PROJECT REPORT

Nutrition Assistant Application

submitted by

Team ID : PNT2022TMID34980

Suriya nishok M -962819205032

Kowshick shalom T -962819205015

Nantha kumar M -962819205021

Hasan afwaaz S -962819205010

TABLE OF CONTENT

1. INTRODUCTION

1.1 Project Overview

Due to being ignorant of healthy food habits, obesity rates, diabetes, sleep apnea, fatty lever, etc. are increasing at an alarming speed which is reflective of the risks to people's health. According to CBS News large study, poor diet is associated with 1 in 5 deaths worldwide which is equivalent to 11 million deaths a year that makes unhealthy eating habits responsible for more deaths than tobacco and high blood pressure. People need to control their daily nutritional intake by eating healthier foods which is the most basic method to avoid these risks. However, although food packaging comes with nutrition labels, it's still not very convenient for people to refer packaged food due to the use of various preservatives which are not good for health.

Therefore we developed a web-based nutrient dashboard system which can analyze real-time images of a meal for nutritional content which can be very handy and improves the dietary habits, and thus helps in maintaining a healthy lifestyle.

By taking the supplied food image, this web application attempts to identify the food and show us the food qualities like ingredients and nutritional values for precise food recognition to determine the nutritional value of the recognized items. Our solution uses a food detection model and food APIs.

1.2 Purpose

Our purpose is that every food vendor and clients will be aware of the meal's quality and the nutrients that are present in the food in an instant. Previously before starting a good diet clients had to study about the nutritional components of each of the food material and calculate the overall nutritional supplements present in an individual food before consuming it. That is too much time consuming and false knowledge in this matter can lead to having unbalanced diet which can be rather cost effective.

But with this application clients can easily get to now about the nutritional components that is present in the food just by using our web application and by clicking a capture of image of the food or upload the food's picture from the galary in the module and the module will give you the nutrients that is present in that food as output.

2. LITERATURE SURVEY

The various methodolies that are all used are discussed as follows:

Prabha et al., 1 proposed a Smart Log system that performs automated nutrition monitoring and meal prediction. The smart sensor board consisting of Piezo Electric sensors is used for nutrition quantification. The nutrient data acquisition is done using Optical Character Recognition and by linking open source Application Program Interfaces (APIs) through barcodes. The meal prediction is done by collecting nutritional value of the leftover food along with the user's feedback on the type of food that is desired. The SR8 database available through the US Department of Agriculture website is also analysed using their API which provides a food report of associated nutrient values for a particular food item and a nutrient report which gives an extensive list of food and their nutrient values for a selected amount of nutrients. The results have been analysed by creating an AttributeRelation File Format which inputs the Waikato Environment for Knowledge Analysis (WEKA) tool which builds a better prediction model and is observed that the Bayesian classifiers provided better results. The open dataset consisted of multiple redundant logs and psychological monitoring mechanisms have not been incorporated which in turn leads to lack of accurate prediction.

B. Manal et al., 2 proposed a machine learning based pipelined approach for predicting the calories from food images. The system takes an image of the food item and passes it through Mathworks Image Processing which extracts the raw features and improves the quality. The image is passed through a compression phase which helps to reduce the number of features using the Principal Component Analysis (PCA) method and scale the subsequent learning phases. The food type classification is done by inputting the compressed image to the classifier. The food size prediction is done by passing the compressed image to a regressor. Calories are predicted by passing the compressed image and predicted values to another regressor. This is based on supervised learning model. The dataset is limited to a small category and the image cannot be diversified.

C. Kohila et al., 3 proposed a calorific value prediction mechanism using image processing and machine learning. The image of the food is transmitted through a

mobile device and it initially undergoes segmentation with Fuzzy C-means Clustering Segmentation which fixes the cluster centre based on the group data unlike the K-means Clustering which can be erroneous if the cluster centre 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.

D. Sangita et al., 4] proposed a nutritional status investigation system based on machine learning. A logical regression model was considered for the major four variables namely BMI (Body Mass Index), HAZ (Height for Age Z score, also known as stunting), WHZ (Weight for Age Z score, also known as

wasting) and WAZ (Weight for Age Z score, also known as underweight) individually. The study predicts the nutrition state of the child in two phases. Phase I pre-processes the dataset using SMOTE Resampling method. Feature extraction is done using machine learning techniques with Entropy based Gain Ratio concept. Phase II uses Nominal logistic regression using iterative reweight least squares algorithm which predicts the characteristic features. The dataset is based on a very specific geographical area and over a particular period of time and it also considers only the basic features.

E. Oscar et al., 5 proposed a menu-match: restaurant-specific food logging from images. An image recognition framework based on the bag of visual words approach which extracts the base features from the images and then encoded with localityconstrained linear coding (LLC). The extracted features are pooled using max-pooling in a rotation-invariant pooling scheme. A regression based method estimates the calories and along with feature representation mapped the feature space to calories using Support Vector Regression. The approach is limited for discrete serving sizes and custom menu and is also dependent on the GPSS of food consumption. The system lacks user customization and requires cost-sensitive learning to directly minimize calorie estimation errors during the training.

F. Kiran et al., 6 proposed a method for measuring the calories and nutrition from food images using machine learning techniques. The images got from the mobile device are pre-processed followed by the segmentation step to extract the colour 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 mixed food images have not been considered.

G. Prabha et al., 7 proposed a Smart Log system based on deep learning for automated nutrition monitoring system in IoT. The proposed system is composed of a smart sensor board along with an application. Automatic Nutrition Quantification is the first step which collects the quantity of nutrient values along with the timestamp and transmits it to the cloud. The nutrient data acquisition is done using Optical Character Recognition in which the on-phone camera captures the FDA-mandated Nutritional Facts Label and the other method is by linking open source Application Program Interfaces (APIs) through barcodes. The meal prediction is done by collecting nutritional value of the leftover food along with the user's feedback on the purpose of the meal. Food Classification is done through an algorithm which is built on a Bayesian or Belief Network (BN) with constraint based method. A multi-layer perceptron neural network is constructed using Stochastic Gradient Descent andhill-climbing search is used for food classification. The system is designed only for the children and the dataset contained multiple redundant logs.

H. Parisa et al., 9 proposed a method for food calorie measurement using deep learning neural network. The system uses food images and the features are extracted using Graph Cut Segmentation. A pre-trained model file is generated with the help of Convolutional Neural Network (CNN). It is loaded into the application and tested to perform image recognition process. The stochastic gradient descent algorithm is used to get an improvement in the cost. The calorie measurement is done by two

approaches namely finger based calorie measurement and calorie measurement using distance estimation. The system lacked analysing mixed food portion and only limited samples were considered.

