

Project Report

NUTRITION ASSISTANT APPLICATION

TEAM ID - PNT2022TMID24575

AJAY SHARMA G - 210419104004

ARJUN RAMPAL M - 210419104017

BIJJAM VENKATA MOHAN REDDY - 210419104031

CHANDRAMOULI V - 210419104038

ELA TEJESHWAR - 210419104050

1. **INTRODUCTION**
 - 1.1 Project Overview
 - 1.2 Purpose
2. **LITERATURE SURVEY**
 - 2.1 Existing problem
 - 2.2 References
 - 2.3 Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
 - 3.1 Empathy Map Canvas
 - 3.2 Ideation & Brainstorming
 - 3.3 Proposed Solution
 - 3.4 Problem Solution fit
4. **REQUIREMENT ANALYSIS**
 - 4.1 Functional requirement
 - 4.2 Non-Functional requirements
5. **PROJECT DESIGN**
 - 5.1 Data Flow Diagrams
 - 5.2 Solution & Technical Architecture
 - 5.3 User Stories
6. **PROJECT PLANNING & SCHEDULING**
 - 6.1 Sprint Planning & Estimation
 - 6.2 Sprint Delivery Schedule
 - 6.3 Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
 - 7.1 Feature 1
 - 7.2 Feature 2
 - 7.3 Database Schema (if Applicable)
8. **TESTING**
 - 8.1 Test Cases
 - 8.2 User Acceptance Testing
9. **RESULTS**
 - 9.1 Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**

Source Code

GitHub & Project Demo Link

CHAPTER 1 - INTRODUCTION

1.1 Project 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.

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.

1.2 Purpose

- To sustain proper diet and to preserve good physique.
- To maintain and lead a healthy lifestyle.
- To get into good physique.
- To get more awareness about the nutrition contents of the feed.
- To get motivated in the journey of fitness.
- To keep count of calories.
- To control cravings, mood swings and stress.
- To improve metabolism.

CHAPTER 2 - LITERATURE SURVEY

2.1 Existing Problem

Some existing platforms are not open source and require subscriptions to avail services. These platforms also do not happen to have the facility of analysing food from a image. Although some applications use certain ML based models they often do not produce accurate results.

These issues led to people restricting their usage of these type of nutrition applications and hance the lifesyle of these people were not enhanced. Our application thrives to fix these problems and improve the overall health of people in general.

2.2 References

S.NO .	Journal Paper Title	Author's Name & Year	Source	Finding
1.	Measuring the calories and nutrition from food images using machine learning techniques	Dr.M.Kiran	Research Gate	The images obtained from the mobile device are pre-processed followed by the segmentation step to extract the color and texture features through K Means clustering. The extracted options are used for food classification using Support Vector Machine (SVM). The food portion volume measurement is done by superimposing a grid of squares onto the image segment which matches the irregular shape of the food images easily. The calorie measurement is done based on the food mass and nutritional tables. The system has limited cuisine varieties and mixed food images have not been considered.
2.	Calorific value prediction	R.Kohila	Research Gate	The image of the food is transmitted through a

	mechanism using image processing and machine learning.			mobile device and it initially undergoes segmentation with Fuzzy C-means Clustering Segmentation which fixes the cluster center based on the group data unlike the K-means Clustering which can be erroneous if the cluster center is not defined properly by the user. The mathematical morphology is utilized as a tool for extracting the image components and the region shape description such as erosion, dilation, opening and closing. Feature extraction is performed to retrieve interesting parts of the image and then calorie measurement is done. It has limited scalability and diversely mixed food images have not been considered.
3.	A survey on nutrition monitoring and dietary management system	Kamaks9hi Priyaa Prakash Dr L Arockiam	Research Gate	A well balanced diet with an estimated nutrient intake is vital for infants and children which reduces the risks of deadly diseases namely cancer, diabetes, obesity and cardiovascular diseases. Unlike adults, infants require some assistance in their food intake. The survey provides valuable insights about the various advancements of IoT in the healthcare industry and the need

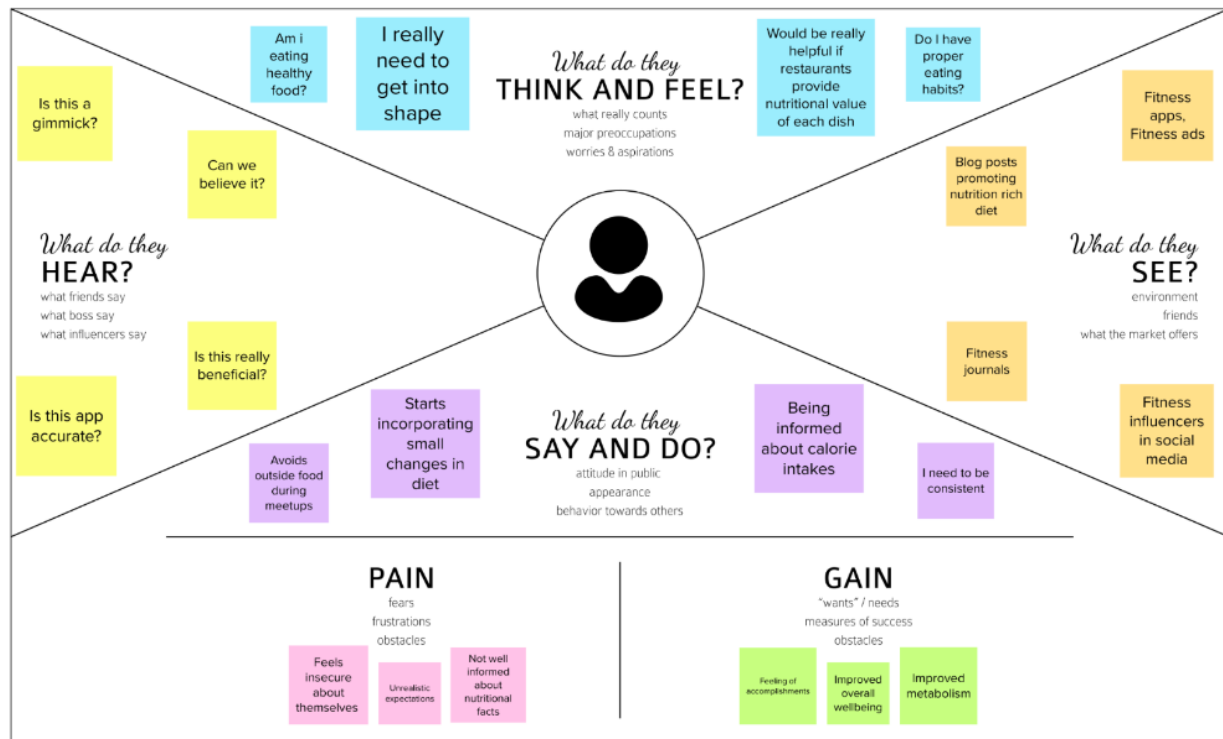
2.3 Problem Statement Definition



Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A health freak and looking for the application to be accurate in analysing the micronutrient and macronutrient.	Sustain proper diet to preserve good physique.	At times it is not possible to monitor my diet.	It is hard to stick to a diet over a long period.	The suggestion given by the application might be difficult to cope up with in the daily routine.
PS-2	A health enthusiast who needs to track their food intake in order to maintain their calorie count.	Maintain and lead a healthy lifestyle.	Sometimes it is not easy to follow the prescriptions of the dietician.	Some meals may induce them while seeing them.	Carvings, mood swings and stress will make it difficult to follow the diet plan.


CHAPTER 3 - IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas






3.2 Ideation & Brainstorming

Step-1: Team Gathering, Collaboration and Select the Problem Statement




Brainstorm & idea prioritization


Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

 10 minutes to prepare
 1 hour to collaborate
 2-8 people recommended

[Share template feedback](#)

 **Before you collaborate**

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.


 10 minutes

A Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.


B Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

C Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) →


 **Define your problem statement**

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.







 5 minutes


PROBLEM

How might we make an obese person healthier?

**Key rules of brainstorming**

To run a smooth and productive session

 Stay in topic.	 Encourage wild ideas.
 Defer judgment.	 Listen to others.
 Go for volume.	 If possible, be visual.



Need some inspiration?

Get a limited version of this template to inspire your work.

[Open example](#) →

Step-2: Brainstorm, Idea Listing and Grouping

2 Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

John Bennett

Get back to physical activity	Engage in physical activity
Get back to physical activity	Engage in physical activity

Chanderanull

Engage in physical activity	Engage in physical activity
Engage in physical activity	Engage in physical activity

Apur Khanna

Engage in physical activity	Engage in physical activity
Engage in physical activity	Engage in physical activity

Nguyen Thanh An Nguyen Bich

Engage in physical activity	Engage in physical activity
Engage in physical activity	Engage in physical activity

Elia Tejedor

Engage in physical activity	Engage in physical activity
Engage in physical activity	Engage in physical activity

3 Group Ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a header card like below. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes

Physical activity

- Exercise regularly
- Stay consistent
- Indulge in sports activity
- Join fitness groups

Tracker

- Use a tracker to monitor your progress
- Use a tracker to monitor your progress

Planning

- Create simple diet plan
- Create diet plan
- Create exercise routine

Dietitian

- Consult a dietitian
- Track your progress
- Stay informed
- Ask for a diet plan

Step-3: Idea Prioritization

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

⌚ 20 minutes



3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Health enthusiasts and people in general need to control their daily calorie intake by eating healthier foods. Even Though, some food packaging has an added nutrition and calorie value,it's not very comfortable to refer to.
2.	Idea / Solution description	People can easily track the Nutrition content and calories of food by scanning real time images of the food which will improve the dietary habits.
3.	Novelty / Uniqueness	This solution has the uniqueness that we can realize real time images of the meal and can easily analyze its nutritional content.
4.	Social Impact / Customer Satisfaction	Can boost the user's lifestyle in a positive way and assist them to follow their dietary recommendation.
5.	Business Model (Revenue Model)	The application will increase the confidence among the people. With the help of social media we can spread word about our application and with the influencers we can attract the normal people.
6.	Scalability of the Solution	Since this is a cloud based application , scalability is much easier. People can access from anywhere at any time to track the calories and nutrition value that will improve a healthy eating pattern.

