

NUTRITION ASSISTANT APPLICATION

PNT2022TMID02492

PROJECT REPORT

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

7.1 SPRINT 1

7.2 SPRINT 2

7.3 SPRINT 3

7.4 SPRINT 4

8. TESTING

8.1 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

1 INTRODUCTION

1.1 PROJECT OVERVIEW

This project aims at building a web App that automatically estimates food attributes such as ingredients and nutritional value by classifying the input image of food. Our method gives accurate food identification and Food API's to give the nutritional value of the identified food.

1.2 PURPOSE

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

2 LITERATURE SURVEY

2.1 EXISTING PROBLEM

Unhealthy diets have been identified as the important causing factor of such diseases. In this context, personalized nutrition emerges as a new research field for providing tailored food intake advice to individuals according to their physical, physiological data, and further personal information. Specifically, in the last few years, several types of research have proposed computational models for personalized food recommendation using nutritional knowledge and user data.

2.2 REFERENCES

Nutritional_biomarkers_and_machine_learning_for_personalized_nutrition_applications_and_health_optimization

[https:// www.researchgate.net/publication/360530930](https://www.researchgate.net/publication/360530930)

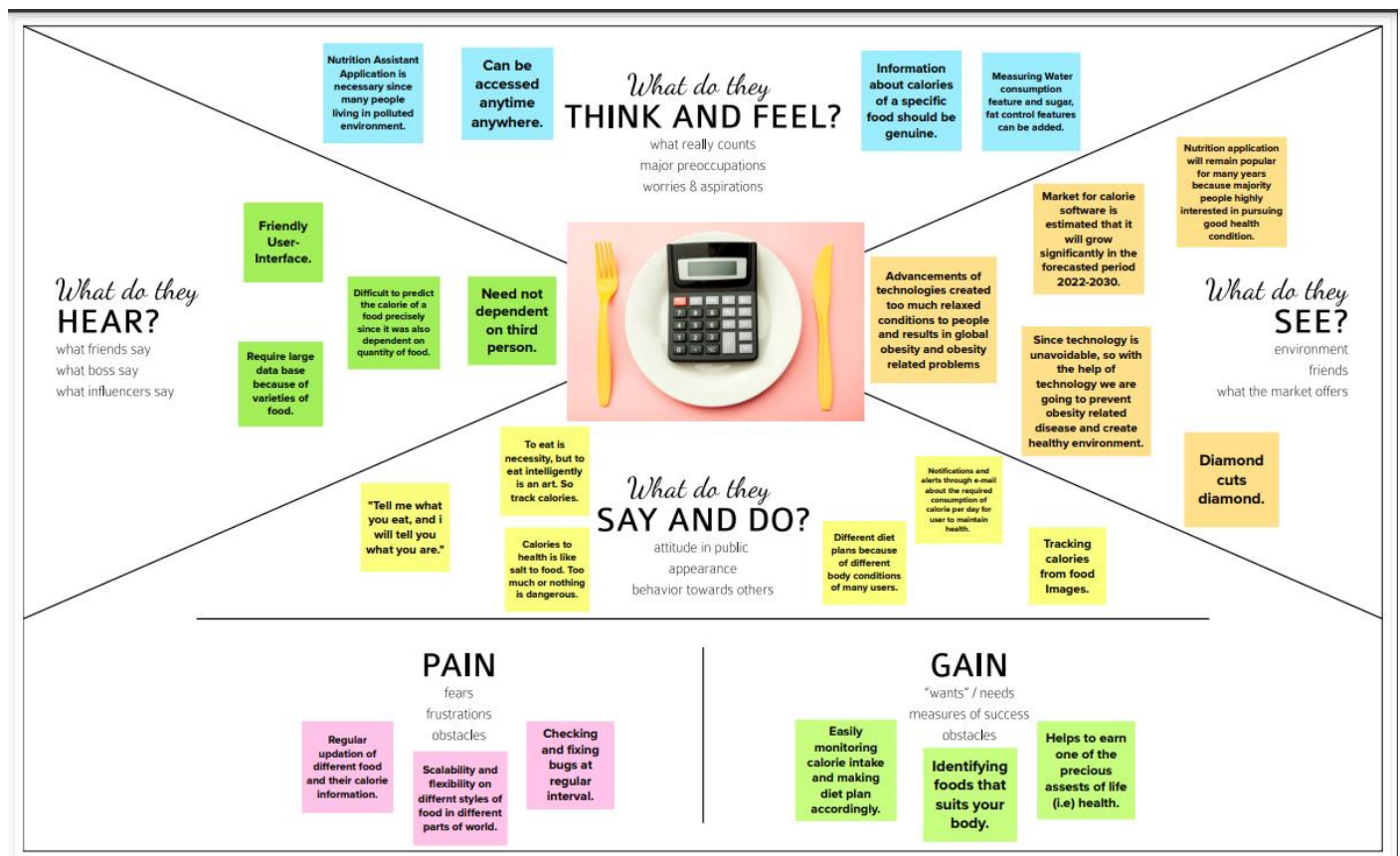
[https:// www.researchgate.net/publication/364203081](https://www.researchgate.net/publication/364203081)