I. Yanchao et al., 10 proposed a calorie estimation model based on deep learning approach which increases the detection accuracy and reduces the error of volume estimation. The image acquisition is done by obtaining the food image using a smartphone. The object detection is done by using Faster Region based Convolutional Neural Networks (Faster R-CNN), which includes Region Proposal Network (RPN) and an Object Detection Network. GrabCut, an image based segmentation algorithm which

depends on optimization by graph cuts is used for image segmentation. Volume Estimation requires calculation of side and top views scaling factor using equations based on the shape types. Finally, calorie estimation is obtained by using the volume and density value of the food mapped using ECUSTFD dataset. The limitation in this system is that it has 20% and more mean error for the discrepancy in the estimated result when compared with the actual values.

J. Bruno et al., 11 proposed an automated food recognition system that provides dietary intervention based on computer vision and machine learning. The unique feature of the system relies in the realization of real-time energy balance with the help of network simulation. Food recognition deals with the challenges in image segmentation, classification and the volume-nutrient estimation. Food segmentation is done on the food image using Otsu's segmentation. Feature extraction is done using Local Binary Patterns (LBP), colour, texture and Scale-Invariant Feature Transform (SFIT). Classification is done using SVM, Bag of Features and K Nearest Neighbours. Weight estimation is done by mapping the nutritional facts from the USDA dataset and then followed by metabolic network modelling. The intervention module provides insightful information regarding the nutrients, eating patterns and alerts for the same are provided. The performance has to be improved and deep learning techniques with wearables could have been incorporated.

K. Chang et al., 12proposed a food image recognition for computer aided dietary assessment based on deep learning techniques. The proposed approach utilized two real-world food image datasets namely (UEC – 256 and Food – 101). The food image recognition is done by a new Convolutional Neural Network (CNN) method based on supervised learning algorithms. The CNN consisted of 3 convolutional layers, 2 sub sampling layers and a fully connected layer. The model was trained for a nonstop period of 2 to 3 days using a server with Nvidia K40 GPU. After the training, the model classified the image in less than a minute. The proposed system lacked real-world data and the accuracy of the measurements have to be improve.

L. Yoke et al., 13 proposed an approach to measure the food nutrition using pocket-size Near Infrared Sensor. A SCiO scanner was used to acquire the nutritional facts using the NIR spectra. The reflected spectrum is uploaded to the SCiO web server through the SCiO application on a smart phone. Based on the varied absorption level of the wavelength, the distinction between the foods is made. The system used 14 off-the-shelf drinks as the input data. The model is trained by correlating features of NIR spectra with energy and carbohydrate contents of different drinks. Finally, the

accuracy of prediction model is assessed. The Partial Least Square (PLS) regression and Support Vector Regression (SVR) were used and the analysis proved that SVR with RBF (Radial Basis Function) performed better.

M. Bahman et al., [32] proposed a smart nutrition monitoring system using heterogeneous Internet of Things Platform. The proposed architecture is based on emerging Fog computing concepts in which pre-processing and lightweight analytics are done by data collection points after which the data is sent to the cloud. The data collection points (i.e., kiosks capture the food image from different angles and pass them to the cloud server which generates a 3D model of the food. The Smart Nutrition Monitoring Engine has 4 components namely Collection Management, Data Analytics, Data store and Visualization. The Collection Management is responsible to upload the data to the cloud server and also for information storage. The Data Analytics component does the statistical analysis and machine learning activities in the system. The data store stores the raw data as well as the processed data. The visualisation module displays charts portraying the food consumption. As a future work, the author aims to adopt new sensors to the system and also a fault-tolerant system thereby increasing the accuracy.

N. Niloofar et al., [33] proposed a mobile platform for nutrition monitoring from spoken data called Speech-to-NutrientInformation (S2NI). The system monitors the food intake using natural language processing through the speech-to-text mobile application. The speech recognition module inputs a stream of sampled speech data and a sequence of words is the required output which is done by sampling, end pointing, feature extraction and template matching. The Natural Language Processing (NLP) module is based on machine learning techniques and this work used Parts of Speech (PoS) tagging which is a supervised machine learning problem. The string matching module uses two methods namely exact matching and approximate matching to match the detected words with the nutrition database. The USDA national database is used. The performance of the various models is analysed at each level. The system achieves 80% accuracy in computing the calorie estimation.

2.1PROBLEM STATEMENT

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.

2.2 References

1.Prabha Sundaravadivel, Kavya Kesavan, Lokeshwar Kesavan, Saraju P. Mohanty, Elias Kougianos and Madhavi Ganapathiraju, “Smart-log: An automated, predictive nutrition monitoring system for infants through the IoT”, IEEE International Conference on Consumer Electronics (ICCE), 2018.

2.Manal Chokr and Shady Elbassuoni, “Calories Prediction from Food Images”, Innovative Applications of Artificial Intelligence Twenty - Ninth IAAI Conference, 2017.

3.Kohila R and Meenakumari R, “Predicting calorific value for mixed food using image processing”, In Proceedings of the 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), 2017, pp. 1–4.

4.Sangita Khare, Kavyashree S, Gupta, D and Jyotishi A, “Investigation of nutritional status of children based on machine learning techniques using Indian demographic and health survey data”, In: 7th International Conference on Advances in Computing and Communications, Vol. 115, 2017, pp. 338–349.

5.Oscar Beijbom , Neel Joshi , Dan Morris , Scott Saponas and Siddharth Khullar, “Menu-Match: Restaurant-Specific Food Logging from Images”, Proceedings of the 2015 IEEE Winter Conference on Applications of Computer Vision, 2015, pp. 844-851, [doi>10.1109/WACV.2015.117].

6.Kiran Abhore and Dawande N A, “Measuring Calories and Nutrition from Food Image”, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 6, 2016.

7.Prabha Sundaravadivel, Kavya Kesavan, Lokeshwar Kesavan, Saraju P. Mohanty and Elias Kougianos, “Smart-Log: A DeepLearning based Automated Nutrition Monitoring System in the IoT”, IEEE Transactions on Consumer Electronics, 2018.

8.Parisa Pouladzadeh, Kuhad P, Peddi S V B, Yassine A and Shirmohammadi S, “Food calorie measurement using deep learning neural network”, Proc. IEEE Int. Instrum. Meas. Technol. Conf. Proc., 2016, pp. 1-6. [28]Yanchao Liang and Jianhua Li, “Deep Learning-Based Food Calorie Estimation Method in Dietary Assessment”, 2018, arXiv preprint arXiv: 1706.04062.