3.4 Problem Solution Fit

Project Title : Nutrition Assistant Application

Project Design Phase - I - Solution Fit

Team ID: PNT2022TMID24575

Define CS, fit into CL	1. CUSTOMER SEGMENT(S) CS All age group people who are careless about their health due to their busy schedule and intake of high-calorie diet.	6. CUSTOMER LIMITATIONS <small>eg. budget, devices</small> CL The customer should provide a clear image for knowing the nutrition content of the food. The app can't provide accurate result if the image is not clear. In some cases, the recipes maybe allergic to their health.	5. AVAILABLE SOLUTIONS <small>PROS & CONS</small> AS The premium features like tracking the calorie intake and creating diet charts is not available to all the users, which is the important feature of an nutrition assistant application.	Explore AS, differentiate
	2. PROBLEMS / PAINS <small>+ ITS FREQUENCY</small> PR The problem and pains of the user are obesity, fear of getting health related issues. They will get frustrated of not getting immediate result and difficult to do tedious work. Lack of confidence due to appearance.	9. PROBLEM ROOT / CAUSE RC It is easy to fall into a trap of eating unhealthy foods which is heavy in calories. Once the nutritional value is replaced by foods high in sugar, bad fats and salt it leads to various health issues so users need to control their daily calorie intake to lead a healthy lifestyle.	7. BEHAVIOR <small>+ ITS INTENSITY</small> BE The behavioral changes in users reflect in their day- to-day life such as they will maintain a proper diet and follow the daily routine in eating and intake of healthy food. So, that it helps them to improve their health.	Focus on PR, tap into BE, understand RC
Focus on PR, tap into BE, understand RC	3. TRIGGERS TO ACT 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.	10. YOUR SOLUTION SL The solution is user can know the nutritional content of the food they are in taking, by taking picture of the food and uploading it in the app. Clarifai's AI-Driven Food Detection Model is used for getting accurate food identification and APIs to give the nutritional value of the identified food.	8. CHANNELS of BEHAVIOR CH ONLINE The application provides a user-friendly environment that enables users to interact to clarify their queries	Extract online & offline CH of BE
	4. EMOTIONS <small>BEFORE / AFTER</small> EM They are scared of declining health, so they get motivated towards eating. Healthy foods and move to healthy lifestyle.		OFFLINE Connecting all the users through offline meeting and giving some complimentary gifts. Conducting offline session by nutrition expert.	
Identify strong TR & EM				

CHAPTER 4 - REQUIREMENT ANALYSIS

4.1 Functional Requirements

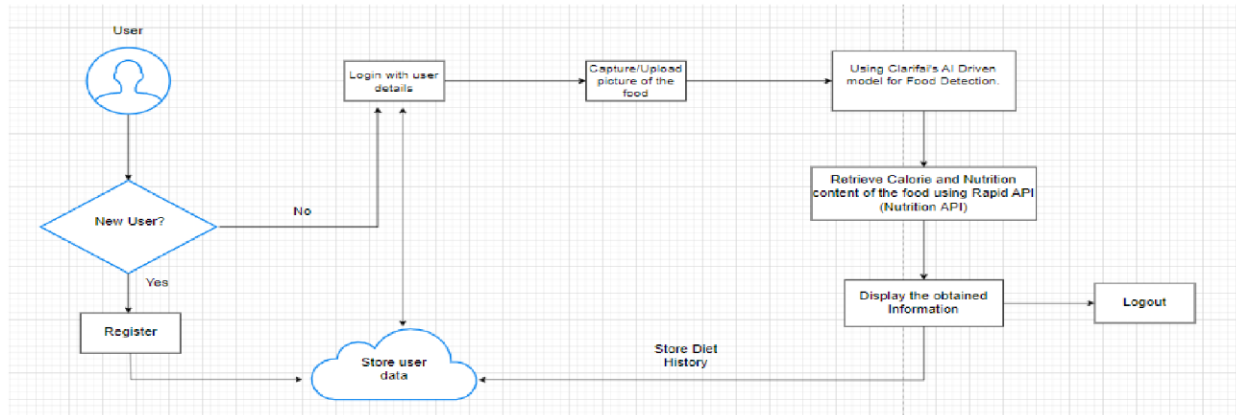
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through E-mail ID or Phone number.
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Profile Completion	Obtain personal details like Height, Weight, Age, Gender, Health issues and calculate BMI.
FR-4	Obtain image of the meal	Upload an image of the meal.
FR-5	Classify the meal	Use the Clarifai API to obtain the name of the food from the uploaded image.
FR-6	Display calorie information	Using the food name obtained from Clarifi API use the Nutrition API (rapid API) to collect the calorie information of the meal and display it to the user.

4.2 Non-Functional Requirements

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Provide user friendly User Interface. Provide an intuitive and simple design.
NFR-2	Security	Provide a comprehensive authorization and authentication scheme for every user.
NFR-3	Reliability	The application must be able to perform without failure or very minimal negligible failure at around 95% of the use cases.
NFR-4	Performance	The response time of every landing page must be as fast as possible and in no case must exceed 5 seconds.
NFR-5	Availability	The services must be available to the users at anyplace and at any time.
NFR-6	Scalability	Provide horizontal or vertical scaling for higher workloads.

CHAPTER 5 - PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture

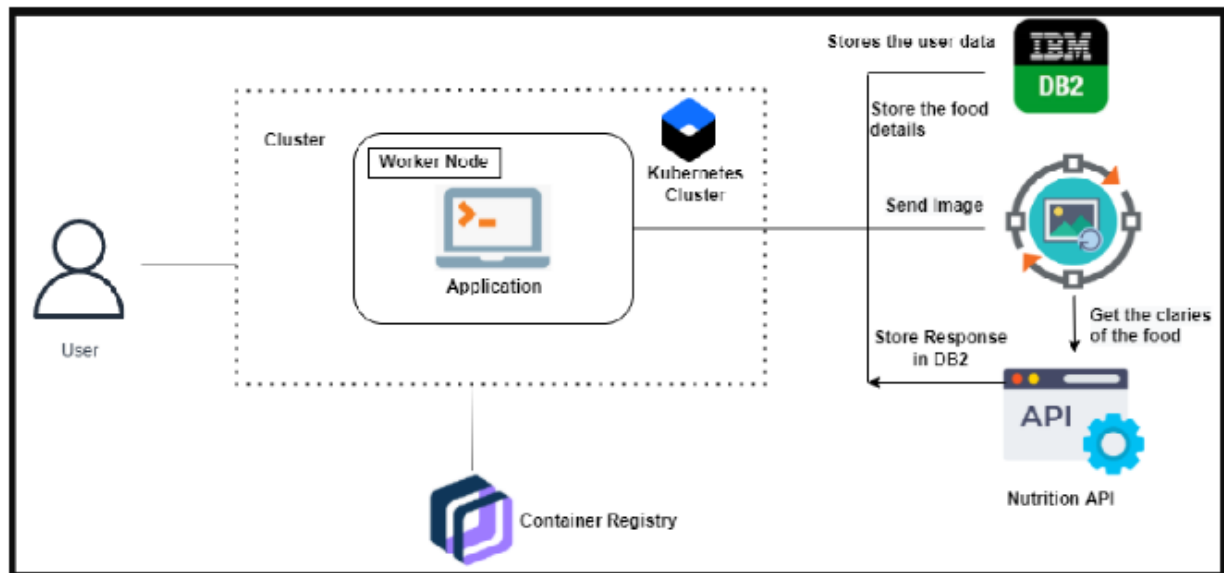


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How the user interacts with the application Eg. Web UI.	HTML, CSS, JavaScript
2.	To get the food nutrition and calorie value	The user will upload the image that contains the food image. Then, the user will then see the food nutrition value that was calculated by the process.	Python, Flask (web Framework), HTML, CSS, JavaScript.
3.	Cloud Database	Database Service Cloud	IBM DB2
4.	Database	Get the user's name, and mail, and store the food calorie value. Data types, Configurations, etc.,	MySQL
5.	External API-1	To predict the image that the user will upload to the upload image page.	Clarifai's AI-driven Food detection Model API
6.	External API-2	Food APIs for the nutritional value of the identified food	Food API
7.	File Storage	File Storage Requirements	IBM Block Storage or Other Storage Services.
8.	Infrastructure	Application Deployment on Local System / Cloud: Local Server Configuration: Cloud Server Configuration:	Local, Cloud Foundry, Kubernetes and Docker.