2.3 PROBLEM STATEMENT

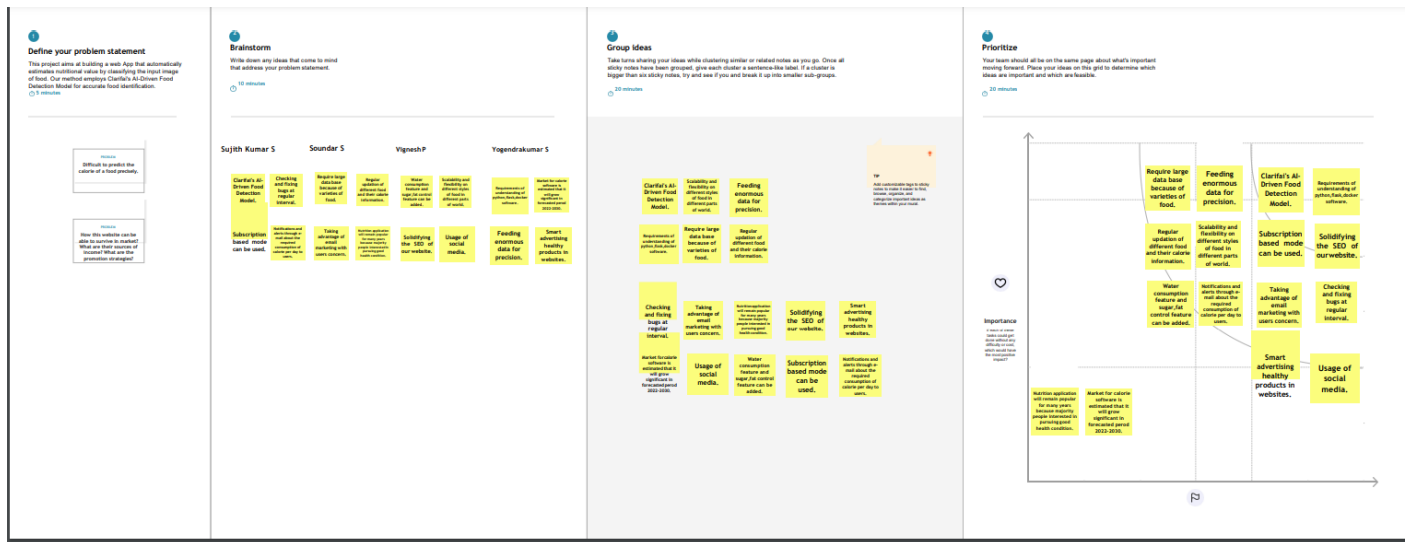
To build a web App that automatically estimates food attributes such as ingredients and nutritional value by classifying the input image of food. Our method gives accurate food identification and Food API's to give the nutritional value of the identified food.

3 IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING



3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To provide nutrition assistance to the user by displaying nutrients of the scanned food.
2.	Idea / Solution description	To display nutrients of the scanned food. To recommend food based on the BMI calculation.
3.	Novelty / Uniqueness	This project is unique for its multi-functionality. In this project, various functions are combined as a single app.
4.	Social Impact / Customer Satisfaction	Our system offers automated personalized visual feedback and recommendations based on individual dietary behavior, phenotype, and preferences.

5.	Business Model (Revenue Model)	It includes cost of equipment, services and fee paid to technology providers by initial development fund and local partners' fund.
6.	Scalability of the Solution	The database base can be updated accordingly. The input details can be changed by the authorized user anytime.

3.4 PROBLEM SOLUTION FIT

TITLE:NUTRITION ASSISTANT APPLICATION

TEAM ID: PNT2022TMDID02492

1. CUSTOMER SEGMENT(S) **CS**

Who is your customer?

The main customers for our project are:

- People who need to follow healthy lifestyle.
- Athletes who need to find them fit enough.
- Patients who conscious about diet routine.

6. CUSTOMER CONSTRAINTS **CC**

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

- Customers might get doubt in the accuracy of the results.
- They may Think ,this is time consuming process.

5. AVAILABLE SOLUTIONS **AS**

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons

ADVANTAGES	DISADVANTAGES
1. Customer can self planned their diet chart with the given daily report of nutritional values of their food.	1.Requires Larger database for different varieties of food.