9 .Bruno Vieira Resende e Silva, Milad Ghiasi Rad, Juan Cui, Megan McCabe and Kaiyue Pan, “A Mobile-Based Diet Monitoring System for Obesity Management”, Journal of Health and Medical Informatics, Vol. 9, Issue 2, 2018, pp. 307. Doi: 10.4172/2157-7420.1000307.

10.Liu C, Cao Y, Luo Y, Chen G, Vokkarane V and Ma Y, “Deepfood: Deep learning-based food image recognition for computeraided dietary assessment”, International Conference on Smart Homes and Health Telematics, 2016, pp. 37-48. doi: https://doi.org/10.1007/978-3-319-39601-9_4.

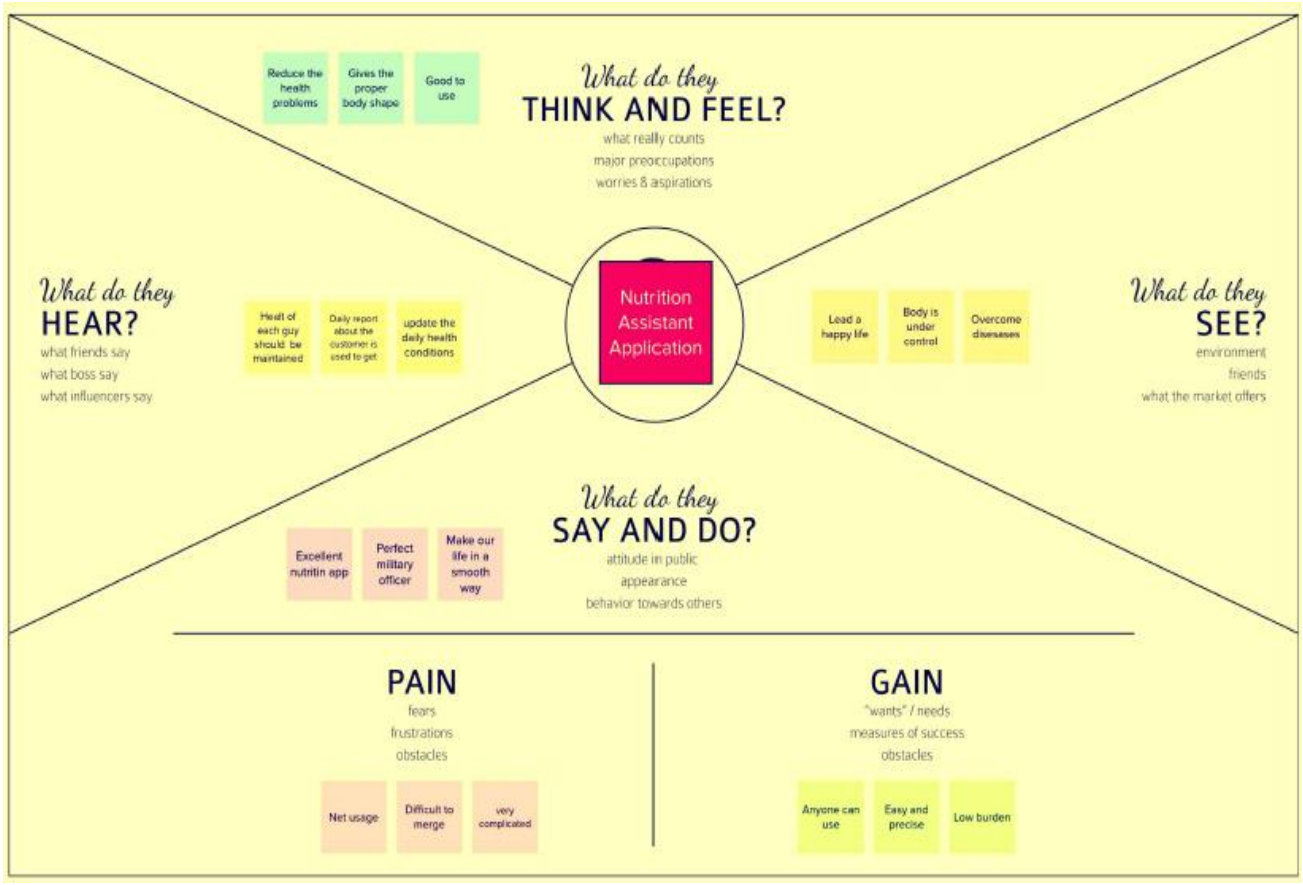
11.Thong Y J, Nguyen T, Zhang Q, Karunanithi M and Yu L, “Predicting food nutrition facts using pocket-size near-infrared sensor”, In2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2017, pp. 742- 745.

12.Javadi B, Calheiros R N, Matawie K M, Ginige A and Cook A, “Smart nutrition monitoring system using heterogeneous Internet of Things platform”, International Conference on Internet and Distributed Computing Systems, 2017, pp. 63-74.

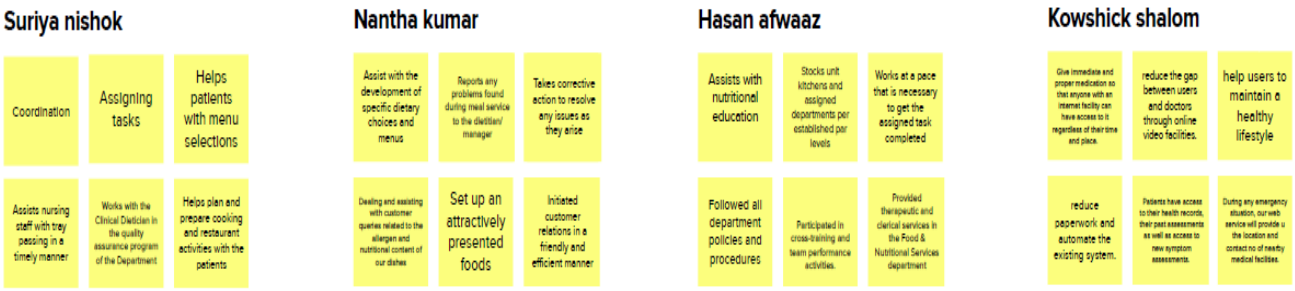
13.Niloofar Hezarjaribi, Cody A, Reynolds, Drew T. Miller, Naomi Chaytor and Hassan Ghasemzadeh, “S2NI: A mobile platform for nutrition monitoring from spoken data”, 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2016, pp. 199.

