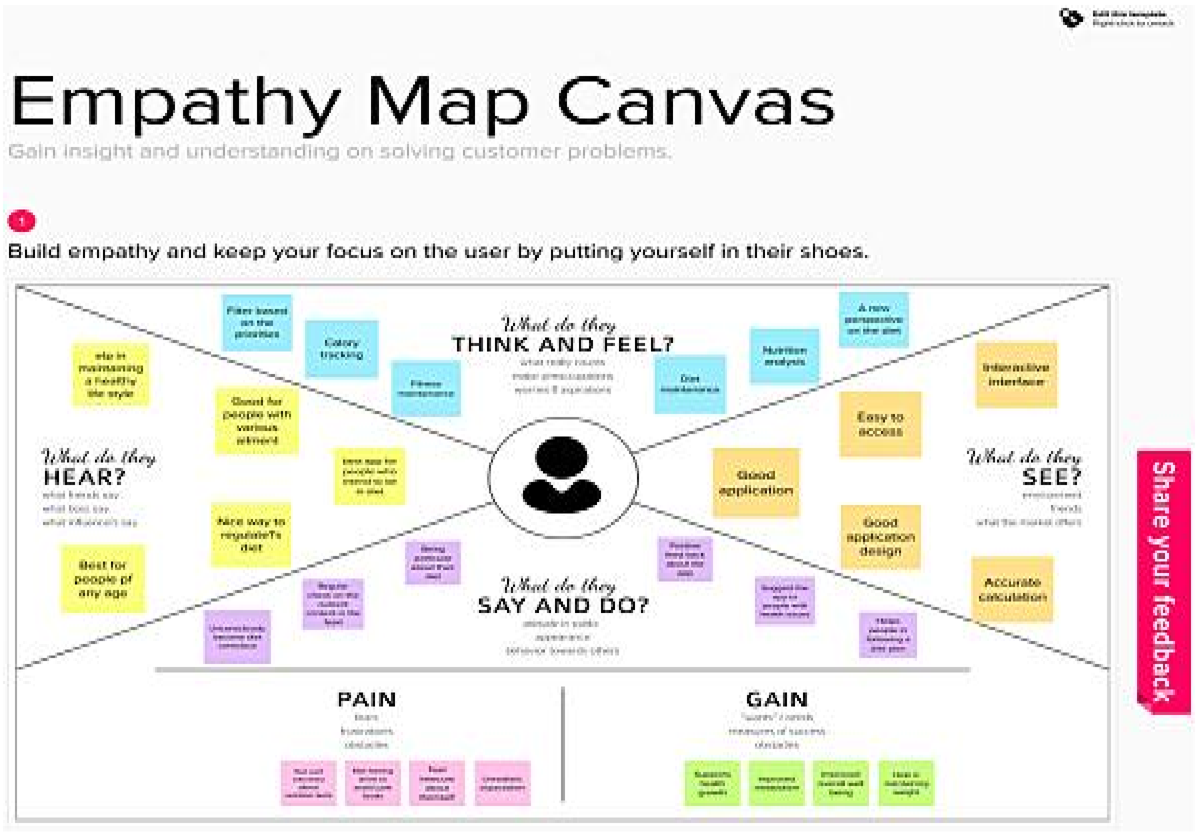
4. P.K. Paul1, P.S. Aithal2, A. Bhuimali3 "**Enhancing Cloud and Big Data Systems for healthy Food and Nutrition Information Systems Practice: A Conceptual Study"2019.**

5. Manju P George, C. A. Kalpana **"Development of a cloud-based solution for effective nutrition intervention in the management of lifestyle diseases"2020.**

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

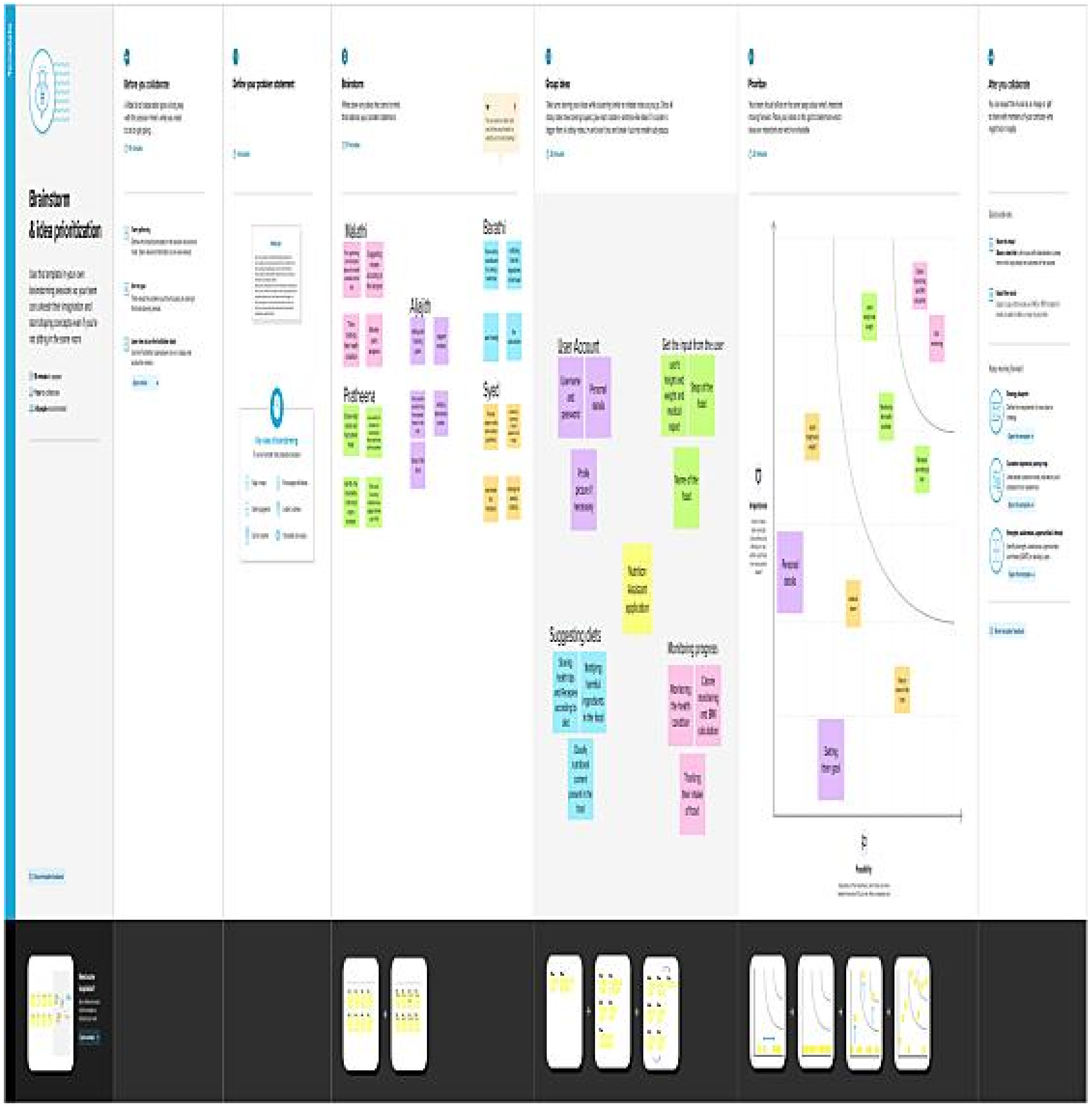
An empathy map is used to gain deeper insights on the customer's interaction with the system. It gives an idea on what the user feels and experiences while using the system, what fears the user has regarding the system, etc. It also specifies how supportive the system environment is and what the users are likely to hear from the people around them regarding the usage of the system



Empathy Map Canvas
Gain insight and understanding on solving customer problems.
Build empathy and keep your focus on the user by putting yourself in their shoes.
What do they THINK AND FEEL?
Filter based on the priorities
Calory tracking
Fitness maintenance
Diet maintenance
Nutrition analysis
A new perspective on the diet
What do they HEAR?
Good for people with various ailment
Nice way to regulate's diet
Best for people of any age
What do they SEE?
Interactive interface
Easy to access
Good application
Good application design
Accurate calculation
What do they SAY AND DO?
PAIN
GAIN
Share your feedback

3.2 Ideation & Brainstorming

Ideation and Brainstorming are performed to generate ideas and solutions. Brainstorming is a group activity unlike ideation.



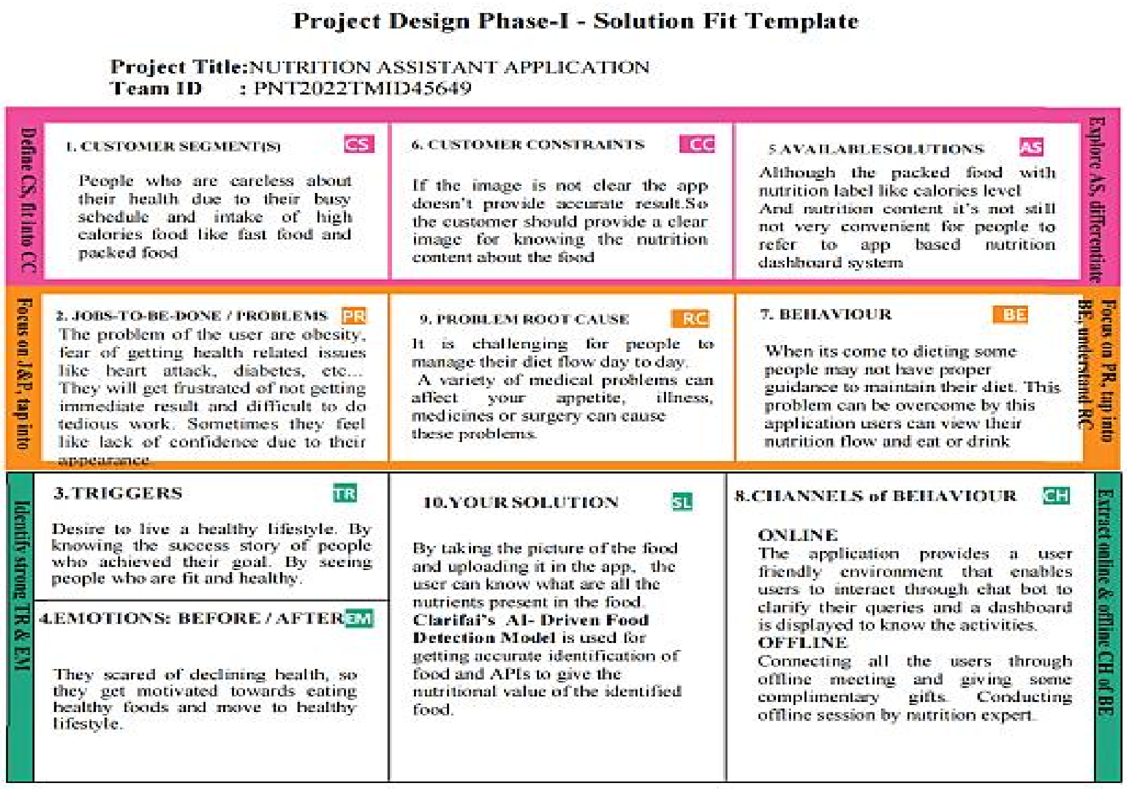
Brainstorm & idea prioritization
Before you collaborate
Define your problem statement
Brainstorm
Group ideas
Prioritize
After you collaborate
Malathi
Bharathi
Allajith
Pratheena
Syed
User Account
Username and password
Personal details
Profile picture if necessary
Get the input from the user
User's height and weight and medical report
Type of the food
Name of the food
Nutrition Assistant application
Suggesting diets
Storing health tips and recipes according to diet
Notifying harmful ingredients in the food
Classify nutritional content present in the food
Monitoring progress
Monitoring the health condition
Calorie monitoring and BMI calculation
Tracking their intake of food
Personal details
Setting their goal
Importance
Feasibility

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Rate of Obesity are increasing at an high speed,due to the ignorance of the proper Nutrition foods, and this leads to risks in 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,some food packaging has an added nutrition and calorie values,but it's not very comfortable to refer.
2.	Idea / Solution description	The solution is user can know the nutritional content of the food they are in taking,by taking picture of the food and upload it in the app.It is used for get accurate food identification and API's to give the nutritional value of the identified food.
3.	Novelty / Uniqueness	Providesauser-friendly environment.provides recipes according to their diet.Provides different ways to access the nutritional information about the food by taking the snap of the food,upload in the gallery and entering manually.
4.	Social Impact / Customer Satisfaction	Getting feedback from the users for enhancement and giving notification on their diet plans and goal tracking.
5.	Business Model (Revenue Model)	Social media is the best way to spread the word about our application.And with the influences we can attract the normal people.Subscription or membership will have extra benefits.

6.	Scalability of the Solution	People can access it from anywhere at anytime to track the calories and nutrition value that will improve a healthy eating pattern.This app will improves the dietary habits and helps in maintaining healthy weight and healthy lifestyle.

