NUTRITION ASSISTANT APPLICATION

Team ID : PNT2022TMID50736

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

The main purpose of this system is to give the clearance of the food details for users to eat. Then the system creates an awareness about obesity in between the peoples.

2. LITERATURE SURVEY

2.1 Existing problem

- In this paper they introduce an application that is capable of classifying foods by capturing the images, provide correct nutritional information of Indian foods and classify the foods based on user's health condition.
- For the purpose of analysis, the user's food intake, calorie values, blood pressure and diabetes data are given. Also helps them to burn the calories by doing exercises or some changes in their daily routines.
- Our existing system consists of two modules; one is image classification module using convolutional neural network and the second module is the development of android application.

2.2 References

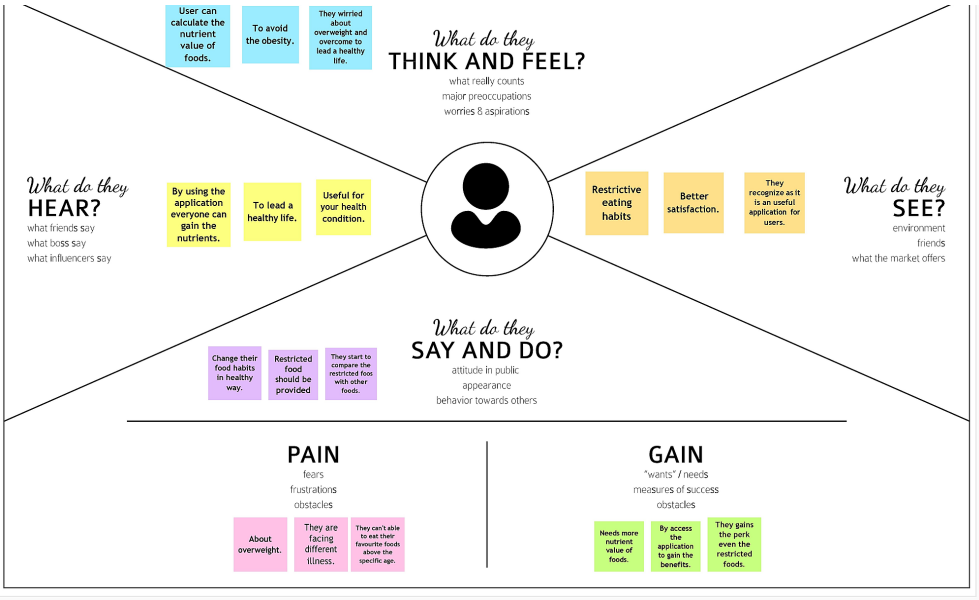
1. Dr. Kavita Sudersanadas, "APPLICATION OF ARTIFICIAL INTELLIGENCE ON NUTRITION ASSESSMENT AND MANAGEMENT", EUROPEAN JOURNAL OF PHARMACEUTICAL AND MEDICAL RESEARCH, volume: 8, issue: 6, pp: 170-174, 2021.
2. Mrs. Karthiyayini J, Prapul Kumar A, Pawan Jenu, Pavan Kumar S, "Food and Nutrition Evaluation for the Visually Impaired", International Journal for Research in Applied Science & Engineering Technology (IJRASET), volume: 8, issue: V, pp: 1893-1896, May 2020.
3. Sathiya T, Surya Prakash B, Thirukkumaran S V, Vijaiarivalagan K, "PREDICTION OF USER'S CALORIE ROUTINE USING CONVOLUTIONAL NEURAL NETWORK ", International Journal of Engineering Applied Sciences and Technology, volume: 5, issue: 3, pp: 189- 195, July 2020.
4. Karthik K, Vignesh K, M. Dhurgadevi,"Android Based Diet Consultant using Rule Pattern-based algorithm", Journal of Science Technology and Research (JSTAR), volume: 2, issue: 1,pp: 120-127, 2021.

2.3 Problem Statement Definition

- Now a days peoples are not eating healthy foods with respect to their health condition. If it continue, it will lead to obesity.
- Obesity in old age is associated with increased morbidity and reducing the quality of life.
- To avoid that the system will detect and recognize the food and evaluate the nutrient values for certain foods.
- In this system, we applied CNN algorithm to the task of food detection and recognition through parameter optimization.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

Problem Statement (PS)	Iam (Customer)	I'm trying to	But	Because	Which makes me feel

PS-1	user of an application	Knows the foodnutrients	I can't predictt he nutrients in food	Doesn't havean efficient system	Better
PS-2	Accessingt he application	Knows thebreakfast, lunch and dinner food and it's nutrient values.	I have astruggle to know the accurate nutrition value.	Can't be sureto say the food's nutrition values manually	Good

1 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.

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. • However, although food packaging comes with nutrition 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.

2 Brainstorm

Write down any ideas that come to mind that address your problem statement.
Remember, the key rules of brainstorming are:

- I want fast response time.
- User want to see the food details immediately
- Expectation from users to store various varities of food.
- Able to see the usage of application through the real time image of food.
- How to access the application even in technical issues
- I want the fovourites option to upload the user's favourite foods.

3 Group ideas

The facilitator should group all the ideas from the brainstorming process (step 2).
After that, you should add your opinions by adding arrows to point ideas into other groups and sticky notes and icons to share your thoughts.

P.Jenifer	M. Kali Gayathri	C. Maragathavalli	P. Selvi
Adapting new technology.	Scanning the real time food image.	Fetching the food and nutrient values from the database.	Helps to know the nutrient values of the foods.
Storing Food images and its nutrient values.	Comparing the scanned food with the stored food.	Retrieving the data.	To identify the food and its details for anytime and anywhere.

3.3 Proposed Solution

Sl.No.	Parameter	Description
1.	Problem Statement (Problem to besolved)	● Now a days peoples are not eatinghealthy foods with respect to their health condition. If it continue, it will load to obesity. 1. To avoid that the system will detect and recognize the food and evaluate the nutrient valuesfor certain foods. 2. In this system,we applied CNN algorithm to the task of food detection and recognition through parameter optimization.
2.	Idea / Solution description	● To store the food and its nutrient details then, scan the real time food and retrieve the corresponding food's nutrient values.
3.	Novelty / Uniqueness	● Clustering the peoples based on theirBMIvalue.
4.	Social Impact/ Customer Satisfaction	● The application which brings theawareness about the obesity in betweenthe peoples.
5.	Business Model(Revenue Model)	● In market, this application gives a benefit across the people health wiseandeconomical wise
6.	Scalability of the Solution	● The application which creates an impactamong the healthy lifestyle.

3.4 Problem Solution fit

Project Title: Nutrition Assistant Application
Team ID: PNT2022TMID50736

Project Design Phase-I - Solution Fit Template

1. CUSTOMER SEGMENT(S)

- **People who are accessing the application those are the users. They can able to view the nutrient values of the foods.**

6. CUSTOMER CONSTRAINTS

- The customers can able to see the certain amount of foods and their nutrient values.

5. AVAILABLE SOLUTIONS

- To provide the nutrient values.

2. JOBS-TO-BE-DONE / PROBLEMS

- Reduce time
- Over weight
- Obesity

9. PROBLEM ROOT CAUSE

- Mostly affected by aged people.

7. BEHAVIOUR

- The customer can predict their obesity level through the over weight.

3. TRRIGGERS

- We trigger the user to maintain their health condition normally through that application.

10. YOUR SOLUTION

- Easy to understand the food and its details by anybody. Such as ingredients and nutrient values.