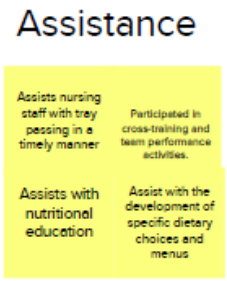
3. IDEATION & PROPOSED SOLUTION

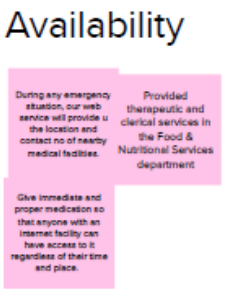
3.1 Empathy Map Canvas

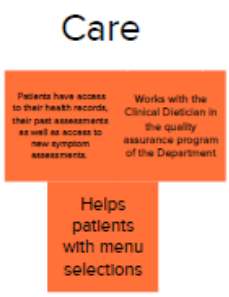


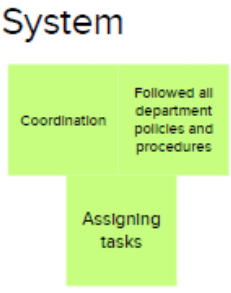
3.2 Ideation & Brainstorming

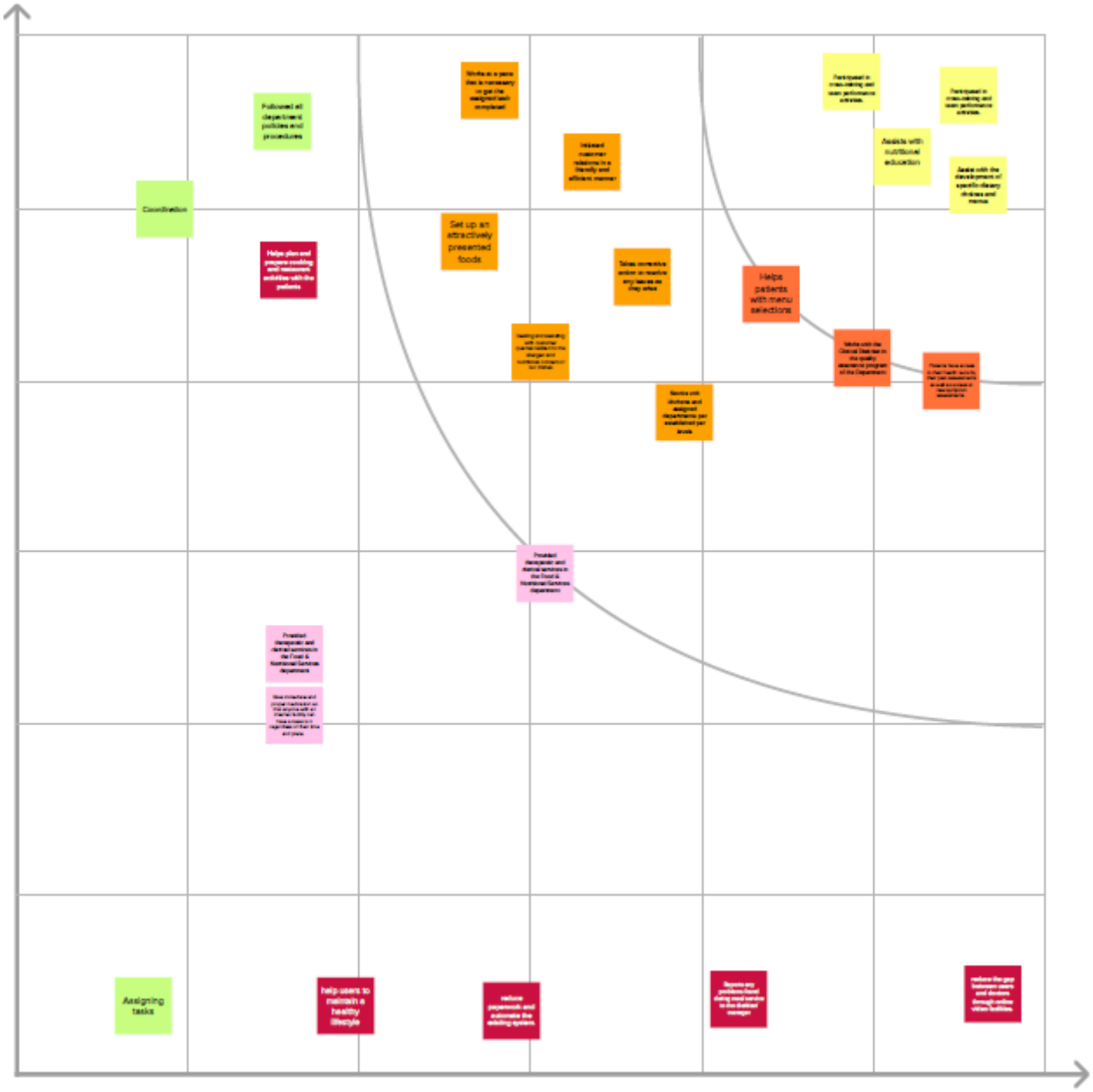












3.3 Proposed Solution

SL. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	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.	Idea / Solution description	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.
3.	Novelty / Uniqueness	A new solution in which the information about the exact nutritional information of inputted image.
4.	Social Impact / Customer Satisfaction	The nutrition assistant impacted society greatly because of the society people is live in unhealthy food routine so this is give the good nutritional information about the food.
5.	Business Model (Revenue Model)	This application is very usefull to the people who are interested or responsibility in their health instead of the gym or trainer investment cost is reduced .There must me the minimal cost of using this web based application.
6.	Scalability of the Solution	This solution is scalable enough to fit the body in healthy way to get nutritional information..

3.4 Problem Solution fit

Nutrition assistant will help to assist	Network connection, less maintenance.	Api will calaculate
Find the exact nutritional information. .	Due to ignorance of healthy food obesity is very high.	Calculate usage and benefits of the api of nutrition .
When the user give the input image and api is calculate the appropriate nutritional content If image is not clear then the application is give exact nutritional content	Using Api/container registery calculate the nutrition about the food image	There will be a assistant to way of the correct nutritional content

4. REQUIREMENT ANALYSIS

4.1 Functional requirements

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	New users need to enter their required details and register in it.
FR-2	User Confirmation	Confirmation via Email confirmation and being able to change the password using forgot password option via Email.
FR-3	Log in	Only registered users are able to log into the page whichis verified in the database and allowed to do the furtheractions.
FR-4	Diet goal setting	Users need to provide their biometrics, mention the chronic diseases and set the short term and long term diet goals to get the food suggestions based on their needs.
FR-5	Uploading images of diets and dietary supplements	Users need to upload the image of the diets, based on that food suggestions are notified. If the diet is not found , then users need to provide extra information about the diet like the ingredients added to it.
FR-6	Display of personalized dietary feedback and nutritional report	The uploaded diet image is processed in the database and corresponding food suggestions are notified to the users as per the request.