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	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
	Profile updation	USN-2	As a user, I have to enter my height, weight and daily activity details.	I can update this information on Dashboard.	High	Sprint-1
	Login	USN-3	As a user, I can login to the application by entering an Email or Phone number and password.	I can access my account/ dashboard.	High	Sprint-2
	Dashboard	USN-4	As a user, I can upload or capture live image of the meal	I can get the nutritional value of that particular meal.	High	Sprint-2
		USN-5	As a user, I can track my daily calorie intake.	I can access my account/ Dashboard.	High	Sprint-3
Administrator	Maintain the Application	USN-6	Maintaining details for users.	I can access the database.	High	Sprint-4

CHAPTER 6 - PROJECT PLANNING & SCHEDULING

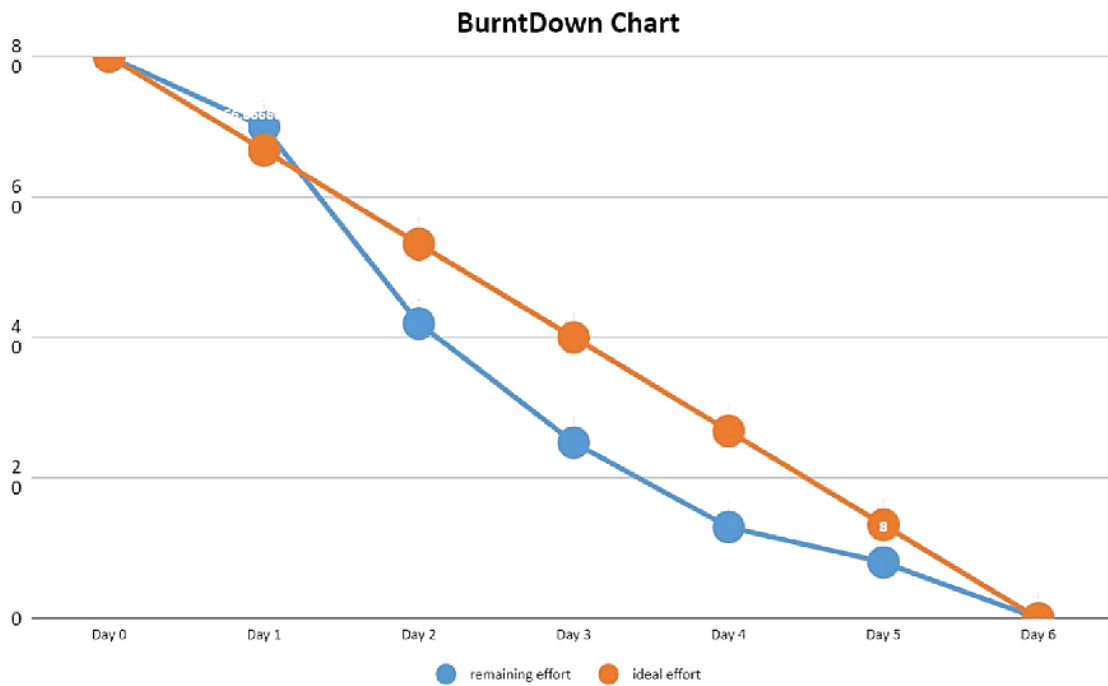
6.1 Sprint Planning & Estimation

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	
Sprint-1	Profile updation	USN-2	As a user, I have to enter my height, weight and daily activity details.	1	High	
Sprint-2	Login	USN-3	As a user, I can login to the application by entering an Email or Phone number and password.	2	High	
Sprint-2	Dashboard	USN-4	As a user, I can upload or capture live image of the meal.	2	High	
Sprint-3		USN-5	As a user, I can track my daily calorie intake.	1	High	
Sprint-4	Maintain the Application	USN-6	Maintaining details for users.	2	High	

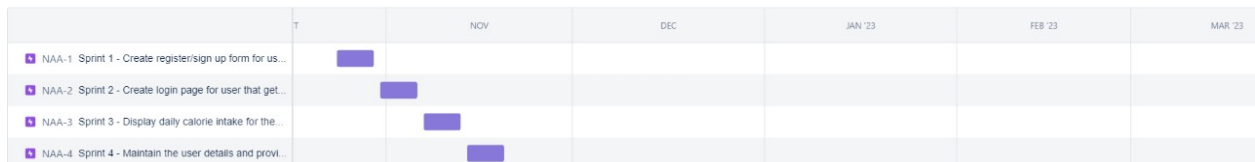
6.2 Sprint Delivery Schedule

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	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	06 Nov 2022	20	06 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

		Initial Estimate	24-Oct	25-Oct	26-Oct	27-Oct	28-Oct	29-Oct
	Sprint 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-3	20	5	5	5	5	0	0
	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

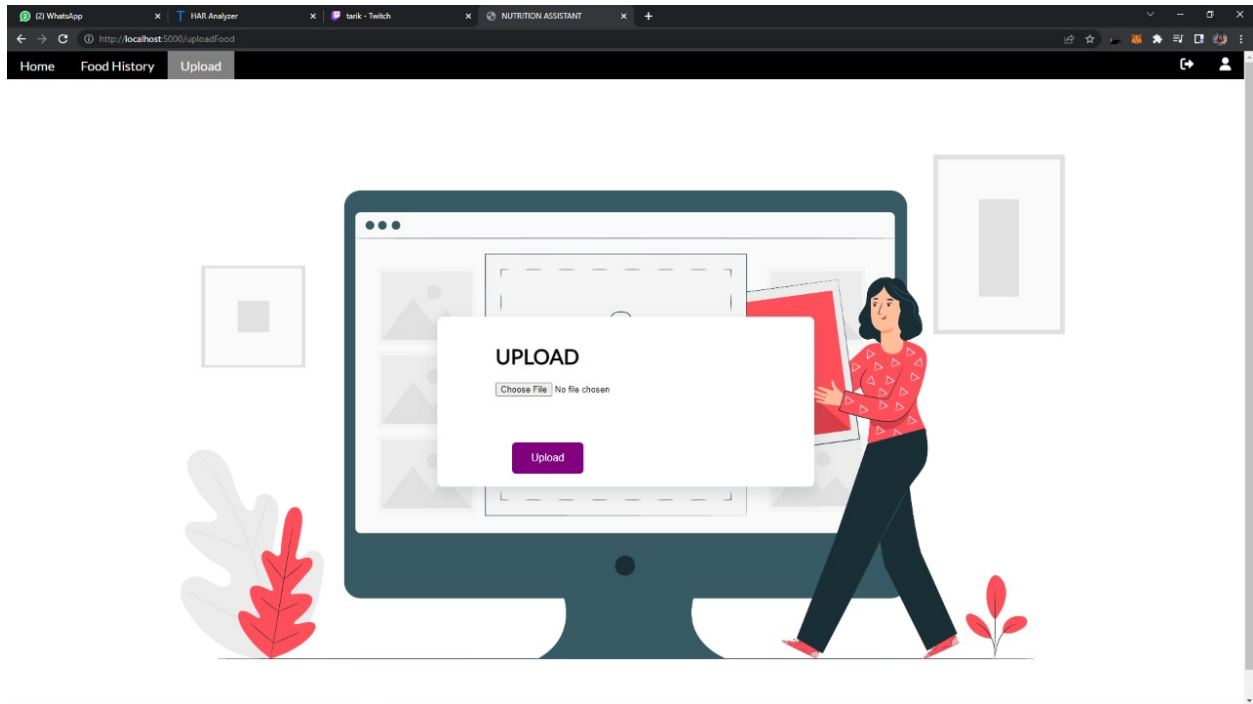


6.3 Reports from JIRA



CHAPTER 7 - CODING & SOLUTION

7.1 Feature 1



Upload.html

```
<html>
  <head>
    <link rel="stylesheet" href="/styles/style.css">
    <script src="https://kit.fontawesome.com/15b6f152f1.js"
crossorigin="anonymous"></script>
    <title>NUTRITION ASSISTANT</title>
  </head>
  <body>
    <ul class="nav-list">
      <li class="nav-item"><a href="/home">Home</a></li>
      <li class="nav-item"><a href="/foodHistory">Food History</a></li>
      <li class="nav-item"><a href="/uploadFood" class="active">Upload</a></li>
```