3.4 Problem Solution fit


Define CS, fit into CC
Explore AS, differentiate
Focus on J&P, tap into
Focus on PR, tap into BE, understand RC
Identify strong TR & EM
Extract online & offline CH of BE

Project Design Phase-I - Solution Fit Template

Project Title:NUTRITION ASSISTANT APPLICATION
Team ID : PNT2022TMID45649

1. CUSTOMER SEGMENT(S) CS

People who are careless about their health due to their busy schedule and intake of high calories food like fast food and packed food

6. CUSTOMER CONSTRAINTS CC

If the image is not clear the app doesn't provide accurate result.So the customer should provide a clear image for knowing the nutrition content about the food

5.AVAILABLESOLUTIONS AS

Although the packed food with nutrition label like calories level And nutrition content it's not still not very convenient for people to refer to app based nutrition dashboard system

2. JOBS-TO-BE-DONE / PROBLEMS PR

The problem of the user are obesity, fear of getting health related issues like heart attack, diabetes, etc... They will get frustrated of not getting immediate result and difficult to do tedious work. Sometimes they feel like lack of confidence due to their appearance

9. PROBLEM ROOT CAUSE RC

It is challenging for people to manage their diet flow day to day.
A variety of medical problems can affect your appetite, illness, medicines or surgery can cause these problems.

7. BEHAVIOUR BE

When its come to dieting some people may not have proper guidance to maintain their diet. This problem can be overcome by this application users can view their nutrition flow and eat or drink

3.TRIGGERS TR

Desire to live a healthy lifestyle. By knowing the success story of people who achieved their goal. By seeing people who are fit and healthy.

4.EMOTIONS: BEFORE / AFTER EM

They scared of declining health, so they get motivated towards eating healthy foods and move to healthy lifestyle.

10.YOUR SOLUTION SL

By taking the picture of the food and uploading it in the app, the user can know what are all the nutrients present in the food. **Clarifai's AI- Driven Food Detection Model** is used for getting accurate identification of food and APIs to give the nutritional value of the identified food.

8.CHANNELS of BEHAVIOUR CH

ONLINE
The application provides a user friendly environment that enables users to interact through chat bot to clarify their queries and a dashboard is displayed to know the activities.
OFFLINE
Connecting all the users through offline meeting and giving some complimentary gifts. Conducting offline session by nutrition expert.

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Project Description:

This Nutrition Assistant Application project is aimed at developing a desktop-based application for estimates food attributes such as ingredients and nutritional value by classifying the input images of food. This application provides efficient knowledge about nutrition content in the food which helps to make their body more healthy and strong. It refers to the system and processes to help the user to analyze the intake of food with the involvement of a Technology system by the information given by every user. This system can be used to store the details of the user's health, calculate BMI of user, update the status of their health condition based on the information provided, and generate health reports weekly or monthly based. This application's major role is to control their daily calorie intake by eating healthier foods, which is the most basic method to avoid obesity and health issue. Without proper diet control, and this will 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 is categorizing individual health condition of the user. This application provides a healthy life to the user.

Scope:

1. **Increase Usability:** It a user-friendly application. In the part of user's just internet is enough to access the news, notification, updates and the other content provided by the admin regarding their health condition.
2. **Maintains good health:** The application will help them personally without going to the doctor. It provide better education of healthy diet and nutrition. It can help in guiding them how to remain healthy and how to take good nutrition.
3. **Health conscious:** This will provide convenience to persons/users who wants to learn about nutrition and other health topics.
4. **Functional Limitations:** 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, updates, etc

Purpose:

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 most important thing in a healthy life. It is more than just obtaining nutrition and calories by food. It's more than just eating the healthy stuff. It's more than just following the fat diet. I believe the purpose of the nutrition is to nourish the body and soul. The food we eat and the way we eat it, is an integral part of the life.It helps the users to eat nutritional rich food which yield to lead a healthy life.

IDENTIFIER	REQUIREMENTS
1. Add health information	This application will allow to add health related information of the user.
2. Delete health information	This application will allow to delete the unwanted details about their health.
3. Categories of nutritional food	The categories of food.
4. View of Dashboard	Application will allow user to view the dashboard containing nutrition details.
5. Mail Notification	This application will allow to send mail notification to user when there are any issues regarding their health.
6. Tracking System	The health can be tracked with this application.
7. Graph analysis	This application will demonstrate health condition by means of nutritional content.
8. Identifying the high calorie food	The high calorie ingredients will be shown via this application.
9. Identifying the low calorie food	The high calorie ingredients will be shown via this application.
10. Passcode	This application has the option to set a passcode to keep their medical reports safe.
11. Add multiple accounts	This application has the option of creating multiple accounts for the users.
12. Update account	This application will allow the user to update their profile.

13. Selection of health report duration	This application has the ability to select the duration for displaying the health report as weekly or monthly.
14. Add an account	This application will allow the user to add their profile.
15. Pupation of nutritional trends	This application will allow constant review of nutritional trends and pupation.
16. PDF report	This application will generate the pdf report of medical analysis.
17. Delete account	This application will allow the user to delete their profile.

4.2 Non-Functional requirements

Security — User's information and their nutritional content are secured.

Performance - The prediction process begins when the image is uploaded. It performs according to the sources of the image and provides the specific nutritional information.

Reliability — The contents provided are based on the nutritional value based on the image. The image upload consistently performs well.

Availability - The image uploaded is used for prediction of nutritional value.

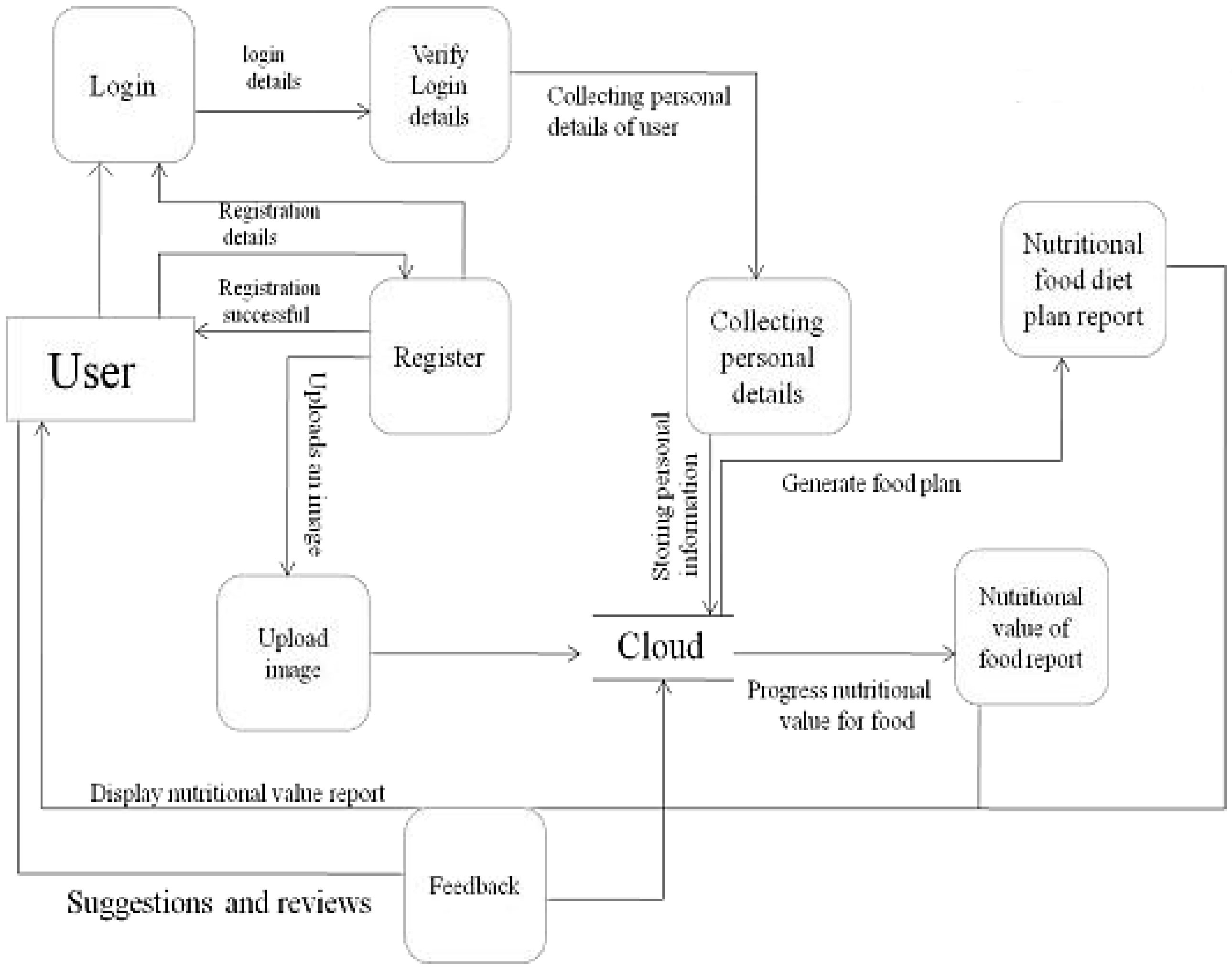
Scalability - Increasing the accuracy of nutritional value prediction of the food image uploaded.

Usability —Nutritional contents are provided based on the image uploaded by Clarifai's Al.

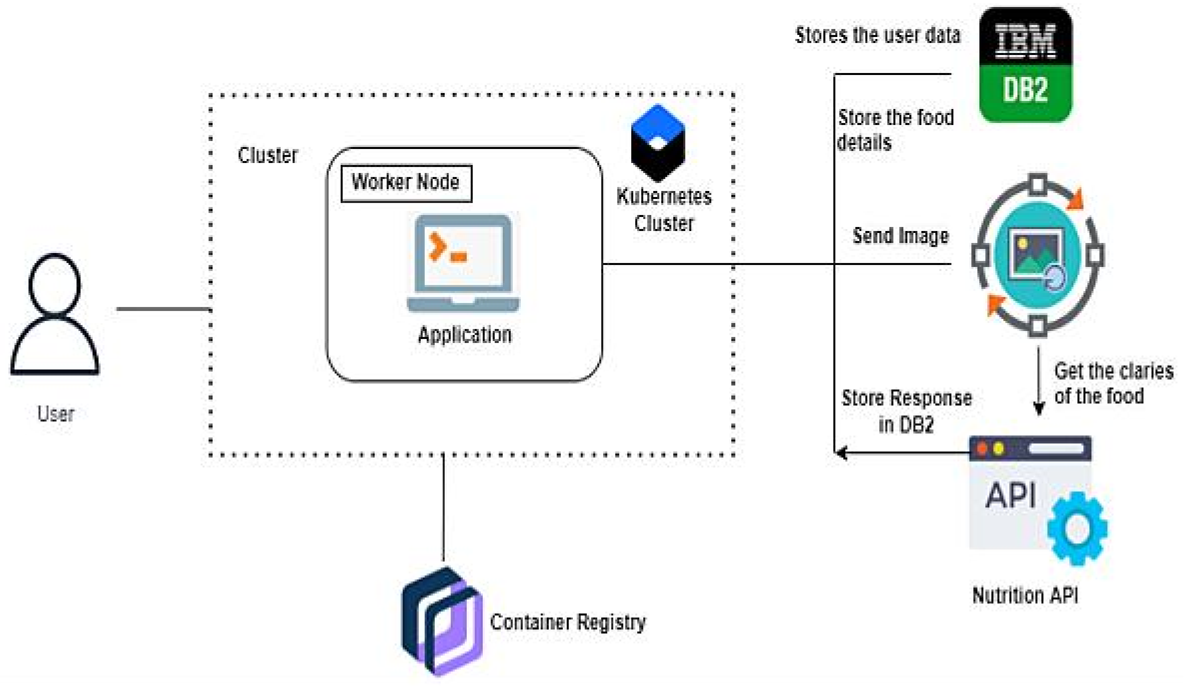
5. PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.