4.2 Non-Functional requirements

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Users can effectively and easily use the system. User interfaces are simple and easily accessed by the users.
NFR-2	Maintainability	Maintainability defines the time required for a solution or its component to be fixed, changed to increase performance or other qualities, or adapted to a changing environment. The mean time to restore the system following a system failure must not be greater than 10 minutes.
NFR-3	Reliability	Reliability specifies how likely the system or its element would run without a failure for a given period of time under predefined conditions. The system must perform without failure in 95 percent of use cases during a month.
NFR-4	Performance	Performance defines how fast a software system or a particular piece of it responds to certain users' actions under a certain workload. The landing page supporting 5000 users per hour must provide 6 second or less response time in a chrome desktop browser.
NFR-5	Availability	Availability describes how likely the system is accessible to a user at a given point in time. While it can be expressed as an expected percentage of successful requests, it may also define it as a percentage of time the system is accessible for operation during some time period. The web dashboard is available at any time whenever the user is accessed.
NFR-6	Scalability	Scalability assesses the highest workloads under whichthe system will still meet the performance requirements. There are two ways to enable the system scale as the workloads get higher: horizontal and

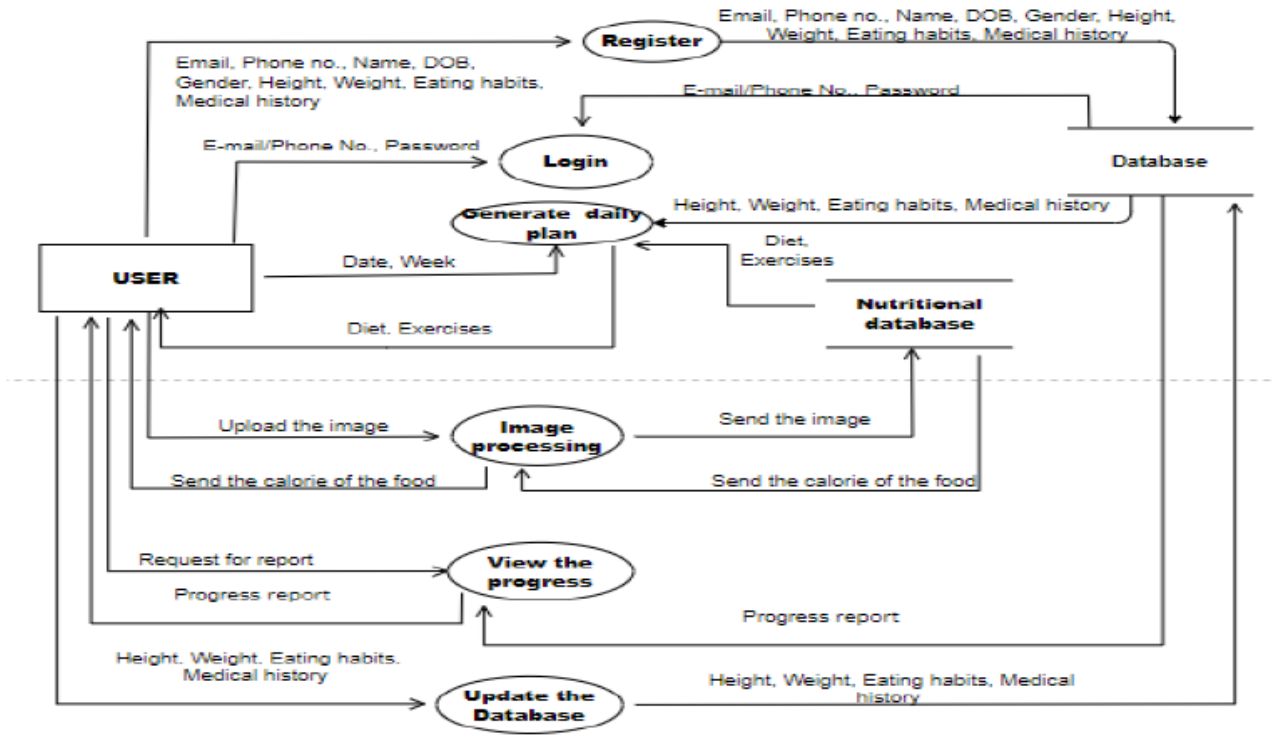
		vertical scaling. The system must be scalable enough to support 1,000,000 visits at the same time while maintaining optimal performance.