8. CHANNELS OF BEHAVIOUR

- **Online**

  We collect the user details in online.

- **Offline**

  They follows the diabetic procedures with respect to the provided guide line.

4. EMOTIONS: BEFORE / AFTER

BEFORE:

- People feels uncomfortable, because of over weight.

AFTER:

- People feels better. Because of reduce the over weight.

4.

REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form
FR-2	Gathering food	Shows the food and its nutrient values

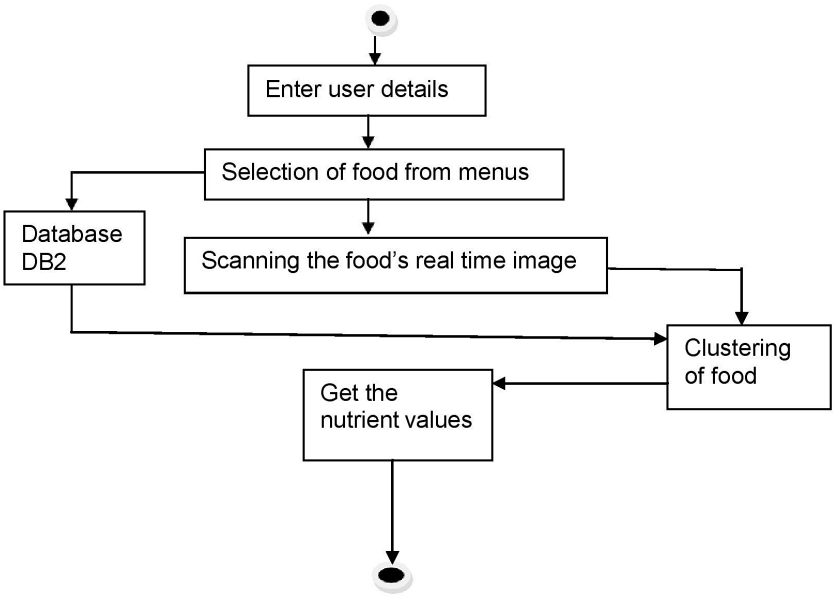
FR-3	Scanning	Scanning the real time image of food
FR-4	Retrieving	Retrieves the food details from the stored data

4.2 Non-Functional requirements

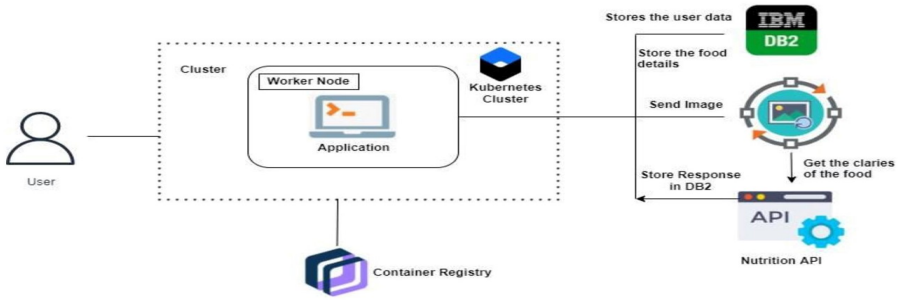
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Ease of use and knows the details immediately.
NFR-2	**Security**	The application whichprotects the dataefficiently over the web.
NFR-3	**Reliability**	The application can be usedin a confidential manner.
NFR-4	**Performance**	The performance of the application is very effective.
NFR-5	**Availability**	The application whichCan be easy to access.
NFR-6	**Scalability**	User access time is less, so that application is scalable.

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



5.3 User Stories

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Create UI to interactwith application	USN-1	As a users, they can interact with application.	1	Low	Kali Gayathri M Jenifer P Maragathavalli CSelviP
Sprint-2	Create IBM DB2 withpython	USN-2	As a user, should connect the database and python for stores the information aboutfood.	1	Medium	Kali Gayathri M Jenifer P Maragathavalli CSelviP
Sprint-3	IntegrateNutrition API	USN-3	As a user, shouldintegrate the nutrition API to see theresults.	1	Low	Kali Gayathri M Jenifer P Maragathavalli CSelviP

Sprin t-4	Output	USN-4	As a user, will seethe result as the selectedfood and its nutrient values.	2	Low	Kali Gayathri M Jenifer P Maragathaval li CSelviP

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Title	Description	Date
Literature Survey & InformationGathering	Literature survey on the selectedproject & gathering information by referring the, technical papers, research publications	24 September 2022
Prepare Empathy Map	Prepare Empathy Map Canvas tocapture the user Pains & Gains, Prepare list of problem statements	25 September 2022
Ideation	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility &importance	20 September 2022
Proposed Solution	Prepare the proposed solution document, which includes thenovelty, feasibility of idea, business model, social impact, scalability of solution, etc	19 September 2022
Problem Solution Fit	Prepare problem solution fit document	20 September 2022
Solution Architecture	Prepare solution architecture document	19 September 2022
Customer Journey	Prepare the customer journey maps to understand the user interactions &experiences with the application	03 October 2022

Functional Requirement	Prepare the functional requirement document	03 October 2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review	03 October 2022
Technology Architecture	Prepare the technology architecture diagram	03 October 2022
Prepare Milestone &Activity List	Prepare the milestones & activity list of the project	22 October 2022
Project Development - Delivery of Sprint-1, 2, 3 , 4	Develop & submitthe developed code by testing it	In Progress…

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	30 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	06 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	15 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