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

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

- Create an user friendly application, where you can able to search healthy foods and obtain the nutrition values of each food that you are scanning.
- The Database of already scanned foods are stored in Memory.
- Show nutrients values of food so that customers get more insights into the food they eat.

9. PROBLEM ROOT CAUSE **RC**

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

- Lack of food education among people which makes them treat food as a hunger satiating tool not as fuel.
- Bad diet culture

7. BEHAVIOUR **BE**

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

- In order to build trust in the accuracy of the predictions, expert doctors and nutritionists can vouch for the exactness of the predictions

3. TRIGGERS **TR**

- Providing the health benefits of food

4. EMOTIONS BEFORE/AFTER:

EMOTIONS BEFORE	EMOTIONS AFTER
Lack of understanding about nutritional value of different foods	Empowered with the knowledge of nutrition which helps them live healthier lives
Inability to achieve their fitness goals due to lack of proper diet	Achieving fitness goals with ease
Unable to live a healthy and strong life	Leading a healthier and happier life

10. YOUR SOLUTION **SL**

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.

To develop an end to end web application that enables customers to understand the food they consume and make better food choices with the detailed nutrition analysis of the food they scanned.

The application aims to promote healthier food choices and make a much more healthier society.

8. CHANNELS of BEHAVIOUR **CH**

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

- Scan the foods they eat everyday
- Review the Nutrition summary of the foods they eat

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

- Make better food choices subconsciously as they can see their reports

4 REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email
FR-3	Food Detection	Scanning by Clarifai's AI-Driven Food Detection Model
FR-4	Nutrients Display	Display nutrients through IBM Cloud
FR-5	User BMI Calculation	Calculating Body Mass Index accurately

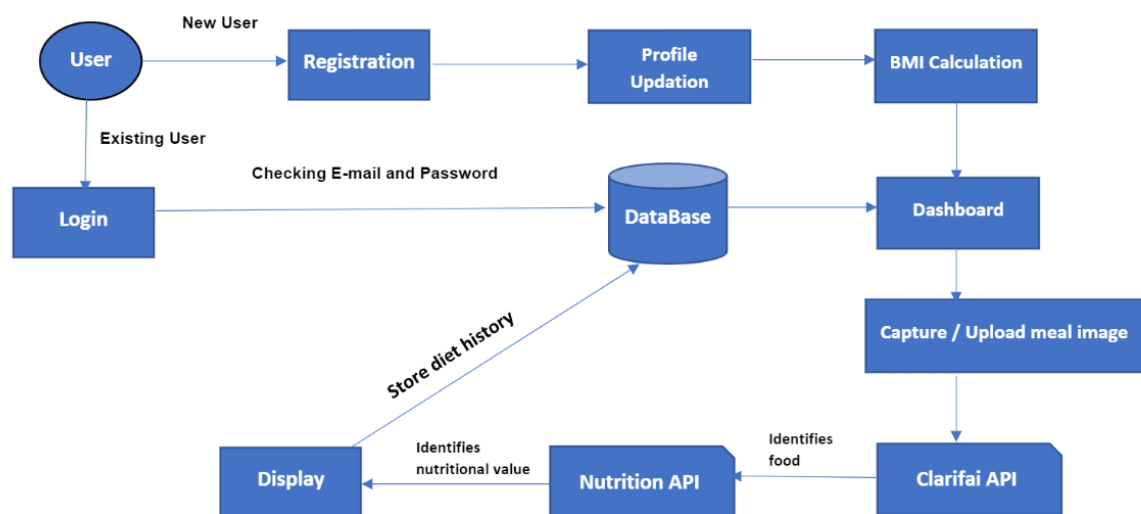
4.2 NON-FUNCTIONAL REQUIREMENTS:

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

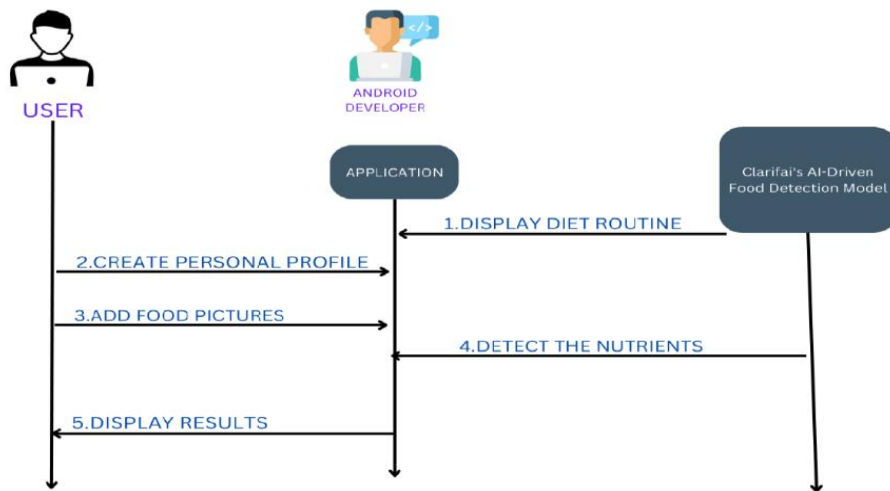
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Usage of Python as a programming language Flask as a Python framework Docker as a software platform IBM cloud as a cloud storage IBM DB2 as a database support
NFR-2	Security	Maintain user credentials such as username and user details in a secure manner.
NFR-3	Reliability	Reliable as it includes accurate BMI calculation and efficient nutrient display.
NFR-4	Performance	Provision of relevant scanning of food and best diet plan which makes the user follow a healthy diet.
NFR-5	Availability	Easily accessible as the user requires only a smart-phone with a good network connection.
NFR-6	Scalability	The database base can be updated accordingly. The input details can be changed by the authorized user anytime.

5 PROJECT DESIGN

5.1 DATA FLOW DIAGRAM



5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

Functional Requirement (Epic)	User Story Number	User Story / Task
User Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.
User Confirmation	USN-2	As a user, I will receive confirmation email once I have registered for the application
Food Detection	USN-3	As a user, I will upload the food image and the food will be detected
Nutrition Display	USN-4	As a user, I can view the nutrition contents of the detected food
User BMI Calculation	USN-5	As a user, I can view my calculated Body Mass Index

6 PROJECT PLANNING & SCHEDULING

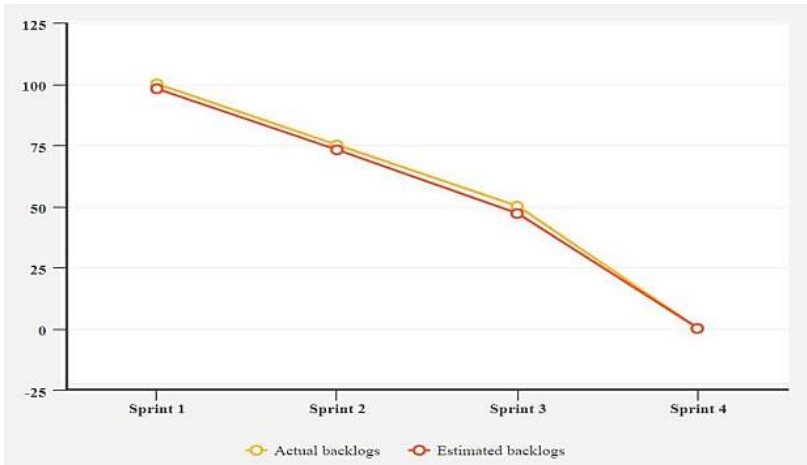
6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	User Panel	USN-1	The user will login into the website and go through the products available on the website.	20	High	Sujith Kumar S Vignesh P Soundar S Yogendrakumar S
Sprint-2	Admin Panel	USN-2	The role of the admin is to check out the database about the stock and have a truck of all the things that the users are purchasing.	20	High	Sujith Kumar S Vignesh P Soundar S Yogendrakumar S
Sprint-3	Chat Bot	USN-3	The user can directly talk to Chatbot regarding the products. Get the recommendations based on information provided by the user	20	High	Sujith Kumar S Vignesh P Soundar S Yogendrakumar S
Sprint-4	Final Delivery	USN-4	Container of applications using docker Kubernetes and development the application. Create the documentation and final submit the application	20	High	Sujith Kumar S Vignesh P Soundar S Yogendrakumar S

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	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	12 Nov 2022

6.3 REPORTS FROM JIRA



7 CODING

7.1 SPRINT

LOGIN.html

```
<!DOCTYPE html>

<html>
  <head>
    <title> Login Page</title>
    <style>
      body{
        background-image: url("image.jpg");
        background-repeat: no-repeat;
        background-size: cover;

      }
    </style>
  </head>
  <body>
    <form action="/Login" method="POST">
      <center> <h1> Login </h1>
      <label>Username:</label>
      <input type="text" placeholder="Enter Username" name="username" required>
      <br></br>
      <label>Password:</label>
      <input type="password" placeholder="Enter password" name="password" required>
      <br></br>
      <button type="submit"> Login</button>
      <br></br>
    </center>
    </form>

  </body>
</html>
```