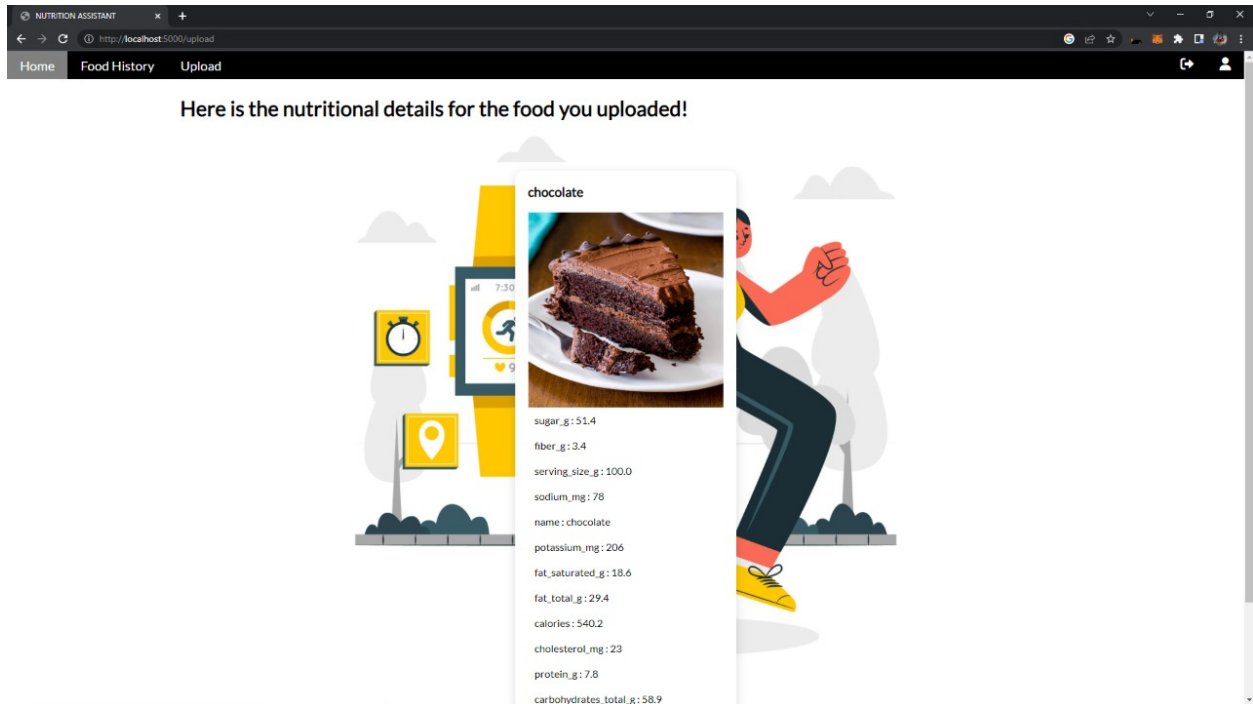
```

        <li class="nav-item-right"><a href="/profile"><i class="fa-solid fa-
user"></i></a></li>
        <li class="nav-item-right" onclick="javascript:return confirm('Are you sure you
want to log out?');"><a href="/logout"><i class="fa-solid fa-right-from-
bracket"></i></a></li>
    </ul>
    <div class="form-wrapper upload-form">
        <form action="/upload" method="POST" class="form" enctype = "multipart/form-
data">
            <h1>UPLOAD</h1>
            <div class="inputContainer">
                <input type="file" name="file" id="imgInp" onchange="previewImage()"
placeholder="Upload image" required>
            </div>
            
            <input type="submit" class="submitBtn" value="Upload">
        </form>
    </div>
</form>
<script>
function previewImage(){
    imgInp = document.getElementById('imgInp')
    img = document.getElementById('preview')
    const [file] = imgInp.files
    if (file) {
        img.src = URL.createObjectURL(file)
        img.style.display = "block"
    }
}
</script>
</body>
</html>

```

7.2 Feature 2

7.2.1 Feature 2.1



Home.html

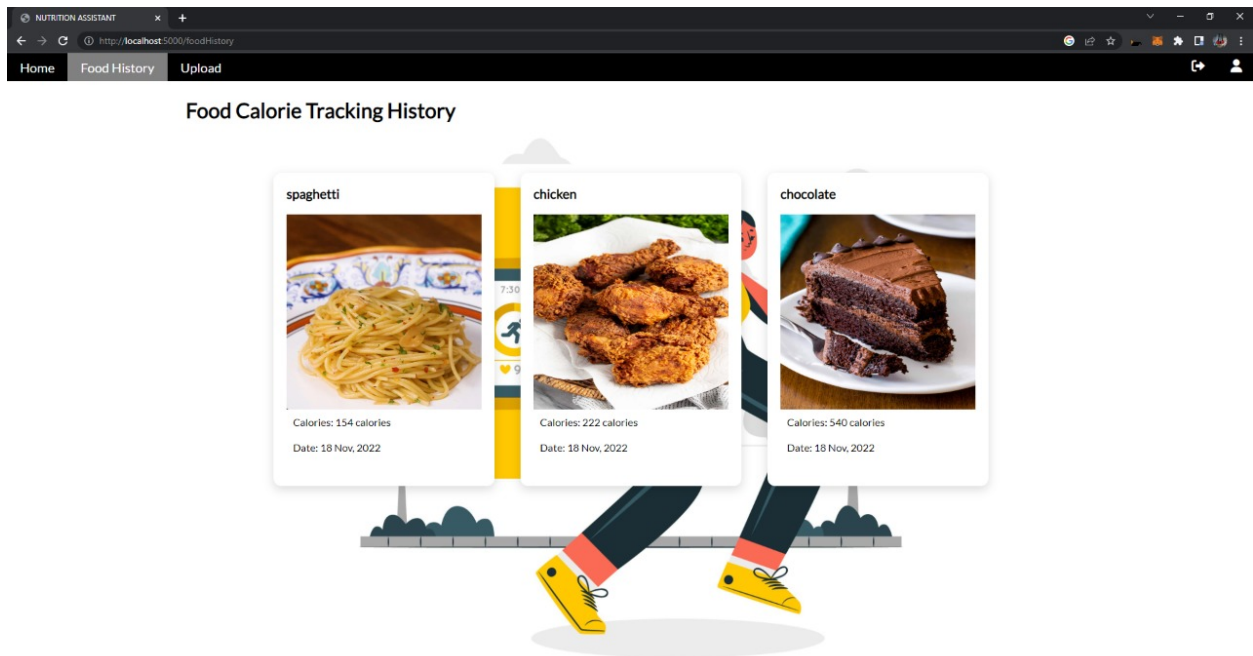
```
<html>
  <head>
    <link rel="stylesheet" href="/styles/style.css">
    <script src="https://kit.fontawesome.com/15b6f152f1.js"
crossorigin="anonymous"></script>
    <title>NUTRITION ASSISTANT</title>
  </head>
  <body>
    <ul class="nav-list">
      <li class="nav-item"><a class="active" href="/home">Home</a></li>
      <li class="nav-item"><a href="/foodHistory">Food History</a></li>
```

```

    <li class="nav-item"><a href="/uploadFood">Upload</a></li>
    <li class="nav-item-right"><a href="/profile"><i class="fa-solid fa-
user"></i></a></li>
    <li class="nav-item-right" onclick="javascript:return confirm('Are you sure you
want to log out?');"><a href="/logout"><i class="fa-solid fa-right-from-
bracket"></i></a></li>
  </ul>
  <h1 style="text-align: center;">Welcome,</h1>
  <h1 style="text-align: center;">Your today's calorie intake is: {{data}} calories</h1>
  
  <script>
    window.watsonAssistantChatOptions = {
      integrationID: "c7403ddb-f48a-4a58-94a2-27346df3efb3", // The ID of this
integration.
      region: "jp-tok", // The region your integration is hosted in.
      serviceInstanceID: "5ffb9075-1402-4989-9682-ed9ea33aac66", // The ID of your
service instance.
      onLoad: function(instance) { instance.render(); }
    };
    setTimeout(function(){
      const t=document.createElement('script');
      t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
      document.head.appendChild(t);
    });
  </script>
</body>
</html>

```

7.2.2 Feature 2.2



FoodHistory.html

```
<html>
  <head>
    <link rel="stylesheet" href="/styles/cardStyle.css">
    <link rel="stylesheet" href="/styles/style.css">
    <script src="https://kit.fontawesome.com/15b6f152f1.js"
crossorigin="anonymous"></script>
    <title>NUTRITION ASSISTANT</title>
  </head>
  <body>
    <ul class="nav-list">
      <li class="nav-item"><a href="/home">Home</a></li>
      <li class="nav-item"><a href="/foodHistory" class="active">Food History</a></li>
      <li class="nav-item"><a href="/uploadFood">Upload</a></li>
      <li class="nav-item-right"><a href="/profile"><i class="fa-solid fa-
user"></i></a></li>
```

```
<li class="nav-item-right" onclick="javascript:return confirm('Are you sure you  
want to log out?');"><a href="/logout"><i class="fa-solid fa-right-from-  
bracket"></i></a></li>
```

```
</ul>
```

```
<div class="container food-history-bg">
```

```
<h2>Food Calorie Tracking History</h2>
```

```
<ul class="cards">
```

```
{% for dict_item in data%}
```

```
<li class="card">
```

```
<div>
```

```
<h3 class="card-title">{{dict_item['FOOD_NAME']}}</h3>
```

```
<div class="card-content">
```

```

```

```
<p class="details">Calories: {{dict_item['CALORIES']}} calories</p>
```

```
<p class="details">Date: {{dict_item['UPLOADED_DATE_TIME']}}</p>
```

```
</div>
```

```
</div>
```

```
</li>
```

```
{%endfor%}
```