7. CODING

7.1 SPRINT- 1

index.html

```
const signUpButton = document.getElementById('signUp');
 const signInButton = document.getElementById('signIn');
 const container = document.getElementById('container');
 const switchone = document.getElementById('c1');
 const switchtwo = document.getElementById("c2");
 const switchthree = document.getElementById('c3');
 const switchfour = document.getElementById('c4');
 const Fswitchone = document.getElementById('l1');
 const Fswitchtwo = document.getElementById("l2");
 const Fswitchthree = document.getElementById('l3');
 const Fswitchfour = document.getElementById('l4');
 const space = document.getElementById('infos');
 var pre_state = 0;
 var stateone = 0;
 var statetwo = 0;
 var statethree = 0;
 signUpButton.addEventListener('click', () => {
 container.classList.add("right-panel-active");
```

```
});
 signInButton.addEventListener('click', () => {
 container.classList.remove("right-panel-active");
});
switchone.addEventListener('click', remover);
switchtwo.addEventListener('click', signin);
switchthree.addEventListener('click', Signup)
switchfour.addEventListener('click', about);
Fswitchone.addEventListener('click', remover);
Fswitchtwo.addEventListener('click', signin);
Fswitchthree.addEventListener('click', Signup)
Fswitchfour.addEventListener('click', about);
function remover() {
 if(pre_state == 1){
 pre_state = 0;
 space.classList.remove("spaceimp");
 document.getElementById("abouts").style.display = "none";
 document.getElementById("logins").style.display = "none";
 document.getElementById("l1").style.display = "flex";
 document.getElementById("l2").style.display = "flex";
 document.getElementById("l3").style.display = "flex";
 document.getElementById("l4").style.display = "flex";
}
}
function div_adder () {
 space.classList.add("spaceimp");
 document.getElementById("abouts").style.display = "none";
 document.getElementById("logins").style.display = "block";
 document.getElementById("l1").style.display = "none";
 document.getElementById("l2").style.display = "none";
 document.getElementById("l3").style.display = "none";
 document.getElementById("l4").style.display = "none";
}
function about_adder () {
 //space.classList.add("spaceimp");
// remover();
 document.getElementById("abouts").style.display = "block";
 document.getElementById("l1").style.display = "none";
 document.getElementById("l2").style.display = "none";
 document.getElementById("l3").style.display = "none";
 document.getElementById("l4").style.display = "none";
}
function signin() {
 if(pre_state == 0) {
 pre_state = 1;
 stateone = 1;
 statetwo = 0;
 statethree = 0;
 container.classList.remove("right-panel-active");
 div_adder();
 }else {
 if(stateone == 0) {
 pre_state = 1;
 stateone = 1;
```

```
 statetwo = 0;
 statethree = 0;
 container.classList.remove("right-panel-active");
 div_adder();
 }else {
 remover();
 }
 }
}
function Signup() {

 if(pre_state == 0) {
 pre_state = 1;
 stateone = 0;
 statetwo = 1;
 statethree = 0;
 container.classList.add("right-panel-active");
 div_adder();
 }else {
 if(statetwo == 0) {
 pre_state = 1;
 stateone = 0;
 statetwo = 1;
 statethree = 0;
 container.classList.add("right-panel-active");
 div_adder();
 }else {
 remover();
 }
 }
}
function about() {
 if(pre_state == 0){
 pre_state = 1;
 stateone = 0;
 statetwo = 0;
 statethree = 3;
 about_adder();
 }else{
 if(statethree == 0){
 remover();
 pre_state = 1;
 stateone = 0;
 statetwo = 0;
 statethree = 3;
 about_adder();
 }else{
 remover();
 }
 }

}
function unvisible(x) {
 if(pre_state == 0) {
```

```
document.getElementById("l1").style.display = "none";
document.getElementById("l2").style.display = "none";
document.getElementById("l3").style.display = "none";
document.getElementById("l4").style.display = "none";
}

}
function visible(x){
if(pre_state == 0) {
document.getElementById("abouts").style.display = "block";
//space.classList.add("spaceimp");
container.classList.add("right-panel-active");
document.getElementById("l1").style.display = "none";
document.getElementById("l2").style.display = "none";
document.getElementById("l3").style.display = "none";
document.getElementById("l4").style.display = "none";
}
}
function unsignin(x) {
if(pre_state == 0){
container.classList.remove("right-panel-active");
space.classList.remove("spaceimp");
document.getElementById("logins").style.display = "none";
document.getElementById("abouts").style.display = "none";
document.getElementById("l1").style.display = "flex";
document.getElementById("l2").style.display = "flex";
document.getElementById("l3").style.display = "flex";
document.getElementById("l4").style.display = "flex";
}

}
function signinOne(x){
if(pre_state == 0) {
container.classList.remove("right-panel-active");
space.classList.add("spaceimp");
document.getElementById("logins").style.display = "block";
document.getElementById("l1").style.display = "none";
document.getElementById("l2").style.display = "none";
document.getElementById("l3").style.display = "none";
document.getElementById("l4").style.display = "none";
}
}
function signinTwo(x){
if(pre_state == 0) {
document.getElementById("logins").style.display = "block";
space.classList.add("spaceimp");
container.classList.add("right-panel-active");
document.getElementById("l1").style.display = "none";
document.getElementById("l2").style.display = "none";
document.getElementById("l3").style.display = "none";
document.getElementById("l4").style.display = "none";
}
function setcon(x) {
if(pre_state == 0) {
```

```
 document.getElementById("abouts").style.display = "block";
 //space.classList.add("spaceimp");
 container.classList.add("right-panel-active");
 document.getElementById("l1").style.display = "none";
 document.getElementById("l2").style.display = "none";
 document.getElementById("l3").style.display = "none";
 document.getElementById("l4").style.display = "none";
 }
 }
}
```

Footer

Index.css

```
@media only screen and (max-width :768px ) {
 .colh {
 height: auto;
 }
 .lists {
 height: 330px;
 overflow: auto;
 flex-direction: column;
 }
 }
```

App.py

```