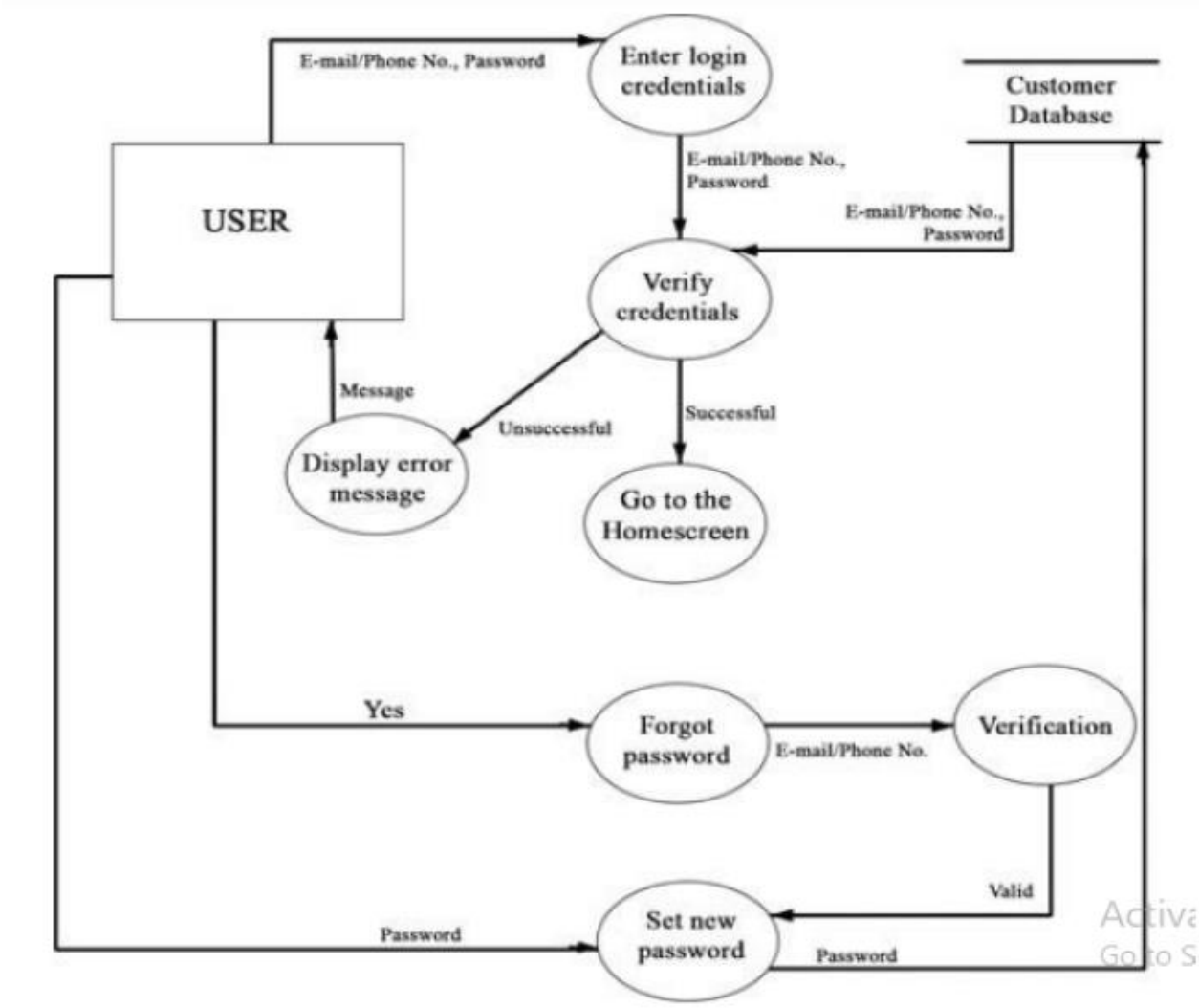
5. PROJECT DESIGN

5.1 Data Flow Diagrams

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

Level -1DFD





Level-2 DFD (LOGIN)

5.2 Solution & Technical Architecture

Solution Architecture:

➤ User interacts with the UI (User Interface) to upload the image as input.

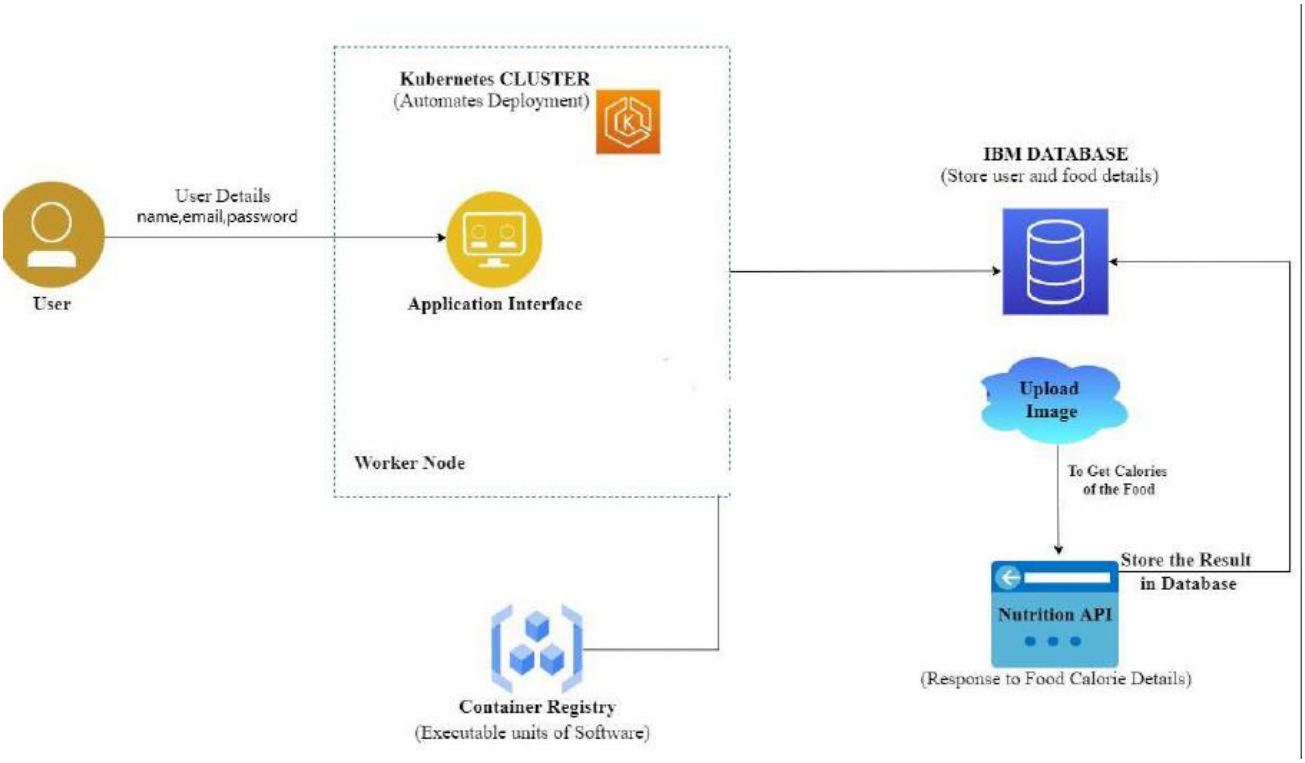
➤ Depending on the different gesture inputs different operations are applied to the input image.

➤ The image can be resized, blur, flip and rectangle.

➤ Once model analyses the gesture, the prediction with operation applied on image is showcased on the UI.

➤ Better execution in accurate results, sensitivity, system architecture design and flexibility of the software.