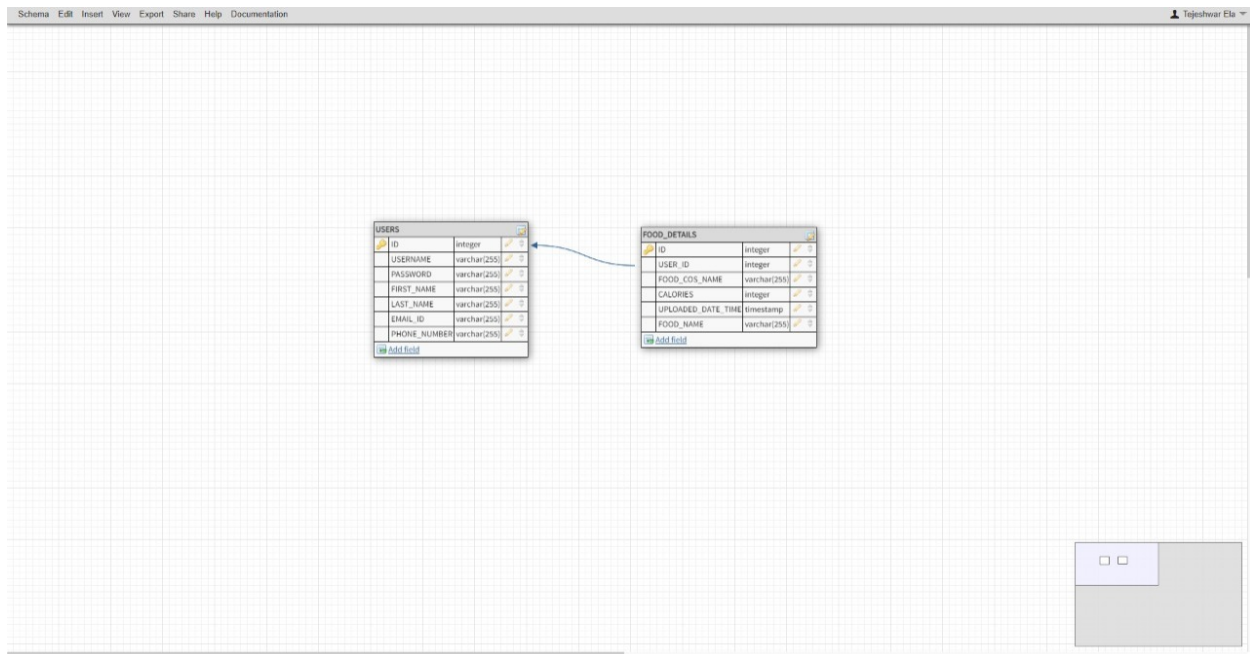
```
</ul>
```

```
</div>
```

```
</body>
```

```
</html>
```


7.3 Database Schema



CHAPTER 8 - TESTING

8.1 Test Cases

- Verify if the buttons in web page are responsive.
- Verify if the UI elements are getting displayed properly.
- Verify if the user can upload files from his system.
- Verify if the output is displayed.
- Verify if the user can login using his credentials.
- Verify if the model predicts the input accurately.
- Verify if the user is getting redirected to home page after sign in.
- Verify if the user can navigate to other pages in navigation bar.
- Verify if the user can exit the home to sign page.

8.2 User Acceptance Testing

Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Skills/Job Recommender Application project at the time of the release to User Acceptance Testing (UAT).

Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved.

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	2	0	0	0	2
Duplicate	1	0	0	0	1
External	0	0	0	0	0
Fixed	3	0	0	0	3
Not Reproduced	2	0	0	0	2
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	8	0	0	0	8

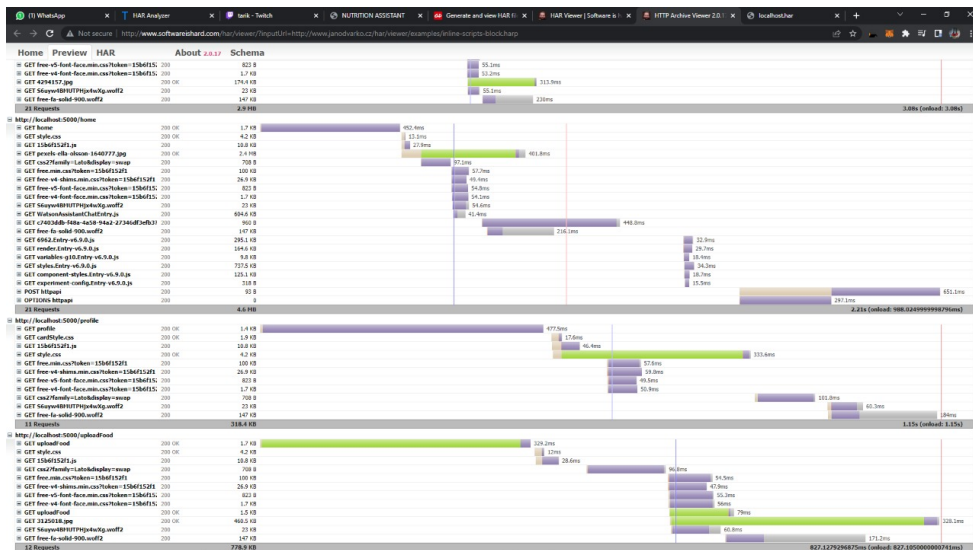
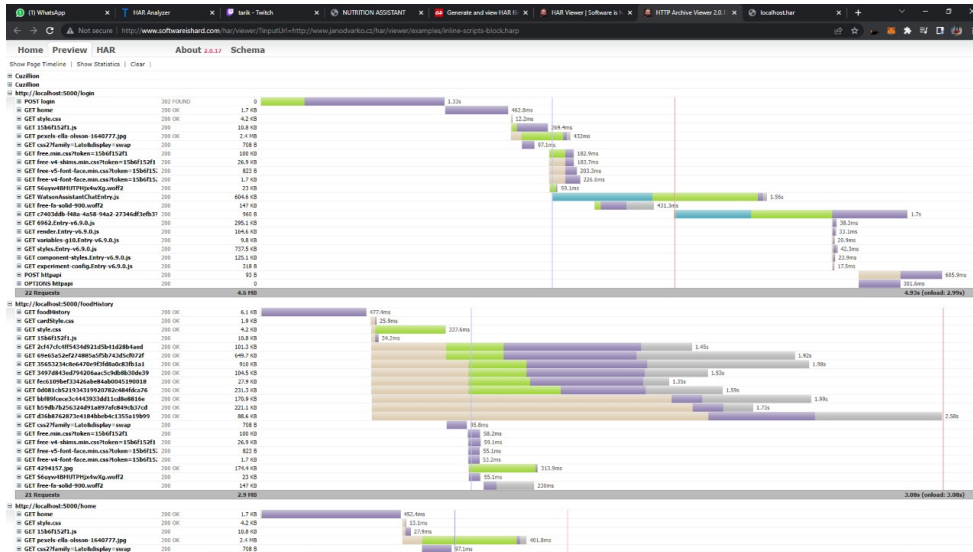
Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested.

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	0	0	0	0
Client Application	3	0	0	3
Security	0	0	0	0
Outsource Shipping	0	0	0	0
Exception Reporting	0	0	0	0
Final Report Output	0	0	0	0
Version Control	0	0	0	0

CHAPTER 9 - RESULTS

9.1 Performance Metrics



CHAPTER 10 - ADVANTAGES & DISADVANTAGES

Advantages:

- ✓ It gives user awareness about calorific value present in the food.
- ✓ User can easily keep track of their calories on a daily basis.
- ✓ Easy user interaction.
- ✓ Easy login process.
- ✓ Can accurately track the nutritional content in the food.
- ✓ User can maintain history of food items they consumed.

Disadvantages:

- ✓ The model lacks diversity in food prediction.
- ✓ Model cannot predict the correct servings of the food.

CHAPTER 11 - CONCLUSION

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 dietitians 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 person.

We developed a cloud based nutrition application which detects the nutrition in food. It clarifies the calories in the food which affects our health.

CHAPTER 12 - FUTURE SCOPE

- In the future user can calculate their maintenance calories and according to their fitness goals if they opt for weight loss the app will suggest calorie deficit foods, if they opt for bulking the app will suggest calorie surplus foods.
- The application can suggest workout routines based on the user's fitness goal.
- The application will enable the user to contact fitness experts.
- For improved accuracy in prediction of food, Deep Learning based models can be developed.

CHAPTER 13 - APPENDIX

Source Code:

Upload.html

```
<html>
  <head>
    <link rel="stylesheet" href="/styles/style.css">
    <script src="https://kit.fontawesome.com/15b6f152f1.js"
crossorigin="anonymous"></script>
    <title>NUTRITION ASSISTANT</title>
  </head>
  <body>
    <ul class="nav-list">
      <li class="nav-item"><a href="/home">Home</a></li>
      <li class="nav-item"><a href="/foodHistory">Food History</a></li>
      <li class="nav-item"><a href="/uploadFood" class="active">Upload</a></li>
      <li class="nav-item-right"><a href="/profile"><i class="fa-solid fa-
user"></i></a></li>
      <li class="nav-item-right" onclick="javascript:return confirm('Are you sure you
want to log out?');"><a href="/logout"><i class="fa-solid fa-right-from-
bracket"></i></a></li>
    </ul>
    <div class="form-wrapper upload-form">
      <form action="/upload" method="POST" class="form" enctype = "multipart/form-
data">
        <h1>UPLOAD</h1>
        <div class="inputContainer">
          <input type="file" name="file" id="imgInp" onchange="previewImage()"
placeholder="Upload image" required>
        </div>
        <img src="#" style="display: none;" alt="" width="300px" height="300px"
```



```

id="preview">
    <input type="submit" class="submitBtn" value="Upload">
  </form>
</div>
</form>
<script>
function previewImage(){
  imgInp = document.getElementById('imgInp')
  img = document.getElementById('preview')
  const [file] = imgInp.files
  if (file) {
    img.src = URL.createObjectURL(file)
    img.style.display = "block"
  }
}
</script>
</body>
</html>