from flask import Flask, render_template, url_for, request
import ibm_db
import sendgrid
from sendgrid.helpers.mail import Mail, Email, To, Content
SENDGRID_API_KEY =
"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" # sendgrid
conn = ibm_db.connect(
 "DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT"

"=32733;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=wqq31800;PWD=tOlYo6K1IKA
c0XhU",
 ", ")
print(conn)
app = Flask(__name__)
app.secret_key = "\xfd{H\xe5<\x95\xf9\xe3\x96.5\xd1\x01O<!\xd5\xa2\xa0\x9fR"
# sendgrid
def send_mail(email):
 sg = sendgrid.SendGridAPIClient(SENDGRID_API_KEY)
 from_email = Email("xxxxxxxxxxxxxxxxxx@gmail.com") # Change to your verified sender
 to_email = To(email) # Change to your recipient
 subject = "Nutrition is a basic human need and a prerequisite for healthy life"
 content = Content("text/plain",
 "Thank you for creating an account on our platform. Now you can utilise our platform "
 "to maintain a healthier life.")
 mail = Mail(from_email, to_email, subject, content)
 # Get a JSON-ready representation of the Mail object
 mail_json = mail.get()
 # Send an HTTP POST request to /mail/send
 response = sg.client.mail.send.post(request_body=mail_json)
 print(response.status_code)
```

```
 print(response.headers)
@app.route('/', methods=['GET', 'POST'])
@app.route('/home', methods=['GET', 'POST'])
def homepage():
 if request.method == 'POST' and 'email' in request.form and 'pass' in request.form:
 return render_template('index.html', error="Wrong Password!")
 return render_template('index.html')
@app.route('/register', methods=['GET', 'POST'])
def register():
 if request.method == 'POST' and 'name' in request.form and 'email' in request.form and 'pass' in
request.form:
 name = request.form['name']
 email_up = request.form['email']
 pass_up = request.form['pass']
 if name == "":
 error = 'Enter a valid Name.'
 return render_template('index.html', error=error)
 if email_up == "":
 error = 'Enter a valid E-mail.'
 return render_template('index.html', error=error)
 if pass_up == "":
 error = 'Enter a valid Password.'
 return render_template('index.html', error=error)
 sql = "SELECT * FROM USER WHERE email =?"
 stmt = ibm_db.prepare(conn, sql)
 ibm_db.bind_param(stmt, 1, email_up)
 ibm_db.execute(stmt)
 account = ibm_db.fetch_assoc(stmt)
 if account:
 return render_template('index.html', error="You are already a member, please login using your
details")
 else:
 try:
 insert_sql = "INSERT INTO USER VALUES ,?,?)"
 prep_stmt = ibm_db.prepare(conn, insert_sql)
 ibm_db.bind_param(prep_stmt, 1, name)
 ibm_db.bind_param(prep_stmt, 2, email_up)
 ibm_db.bind_param(prep_stmt, 3, pass_up)
 ibm_db.execute(prep_stmt)
 send_mail(email_up)
 return render_template('index.html', error="Successfully created")
 except ibm_db.stmt_error:
 print(ibm_db.stmt_error())
 return render_template('index.html', error="Failed to create Account")
 return render_template('index.html')
if __name__ == '__main__':
 app.debug = True
 app.run()
prediction
from flask import Flask, render_template, url_for, request
import ibm_db
import sendgrid
from sendgrid.helpers.mail import Mail, Email, To, Content
SENDGRID_API_KEY =
```

```
"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" # sendgrid
conn = ibm_db.connect(
 "DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT"

"=32733;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=wqq31800;PWD=tOlYo6K1IKA
c0XhU",
 ", ")
print(conn)
app = Flask(__name__)
app.secret_key = "\xfd{H\xe5<\x95\xf9\xe3\x96.5\xd1\x01O<!\xd5\xa2\xa0\x9fR"
# sendgrid
def send_mail(email):
 sg = sendgrid.SendGridAPIClient(SENDGRID_API_KEY)
 from_email = Email("Ram@gmail.com") # Change to your verified sender
 to_email = To(email) # Change to your recipient
 subject = "Nutrition is a basic human need and a prerequisite for healthy life"
 content = Content("text/plain",
 "Thank you for creating an account on our platform. Now you can utilise our platform "
 "to maintain a healthier life.")
 mail = Mail(from_email, to_email, subject, content)
 # Get a JSON-ready representation of the Mail object
 mail_json = mail.get()
 # Send an HTTP POST request to /mail/send
 response = sg.client.mail.send.post(request_body=mail_json)
 print(response.status_code)
 print(response.headers)
@app.route('/', methods=['GET', 'POST'])
@app.route('/home', methods=['GET', 'POST'])
def homepage():
 if request.method == 'POST' and 'email' in request.form and 'pass' in request.form:
 return render_template('index.html', error="Wrong Password!")
 return render_template('index.html')
@app.route('/register', methods=['GET', 'POST'])
def register():
 if request.method == 'POST' and 'name' in request.form and 'email' in request.form and 'pass' in
request.form:
 name = request.form['name']
 email_up = request.form['email']
 pass_up = request.form['pass']
 if name == "":
 error = 'Enter a valid Name.'
 return render_template('index.html', error=error)
 if email_up == "":
 error = 'Enter a valid E-mail.'
 return render_template('index.html', error=error)
 if pass_up == "":
 error = 'Enter a valid Password.'
 return render_template('index.html', error=error)
 sql = "SELECT * FROM USER WHERE email =?"
 stmt = ibm_db.prepare(conn, sql)
 ibm_db.bind_param(stmt, 1, email_up)
 ibm_db.execute(stmt)
 account = ibm_db.fetch_assoc(stmt)
```

```
    if account:
    return render_template('index.html', error="You are already a member, please login using your
   details")
    else:
    try:
    insert_sql = "INSERT INTO USER VALUES ,?,?)"
    prep_stmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(prep_stmt, 1, name)
    ibm_db.bind_param(prep_stmt, 2, email_up)
    ibm_db.bind_param(prep_stmt, 3, pass_up)
    ibm_db.execute(prep_stmt)
    send_mail(email_up)
    return render_template('index.html', error="Successfully created")
    except ibm_db.stmt_error:
    print(ibm_db.stmt_error())
    return render_template('index.html', error="Failed to create Account")
    return render_template('index.html')
   if __name__ == '__main__':
    app.debug = True
    app.run()
```