Login
login details
Verify Login details
Collecting personal details of user
Registration details
Registration successful
User
Register
Uploads an image
Collecting personal details
Storing personal information
Nutritional food diet plan report
Generate food plan
Upload image
Cloud
Nutritional value of food report
Progress nutritional value for food
Display nutritional value report
Feedback
Suggestions and reviews

5.2 Technical Architecture



5.3 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 usercan registerthe applicationbyentering email,password,andconfirm password.	I can access my account / dashboard	High	Sprint
	Registration	USN-2	As a user,will receive confirmation email once user have registered for the application	I can receive confirmation email & click confirm	High	Sprint
	Login	USN-3	As a user can log into the application by entering email & password	I can login when passwordand email are correct	High	Sprint
	Collecting personal details	USN-4	As a user can provide a personal information for processing	I can enter the personal details	Medium	Sprint
	Upload image	USN-5	As a user can upload an image for the processing of food.	I can upload a food image.	High	Sprint
	Feedback	USN-6	As a user can give feedback	I can give feedback about the application	Low	Sprint
Cloud	Nutritional value of report	USN-7	In cloud the food image is processed and provides the nutritional value of food.	It gives the nutritional value of food.	High	Sprint
	Nutritional Fooddiet plan report	USN-8	In cloud the food diet plan based on nutritional value is generated based on the personal information provided by the user.	It provides the diet nutritional plan.	Medium	Sprint

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

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	Aljajith S Malathi T Pratheena G Syedsafiullah S Barathi S
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application.	1	High	Aljajith S Malathi T Pratheena G Syedsafiullah S Barathi S
Sprint-1	User details	USN-3	As a user, I can log into the application by entering email & password.	1	High	Aljajith S Malathi T Pratheena G Syedsafiullah S Barathi S
Sprint-2	Login	USN-4	As a user, I can fill the Details.	2	High	Aljajith S Malathi T Pratheena G Syedsafiullah S Barathi S

Sprint -3	Push notification	USN-5	As a user, I can fill the Details.	2	Medium	Aljajith S Malathi T Pratheena G Syedsafiullah S Barathi S
Sprint -4	Shown the nutrition Recipe for scanned food	USN-6	As a user, I can scan the food an get the nutrition details and recipe for related scanned	1	High	Aljajith S Malathi T Pratheena G Syedsafiullah S Barathi S

6.2 Sprint Delivery Schedule

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$Av = \frac{\text{Sprint duration}20}{\text{Velocity}10}$$

Average Velocity = Story Points per Day

Sprint Duration = Number of

(Duration) days per Sprint

Velocity = Points per Sprint

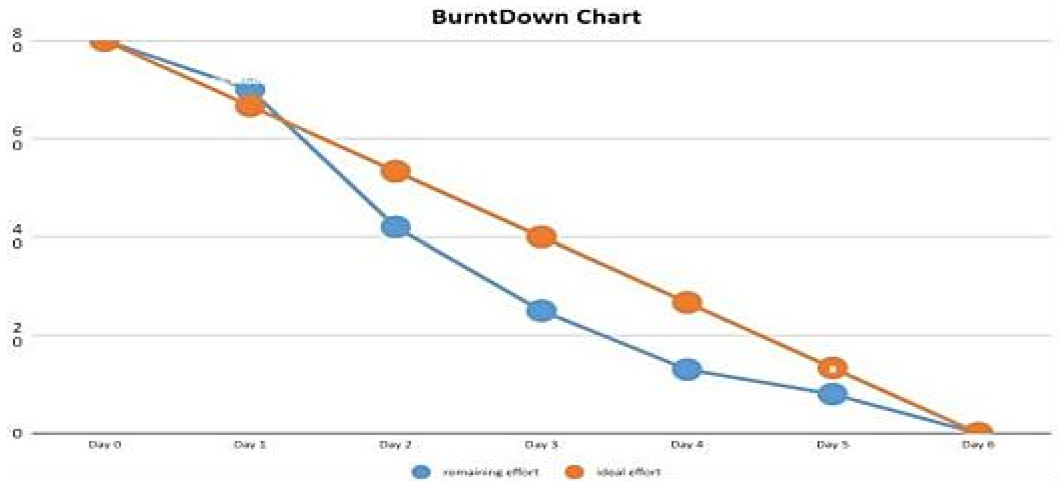
20 AV=≈ 46

Therefore, the **AVERAGE VELOCITY IS 4 POINTS PER SPRINT**

Burn down Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

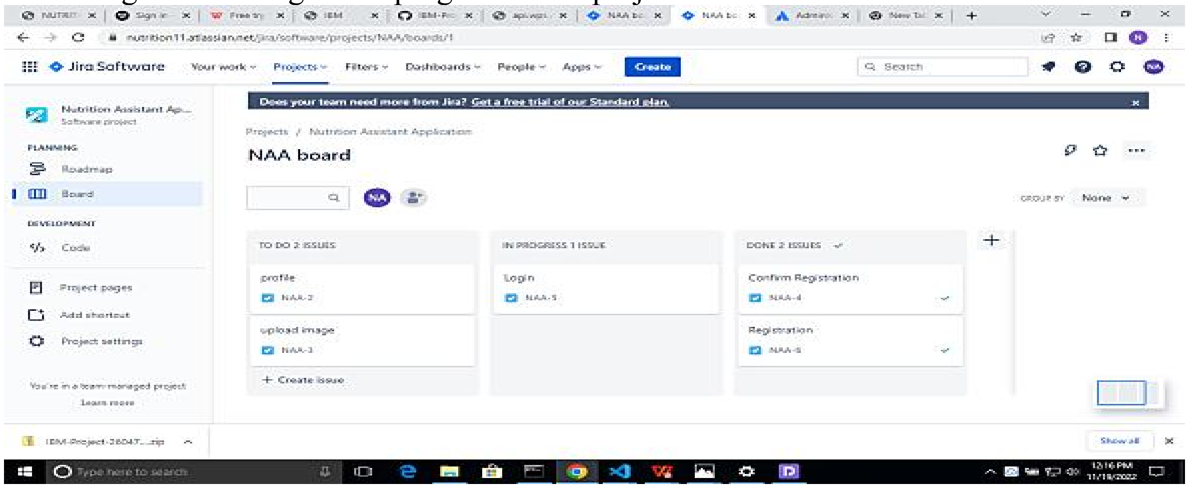
	Initial estimate						
Spring number	Day 0	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
Sprint-1	20	0	10	5	3	1	1
Sprint-2	20	2	10	4	1	1	2
Sprint-4	20	3	3	3	3	3	5
Remaining effort	80	70	42	25	13	8	0
Ideal effort	80	66.66666667	53.33333333	40	26.66666667	13.33333333	0



6.3 Reports from JIRA

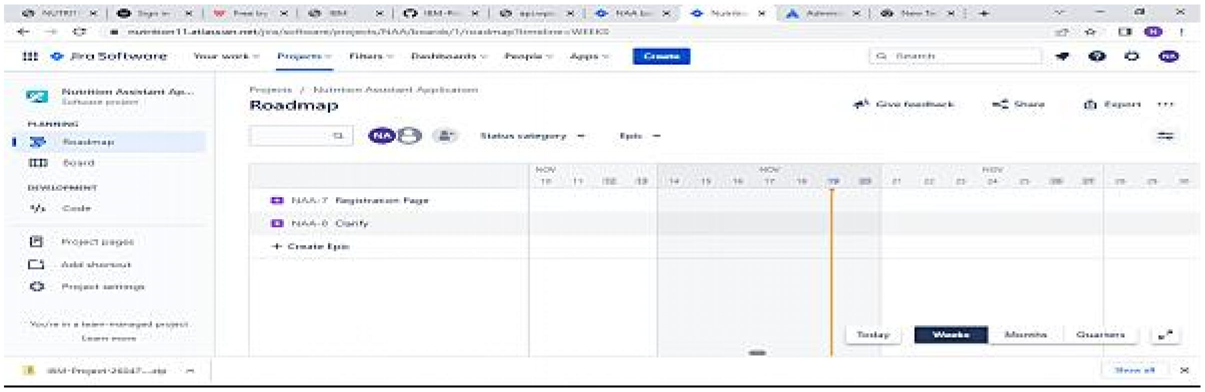
Board

A board reflects your team's process, tracking the status of work. The columns on the board represent the status of your team's issues. The visual representation of the work helps in discussing and tracking of the progress of the project from start to finish.



Roadmap

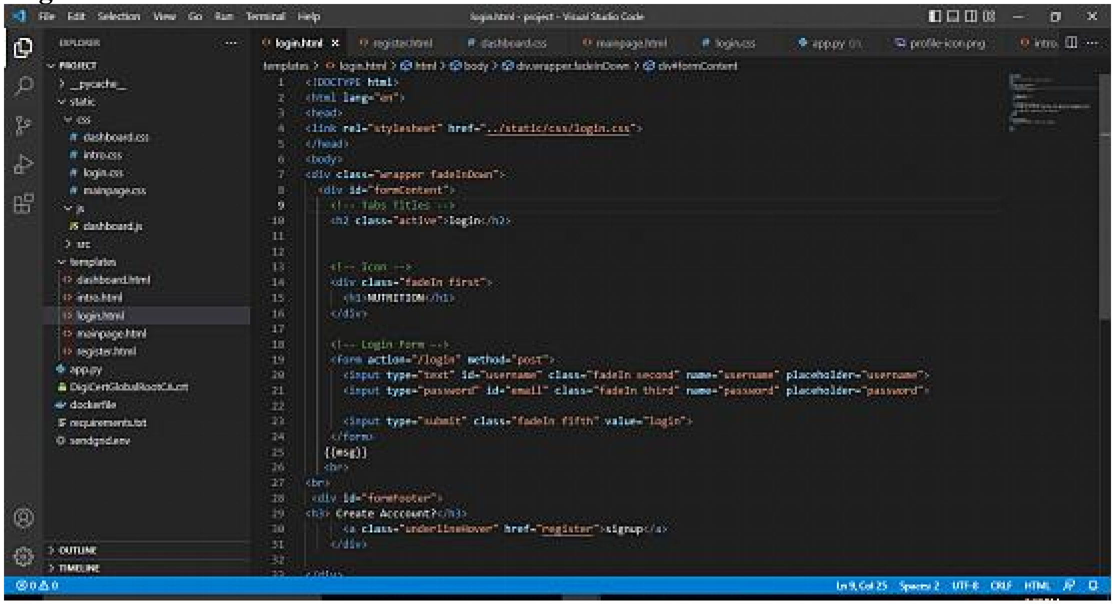
A roadmap offers quick and easy planning that helps teams better manage their dependencies and track progress on the big picture in real-time.



7. CODING & SOLUTIONING

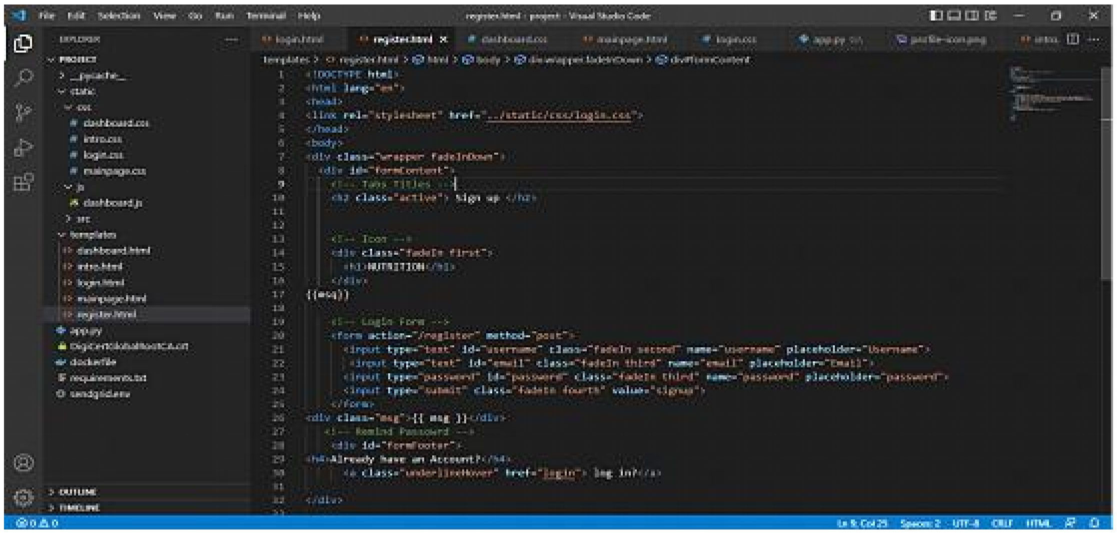
7.1 Feature

Login.html



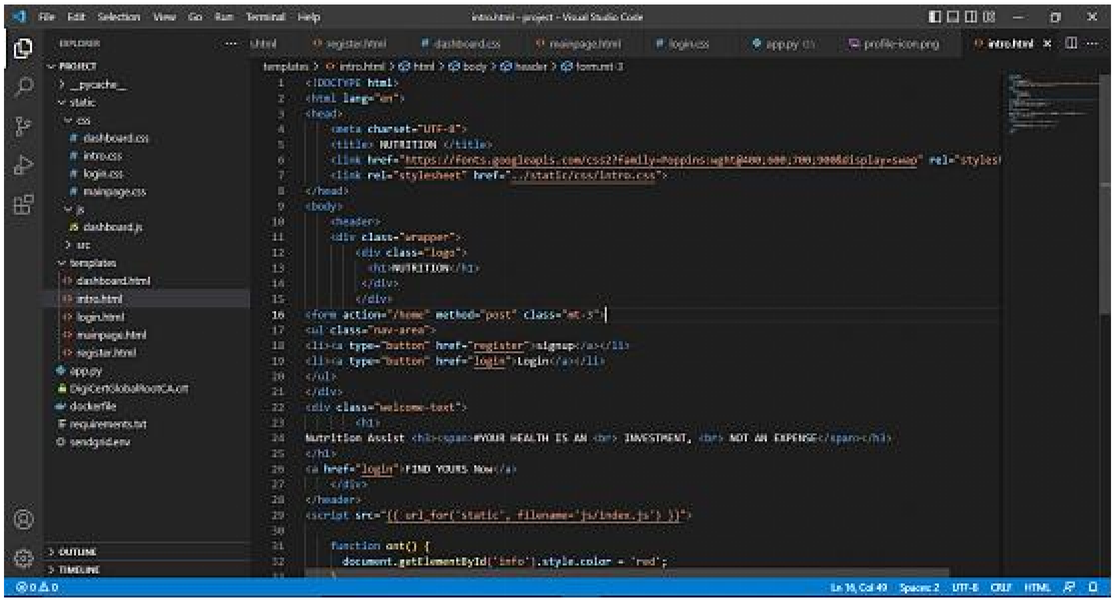
```html
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="stylesheet" href="../static/css/login.css">
</head>
<body>
<div class="wrapper fadeInDown">
  <div id="formContent">
    <!-- Tabs Titles -->
    <h2 class="active">login</h2>


    <!-- Icon -->
    <div class="fadeIn first">
      <h1>NUTRITION</h1>
    </div>

    <!-- Login Form -->
    <form action="/login" method="post">
      <input type="text" id="username" class="fadeIn second" name="username" placeholder="username">
      <input type="password" id="email" class="fadeIn third" name="password" placeholder="password">

      <input type="submit" class="fadeIn fifth" value="login">
    </form>
   {{msg}}
   <br>
<br>
 <div id="formFooter">
<h3> Create Account?</h3>
      <a class="underlineHover" href="register">signup</a>
    </div>

</div>
```