```

Home.html

```

<html>
  <head>
    <link rel="stylesheet" href="/styles/style.css">
    <script src="https://kit.fontawesome.com/15b6f152f1.js"
crossorigin="anonymous"></script>
    <title>NUTRITION ASSISTANT</title>
  </head>
  <body>
    <ul class="nav-list">
      <li class="nav-item"><a class="active" href="/home">Home</a></li>
      <li class="nav-item"><a href="/foodHistory">Food History</a></li>
      <li class="nav-item"><a href="/uploadFood">Upload</a></li>
      <li class="nav-item-right"><a href="/profile"><i class="fa-solid fa-
user"></i></a></li>
      <li class="nav-item-right" onclick="javascript:return confirm('Are you sure you
want to log out?');"><a href="/logout"><i class="fa-solid fa-right-from-

```

```

    bracket"></i></a></li>
  </ul>
  <h1 style="text-align: center;">Welcome,</h1>
  <h1 style="text-align: center;">Your today's calorie intake is: {{data}} calories</h1>
  
  <script>
    window.watsonAssistantChatOptions = {
      integrationID: "c7403ddb-f48a-4a58-94a2-27346df3efb3", // The ID of this
integration.
      region: "jp-tok", // The region your integration is hosted in.
      serviceInstanceID: "5ffb9075-1402-4989-9682-ed9ea33aac66", // The ID of your
service instance.
      onLoad: function(instance) { instance.render(); }
    };
    setTimeout(function(){
      const t=document.createElement('script');
      t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
      document.head.appendChild(t);
    });
  </script>
</body>
</html>

```

FoodHistory.html

```

<html>
  <head>
    <link rel="stylesheet" href="/styles/cardStyle.css">
    <link rel="stylesheet" href="/styles/style.css">
    <script src="https://kit.fontawesome.com/15b6f152f1.js"
crossorigin="anonymous"></script>
    <title>NUTRITION ASSISTANT</title>
  </head>
  <body>
    <ul class="nav-list">

```

```

<li class="nav-item"><a href="/home">Home</a></li>
<li class="nav-item"><a href="/foodHistory" class="active">Food History</a></li>
<li class="nav-item"><a href="/uploadFood">Upload</a></li>
<li class="nav-item-right"><a href="/profile"><i class="fa-solid fa-
user"></i></a></li>
    <li class="nav-item-right" onclick="javascript:return confirm('Are you sure you
want to log out?');"><a href="/logout"><i class="fa-solid fa-right-from-
bracket"></i></a></li>
</ul>
<div class="container food-history-bg">
    <h2>Food Calorie Tracking History</h2>
    <ul class="cards">
        {% for dict_item in data%}
        <li class="card">
            <div>
                <h3 class="card-title">{{dict_item['FOOD_NAME']}}</h3>
                <div class="card-content">
                    
                    <p class="details">Calories: {{dict_item['CALORIES']}} calories</p>
                    <p class="details">Date: {{dict_item['UPLOADED_DATE_TIME']}}</p>
                </div>
            </div>
        </li>
        {%endfor%}
    </ul>
</div>
</body>
</html>

```

Signup.html

```

<html>
<head>
    <link rel="stylesheet" href="/styles/style.css">
    <title>NUTRITION ASSISTANT</title>

```

```

</head>
<body>
  <div class="form-wrapper signup-form">
    <form action="/login" method="POST" class="form">
      <h1>SIGN UP</h1>
      <div class="inputContainer">
        <input type="text" class="input" name="first_name" placeholder="First Name"
required>
        <label for="" class="label">First Name</label>
      </div>
      <div class="inputContainer">
        <input type="text" class="input" name="last_name" placeholder="Last Name"
required>
        <label for="" class="label">Last Name</label>
      </div>
      <div class="inputContainer">
        <input type="text" class="input" name="email_id" placeholder="Email ID"
required>
        <label for="" class="label">Email ID</label>
      </div>
      <div class="inputContainer">
        <input type="text" class="input" name="phone_number"
placeholder="Contact" required>
        <label for="" class="label">Contact</label>
      </div>
      <div class="inputContainer">
        <input type="text" class="input" name="username" placeholder="Username"
required>
        <label for="" class="label">Username</label>
      </div>
      <div class="inputContainer">
        <input type="password" class="input" id="password" name="password"
placeholder="Password" required>
        <label for="" class="label">Password</label>
        <span id="togglePassword" onclick="showPassword()" style="position:
absolute;right: -85px;top: 10px;"><i class="far fa-eye" style="cursor:
pointer;"></i></span>

```

```

        </div>
        <input type="submit" class="submitBtn" value="Sign Up">
        <a href="/loginForm" style="text-decoration: none;"><input type="button"
class="submitBtn" value="I'm already a member"></a>
    </form>
</div>
<script>
    function showPassword(){
        var x = document.getElementById("password");
        if(x.type == "password"){
            x.type = "text";
        }
        else{
            x.type = "password";
        }
    }
</script>
</body>
</html>

```

Login.html

```

<html>
    <head>
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/sweetalert/2.1.0/sweetalert.min.js"></scrip
t>
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.13.0/css/all.min.css">
        <link rel="stylesheet" href="/styles/style.css">
        <title>NUTRITION ASSISTANT</title>
    </head>
    <body>
        <input type="hidden" id="status" value="{{status}}">
        <div class="form-wrapper login-form">
            <form action="/login" method="POST" class="form">
                <h1>LOGIN</h1>

```

```

    <div class="inputContainer">
        <input type="text" class="input" name="username" placeholder="Username"
required>
        <label for="" class="label">Username</label>
    </div>
    <div class="inputContainer">
        <input type="password" class="input" id="password" name="password"
placeholder="Password" required>
        <label for="" class="label">Password</label>
        <span id="togglePassword" onclick="showPassword()" style="position:
absolute;right: -85px;top: 10px;"><i class="far fa-eye" style="cursor:
pointer;"></i></span>
    </div>
    <input type="submit" class="submitBtn" value="Login">
</form>
</div>
<script>
    var status = document.getElementById("status").value;
    console.log(status)
    if(status == "incorrectPassword"){
        swal("Sorry!", "Wrong username or password!", "error");
    }
    else if(status == "unknownFailure"){
        swal("Sorry!", "Incorrect credentials, please try again!", "error");
    }
</script>
<script>
    function showPassword(){
        var x = document.getElementById("password");
        if(x.type == "password"){
            x.type = "text";
        }
        else{
            x.type = "password";
        }
    }
</script>

```

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

Profile.html

```
<html>
  <head>
    <link rel="stylesheet" href="/styles/cardStyle.css">
    <script src="https://kit.fontawesome.com/15b6f152f1.js"
crossorigin="anonymous"></script>
    <link rel="stylesheet" href="/styles/style.css">
    <title>NUTRITION ASSISTANT</title>
  </head>
  <body>
    <ul class="nav-list">
      <li class="nav-item"><a href="/home">Home</a></li>
      <li class="nav-item"><a href="/foodHistory">Food History</a></li>
      <li class="nav-item"><a href="/uploadFood">Upload</a></li>
      <li class="nav-item-right"><a href="/profile" class="active"><i class="fa-solid fa-
user"></i></a></li>
      <li class="nav-item-right" onclick="javascript:return confirm('Are you sure you
want to log out?');"><a href="/logout"><i class="fa-solid fa-right-from-
bracket"></i></a></li>
    </ul>
    <div class="container">
      <h2>User Profile</h2>
      <div class="card">
        <h3 class="card-title">WELCOME {{data['FIRST_NAME']}}! <br>Here is your
profile details!</h3>
        <div class="card-content">
          <p>First name: {{data['FIRST_NAME']}}</p>
          <p>Last name: {{data['LAST_NAME']}}</p>
          <p>Username: {{data['USERNAME']}}</p>
          <p>Contact: {{data['PHONE_NUMBER']}}</p>
          <p>Email ID: {{data['EMAIL_ID']}}</p>
        </div>
      </div>
    </div>
```

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

FoodDetail.html

```
<html>
  <head>
    <link rel="stylesheet" href="/styles/cardStyle.css">
    <link rel="stylesheet" href="/styles/style.css">
    <script src="https://kit.fontawesome.com/15b6f152f1.js"
crossorigin="anonymous"></script>
    <title>NUTRITION ASSISTANT</title>
  </head>
  <body>
    <ul class="nav-list">
      <li class="nav-item"><a class="active" href="/home">Home</a></li>
      <li class="nav-item"><a href="/foodHistory">Food History</a></li>
      <li class="nav-item"><a href="/uploadFood">Upload</a></li>
      <li class="nav-item-right"><a href="/profile"><i class="fa-solid fa-
user"></i></a></li>
      <li class="nav-item-right" onclick="javascript:return confirm('Are you sure you
want to log out?');"><a href="/logout"><i class="fa-solid fa-right-from-
bracket"></i></a></li>
    </ul>
    <div class="container food-history-bg">
      <h2>Here is the nutritional details for the food you uploaded!</h2>
      <ul class="cards">
        <li class="card">
          <div>
            <h3 class="card-title">{{targetFoodItem}}</h3>
            <div class="card-content">
              
              {%for key, value in data['items'][0].items()%}
                <p class="details">{{key}} : {{value}}</p>
              {%endfor%}
              <a href="/foodHistory" style="text-decoration: none;"><input type="button"
```



```
value="Go to Food History" class="submitBtn"></a>
    </div>
</div>
</li>
</ul>
</div>
</body>
</html>
```