Upload.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Upload Face with ID</title>

</head>

<body>


<div class="container">

<div class="row">

<div class="col">


<h1>Upload Face (filename = face's name (i.e. John_Smith.jpg)</h1>

<hr>


<form action="/upload-image" method="POST" enctype="multipart/form-data">
```

```
<div class="form-group">

<label>Select image</label>

<div class="custom-file">

<input type="file" class="custom-file-input" name="image"


id="image">

<label class="custom-file-label" for="image">Select image...</label>

</div>

</div>

<button type="submit" class="btnbtn-primary">Upload</button>

</form>


</div>

</div>

</div>


<imgsrc="{{ uploaded_image }}">


</body>

</html> Upload.py  importos


from flask import Flask, redirect, jsonify, request, url_for, render_template, flash


app = Flask(__name__)


app.config["IMAGE_UPLOADS"] = "C:/Flask/Upload/"



@app.route("/") def home():
```

returnrender_template("index.html")

Route to upload image

@app.route('/upload-image', methods=['GET', 'POST'])
defupload_image(): ifrequest.method == "POST": ifrequest.files:
image = request.files["image"]

print(image + "Uploaded to Faces")

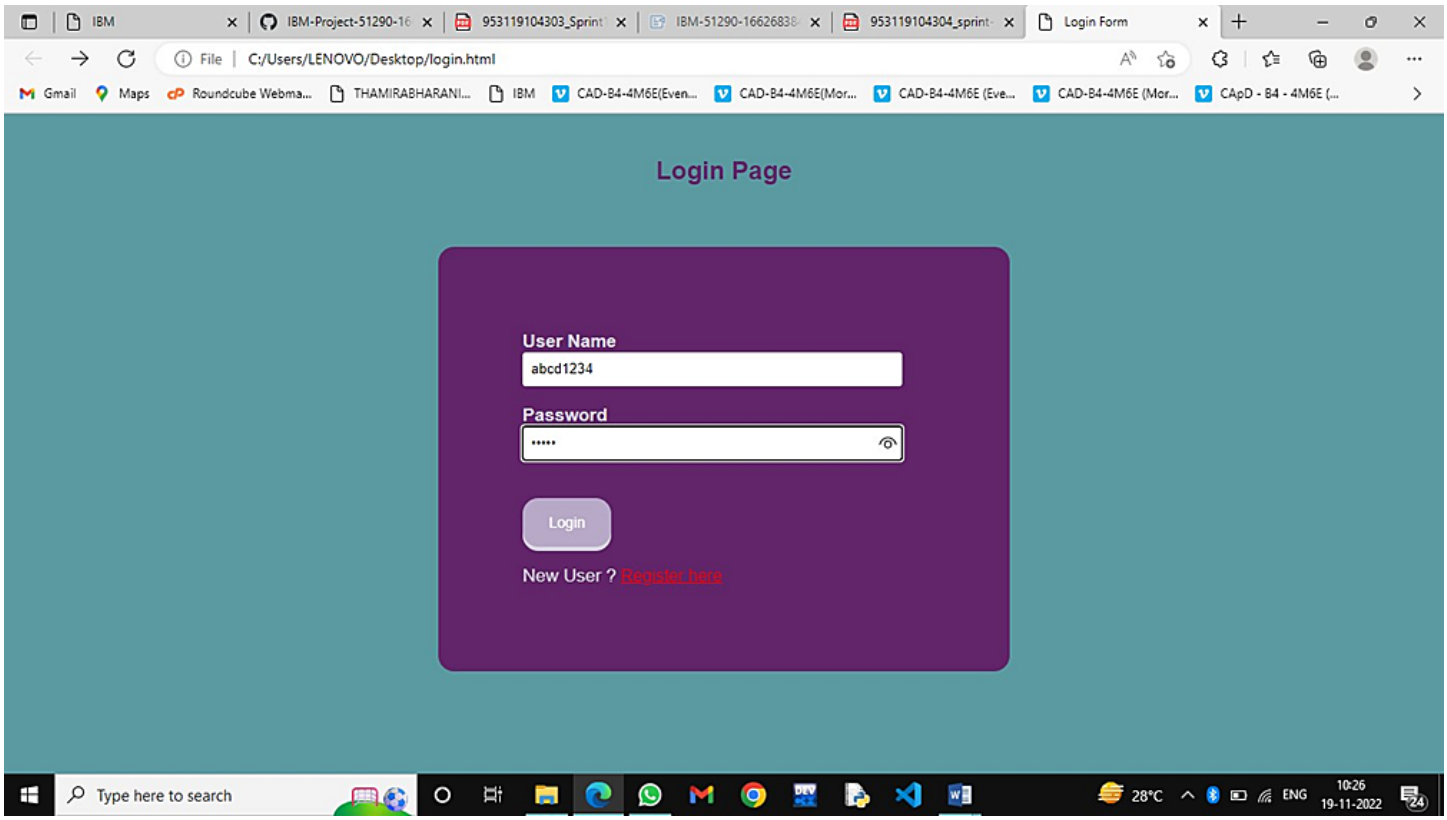
flash('Image successfully Uploaded to Faces.')

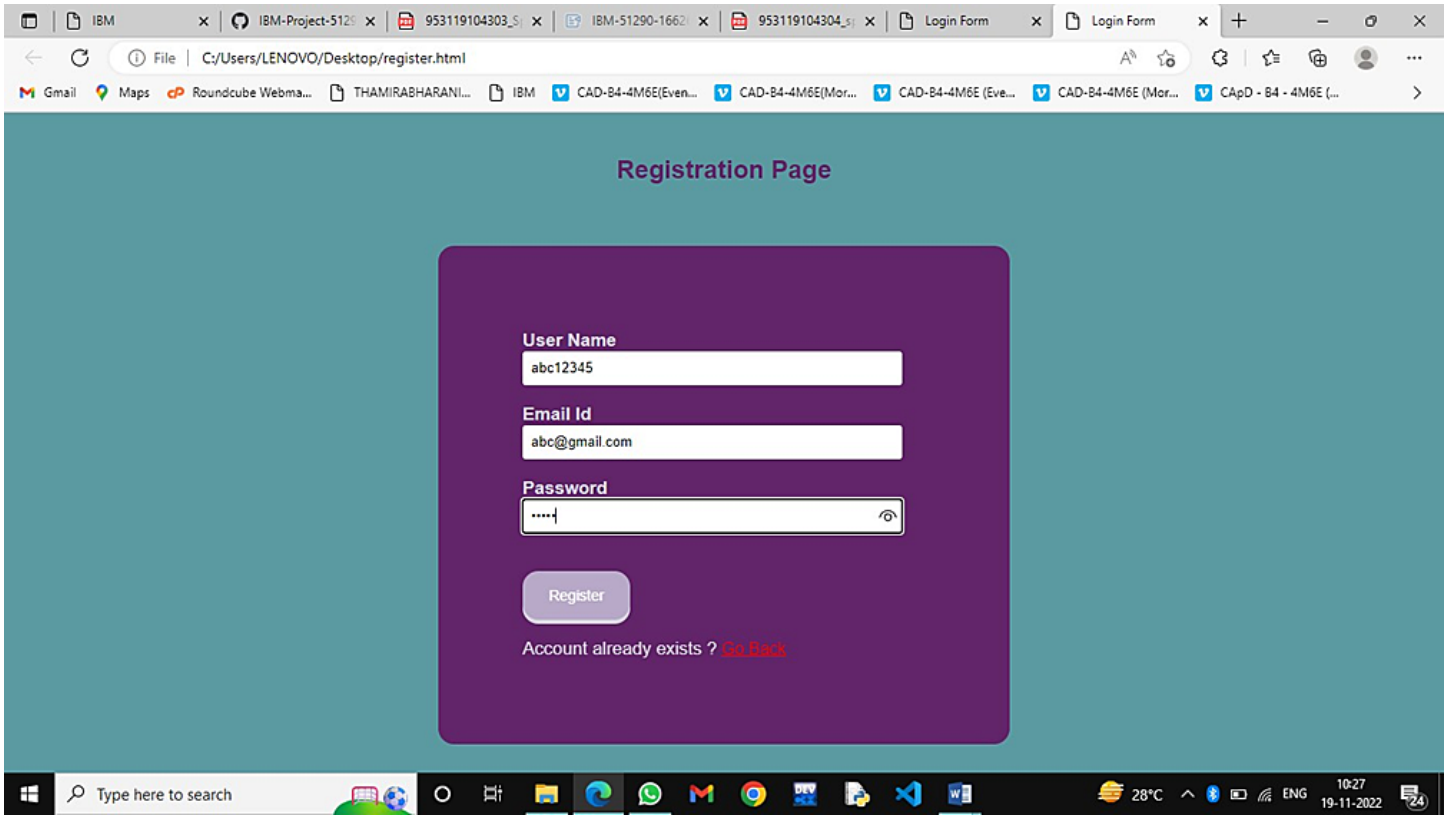
image.save(os.path.join(app.config["IMAGE_UPLOADS"], image.filename)) filename =
os.path.join(app.config["IMAGE_UPLOADS"], image.filename) print("stored as:" + filename)

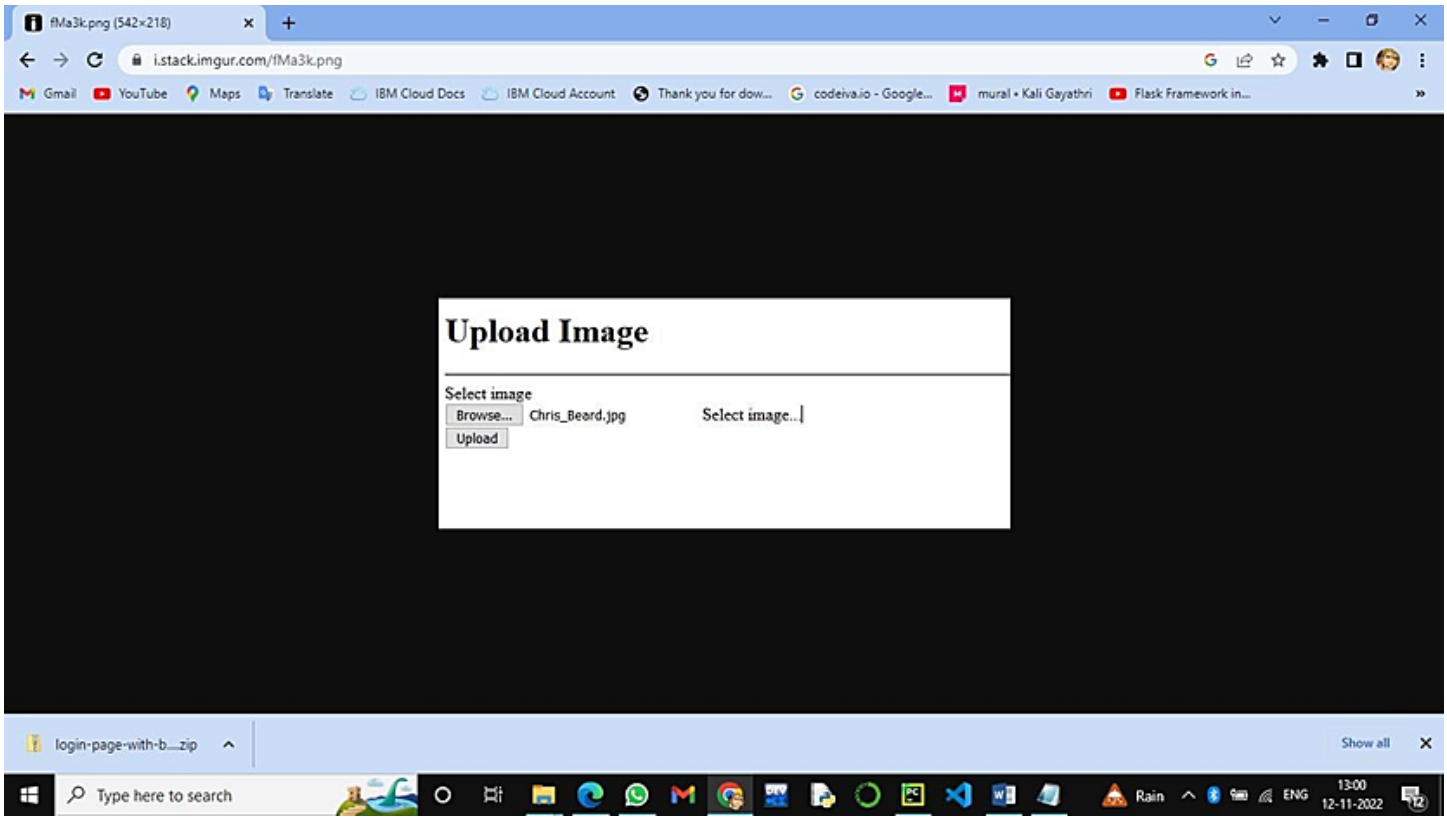
returnrender_template("upload_image.html",
uploaded_image=filename) returnrender_template("upload_image.html") if
__name__ == "__main__":

app.run()

output:




Registration Page
User Name
abc12345
Email Id
abc@gmail.com
Password
Register
Account already exists ? Go Back


Upload Image
Select image
Browse...
Chris_Beard.jpg
Select image...
Upload

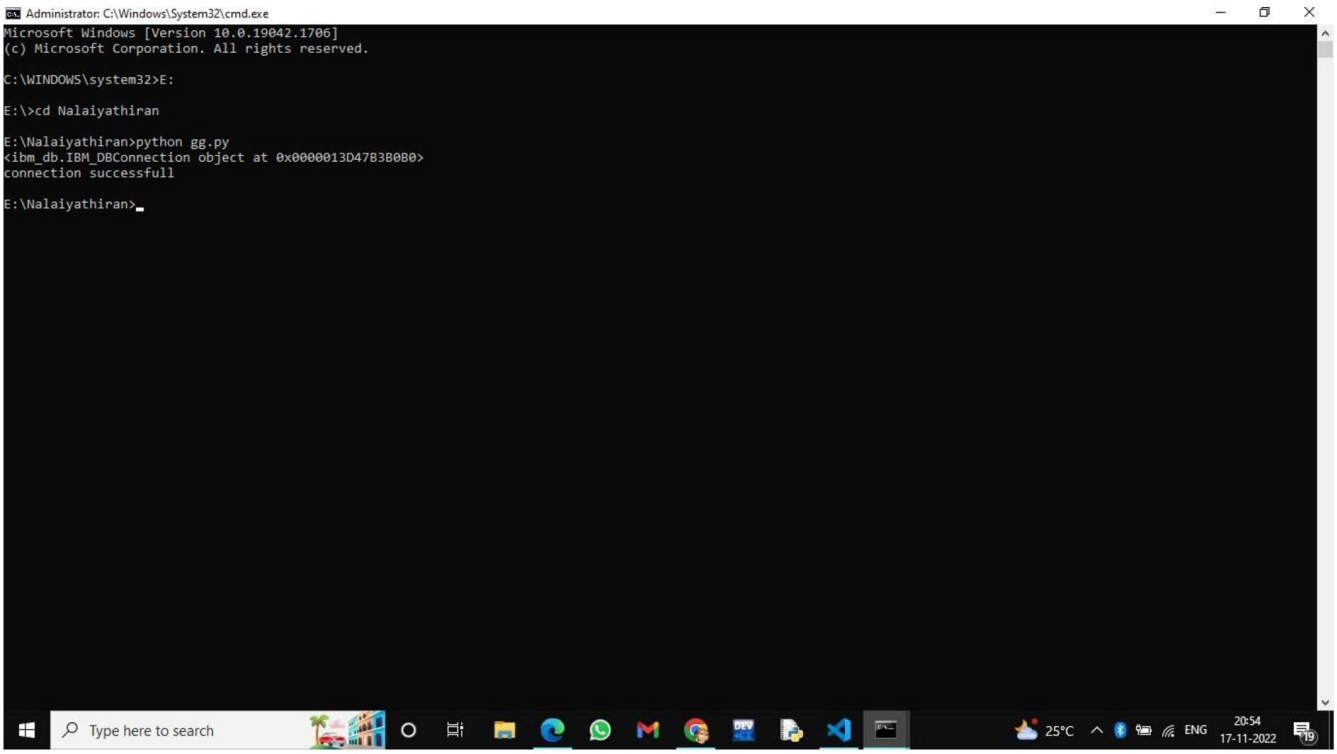
7.2 SPRINT-2

DB2 connection with python

gg.py