Register.html



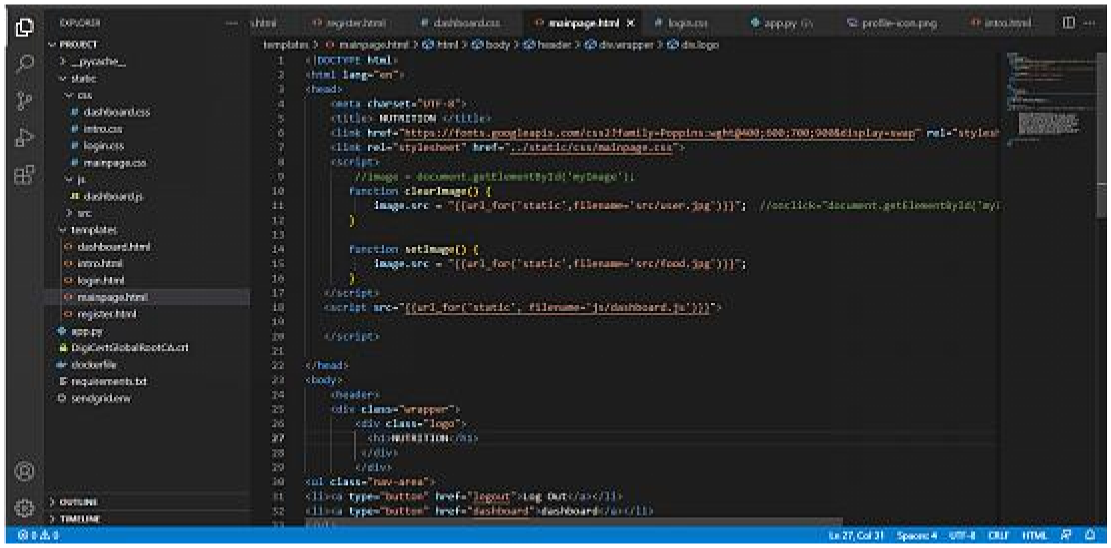
```html
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="stylesheet" href="../static/css/login.css">
</head>
<body>
<div class="wrapper fadeInDown">
  <div id="formContent">
    <!-- Tabs Titles -->
    <h2 class="active"> Sign up </h2>


    <!-- Icon -->
    <div class="fadeIn first">
      <h1>NUTRITION</h1>
    </div>
{{msg}}

    <!-- Login Form -->
    <form action="/register" method="post">
      <input type="text" id="username" class="fadeIn second" name="username" placeholder="Username">
       <input type="text" id="email" class="fadeIn third" name="email" placeholder="Email">
      <input type="password" id="password" class="fadeIn third" name="password" placeholder="password">
      <input type="submit" class="fadeIn fourth" value="signup">
    </form>
<div class="msg">{{ msg }}</div>
   <!-- Remind Passowrd -->
    <div id="formFooter">
<h4>Already have an Account?</h4>
      <a class="underlineHover" href="login"> log in?</a>

</div>
```

Intro.html



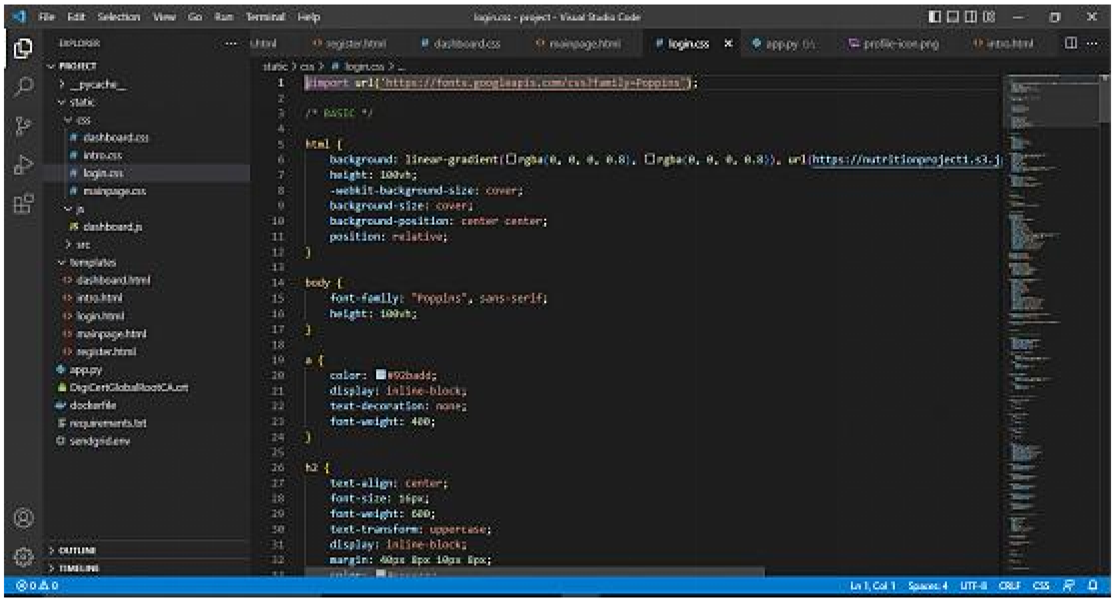
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title> NUTRITION </title>
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700;900&display=swap" rel="styles
    <link rel="stylesheet" href="../static/css/intro.css">
</head>
<body>
    <header>
    <div class="wrapper">
        <div class="logo">
         <h1>NUTRITION</h1>
        </div>
        </div>
<form action="/home" method="post" class="mt-5">
<ul class="nav-area">
<li><a type="button" href="register">signup</a></li>
<li><a type="button" href="login">Login</a></li>
</ul>
</div>
<div class="welcome-text">
        <h1>
Nutrition Assist <h1><span>#YOUR HEALTH IS AN <br> INVESTMENT, <br> NOT AN EXPENSE</span></h1>
</h1>
<a href="login">FIND YOURS Now</a>
    </div>
</header>
<script src="{{ url_for('static', filename='js/index.js') }}">

    function ont() {
     document.getElementById('info').style.color = 'red';
```

Mainpage.html



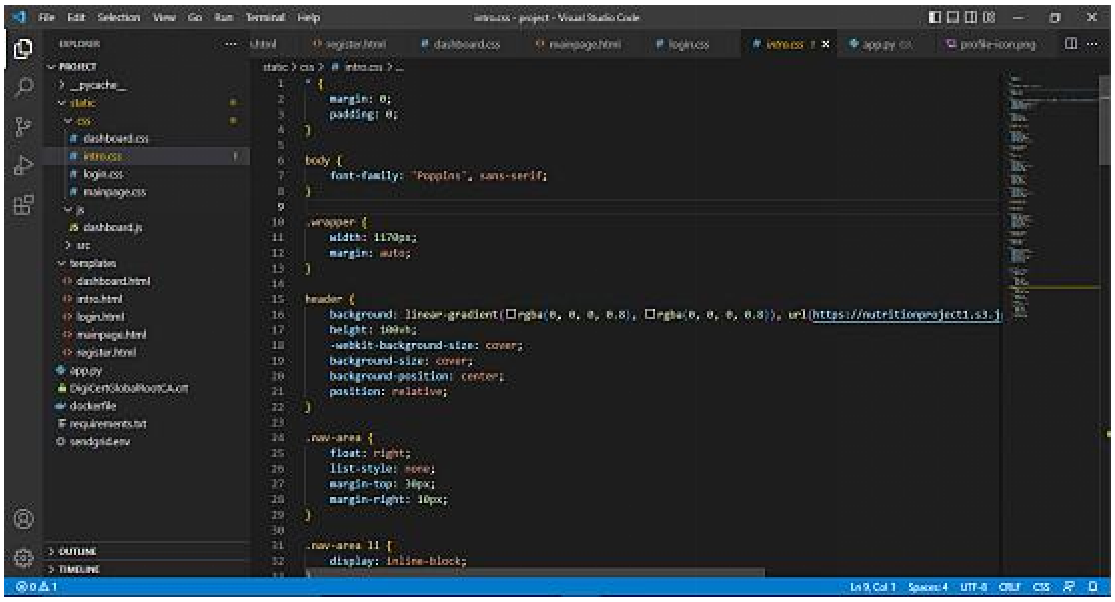
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title> NUTRITION </title>
    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700;900&display=swap" rel="styles
    <link rel="stylesheet" href="../static/css/mainpage.css">
    <script>
      //image = document.getElementById('myImage');
      function clearImage() {
          image.src = "{{url_for('static',filename='src/user.jpg')}}";  //onclick="document.getElementById('my
      }

      function setImage() {
          image.src = "{{url_for('static',filename='src/food.jpg')}}";
      }
   </script>
   <script src="{{url_for('static', filename='js/dashboard.js')}}">

   </script>

</head>
<body>
    <header>
    <div class="wrapper">
        <div class="logo">
         <h1>NUTRITION</h1>
        </div>
        </div>
<ul class="nav-area">
<li><a type="button" href="logout">Log Out</a></li>
<li><a type="button" href="dashboard">dashboard</a></li>
```

Login.css



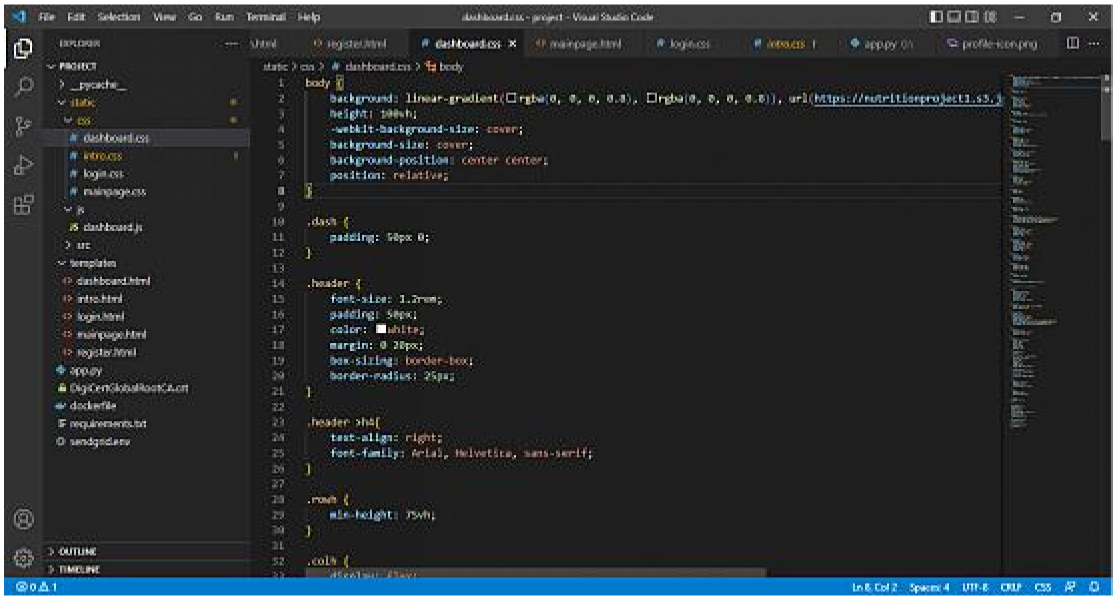
```css
@import url('https://fonts.googleapis.com/css?family=Poppins');

/* BASIC */

html {
    background: linear-gradient(rgba(0, 0, 0, 0.8), rgba(0, 0, 0, 0.8)), url(https://nutritionproject1.s3.j
    height: 100vh;
    -webkit-background-size: cover;
    background-size: cover;
    background-position: center center;
    position: relative;
}

body {
    font-family: "Poppins", sans-serif;
    height: 100vh;
}

a {
    color: #92badd;
    display: inline-block;
    text-decoration: none;
    font-weight: 400;
}

h2 {
    text-align: center;
    font-size: 16px;
    font-weight: 600;
    text-transform: uppercase;
    display: inline-block;
    margin: 40px 8px 10px 8px;
```