CardStyle.css

```
:root {
  --red: #ef233c;
  --darkred: #c00424;
  --platinum: #e5e5e5;
  --black: #2b2d42;
  --white: #fff;
  --thumb: #edf2f4;
}

* {
  box-sizing: border-box;
  padding: 0;
  margin: 0;
}

.container {
  max-width: 1400px;
  padding: 25px 15px 25px 15px;
  margin: 0 auto;
  height: calc(100% - 43px);
  width: 100%;
}

.food-history-bg{
  background-image: url('4294157.jpg');
  background-size: contain;
```

```
background-repeat: no-repeat;
background-position: center;
}
```

```
h2 {
  font-size: 32px;
  margin-bottom: 1em;
}
```

```
.cards {
  display: flex;
  padding: 25px 0px;
  list-style: none;
  scroll-snap-type: x mandatory;
  flex-wrap: wrap;
  justify-content: center;
}
```

```
.details{
  padding: 10px;
}
```

```
.card {
  display: flex;
  flex-direction: column;
  padding: 20px;
  max-width: 450px;
  background: var(--white);
  border-radius: 12px;
  box-shadow: 0 5px 15px rgb(0 0 0 / 15%);
  margin: 20px;
}
```

```
.card:hover {
  color: var(--white);
  background: #ffc801;
}
```

```
.card .card-title {  
  font-size: 20px;  
}
```

```
.card .card-content {  
  margin: 20px 0;  
  max-width: 85%;  
}
```

```
.card .card-link-wrapper {  
  margin-top: auto;  
}
```

```
.card .card-link {  
  display: inline-block;  
  text-decoration: none;  
  color: white;  
  background: var(--red);  
  padding: 6px 12px;  
  border-radius: 8px;  
  transition: background 0.2s;  
}
```

```
.card:hover .card-link {  
  background: #4acc00;  
}
```

```
.cards::-webkit-scrollbar {  
  height: 12px;  
}
```

```
.cards::-webkit-scrollbar-thumb,  
.cards::-webkit-scrollbar-track {  
  border-radius: 92px;  
}
```

```
.cards::-webkit-scrollbar-thumb {  
  background: var(--darkred);  
}
```

```
.cards::-webkit-scrollbar-track {  
  background: var(--thumb);  
}
```

Style.css

```
@charset "UTF-8";
```

```
@import url('https://fonts.googleapis.com/css2?family=Lato&display=swap');
```

```
/* Get rid of all default margins/paddings. Set typeface */
```

```
body {  
  box-sizing: border-box;  
  margin: 0;  
  padding: 0;  
  background-color: white;  
  font-family: "lato", sans-serif;  
}
```

```
.form-wrapper {  
  display: flex;  
  margin: 0;  
  padding: 0;  
  height: 100%;  
  align-items: center;  
  justify-content: center;  
}
```

```
.upload-form{  
  background-image: url('3125018.jpg');  
  background-size: contain;  
  background-repeat: no-repeat;  
  background-position: center;  
}
```

```
.signup-form{
  background-image: url('5912.jpg');
  background-size: contain;
  background-repeat: no-repeat;
  background-position: center;
}
```

```
.login-form{
  background-image: url('nutritionist_4.jpg');
  background-size: contain;
  background-repeat: no-repeat;
  background-position: center;
}
```

```
/* Blinking style*/
.blink {
  color: red;
  text-align: center;
  animation: blink 1s steps(1, end) infinite;
}
```

```
@keyframes blink {
  0% {
    opacity: 1;
  }
  50% {
    opacity: 0;
  }
  100% {
    opacity: 1;
  }
}
```

```
/* Puts the form in the center both horizontally and vertically. Sets its height to 100% of
the viewport's height */
```

```
.signupFrm {  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  height: 100vh;  
}
```

```
.form {  
  background-color: white;  
  width: 400px;  
  border-radius: 8px;  
  padding: 20px 90px;  
  box-shadow: 0 10px 25px rgba(92, 99, 105, .2);  
}
```

```
.searchForm {  
  background-color: white;  
  width: inherit;  
  border-radius: 8px;  
  padding: 20px 90px;  
}
```

```
.title {  
  font-size: 50px;  
  margin-bottom: 50px;  
}
```

```
.userProfile{  
  width: 400px;  
  display: block;  
  box-shadow: 0 10px 25px rgba(92, 99, 105, .2);  
  margin: 0 auto;  
  padding: 10px 20px 10px 20px;  
}
```

```
.inputContainer {
```

```
position: relative;
height: 45px;
width: 90%;
margin-bottom: 17px;
}
```

```
.giftCardDetails{
    text-align: left;
}
```

```
/* Style the inputs */
```

```
.input {
    position: absolute;
    top: 0px;
    left: 0px;
    height: 100%;
    width: 100%;
    border: 1px solid #DADCE0;
    border-radius: 7px;
    font-size: 16px;
    padding: 0 20px;
    outline: none;
    background: none;
    z-index: 1;
}
```

```
/* Hide the placeholder texts (a) */
```

```
::placeholder {
    color: transparent;
}
```

```
/* Styling text labels */
```

```
.label {
```

```
position: absolute;
top: 15px;
left: 15px;
padding: 0 4px;
background-color: white;
color: #DADCE0;
font-size: 16px;
transition: 0.5s;
z-index: 0;
}
```

```
.balanceBtn {
display: grid;
margin: auto;
padding: 15px 30px;
border: none;
background-color: purple;
color: white;
border-radius: 6px;
cursor: pointer;
font-size: 16px;
margin-top: 30px;
}
```

```
.balanceBtn:hover {
background-color: #9867C5;
transform: translateY(-2px);
}
```

```
.submitBtn {
display: inline;
margin-left: 25px;
padding: 15px 30px;
border: none;
background-color: purple;
color: white;
border-radius: 6px;
```



```
    cursor: pointer;
    font-size: 16px;
    margin-top: 30px;
}
```

```
.submitBtn:hover {
    background-color: #9867C5;
    transform: translateY(-2px);
}
```

```
.input:focus + .label {
    top: -7px;
    left: 3px;
    z-index: 10;
    font-size: 14px;
    font-weight: 600;
    color: purple;
}
```

```
.input:focus {
    border: 2px solid purple;
}
```

```
.input:not(:placeholder-shown)+ .label {
    top: -7px;
    left: 3px;
    z-index: 10;
    font-size: 14px;
    font-weight: 600;
}
```

```
.nav-list {
    list-style-type: none;
    margin: 0;
    padding: 0px;
    overflow: hidden;
    background-color: black;
```

```
height: 43px;  
}
```

```
.nav-item {  
    float: left;  
}
```

```
.nav-item-right a{  
    float:right;  
    color: white;  
    font-size:20px;  
    text-align: center;  
    padding: 10px 20px;  
}
```

```
.nav-item a {  
    display: block;  
    color: white;  
    font-size:20px;  
    text-align: center;  
    padding: 10px 20px;  
    text-decoration: none;  
}
```

```
.active{  
background-color: gray;  
color: white;  
}
```

```
.nav-item a:hover {  
    background-color: purple;  
    color: white;  
}
```

```
.nav-item-right a:hover {  
    background-color: purple;  
    color: white;
```

```
}
```

```
.home-page-image{  
    width: 100%;  
    padding-top: 25px;  
}
```

App.py

```
import ibm_boto3, ibm_db  
import uuid  
import requests, json  
import os  
from sendgrid import SendGridAPIClient  
from sendgrid.helpers.mail import Mail  
from datetime import datetime  
from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel  
from clarifai_grpc.grpc.api import service_pb2_grpc  
from clarifai_grpc.grpc.api import service_pb2, resources_pb2  
from clarifai_grpc.grpc.api.status import status_code_pb2  
from ibm_botocore.client import Config, ClientError  
from passlib.hash import sha256_crypt  
from flask import Flask, render_template, request, redirect, url_for, session,  
send_from_directory  
stub = service_pb2_grpc.V2Stub(ClarifaiChannel.get_grpc_channel())  
SENDGRID_API_KEY = 'SG._dXVE8-  
ORDew98pp8Ggusw.Os9Re3Z1cw9aErSRamKtaNayrA6fPDyk5f8_MHLdeGo'  
conn= ibm_db.connect("DATABASE=bludb;HOSTNAME=21fecfd8-47b7-4937-840d-  
d791d0218660.bs2io90l08kqb1od8lclg.databases.appdomain.cloud;PORT=31864;SECU  
RITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=wtm19331;PWD=PqN1q  
LnqGxUVePyP",;")  
COS_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud"  
COS_API_KEY_ID = "6j6IEFB66TlzbqHpfBUaaNA9ej3_dCjYlOWzDb3dBvt5"  
COS_INSTANCE_CRN = "crn:v1:bluemix:public:cloud-object-  
storage:global:a/040e6e55879742a39df9744b96d97e3f:ebaca05e-1864-4b3c-9aa1-  
e363d67806aa::"  
cos = ibm_boto3.resource("s3",
```

```

    ibm_api_key_id=COS_API_KEY_ID,
    ibm_service_instance_id=COS_INSTANCE_CRN,
    config=Config(signature_version="oauth"),
    endpoint_url=COS_ENDPOINT
)
app = Flask(__name__)
app.secret_key = "XYZ"

@app.route('/styles/<path:path>')
def send_report(path):
    return send_from_directory('styles', path)

@app.route('/signUpForm')
def form():
    return render_template('signup.html')

@app.route('/home')
def home():
    id = session['id']
    totalCalories = 0
    resultSet = []
    date = datetime.today().strftime('%Y-%m-%d')
    select_sql= "SELECT * FROM FOOD_DETAILS WHERE USER_ID= '"+str(id)+"' AND
DATE(UPLOADED_DATE_TIME) = '"+date+"'"
    select_stmt= ibm_db.exec_immediate(conn, select_sql)
    dictionary = ibm_db.fetch_assoc(select_stmt)
    while dictionary != False:
        resultSet.append(dictionary)
        dictionary = ibm_db.fetch_assoc(select_stmt)
    print(resultSet)
    for i in range(len(resultSet)):
        totalCalories += resultSet[i]['CALORIES']
    return render_template('home.html', data=totalCalories)

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