```
from flask import Flask
import ibm_db
conn = ibm_db.connect( "DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURIT
Y=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=wqq31800;PWD=tOlYo6K1I
KAc0XhU",
    '', '')
print(conn)
print("connection successfull")
```

Output:

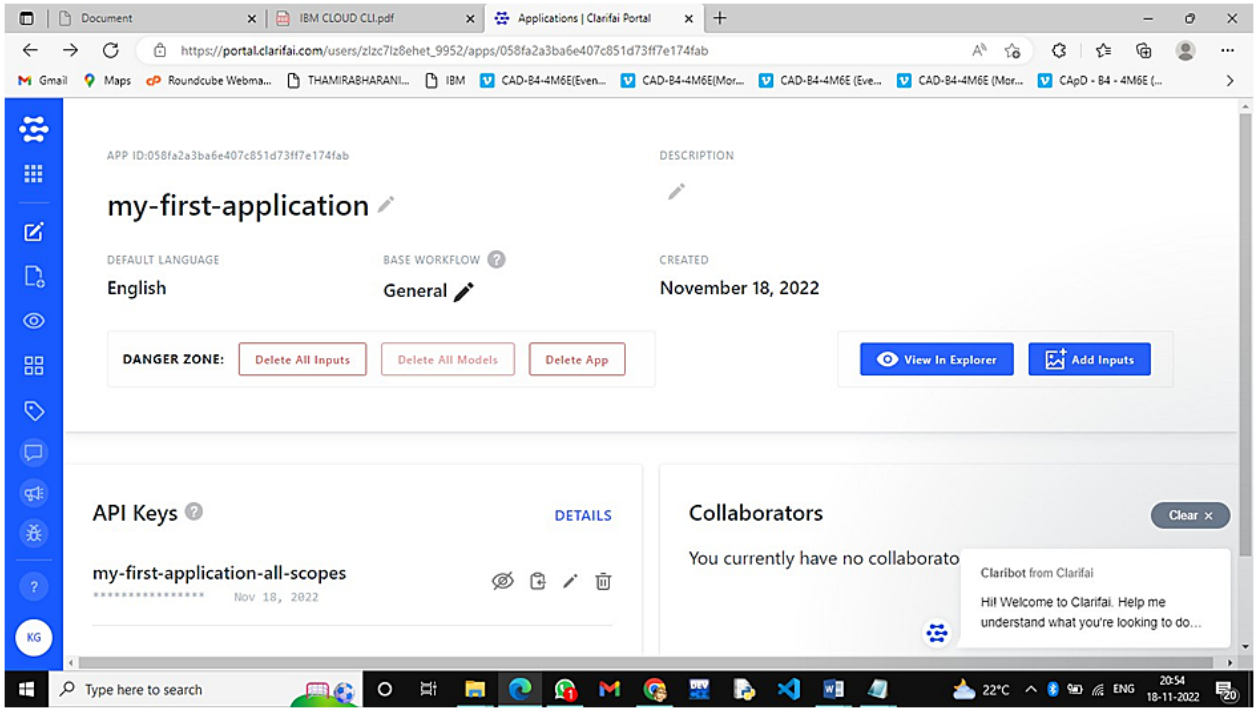


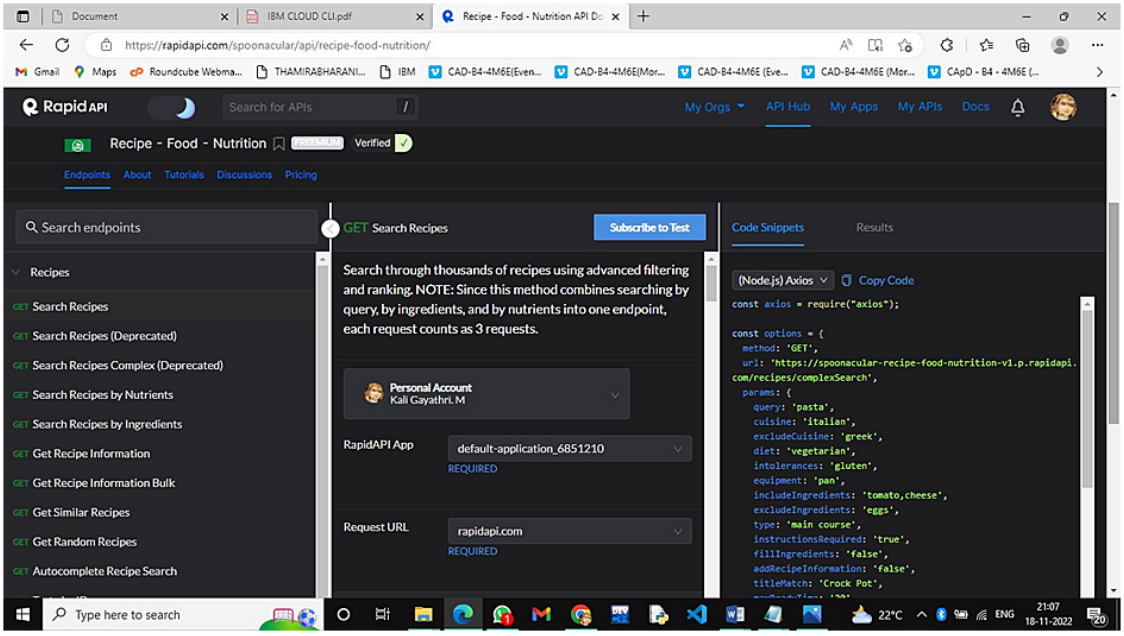
7.3 SPRINT-3

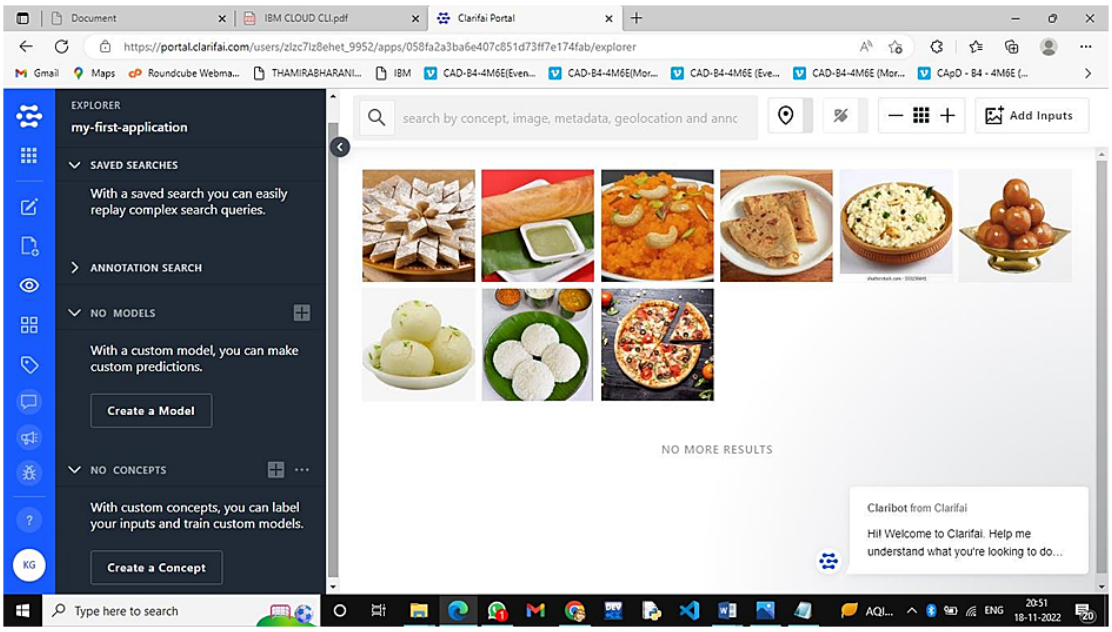
Integration of nutrition API with python code:

```
const axios = require("axios");
const options = {
 method: 'GET',
 url: 'https://spoonacular-recipe-food-nutritionv1.p.rapidapi.com/recipes/1003464/nutritionWidget.json',
 headers: {
 'X-RapidAPI-Key': '76869089cemsh655bcd10f071ae9p127677jsn48ace4b75040',
 'X-RapidAPI-Host': 'spoonacular-recipe-food-nutrition-v1.p.rapidapi.com'
```

```
 }
};
axios.request(options).then(function (response) {
console.log(response.data);
}).catch(function (error) {
console.error(error);
});
```



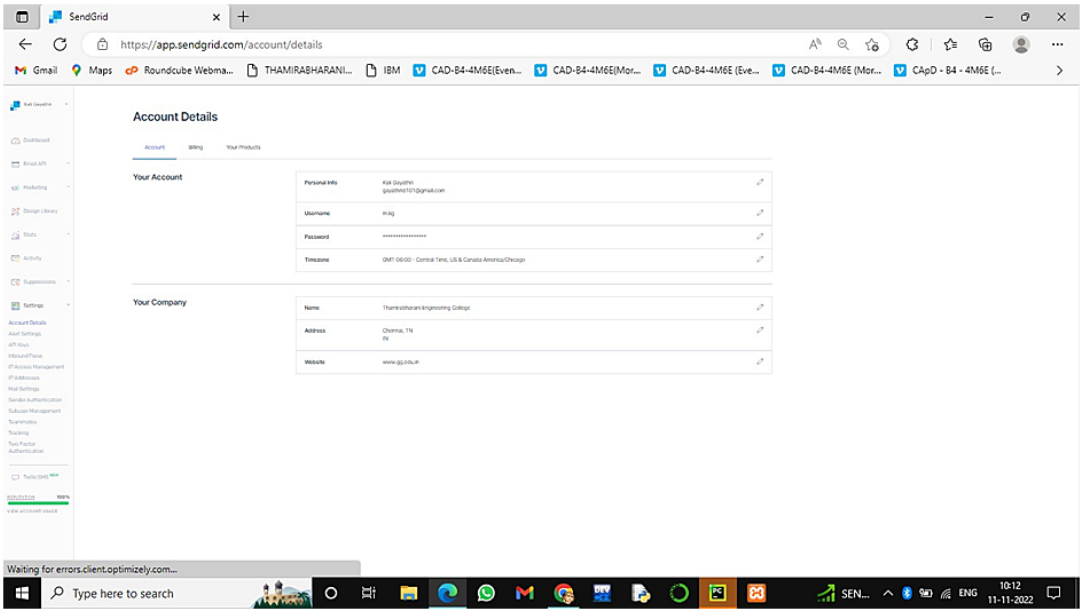

Document
IBM CLOUD CLI.pdf
Recipe - Food - Nutrition API Do
https://rapidapi.com/spoonacular/api/recipe-food-nutrition/
RapidAPI
Search for APIs
My Orgs
API Hub
My Apps
My APIs
Docs
Recipe - Food - Nutrition
Verified
Endpoints
About
Tutorials
Discussions
Pricing
Search endpoints
Recipes
GET Search Recipes
GET Search Recipes (Deprecated)
GET Search Recipes Complex (Deprecated)
GET Search Recipes by Nutrients
GET Search Recipes by Ingredients
GET Get Recipe Information
GET Get Recipe Information Bulk
GET Get Similar Recipes
GET Get Random Recipes
GET Autocomplete Recipe Search
GET Search Recipes
Subscribe to Test
Search through thousands of recipes using advanced filtering and ranking. NOTE: Since this method combines searching by query, by ingredients, and by nutrients into one endpoint, each request counts as 3 requests.
Personal Account
Kali Gayathri. M
RapidAPI App
default-application_6851210
REQUIRED
Request URL
rapidapi.com
REQUIRED
Code Snippets
Results
(Node.js) Axios
Copy Code
const axios = require("axios");
const options = {
method: 'GET',
url: 'https://spoonacular-recipe-food-nutrition-v1.p.rapidapi.com/recipes/complexSearch',
params: {
query: 'pasta',
cuisine: 'italian',
excludeCuisine: 'greek',
diet: 'vegetarian',
intolerances: 'gluten',
equipment: 'pan',
includeIngredients: 'tomato,cheese',
excludeIngredients: 'eggs',
type: 'main course',
instructionsRequired: 'true',
fillIngredients: 'false',
addRecipeInformation: 'false',
titleMatch: 'Crock Pot',
Type here to search
22°C
ENG
21:07
18-11-2022

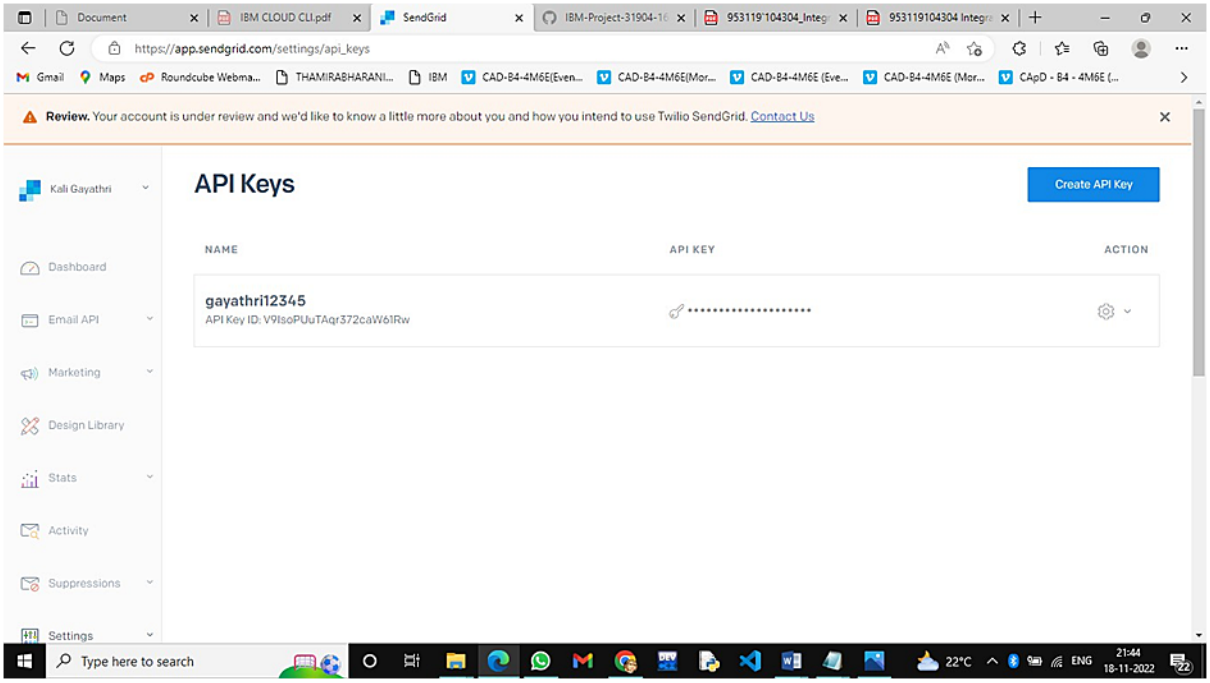

Document
IBM CLOUD CLI.pdf
Clarifai Portal
https://portal.clarifai.com/users/zlzc7lz8ehet_9952/apps/058fa2a3ba6e407c851d73ff7e174fab/explorer
EXPLORER
my-first-application
SAVED SEARCHES
With a saved search you can easily replay complex search queries.
ANNOTATION SEARCH
NO MODELS
With a custom model, you can make custom predictions.
Create a Model
NO CONCEPTS
With custom concepts, you can label your inputs and train custom models.
Create a Concept
search by concept, image, metadata, geolocation and annc
Add Inputs
NO MORE RESULTS
Claribot from Clarifai
Hi! Welcome to Clarifai. Help me understand what you're looking to do...
Type here to search
ENG
20:51
18-11-2022

7.4 SPRINT-4

Sendgrid Integration with python:

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

import sendgrid
from sendgrid.helpers.mail import Mail, Email, To, Content


SENDGRID_API_KEY =
"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" # sendgrid

conn = ibm_db.connect( "DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURITY=SSL;SSLS
erverCertificate=DigiCertGlobalRootCA.crt;UID=wqq31800;PWD=tOlYo6K1IKAc0XhU",
", ")


print(conn)


app = Flask(   name    )


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




# sendgrid
def send_mail(email):
sg = sendgrid.SendGridAPIClient(SENDGRID_API_KEY) from_email = Email("Ram@gmail.com") #
Change to your verified sender
to_email = To(email) # Change to your recipient
subject = "Nutrition is a basic human need and a prerequisite for healthy life" content =
Content("text/plain",
"Thank you for creating an account on our platform. Now you can utilise our platform " "to maintain a
healthier life.")

mail = Mail(from_email, to_email, subject, content)
```





8.CONCLUSION

Nutrition is a key component in the treatment plan for individuals with pressure ulcer, diabetic ulcers, or chronic wounds. Early identification of under nutrition and the correction of nutritional deficits promote healing and improve the patient's quality of life. The use of a nutritional screening tool highlights those at risk of nutritional deficiency. Age-appropriate protein and energy needs should be the minimum provided, and nutritional supplements or enteric feeding should be considered if minimum goal is not achived.

9.APPENDIX

Source Code

- Sprint-1
- Sprint-2
- Sprint-3
- Sprint-4

GitHub & Project Demo Link

https://github.com/IBM-EPBL/IBM-Project-51290-1660977444

https://screenapp.io/#/shared/680ced85-d7c9-4836-ac75-d74292d4dc77