Intro.css



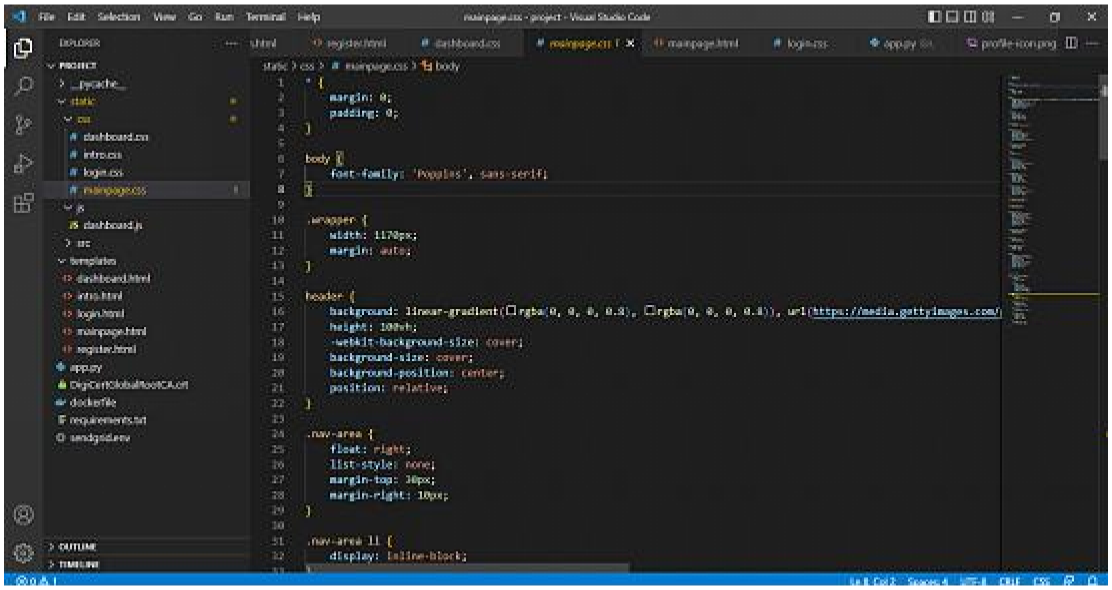
```css
* {
    margin: 0;
    padding: 0;
}

body {
    font-family: "Poppins", sans-serif;
}

.wrapper {
    width: 1170px;
    margin: auto;
}

header {
    background: linear-gradient(rgba(0, 0, 0, 0.8), rgba(0, 0, 0, 0.8)), url(https://nutritionproject1.s3.j
    height: 100vh;
    -webkit-background-size: cover;
    background-size: cover;
    background-position: center;
    position: relative;
}

.nav-area {
    float: right;
    list-style: none;
    margin-top: 30px;
    margin-right: 10px;
}

.nav-area li {
    display: inline-block;
```

Dashboard.css



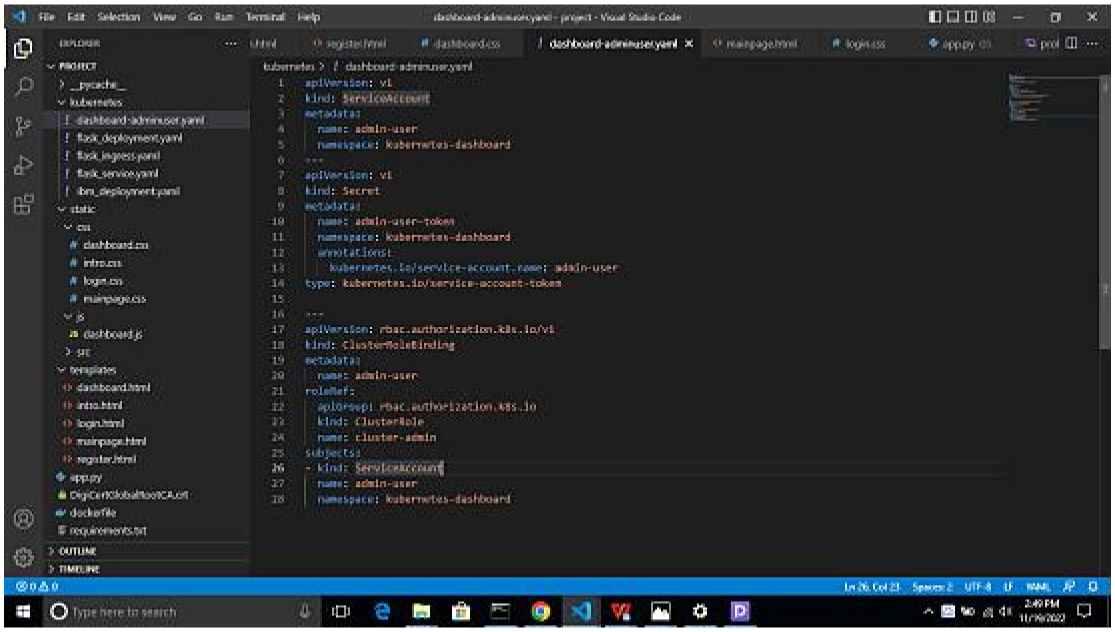
```
body {
    background: linear-gradient(rgba(0, 0, 0, 0.8), rgba(0, 0, 0, 0.8)), url(https://nutritionproject1.s3.
    height: 100vh;
    -webkit-background-size: cover;
    background-size: cover;
    background-position: center center;
    position: relative;
}

.dash {
    padding: 50px 0;
}

.header {
    font-size: 1.2rem;
    padding: 50px;
    color: white;
    margin: 0 20px;
    box-sizing: border-box;
    border-radius: 25px;
}

.header >h4{
    text-align: right;
    font-family: Arial, Helvetica, sans-serif;
}

.rowh {
    min-height: 75vh;
}

.colh {
    display: flex;
```

Mainpage.css



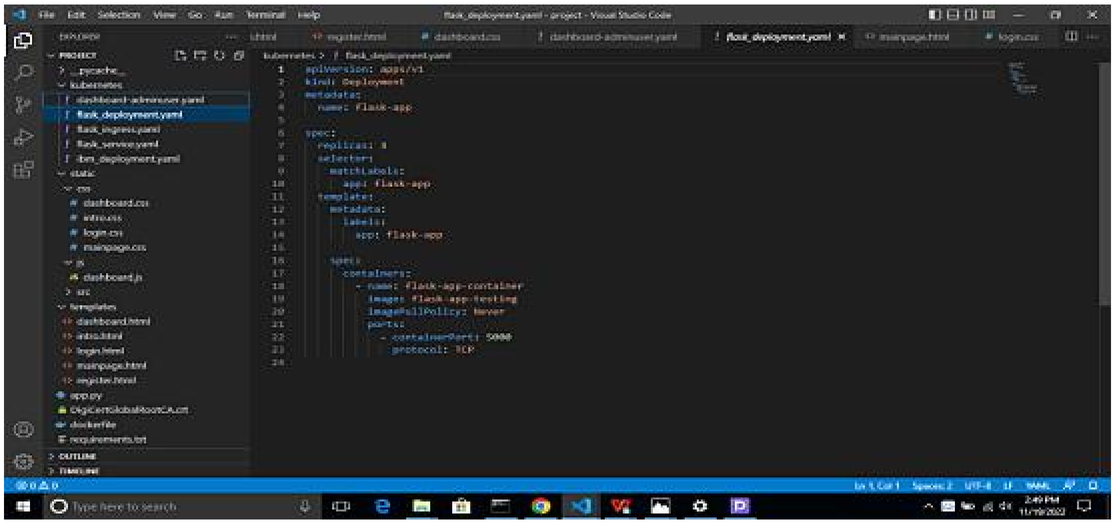
```
* {
    margin: 0;
    padding: 0;
}

body {
    font-family: 'Poppins', sans-serif;
}

.wrapper {
    width: 1170px;
    margin: auto;
}

header {
    background: linear-gradient(rgba(0, 0, 0, 0.3), rgba(0, 0, 0, 0.8)), url(https://media.gettyimages.com/
    height: 100vh;
    -webkit-background-size: cover;
    background-size: cover;
    background-position: center;
    position: relative;
}

.nav-area {
    float: right;
    list-style: none;
    margin-top: 30px;
    margin-right: 10px;
}

.nav-area li {
    display: inline-block;
```

Dashboard adminuser.yaml



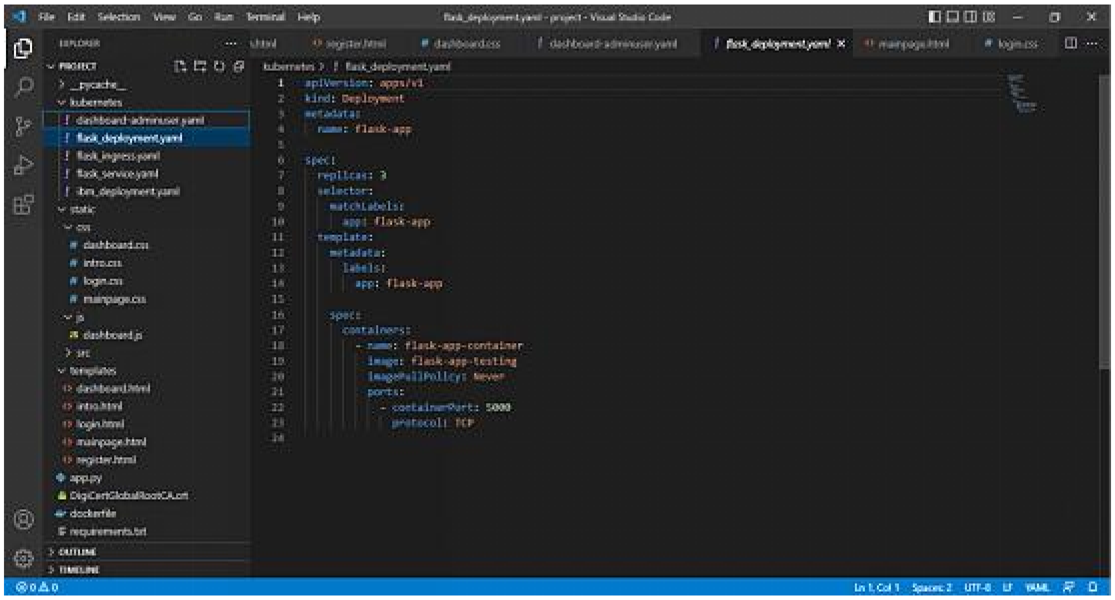
```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard
---
apiVersion: v1
kind: Secret
metadata:
  name: admin-user-token
  namespace: kubernetes-dashboard
  annotations:
    kubernetes.io/service-account.name: admin-user
type: kubernetes.io/service-account-token

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin-user
  namespace: kubernetes-dashboard
```

flask_development.yaml



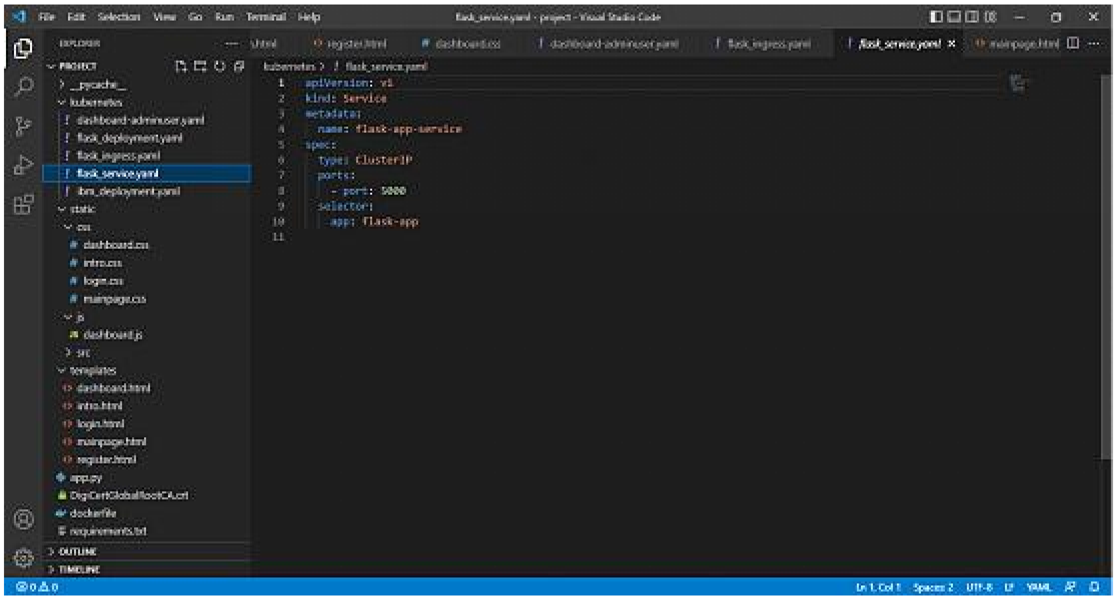
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: flask-app

spec:
  replicas: 8
  selector:
    matchLabels:
      app: flask-app
  template:
    metadata:
      labels:
        app: flask-app

    spec:
      containers:
        - name: flask-app-container
          image: flask-app-testing
          imagePullPolicy: Never
          ports:
            - containerPort: 5000
              protocol: TCP
```

flask_ingress.yaml



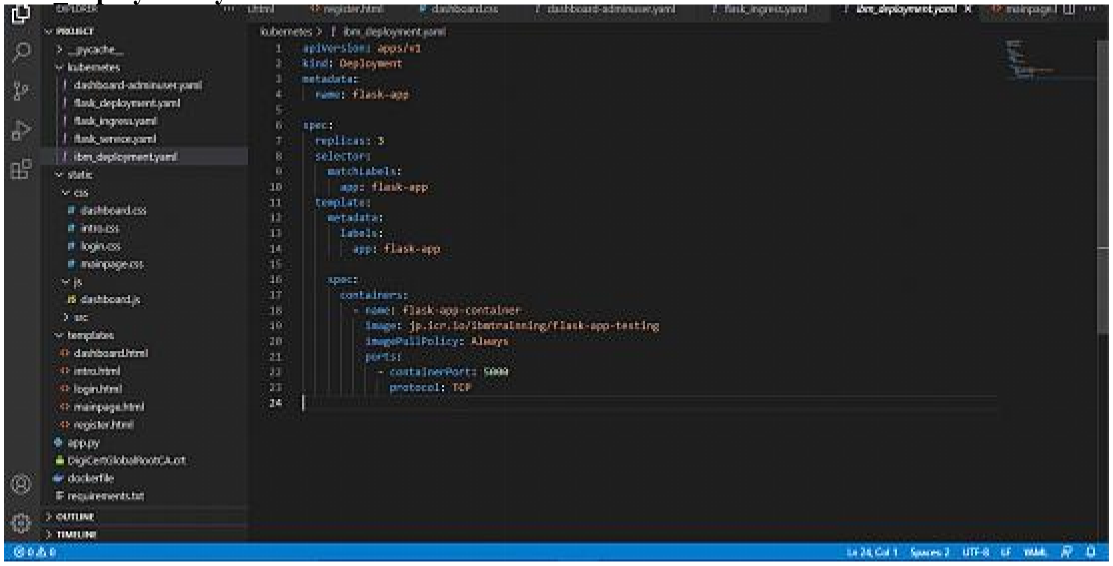
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: flask-app

spec:
  replicas: 3
  selector:
    matchLabels:
      app: flask-app
  template:
    metadata:
      labels:
        app: flask-app

    spec:
      containers:
        - name: flask-app-container
          image: flask-app-testing
          imagePullPolicy: Never
          ports:
            - containerPort: 5000
              protocol: TCP
```