```

```

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

@app.route('/signup', methods = ['POST', 'GET'])
def signup():
    first_name = request.form['first_name']
    last_name = request.form['last_name']
    email_id = request.form['email_id']
    phone_number = request.form['phone_number']
    username = request.form['username']
    password = request.form['password']
    password = sha256_crypt.hash(password)
    insert_sql= "INSERT INTO WTM19331.USERS
(USERNAME,PASSWORD,FIRST_NAME,LAST_NAME,EMAIL_ID,PHONE_NUMBER)
VALUES('"+username+"', '"+password+"', '"+first_name+"', '"+last_name+"', '"+email_id+"',
 '"+phone_number+"')"
    print(insert_sql)
    insert_stmt= ibm_db.exec_immediate(conn,insert_sql)
    message = Mail(
        from_email='elatejeshwarcse2019@citchennai.net',
        to_emails=email_id,
        subject='Hello from Nutrition Assistant Team!',
        html_content='<strong> We are delighted to have you on board and hope to help
you in your day to day fitness and food intake goals!</strong>')
    try:
        sg = SendGridAPIClient(SENDGRID_API_KEY)
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e.message)
    return render_template('login.html')

@app.route('/logout', methods=['POST', 'GET'])

```

```
def logout():
    session.pop('id')
    return render_template('login.html')
```

```
@app.route('/login', methods = ['POST', 'GET'])
```

```
def login():
    username = request.form['username']
    password = request.form['password']
    select_sql= "SELECT * FROM USERS WHERE USERNAME = '"+username+"'"
    select_stmt= ibm_db.exec_immediate(conn, select_sql)
    dictionary = ibm_db.fetch_assoc(select_stmt)
    if(dictionary):
        if(sha256_crypt.verify(password, dictionary['PASSWORD'])):
            session['id'] = dictionary['ID']
            return redirect(url_for('home'))
        else:
            return render_template('login.html', status="incorrectPassword")
    else:
        return render_template('login.html', status="unknownFailure")
```

```
@app.route('/foodHistory')
```

```
def foodHistory():
    id = session['id']
    resultSet = []
    date = datetime.today().strftime('%Y-%m-%d')
    select_sql= "SELECT * FROM FOOD_DETAILS WHERE USER_ID= '"+str(id)+"' ORDER BY
UPLOADED_DATE_TIME DESC"
    select_stmt= ibm_db.exec_immediate(conn, select_sql)
    dictionary = ibm_db.fetch_assoc(select_stmt)
    while dictionary != False:
        resultSet.append(dictionary)
        dictionary = ibm_db.fetch_assoc(select_stmt)
    for i in range(len(resultSet)):
        date_time = resultSet[i]['UPLOADED_DATE_TIME']
        d = date_time.strftime("%d %b, %Y")
        resultSet[i]['UPLOADED_DATE_TIME'] = d
    print(resultSet)
```

```
return render_template('foodHistory.html', data=resultSet)
```

```
@app.route('/profile', methods=['POST','GET'])
```

```
def profile():
```

```
    id = session['id']
```

```
    select_sql= "SELECT * FROM USERS WHERE ID = '"+str(id)+"'"
```

```
    select_stmt= ibm_db.exec_immediate(conn, select_sql)
```

```
    dictionary = ibm_db.fetch_assoc(select_stmt)
```

```
    if(dictionary):
```

```
        print(dictionary)
```

```
        return render_template('profile.html', data=dictionary)
```

```
@app.route('/upload', methods=['POST', 'GET'])
```

```
def upload():
```

```
    if(request.method == 'POST'):
```

```
        bucket = 'nutrition-assistant-food-details'
```

```
        name_file = uuid.uuid4().hex
```

```
        f = request.files['file']
```

```
        print(f)
```

```
        multi_part_upload(bucket, name_file, f)
```

```
        id = session['id']
```

```
        image_url = 'https://s3.jp-tok.cloud-object-  
storage.appdomain.cloud/'+bucket+'/'+name_file
```

```
        #Clarifai API Call
```

```
        CLARIFAI_API_KEY = "517365bb2aa54f1e81b10c6beb05af4a"
```

```
        APPLICATION_ID = "search-test"
```

```
        metadata = (("authorization", f"Key {CLARIFAI_API_KEY}"),)
```

```
        apiRequest = 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="food-item-v1-recognition",
```

```
            user_app_id=resources_pb2.UserAppIDSet(app_id=APPLICATION_ID),
```

```
            inputs=[
```

```
                resources_pb2.Input(
```

```
                    data=resources_pb2.Data(image=resources_pb2.Image(url=image_url))
```

```
                )
```

```
            ],
```

```

)
response = stub.PostModelOutputs(apiRequest, metadata=metadata)
if response.status.code != status_code_pb2.SUCCESS:
    print(response)
    raise Exception(f"Request failed, status code: {response.status}")
print(response.outputs[0].data.concepts[0].name)
targetFoodItem = response.outputs[0].data.concepts[0].name
#RapidAPI Calorieninjas API Call
calorieApiURL = "https://calorieninjas.p.rapidapi.com/v1/nutrition"
querystring = {"query":targetFoodItem}
headers = {
    "X-RapidAPI-Key":
'4c91d633camsh5f6167437f10c9bp1d5217jsnf46fc8764733',
    "X-RapidAPI-Host": 'calorieninjas.p.rapidapi.com'
}
calorieApiResponse = requests.request("GET", calorieApiURL, headers=headers,
params=querystring)
calorieApiResponseDict = json.loads(calorieApiResponse.text)
print(calorieApiResponseDict)
targetFoodCalories = calorieApiResponseDict['items'][0]['calories']
insert_sql = "INSERT INTO WTM19331.FOOD_DETAILS
(USER_ID,FOOD_COS_NAME,CALORIES,UPLOADED_DATE_TIME,FOOD_NAME)
VALUES('"+str(id)+"', '"+name_file+"', '"+str(targetFoodCalories)+"', now(),
 '"+targetFoodItem+"')"
print(insert_sql)
insert_stmt= ibm_db.exec_immediate(conn,insert_sql)
return render_template('food_detail.html', data=calorieApiResponseDict,
image_url=image_url, targetFoodItem=targetFoodItem)

if(request.method == 'GET'):
    return render_template('upload.html')

def multi_part_upload(bucket_name, item_name, file):
    print(cos)
    print(bucket_name, item_name, file)
    try:
        print('Starting file transfer for ',item_name,' to bucket ',bucket_name)

```



```

#set 5MB chunks
part_size = 1024 * 1024 * 5

#set threshold to 15MB
file_threshold = 1024 * 1024 * 15

#set the transfer threshold and chunk size
transfer_config = ibm_boto3.s3.transfer.TransferConfig(
    multipart_threshold= file_threshold,
    multipart_chunksize= part_size
)

#The upload_fileobj method will automatically exwcute a multi-part upload in 5MB
chunks
cos.Object(bucket_name, item_name).upload_fileobj(
    Fileobj = file,
    Config = transfer_config
)
print("Transfer for [0] completed!\n".format(item_name))
except ClientError as be:
    print("CLIENT ERROR: ",be)
except Exception as e:
    print("Unable to complete multipart upload : ", e)

app.run(host='localhost', port=5000)
if __name__ == '__main__':
    app.debug = True
    app.run()

```

Deployment.yaml

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: nutrition-app-1
spec:

```

```
replicas: 1
selector:
  matchLabels:
    app: nutrition-app
template:
  metadata:
    labels:
      app: nutrition-app
spec:
  containers:
  - name: nutrition-app
    image: icr.io/nutritionass_namespace/chandru1661/nutrition-app
    command: ["/bin/sh"]
    args: ["-c", "while true; do echo Done Deploying sv-premier; sleep 3600;done"]
    imagePullPolicy: Always
    ports:
    - containerPort: 5000
```

Github Link : <https://github.com/IBM-EPBL/IBM-Project-49998-1660887628>

Demo Link : <https://drive.google.com/file/d/1mleTslILG3JrrGxwhetVs-sx-r8Tax2b/view?usp=sharing>