REGISTER.HTML

```
!DOCTYPE html>

<html>
  <head>
    <title>Register page</title>
    <style>
      body{
```

```

        background-image: url("Naa.jpg");
        background-repeat:no-repeat;
        background-size: 100% 100%;

    }
</style>
</head>
<body>
    <form action="/Register" method="POST">
        <center> <h1>Singup</h1>
        <label><b>username:</b></label>
        <input type="text" placeholder="Enter Username" name="username" required>
        <br></br>
        <label><b>password:</b></label>
        <input type="password" placeholder="Enter password" name="password" required>
        <br></br>
        <label><b>phonenumber:</b></label>
        <input type="phonenumber" placeholder="Enter phone number " name="phonenumber" required>
        <br></br>
        <label><b>emailid:</b></label>
        <input type="emailid" placeholder="Enter email id" name="emailid" required>
        <br></br>
        <button type="submit"> submit</button>
        <br></br>
    </center>
    </form>

</body>
</html>

```

DASH.HTML

```

<!DOCTYPE html>

<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
    font-family: "Lato", sans-serif;
    background-image: url("image.jpg");
    background-repeat: no-repeat;
    background-image: cover;
}

```

```

.sidenav {
  height: 100%;
  width: 0;
  position: fixed;
  z-index: 1;
  top: 0;
  left: 0;
  background-color: #111;
  overflow-x: hidden;
  transition: 0.5s;
  padding-top: 60px;
}

.sidenav a {
  padding: 8px 8px 8px 32px;
  text-decoration: none;
  font-size: 25px;
  color: #818181;
  display: block;
  transition: 0.3s;
}

.sidenav a:hover {
  color: #f1f1f1;
}

.sidenav .closebtn {
  position: absolute;
  top: 0;
  right: 25px;
  font-size: 36px;
  margin-left: 50px;
}

@media screen and (max-height: 450px) {
  .sidenav {padding-top: 15px;}
  .sidenav a {font-size: 18px;}
}
</style>
</head>
<body>
<form action="/dash" method="POST">
<div id="mySidenav" class="sidenav">

```

```

<a href="javascript:void(0)" class="closebtn" onclick="closeNav()">&times;</a>
<a href="Login.html">Home</a>
<a href="Register.html">Register</a>
<a href="upload.html">Upload Image</a>
<a href="#">Food Items</a>
<a href="BMI_Calculation.html">BMI Calculation</a>
<a href="ref.html">Logout</a>
</div>
</form>

<span style="font-size:30px;cursor:pointer" onclick="openNav()">&#9776; Menubar</span>

<script>
function openNav() { document.getElementById("mySidenav").style.width
= "250px";
}

function closeNav() {
document.getElementById("mySidenav").style.width = "0";
}
</script>

</body>
</html>

```

7.2 SPRINT 2

UPOLOAD.HTML

```

<!DOCTYPE html>

<html>
<head>
  <title>select the file</title>
  <style>
    body{
      background-color: #ffb6c1;
    }
  </style>
</head>
<body>
<form action="/upload" method="POST">
<center>
<label for="myfile">Select a file:</label>

```

```

<input type="file" id="myfile" name="myfile" /> <br/><br/>
<input type="submit" value="submit" />
</center>

```

```

<P><b>This page helpful for check the nutrition value</b></P>
</form>
</body>
</html>

```

7.3 SPRINT 3

```

import streamlit as st
from PIL import Image
from keras_preprocessing.image import img_to_array
import numpy as np
from keras.models import load_model
import requests
from bs4 import BeautifulSoup

model = load_model('FV.h5')
labels = {0: 'apple', 1: 'banana', 2: 'beetroot', 3: 'bell pepper', 4: 'cabbage', 5: 'capsicum', 6: 'carrot', 7: 'cauliflower', 8: 'chilli pepper', 9: 'corn', 10: 'cucumber', 11: 'eggplant', 12: 'garlic', 13: 'ginger', 14: 'grapes', 15: 'jalepeno', 16: 'kiwi', 17: 'lemon', 18: 'lettuce', 19: 'mango', 20: 'onion', 21: 'orange', 22: 'paprika', 23: 'pear', 24: 'peas', 25: 'pineapple', 26: 'pomegranate', 27: 'potato', 28: 'raddish', 29: 'soy beans', 30: 'spinach', 31: 'sweetcorn', 32: 'sweetpotato', 33: 'tomato', 34: 'turnip', 35: 'watermelon'}

fruits = ['Apple','Banana','Bello Pepper','Chilli Pepper','Grapes','Jalepeno','Kiwi','Lemon','Mango','Orange','Paprika','Pear','Pineapple','Pomegranate','Watermelon'] vegetables = ['Beetroot','Cabbage','Capsicum','Carrot','Cauliflower','Corn','Cucumber','Eggplant','Ginger','Lettuce','Onion','Peas','Potato','Raddish','Soy Beans','Spinach','Sweetcorn','Sweetpotato','Tomato','Turnip']

def fetch_calories(prediction):
    url = 'https://www.google.com/search?q=calories in ' + prediction
    req = requests.get(url).text
    scrap = BeautifulSoup(req, 'html.parser')
    calories = scrap.find("div", class_="BNeawe iBp4i AP7Wnd").text
    return calories

def processed_img(img_path):
    img=load_img(img_path,target_size=(224,224,3))
    img=img_to_array(img)
    img=img/255
    img=np.expand_dims(img,[0])
    answer=model.predict(img)