Flask_service.yaml



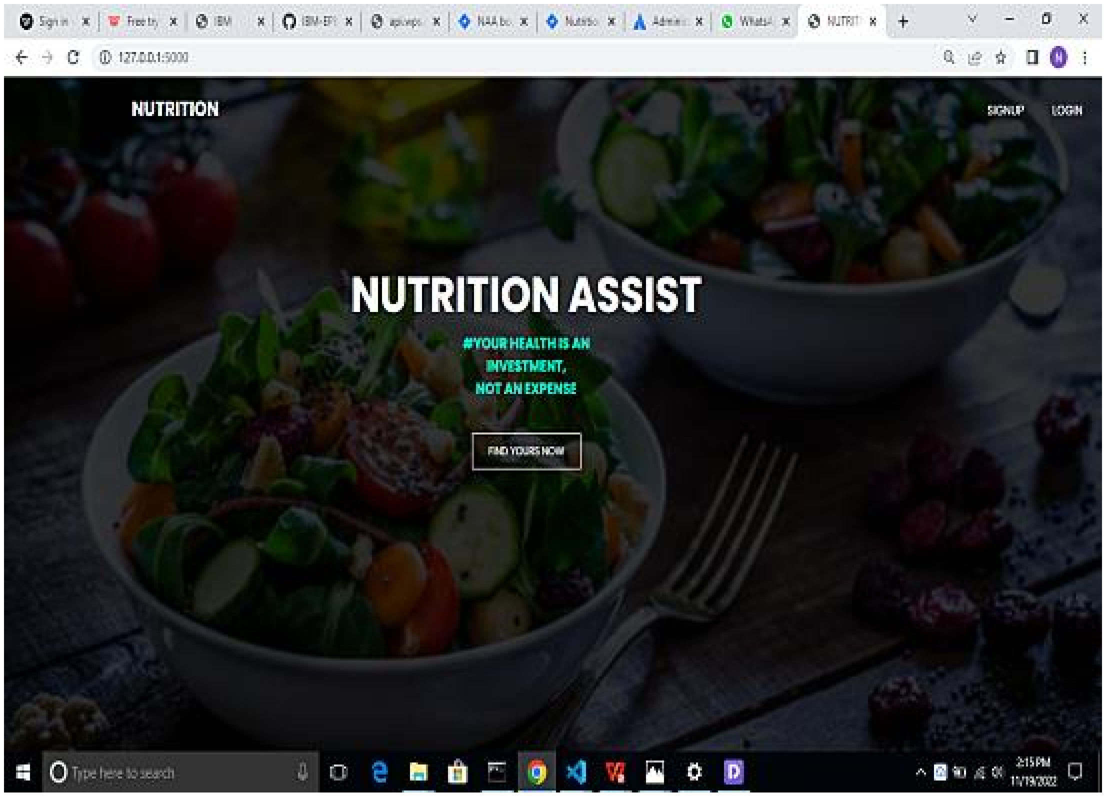
```
apiVersion: v1
kind: Service
metadata:
  name: flask-app-service
spec:
  type: ClusterIP
  ports:
    - port: 5000
  selector:
    app: flask-app
```

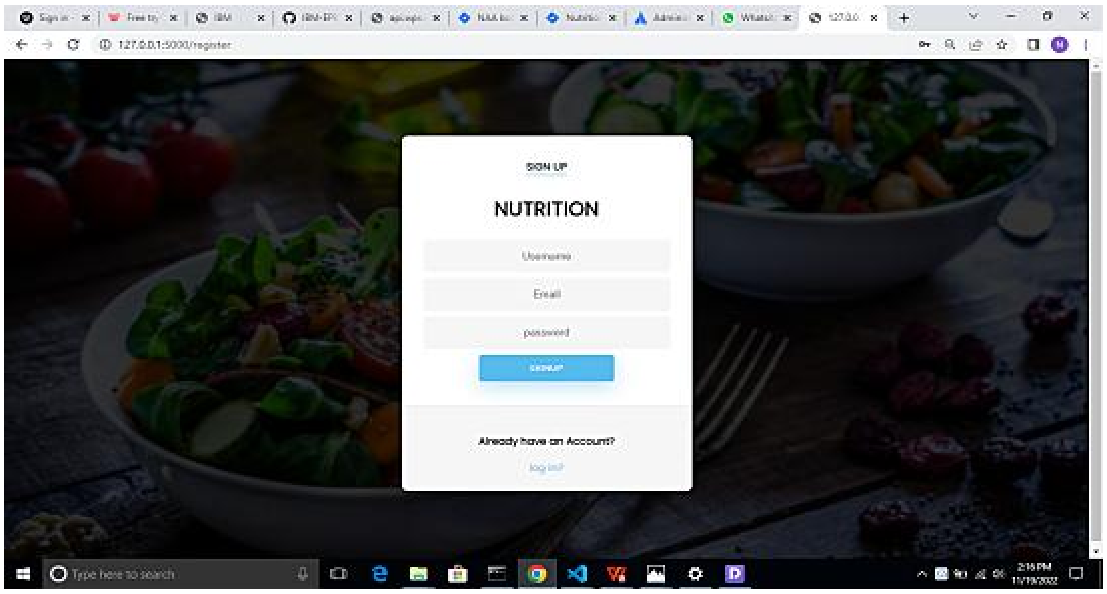
Ibm_deployment.yaml

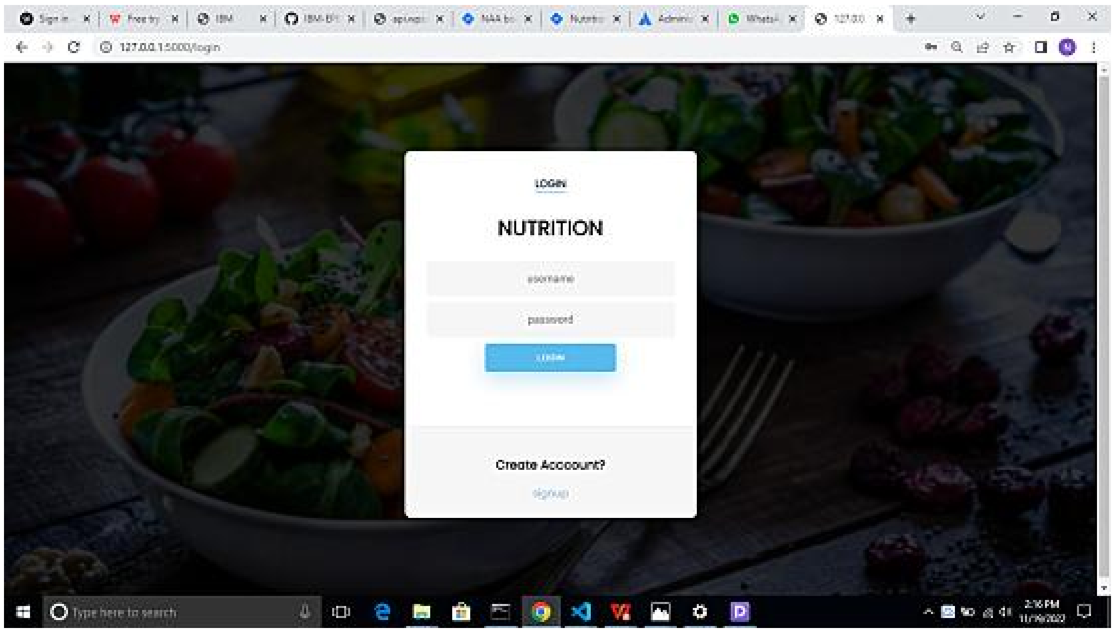


```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: flask-app

spec:
  replicas: 3
  selector:
    matchLabels:
      app: flask-app
  template:
    metadata:
      labels:
        app: flask-app

    spec:
      containers:
        - name: flask-app-container
          image: jp.icr.io/ibmtraining/flask-app-testing
          imagePullPolicy: Always
          ports:
            - containerPort: 5000
              protocol: TCP

```

Result




127.0.0.1:5000/register
SIGN UP
NUTRITION
Username
Email
password
SIGNUP
Already have an Account?
log in?
Type here to search


127.0.0.1:5000/login
LOGIN
NUTRITION
username
password
LOGIN
Create Acccount?
signup
Type here to search

8. TESTING

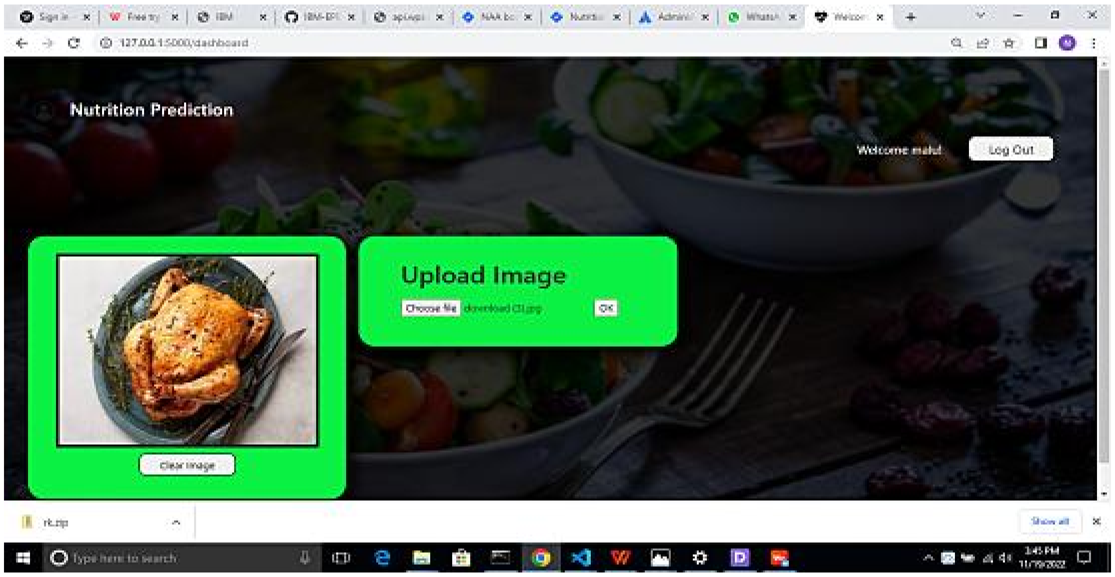
8.1 User acceptance test

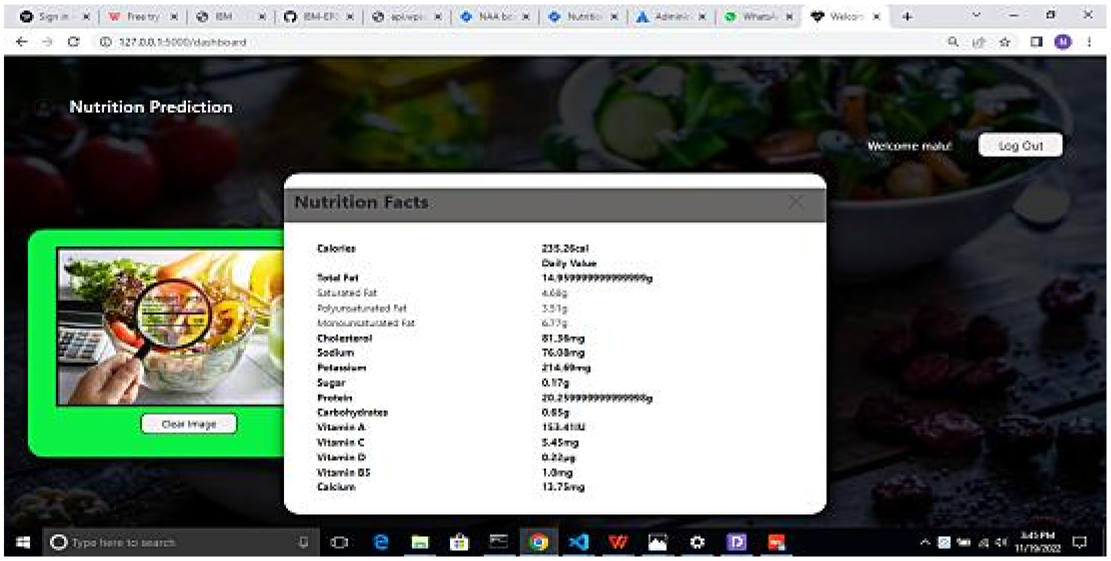
Before deploying the software application to a production environment the end user or client performs a type of testing known as user acceptance testing, or UAT to ensure whether the software functionalities serve the purpose of development.