Solution Architecture Diagram:



Technology Architecture Diagram:

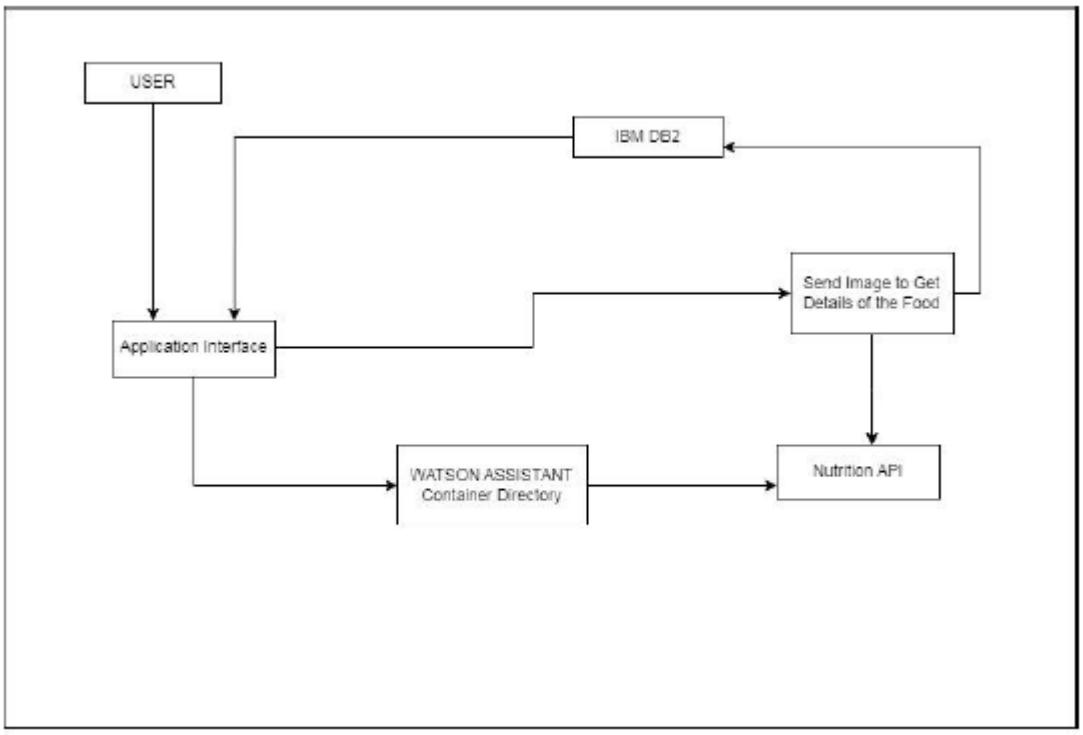

USER
IBM DB2
Send Image to Get Details of the Food
Application Interface
WATSON ASSISTANT Container Directory
Nutrition API

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1	User Interface	User interacts with application such as Web UI, Mobile App.	HTML, CSS
2	Application Logic-1	Logic for a process in the application of nutrition assistant	Python
3	Application Logic-2	Logic for a process in the application for the container.	IBM Watson STT service
4	Application Logic-3	Logic for a process in the application for creating ChatBot	IBM Watson Assistant
5	Database	Data Type, Configurations etc.	MySQL
6	File Storage	File storage stores the file related to the nutrition	IBM Block Storage or Other Storage Service or Local Filesystem
7	External API-1	Purpose of External API used in the application which interacts with file Storage	IBM Weather API, etc.
8	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration:XAMPP Cloud Server Configuration : IBM Cloud	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1	Open-Source Frameworks	IBM Cloud Foundry is the Open-Source Framework used to build, deploy and run of the application.	Cloud Foundry
2	Security Implementations	Web Application Firewall.	Encryptions, OWASP etc.
3	Scalable Architecture	It is highly scalable and always meets up with the user's demand.	No Technology used
4	Availability	Available on the web and can be used anywhere and anytime.	Python
5	Performance	The overall Performance will be good and user Friendly.	The software requirement as Python, Docker, Flask

5.3 User stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a 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
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm and enter my details	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail/Phone no.	I can access my account and enter biometrics	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can enter my biometrics and access my dashboard	High	Sprint-1
	Dashboard	USN-6	As a user, I can generate daily plan	I can receive daily daily diet plan and exercise	Medium	Sprint-2

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
		USN-7	As a user, I can upload images of the food	I can receive the calorie content of the food	High	Sprint-1
Customer (Web user)		USN-8	As a user, I can request for my progress report	I can able to view my progress	Medium	Sprint-2
Administrator		USN-9	As an administrator, I can manage the details of the user	I can give access to the user	High	Sprint-1
		USN-10	As an administrator,I can manage the calorie content of the food in database	I can modify the calorie content of the food	High	Sprint

6.PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

User Story Number	User Story / Task	Story Points	Priority	Team Members
USN-1	The system should provide diets for the user.	10	High	Nantha kumar Suriya nishok
USN-2	The information of each user should be realistic and it should be stored in the database.	10	High	Nantha kumar Suriya nishok
USN-3	The camera of the user's should be in high quality.	10	High	Nantha kumar Suriya nishok
USN-4	The photage of the image should be identified by API	10	High	Nantha kumar Suriya nishok
USN-5	After detecting the food by API , the nutrients in the food is analyzed.	10	High	Kowshick shalom Hasan afwaaz
USN-6	The exact quality of the food is determined.	20	High	Kowshick shalom Hasan afwaaz
USN-7	Sending Information	5	High	Kowshick shalom Hasan afwaaz
USN-8	Whether the capture food is good for that user or not.	15	High	Kowshick shalom Hasan afwaaz

6.2 Sprint Delivery Schedule

Project Tracker, Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint 1	20	8 days	27-10-2022	3-11-2022	20	3-11-2022
Sprint 2	20	8 days	5-11-2022	12-11-2022	20	12-11-2022
Sprint 3	30	8 days	14-11-2022	21-11-2022	30	21-11-2022
Sprint 4	20	8 days	23-11-2022	30-11-2022	20	30-11-2022

Velocity:

Let's calculate the team's average velocity (AV) per iteration unit (story points per day).