```

```

y_class = answer.argmax(axis=-1)
print(y_class)
y = " ".join(str(x) for x in y_class)
y = int(y)
res = labels[y]
print(res)
return res.capitalize()

def run():
    st.title("Fruits-Classification")
    img_file = st.file_uploader("Choose an Image", type=["jpg", "png"]) if
    img_file is not None:
        img = Image.open(img_file).resize((250,250))
        st.image(img,use_column_width=False)
        save_image_path = './upload_images/'+img_file.name
        with open(save_image_path, "wb") as f:
            f.write(img_file.getbuffer())

    # if st.button("Predict"):
    if img_file is not None:
        result= processed_img(save_image_path)
        print(result)
        if result in vegetables:
            st.info("**Category: Vegetables**")
        else:
            st.info("**Category : Fruit**")
        st.success("**Predicted : "+result+"**")
        cal = fetch_calories(result)
        if cal:
            st.warning("**+cal+'(100 grams)**")

run()

```

7.4 SPRINT 4

BMI CALCULATION.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">
    <title>Document</title>
    <style>
        body{
            background-image: url("image.jpg");
            background-repeat: no-repeat;

```



```
        background-image: cover;
    }
</style>
</head>
<body>
    <form action="/BMI_Calculation" method="POST">
    <div class="calculator-container">
        <center>
            <h1>BMI CALCULATOR</h1>
            <label>Height:</label>
            <input class="Height-input-field" type="text">
            <br></br>
            <label>Weight:</label>
            <input class="Weight-input-field" type="text"><br>
            <button class="calculate"> Calculate</button>
        </div>
        <h3 class="result"></h3>
        <p class="result-statement"></p>
        <script src="script.js"></script>
        </center>
    </form>
</body>
</html>
```

7.5 USER ACCEPTANCE TESTING

7.5.1 Purpose of Document

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

7.5.2 Defect Analysis

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

Resolution	Severity1	Severity2	Severity3	Severity4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37

Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

7.5.3 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	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

8 RESULTS

8.1 PERFORMANCE METRICS

Performance metrics are known as numbers and data representing in the project abilities, action and overall quality. Various forms of performance metrics include content of project idea to success in organization's

9 ADVANTAGES AND DISADVANTAGES

Advantages:

The application reduces manual work.

The user would be able to maintain healthy diet.

The application saves time.

Disadvantages:

It requires manual entry of user details.

The application requires camera with more clarity.

10 CONCLUSION

The development of Nutrition Assistant Application was a good learning experience. Developing this application has given clear idea and knowledge about the cloud application development. This help in familiarising with Cloud and Docker concepts. This gave us insights into real-time software engineering.

11 FUTURE SCOPE

The scope of a Nutrition Assistant Application can cover many needs including prediction of nutrition in a mixed food items. The application may include meal planning according to the user location. The accuracy of the prediction and the variety of input comparison may be increased.

12 APPENDIX

```
<!DOCTYPE html>
<html>
  <head>
    <title> Login Page</title>
    <style>
      body{
        background-image: url("image.jpg");
        background-repeat: no-repeat;
        background-image: cover;

      }
    </style>
  </head>
  <body>
    <form action="/Login" method="POST">
      <center> <h1> Login</h1>
      <label>Username:</label>
      <input type="text" placeholder="Enter Username" name="username" required>
      <br></br>
      <label>Password:</label>
      <input type="password" placeholder="Enter password " name="password" required>
      <br></br>
      <button type="submit"> Login</button>
      <br></br>
    </center>
  </form>

  </body>
</html>
<!DOCTYPE html>
```

```

<html>
  <head>
    <title>Register page</title>
    <style>
      body{
        background-image: url("Naa.jpg");
        background-repeat:no-repeat;
        background-size: 100% 100%;