9. RESULT

9.1 Performance Metrics

Metrics are a baseline for performance tests. Monitoring the correct parameters will help you detect areas that require increased attention and find ways to improve them.





Nutrition Facts

Calories	235.26cal
	Daily Value
Total Fat	14.959999999999999g
Saturated Fat	4.68g
Polyunsaturated Fat	3.51g
Monounsaturated Fat	6.77g
Cholesterol	81.36mg
Sodium	76.08mg
Potassium	214.69mg
Sugar	0.17g
Protein	20.259999999999998g
Carbohydrates	0.65g
Vitamin A	153.41IU
Vitamin C	5.45mg
Vitamin D	0.22µg
Vitamin B5	1.0mg
Calcium	13.75mg

10. ADVANTAGES & DISADVANTAGES

Advantages:

1. Early detection of health problems.
2. Easy to know about the nutrition values.
3. User friendly and gives accurate suggestions.

Disadvantages:

1. Requires training the system with a large dataset.
2. Works only on the pretrained Images.
3. Users may not have time to upload the image before eating.
4. The image uploaded should be clear to get accurate results

11. CONCLUSION

Hence a system that takes in images as user input, analyzes those and identifies the Nutritional content,and gives all the ingredients present in the image with its nutritional content.

12. FUTURE SCOPE

The system must be trained with numerous images of food and suggest some healthy recipes for the same . Help users to connect with other users and share their feedback .

13. APPENDIX

Source Code

Dashboard.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi
" crossorigin="anonymous">
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw
3" crossorigin="anonymous"></script>
    <link rel="stylesheet" href="{{url_for('static', filename='css/dashboard.css')}}">
    <link rel="icon" href="{{ url_for( 'static', filename = 'src/cardiogram.png')}}">
    <title>Welcome {{session.username}}</title>
</head>


<body>

    {% if msg %}
    <div class="msg bg-info" style="padding: 0px 0 0px 50px;margin: 20px 20px 0
20px;border-radius: 20px;">
        <h4>{{msg}}</h4>
    </div>
    {% endif %}


    <div class="container-fluid dash">
        <div class="header p-3">
            <h3><img src="{{url_for('static', filename='src/user.jpg')}}" alt="ico"
width="50px" height="50px">  Nutrition Prediction</h3>
            <div style="display:flex; justify-content: right;align-items: center;color:
white;">Welcome {{session.username}}!  <form action="" method="post"
enctype="multipart/form-data"><button type="submit" name="logout"
class="combutton btns">Log Out</button></form>
            </div>
        </div>
    </div>
            </div>
            <div class="col-lg-8 row colh">
                <div class="row normsize">
```

```
                    <div class="col-lg normsize roudcorner comcolor">
 <div class="comflex-col">
                            <img id="myImage" class="normsize" style="border: 5px solid
rgb(25, 25, 25);;" src="{{url_for('static', filename='src/food.jpg')}}" alt="food"
width="300" height="300">
                            <button class="combutton btns" onclick="setImage()" >Clear
Image</button>
                        </div>
                    </div>
                    <div class="col-lg normsize roudcorner comcolor">
                        <div class="comflex lesssize normpadding">
                            <div>
                                <h1>Upload Image</h1>
                                <form action="{{url_for('upload_file')}}" method="post"
enctype="multipart/form-data">
                                    <input type=file onchange="readURL(this);" name="file">
                                    <input style="margin: 10px 0px;" onclick="setImage()"
type=submit value=OK name="OK">
                                </form>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
    {% if data %}
    <dic class="container-fluid float">
        <div class="containers floatcontainer ">
            <div class="box1">
                <div class="close">
                    <a href="{{url_for('upload_file',methods='POST')}}"
class="closes"></a>
                </div>
            </div>
            <div style="background-color: rgb(105, 102, 102);margin-top: 25px;font-
size: 30px;font-weight: bold;padding-left: 15px;"><p>Nutrition Facts</p></div>
            <div class="box2">
                <div class="bcol">
                    <table style="width:100%;">
                        <tr>
                          <th>Calories</th>
                          <th>{{data[0]}}{{unit[0]}}</th>
                        </tr>
                        <tr>
                          <th></th>
                          <th>Daily Value</th>
                        </tr>
                        <tr>
                          <th>Total Fat</th>
                          <th>{{data[1]}}{{unit[1]}}</th>
                        </tr>
                        <tr>
                            <td>Saturated Fat</td>
    <td>{{data[2]}}{{unit[2]}}</td>
</tr>
<tr>
    <td>Polyunsaturated Fat</td>
    <td>{{data[3]}}{{unit[3]}}</td>
</tr>
<tr>
    <td>Monounsaturated Fat</td>
    <td>{{data[4]}}{{unit[4]}}</td>
</tr>
<tr>
    <th>Cholesterol</th>
    <th>{{data[5]}}{{unit[5]}}</th>
</tr>
<tr>
    <th>Sodium</th>
    <th>{{data[6]}}{{unit[6]}}</th>
</tr>
<tr>
    <th>Potassium</th>
    <th>{{data[7]}}{{unit[7]}}</th>
</tr>
<tr>
    <th>Sugar</th>
    <th>{{data[8]}}{{unit[8]}}</th>
  </tr>
  <tr>
  <th>Protein</th>
  <th>{{data[9]}}{{unit[9]}}</th>
</tr>
<tr>
  <th>Carbohydrates</th>
  <th>{{data[10]}}{{unit[10]}}</th>
</tr>
<tr>
  <th>Vitamin A</th>
  <th>{{data[11]}}{{unit[11]}}</th>
</tr>
<tr>
  <th>Vitamin C</th>
  <th>{{data[12]}}{{unit[12]}}</th>
</tr>
<tr>
  <th>Vitamin D</th>
  <th>{{data[13]}}{{unit[13]}}</th>
</tr>
<tr>
  <th>Vitamin B5</th>
  <th>{{data[14]}}{{unit[14]}}</th>
</tr>
<tr>
  <th>Calcium</th>
  <th>{{data[15]}}{{unit[15]}}</th>
</tr>
```

```
                </table>
             </div>

          </div>

       </div>
    </dic>
    {% endif %}


    <script>
         //image = document.getElementById('myImage');
        function clearImage() {
            image.src = "{{url_for('static',filename='src/user.jpg')}}";
//onclick="document.getElementById('myImage').src='src/omplate.png'"
        }

        function setImage() {
            image.src = "{{url_for('static',filename='src/food.jpg')}}";
        }
    </script>
    <script src="{{url_for('static', filename='js/dashboard.js')}}">

    </script>
</body>
</html>
```

Register.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="stylesheet" href="../static/css/login.css">
</head>
<body>
<div class="wrapper fadeInDown">
  <div id="formContent">
    <!-- Tabs Titles -->
    <h2 class="active"> Sign up </h2>


    <!-- Icon -->
    <div class="fadeIn first">
      <h1>NUTRITION</h1>
    </div>
{{msq}}

    <!-- Login Form -->
    <form action="/register" method="post">
      <input type="text" id="username" class="fadeIn second" name="username"
placeholder="Username">
       <input type="text" id="email" class="fadeIn third" name="email"
placeholder="Email">
      <input type="password" id="password" class="fadeIn third" name="password"
placeholder="password">
      <input type="submit" class="fadeIn fourth" value="signup">
    </form>
<div class="msg">{{ msg }}</div>
   <!-- Remind Passowrd -->
    <div id="formFooter">
<h4>Already have an Account?</h4>
      <a class="underlineHover" href="login"> log in?</a>

</div>

  </div>
</div>
</body>
</html>
```

Login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="stylesheet" href="../static/css/login.css">
</head>
<body>
<div class="wrapper fadeInDown">
  <div id="formContent">
    <!-- Tabs Titles -->
    <h2 class="active">login</h2>
    <!-- Icon -->
    <div class="fadeIn first">
      <h1>NUTRITION</h1>
    </div>

    <!-- Login Form -->
    <form action="/login" method="post">
```

```
     <input type="text" id="username" class="fadeIn second" name="username"
placeholder="username">
     <input type="password" id="email" class="fadeIn third" name="password"
placeholder="password">

     <input type="submit" class="fadeIn fifth" value="login">
   </form>
  {{msg}}
  <br>
<br>
 <div id="formFooter">
<h3> Create Acccount?</h3>
     <a class="underlineHover" href="register">signup</a>
   </div>

</div>
</body>
</html>
```

Mainpage.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title> NUTRITION </title>
   <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700;900
&display=swap" rel="stylesheet">
   <link rel="stylesheet" href="../static/css/mainpage.css">
   <script>
       //image = document.getElementById('myImage');
      function clearImage() {
         image.src             =             "{{url_for('static',filename='src/user.jpg')}}";
//onclick="document.getElementById('myImage').src='src/omplate.png'"
      }

      function setImage() {
         image.src = "{{url_for('static',filename='src/food.jpg')}}";
      }
   </script>
   <script src="{{url_for('static', filename='js/dashboard.js')}}">

   </script>

</head>
<body>
    <header>
    <div class="wrapper">
        <div class="logo">
          <h1>NUTRITION</h1>
         </div>
        </div>
<ul class="nav-area">
<li><a type="button" href="logout">Log Out</a></li>
<li><a type="button" href="dashboard">dashboard</a></li>
</ul>
</div>
<div class="welcome-text">
        <h1>
    Welcome {{session.username}}!<h3> <span>Check Nutrition Values of food
by click the dashboard</span></h3></h1>
</div>

            <p style="color:white;padding-top: 350px;">
              Nutritional support is the provision of adequate nutrients to maintain a
              healthy body weight and avoid malnutrition. The continuous delivery of
              high-quality and cost-effective nutritional care to patients has been
              shown to be an increasingly difficult task. It is observed that dieticians
              are requested to carry out the nutritional assessment, to manually
              calculate the nutritional needs and to design the everyday meal plan for
              each patient. In most cases, these time-consuming tasks are not
completed
              due to lack of time or inadequate number of personnel. Development of a
              computer assisted information tool with cloud-based on-line diet
              consultation module and comparison of its efficacy with one- to-one
              counselling would be efficiently utilized for client education intervention
              programs. The nutrient content calculation was planned to undertake
with
              commonly consumed traditional as well as junk foods
```

```
             </p>

    <form action="/mainpage" method="post">

</header>
</body>
</html>
```

Intro.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title> NUTRITION </title>
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700;900
&display=swap" rel="stylesheet">
    <link rel="stylesheet" href="../static/css/intro.css">
</head>
<body>
    <header>
    <div class="wrapper">
        <div class="logo">
          <h1>NUTRITION</h1>
         </div>
        </div>
<form action="/home" method="post" class="mt-3">
<ul class="nav-area">
<li><a type="button" href="register">signup</a></li>
<li><a type="button" href="login">Login</a></li>
</ul>
</div>
<div class="welcome-text">
        <h1>
Nutrition Assist <h3><span>#YOUR HEALTH IS AN <br> INVESTMENT, <br>
NOT AN EXPENSE</span></h3>
</h1>
<a href="login">FIND YOURS Now</a>
    </div>
</header>
```

```
<script src="{{ url_for('static', filename='js/index.js') }}">

    function ont() {
      document.getElementById('info').style.color = 'red';
    }
  </script>
</body>
</html>
```

App.py

```
import binascii
import math
import random
import requests as res
import secrets
import time
from base64 import urlsafe_b64encode as b64e, urlsafe_b64decode as b64d
from time import strftime, localtime
import re
import ibm_db
import sendgrid
from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel
from clarifai_grpc.grpc.api import resources_pb2, service_pb2, service_pb2_grpc
from clarifai_grpc.grpc.api.status import status_code_pb2
from cryptography.fernet import InvalidToken
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
from flask import Flask, render_template, request, session, redirect
from sendgrid import SendGridAPIClient
from markupsafe import escape
from sendgrid.helpers.mail import Mail, Email, To, Content

# clarifai
YOUR_CLARIFAI_API_KEY = "1b93f2034d514024a01206328d16a000"
YOUR_APPLICATION_ID = "Nutrition_assistant1"
channel = ClarifaiChannel.get_json_channel()
stub = service_pb2_grpc.V2Stub(channel)
metadata = (("authorization", f"Key {YOUR_CLARIFAI_API_KEY}"),)

# sendgrid
```

```
SENDGRID_API_KEY =
"xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"

# rapid API
url = "https://spoonacular-recipe-food-nutrition-
v1.p.rapidapi.com/recipes/parseIngredients"
querystring = {"includeNutrition": "true"}
headers = {"content-type": "application/x-www-form-urlencoded",
        "X-RapidAPI-Key":
"cce727c5c8msha93ab918e2a4963p137240jsn9e783821969c",
        "X-RapidAPI-Host": "spoonacular-recipe-food-nutrition-v1.p.rapidapi.com"
        }

ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'jfif'}

KEY = "2480387791346406708896352768923 1"

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=19af6446-6171-
4641-8aba-
9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30699;SE
CURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=yyx69722;
PWD=2YqarEmzriL08SP7","","")

print(conn)

app = Flask(__name__)

app.secret_key = "\xfd{H\xe5<\x95\xf9\xe3\x96.5\xd1\x01O<!\xd5\xa2\xa0\x9fR"