      }
    </style>
  </head>
  <body>
    <form action="/Register" method="POST">
      <center> <h1>Singup</h1>
      <label><b>username:</b></label>
      <input type="text" placeholder="Enter Username" name="username" required>
      <br></br>
      <label><b>password:</b></label>
      <input type="password" placeholder="Enter password " name="password" required>
      <br></br>
      <label><b>phonenummer:</b></label>
      <input type="phonenummer" placeholder="Enter phonenummer " name="phonenummer" required>
      <br></br>
      <label><b>emailid:</b></label>
      <input type="emailid" placeholder="Enter emailid" name="emailid" required>
      <br></br>
      <button type="submit"> submit</button>
      <br></br>
    </center>
  </form>

  </body>
</html>
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
  font-family: "Lato", sans-serif;
  background-image: url("image.jpg");
  background-repeat: no-repeat;
  background-image: cover;
}

.sidenav {
  height: 100%;
  width: 0;
  position: fixed;

```

```
z-index: 1;
top: 0;
left: 0;
background-color: #111;
overflow-x: hidden;
transition: 0.5s;
padding-top: 60px;
}
```

```
.sidenav a {
padding: 8px 8px 8px 32px;
text-decoration: none;
font-size: 25px;
color: #818181;
display: block;
transition: 0.3s;
}
```

```
.sidenav a:hover {
color: #f1f1f1;
}
```

```
.sidenav .closebtn {
position: absolute;
top: 0;
right: 25px;
font-size: 36px;
margin-left: 50px;
}
```

```
@media screen and (max-height: 450px) {
.sidenav {padding-top: 15px;}
.sidenav a {font-size: 18px;}
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<form action="/dash" method="POST">
```

```
<div id="mySidenav" class="sidenav">
```

```
<a href="javascript:void(0)" class="closebtn" onclick="closeNav()">&times;</a>
```

```
<a href="Login.html">Home</a>
```

```
<a href="Register.html">Register</a>
```

```
<a href="upload.html">Upload Image</a>
```

```
<a href="#">Food Items</a>
```

```
<a href="BMI_Calculation.html">BMI Calculation</a>
```

```
<a href="ref.html">Logout</a>
```

```
</div>
```

```
</form>
```

```
<span style="font-size:30px;cursor:pointer" onclick="openNav()">#9776; Menubar</span>
```

```
<script>
function openNav() {
    document.getElementById("mySidenav").style.width = "250px";
}

function closeNav() { document.getElementById("mySidenav").style.width
    = "0";
}
</script>
```

```
</body>
</html>
<!DOCTYPE html>
<html>
<head>
    <title>select the file</title>
    <style>
        body{
            background-color: #ffb6c1;
        }
    </style>
</head>
<body>
<form action="/upload" method="POST">
<center>
<label for="myfile">Select a file:</label>
<input type="file" id="myfile" name="myfile" /> <br/><br/>
<input type="submit" value="submit" />
</center>
```

```
<P><b>This page helpful for check the nutrition value</b></P>
</form>
</body>
</html>
```

```
from tkinter.tix import Meter
```

```
import ibm_db
from flask import Flask, redirect, render_template, request, session, url_for
```

```
app=Flask(__name__)
```

```
conn=ibm_db.connect('DATABASE=bludb;HOSTNAME=b1bc1829-6f45-4cd4-bef4-
10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32304;SECURITY=SSL;SSLServerCertificate
=certi.crt;UID=spy48271;PWD=80QbK1bDsTDY3NOO;','')
```

```

@app.route("/")
def index():
    return render_template("dash.html")


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


@app.route("/Register",methods=["GET","POST"])
def Register():
    if request.method=="POST":
        username=request.form['username']
        password=request.form['password']
        phonenumber=request.form['phonenumber']
        emailid=request.form['emailid']
        sql="Insert INTO REGISTER VALUES(?,?,?,?)"
        stmt=ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.bind_param(stmt,3,phonenumber)
        ibm_db.bind_param(stmt,4,emailid)
        ibm_db.execute(stmt)
        return render_template("Login.html")


@app.route("/Login",methods=["GET","POST"])
def Login():
    if request.method=="POST":
        Username=request.form['username']
        Password=request.form['password']
        sql="Insert INTO LOGIN VALUES(?,?)"
        stmt=ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,Username)
        ibm_db.bind_param(stmt,2>Password)
        ibm_db.execute(stmt)
        return render_template("upload.html")


@app.route("/upload",methods=["GET","POST"])
def upload():
    if request.method=="POST":
        myfile=request.form['myfile']
        sql="Insert INTO UPLOAD VALUES(?)"
        stmt=ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,myfile)
        ibm_db.execute(stmt)
        return render_template("BMI_Calculation.html")

```



```
@app.route("/BMI_Calculation",methods=["GET","POST"]) def
BMI_Calculation():
```

```
    if request.method=="POST":
        Height=request.form['Height']
        Weight=request.form['Weight']
        sql="Insert INTO BMI_CALCULATION VALUES(?,?)"
        stmt=ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,Height)
        ibm_db.bind_param(stmt,2,Weight)
        ibm_db.execute(stmt)
        return render_template("ref.html")
```

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

```
import streamlit as st
from PIL import Image
from keras_preprocessing.image import img_to_array
import numpy as np
from keras.models import load_model
import requests
from bs4 import BeautifulSoup
```

```
model = load_model('FV.h5')
labels = {0: 'apple', 1: 'banana', 2: 'beetroot', 3: 'bell pepper', 4: 'cabbage', 5: 'capsicum', 6: 'carrot', 7: 'cauliflower', 8: 'chilli
pepper', 9: 'corn', 10: 'cucumber', 11: 'eggplant', 12: 'garlic', 13: 'ginger', 14: 'grapes', 15: 'jalepeno', 16: 'kiwi', 17: 'lemon',
18: 'lettuce',
        19: 'mango', 20: 'onion', 21: 'orange', 22: 'paprika', 23: 'pear', 24: 'peas', 25: 'pineapple', 26: 'pomegranate', 27:
'potato', 28: 'raddish', 29: 'soy beans', 30: 'spinach', 31: 'sweetcorn', 32: 'sweetpotato', 33: 'tomato', 34: 'turnip', 35: 'watermelon'}
```

```
fruits = ['Apple','Banana','Bello Pepper','Chilli
Pepper','Grapes','Jalepeno','Kiwi','Lemon','Mango','Orange','Paprika','Pear','Pineapple','Pomegranate','Watermelon'] vegetables
= ['Beetroot','Cabbage','Capsicum','Carrot','Cauliflower','Corn','Cucumber','Eggplant','Ginger','Lettuce','Onion','Peas','Potato','Ra
ddish','Soy Beans','Spinach','Sweetcorn','Sweetpotato','Tomato','Turnip']
```

```
def fetch_calories(prediction):
    url = 'https://www.google.com/search?&q=calories in ' + prediction
    req = requests.get(url).text
    scrap = BeautifulSoup(req, 'html.parser')
    calories = scrap.find("div", class_="BNeawe iBp4i AP7Wnd").text
    return calories
```

```
def processed_img(img_path):
    img=load_img(img_path,target_size=(224,224,3))
```

```

img=img_to_array(img)
img=img/255
img=np.expand_dims(img,[0])
answer=model.predict(img)
y_class = answer.argmax(axis=-1)
print(y_class)
y = " ".join(str(x) for x in y_class)
y = int(y)
res = labels[y]
print(res)
return res.capitalize()

```

```

def run():
    st.title("Fruits-Classification")
    img_file = st.file_uploader("Choose an Image", type=["jpg", "png"])
    if img_file is not None:
        img = Image.open(img_file).resize((250,250))
        st.image(img,use_column_width=False)
        save_image_path = './upload_images/'+img_file.name
        with open(save_image_path, "wb") as f:
            f.write(img_file.getbuffer())

    # if st.button("Predict"):
    if img_file is not None:
        result= processed_img(save_image_path)
        print(result)
        if result in vegetables:
            st.info("**Category : Vegetables**")
        else:
            st.info("**Category : Fruit**")
        st.success("***Predicted : "+result+"**")
        cal = fetch_calories(result)
        if cal:
            st.warning("**'+cal+'(100 grams)**")

```

```

run()
import requests
from bs4 import BeautifulSoup

# def get_weather(place):
url='https://www.google.com/search?&q=calories in '+dal makhani'
req=requests.get(url).text
print(req)
scrap=Beautifulsoup(req,'html.parser')
tmp = scrap.find("div", class_="BNeawe iBp4i AP7Wnd").text
print(tmp)
<!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">
<title>Document</title>
<style>
  body{
    background-image: url("image.jpg");
    background-repeat: no-repeat;
    background-size: cover;
  }
</style>
</head>
<body>
  <form action="/BMI_Calculation" method="POST">
  <div class="calculator-container">
    <center>
      <h1>BMI CALCULATOR</h1>
      <label>Height:</label>
      <input class="Height-input-field" type="text">
      <br></br>
      <label>Weight:</label>
      <input class="Weight-input-field" type="text"><br>
      <button class="calculate"> Calculate</button>
    </div>
    <h3 class="result"></h3>
    <p class="result-statement"></p>
    <script src="script.js"></script>
    </center>
  </form>
</body>
</html>
```

GITHUB LINK

<https://github.com/IBM-EPBL/IBM-Project-20478-1659720339>