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

def send_confirmation_mail(user, email):
   message = email(
      from_email = "nutritionapplication1@gmail.com",
      to_emails = email,
      subject = "Congrats! Your Account was created Successfully",
      html_content = f"<strong>Congrats {user}!</strong><br>Account Created
with username {email}"
   )
```

```
    SENDGRID_API_KEY='SG.J2dZQyDITtGMgU-I1s7Nvw._iDky0C2fBHQ-
073TmWnOIwKYekJGsFAAbphUtxjFwI'
    try:
        sg = SendGridAPIClient(SENDGRID_API_KEY)
        response = sg.send(message)
        print(response.status_code,response.body)
        #print(response.body)
        #print(response.headers)
    except Exception as e:
        print(f"Some error in sendgrid, {e}")



@app.route('/login', methods =['GET', 'POST'])
def login():
    msg = ''
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        sql = "SELECT * FROM Database 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['id'] = account['id']
            session['username'] = username
            msg = 'Logged in successfully !'
            return render_template('mainpage.html', msg = msg)
        else:
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg = msg)

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



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


    if request.method=='POST':
        username = request.form['username']
        password = request.form['password']
        email = request.form['email']
        print(username ,password)
        sql = "SELECT * FROM Database 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:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        elif not username or not password or not email:
            msg = 'Please fill out the form !'
        else:
          insert_sql = "INSERT INTO Database 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.execute(prep_stmt)
          msg = 'You have successfully registered !'
    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('register.html', msg = msg)
```

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


@app.route('/dashboard', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        # check if the post request has the file part
        if 'logout' in request.form:
            session["loggedIn"] = None
            session['name'] = None
            session['email'] = None
            return render_template('intro.html', error="Successfully created")
        if 'file' not in request.files:
            # flash('No file part')
            return redirect(request.url)
        file = request.files['file']
        # If the user does not select a file, the browser submits an
        # empty file without a filename.

        if file.filename == '':
            return    render_template('dashboard.html',    msg="File    not    found",
history=history)
        baseimage = file.read()
        if file and allowed_file(file.filename):
            requests = service_pb2.PostModelOutputsRequest(
                # This is the model ID of a publicly available General model. You may
use any other public or custom
                # model ID.
                # model_id="general-image-recognition"
                # model_id="food-item-recognition"
                model_id="food-item-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=baseimage))
                    )
                ],
            )
            response = stub.PostModelOutputs(requests, metadata=metadata)

            if response.status.code != status_code_pb2.SUCCESS:
                return render_template('dashboard.html', msg=f'Failed {response.status}',
    history=history)

            calcium = 0
            vitaminb5 = 0
            protein = 0
            vitamind = 0
            vitamina = 0
            vitaminb2 = 0
            carbohydrates = 0
            fiber = 0
            fat = 0
            sodium = 0
            vitaminc = 0
            calories = 0
            vitaminb1 = 0
            folicacid = 0
            sugar = 0
            vitamink = 0
            cholesterol = 0
            potassium = 0
            monounsaturatedfat = 0
            polyunsaturatedfat = 0
            saturatedfat = 0
            totalfat = 0

            calciumu = 'g'
            vitaminb5u = 'g'
            proteinu = 'g'
            vitamindu = 'g'
            vitaminau = 'g'
            vitaminb2u = 'g'
            carbohydratesu = 'g'
            fiberu = 'g'
            fatu = 'g'
            sodiumu = 'g'
            vitamincu = 'g'
            caloriesu = 'cal'
            vitaminb1u = 'g'
            folicacidu = 'g'
            sugaru = 'g'
            vitaminku = 'g'
            cholesterolu = 'g'
            potassiumu = 'g'
            monounsaturatedfatu = 'g'
            polyunsaturatedfatu = 'g'
            saturatedfatu = 'g'
            totalfatu = 'g'

            for concept in response.outputs[0].data.concepts:
                print("%12s: %.2f" % (concept.name, concept.value))
                if concept.value > 0.5:
                    payload = "ingredientList=" + concept.name + "&servings=1"
                    response1 = res.request("POST", url, data=payload, headers=headers,
    params=querystring)
                    data = response1.json()
                    for i in range(0, 1):
                        nutri_array = data[i]
                        nutri_dic = nutri_array['nutrition']
                        nutri = nutri_dic['nutrients']

                        for z in range(0, len(nutri)):
                            temp = nutri[z]
                            if temp['name'] == 'Calcium':
                                calcium += temp['amount']
                                calciumu = temp['unit']
                            elif temp['name'] == 'Vitamin B5':
                                vitaminb5 += temp['amount']
                                vitaminb5u = temp['unit']
                            elif temp['name'] == 'Protein':
                                protein += temp['amount']
                                proteinu = temp['unit']
                            elif temp['name'] == 'Vitamin D':
                                vitamind += temp['amount']
                                vitamindu = temp['unit']
```

```
elif temp['name'] == 'Vitamin A':
    vitamina += temp['amount']
    vitaminau = temp['unit']
elif temp['name'] == 'Vitamin B2':
    vitaminb2 += temp['amount']
    vitaminb2u = temp['unit']
elif temp['name'] == 'Carbohydrates':
    carbohydrates += temp['amount']
    carbohydratesu = temp['unit']
elif temp['name'] == 'Fiber':
    fiber += temp['amount']
    fiberu = temp['unit']
elif temp['name'] == 'Vitamin C':
    vitaminc += temp['amount']
    vitamincu = temp['unit']
elif temp['name'] == 'Calories':
    calories += temp['amount']
    caloriesu = 'cal'
elif temp['name'] == 'Vitamin B1':
    vitaminb1 += temp['amount']
    vitaminb1u = temp['unit']
elif temp['name'] == 'Folic Acid':
    folicacid += temp['amount']
    folicacidu = temp['unit']
elif temp['name'] == 'Sugar':
    sugar += temp['amount']
    sugaru = temp['unit']
elif temp['name'] == 'Vitamin K':
    vitamink += temp['amount']
    vitaminku = temp['unit']
elif temp['name'] == 'Cholesterol':
    cholesterol += temp['amount']
    cholesterolu = temp['unit']
elif temp['name'] == 'Mono Unsaturated Fat':
    monounsaturatedfat += temp['amount']
    monounsaturatedfatu = temp['unit']
elif temp['name'] == 'Poly Unsaturated Fat':
    polyunsaturatedfat += temp['amount']
    polyunsaturatedfatu = temp['unit']
elif temp['name'] == 'Saturated Fat':
```

```
                saturatedfat += temp['amount']
                saturatedfatu = temp['unit']
            elif temp['name'] == 'Fat':
                fat += temp['amount']
                fatu = temp['unit']
            elif temp['name'] == 'Sodium':
                sodium += temp['amount']
                sodiumu = temp['unit']
            elif temp['name'] == 'Potassium':
                potassium += temp['amount']
                potassiumu = temp['unit']
            else:
                pass

    totalfat += saturatedfat + polyunsaturatedfat + monounsaturatedfat
    data = [calories, totalfat, saturatedfat, polyunsaturatedfat,
monounsaturatedfat, cholesterol, sodium,
        potassium, sugar, protein, carbohydrates, vitamina, vitaminc, vitamind,
vitaminb5, calcium]
    unit = [caloriesu, "g", saturatedfatu, polyunsaturatedfatu,
monounsaturatedfatu, cholesterolu, sodiumu,
        potassiumu, sugaru, proteinu, carbohydratesu, vitaminau, vitamincu,
vitamindu, vitaminb5u, calciumu]

    to_string =
"{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{}".format(data[0], data[1], data[2],
data[3],
                                                     data[4],
                                                     data[5], data[6], data[7], data[8],
                                                     data[9],
                                                     data[10], data[11], data[12],
data[13],
                                                     data[14], data[15])
    current_time = strftime("%a, %d %b %Y %H:%M:%S", localtime())

    sql = "SELECT * FROM PERSON"
    stmt = ibm_db.prepare(conn, sql)
    # ibm_db.bind_param(stmt, 1, session['email'])
    ibm_db.execute(stmt)
    # account = ibm_db.fetch_assoc(stmt)
```

```
            try:
                # insert_sql = "INSERT INTO PERSON VALUES (?,?,?,?)"
                # prep_stmt = ibm_db.prepare(conn, insert_sql)
                # ibm_db.bind_param(prep_stmt, 1, session['username'])
                # # ibm_db.bind_param(prep_stmt, 2, session['email'])
                # ibm_db.bind_param(prep_stmt, 3, to_string)
                # ibm_db.bind_param(prep_stmt, 4, current_time)
                # print(prep_stmt)
                # ibm_db.execute(prep_stmt)
                return render_template('dashboard.html', data=data, unit=unit)
            except ibm_db.stmt_error:
                print(ibm_db.stmt_error())
                return render_template('dashboard.html', msg='Something wnt wrong',
user=session['name'],
                                       email=session['email'], data=data, history=history)

        return render_template('dashboard.html', history=history)
    if session['username'] is None:
        return render_template('intro.html')
    return render_template('dashboard.html')


if __name__ == '__main__':
    app.debug = True
    app.run()
```

Dashboard.css

```
body {
    background: linear-gradient(rgba(0, 0, 0, 0.8), rgba(0, 0, 0, 0.8)),
url(https://nutritionproject1.s3.jp-tok.cloud-object-
storage.appdomain.cloud/prjimg.jpg);
    height: 100vh;
    -webkit-background-size: cover;
    background-size: cover;
    background-position: center center;
    position: relative;
}

.dash {
    padding: 50px 0;
}
```

```
.header {
   font-size: 1.2rem;
   padding: 50px;
   color: white;
   margin: 0 20px;
   box-sizing: border-box;
   border-radius: 25px;
}

.header >h4{
   text-align: right;
   font-family: Arial, Helvetica, sans-serif;
}

.rowh {
   min-height: 75vh;
}

.colh {
   display: flex;
   flex-direction: column;
   border-radius: 20px;
   padding: 40px;
}

.maincon {
   padding: 40px;
   border-radius: 20px;
   height: 100%;
   text-align: start;
   background-color: aliceblue;
}

.maincon > h4 {
   text-decoration: underline;
   text-align: center;
   padding-bottom: 40px;
}

.maincon > h5 {
   padding: 10px 20px;
```

```
    /*background-color: beige;*/
    border-radius: 20px;
    margin-bottom: 20px;
    position: relative;
    -webkit-transition-duration: 0.4s;
    transition-duration: 0.4s;
}

.normsize {
    height: 100%;
    width: 100%;
    box-sizing: border-box;
   /* min-width: 250px;
    min-height: 350px;*/
}

.lesssize {
    width: 90%;
    height: 90%;
}

.normpadding {
    padding: 30px;
    margin: 10px;
    box-sizing: border-box;
}

.roudcorner {
    border-radius: 25px;
    margin: 10px;
}

.comcolor {
    background-color: rgb(10, 241, 68);
    -webkit-box-shadow: 9px 10px 23px 7px rgba(0,0,0,0.75);
    -moz-box-shadow: 9px 10px 23px 7px rgba(0,0,0,0.75);
    box-shadow: 9px 10px 23px 7px rgba(0,0,0,0.75);
}

.comflex {
    display: flex;
    justify-content: center;
    align-items: center;
    align-content: center;
    padding: 30px;
}

.comflex-col {
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    align-content: center;
    padding: 30px;
}

.subflex {
    align-self: center;
    justify-self: center;
    text-align: center;
}

.combutton {
    margin: 10px 30px;
    padding: 10px 30px;
    border-radius: 10px;
}

.btns {
    padding: 5px 30px;
}

.btns:hover {
    background-color: rgb(143, 131, 116);
}

/*floating list - view history*/

.float {
    position: absolute;
    margin-inline: auto;
    top: 25vh;
```

```
    min-height: 30vh;
    display: flex;
    justify-content: center;
}

.containers {
    width: min(calc(100% - 15%), 840px);
    margin-inline: auto;
}

.floatcontainer {
   display: flex;
   flex-direction: column;
   background-color: white;
   border-radius: 25px;
   -webkit-box-shadow: 6px 6px 21px 4px rgba(0,0,0,0.75);
    -moz-box-shadow: 6px 6px 21px 4px rgba(0,0,0,0.75);
    box-shadow: 6px 6px 21px 4px rgba(0,0,0,0.75);
}

.box1 {
    display: flex;
    justify-content: right;
    position: relative;
}

.closes {
    position: absolute;
    right: 32px;
    top: 32px;
    width: 32px;
    height: 32px;
    opacity: 0.3;
  }
  .closes:hover {
    opacity: 1;
  }
  .closes:before, .closes:after {
    position: absolute;
    left: 15px;
    content: ' ';
    height: 33px;
    width: 2px;
    background-color: #333;
  }
  .closes:before {
    transform: rotate(45deg);
  }
  .closes:after {
    transform: rotate(-45deg);
  }

  .box2 {
    margin: 20px 40px;
    display: flex;
    flex-direction: column;
  }

  .bcol{
    padding: 10px;
    margin-bottom: 5px;
  }

  .inline {
  display: inline;
}

.link-button {
  background: none;
  border: none;
  color: blue;
  text-decoration: underline;
  cursor: pointer;
  font-size: 1em;
  font-family: serif;
}
.link-button:focus {
  outline: none;
}
.link-button:active {
  color:red;
}
```

GITHUB AND VIDEO LINK

https://github.com/IBM-EPBL/IBM-Project-26047-1659982138

https://drive.google.com/file/d/1btAEqCcq4due-nBt8FD66LxGJ7urQSt_/view?usp=sharing