AV = Velocity / Sprint Duration

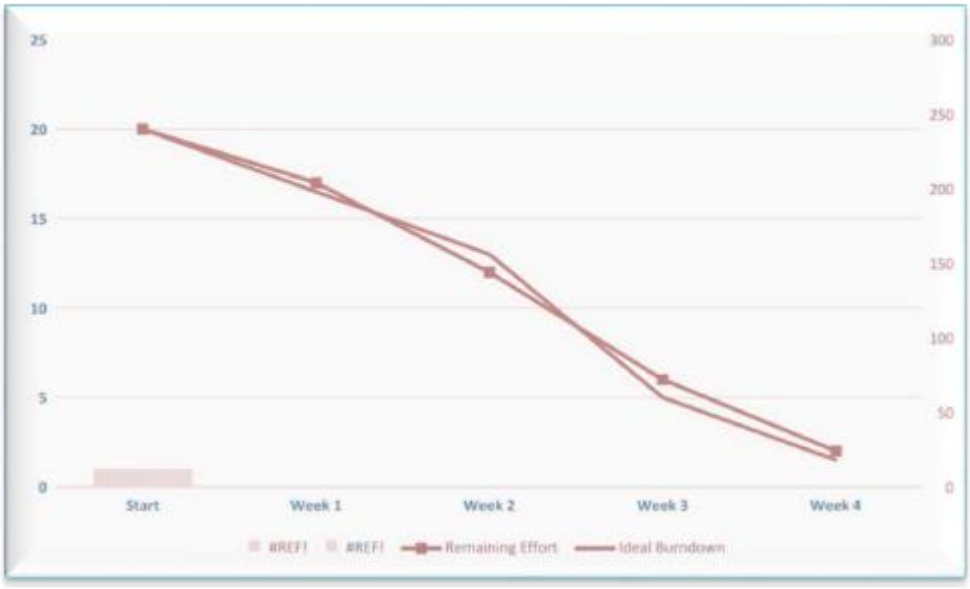
= 20 / 8

= 2.5

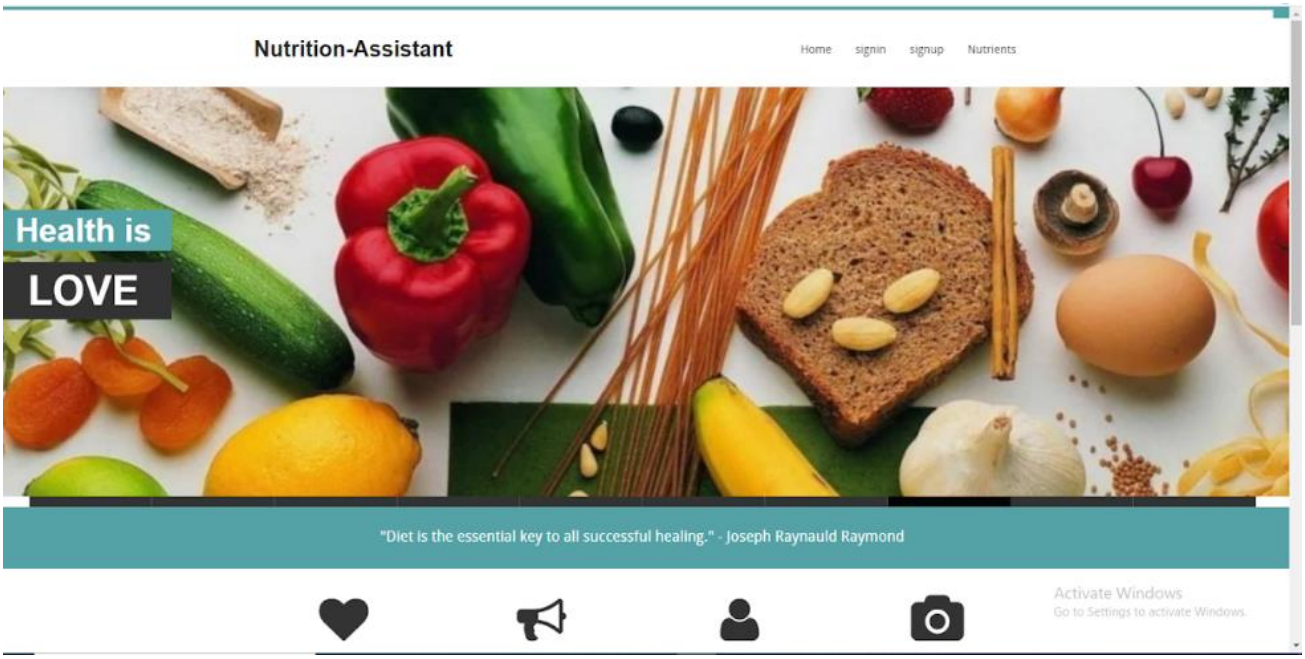
AV = 30 / 8

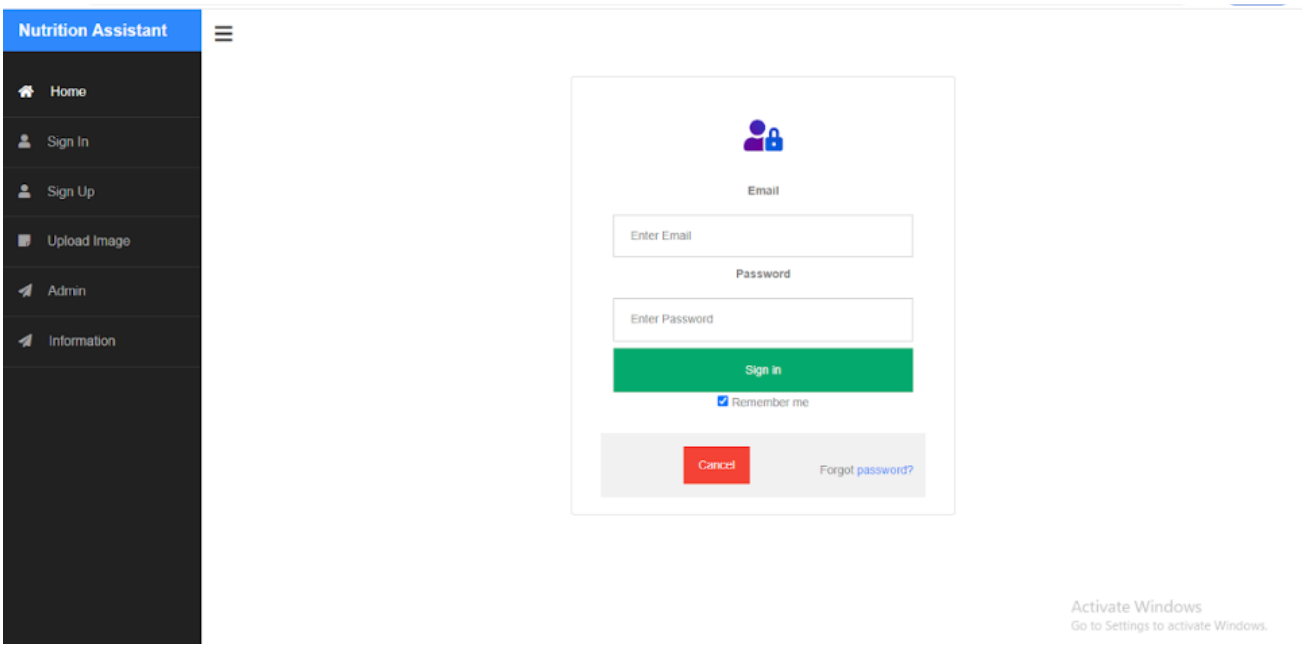
= 3.75

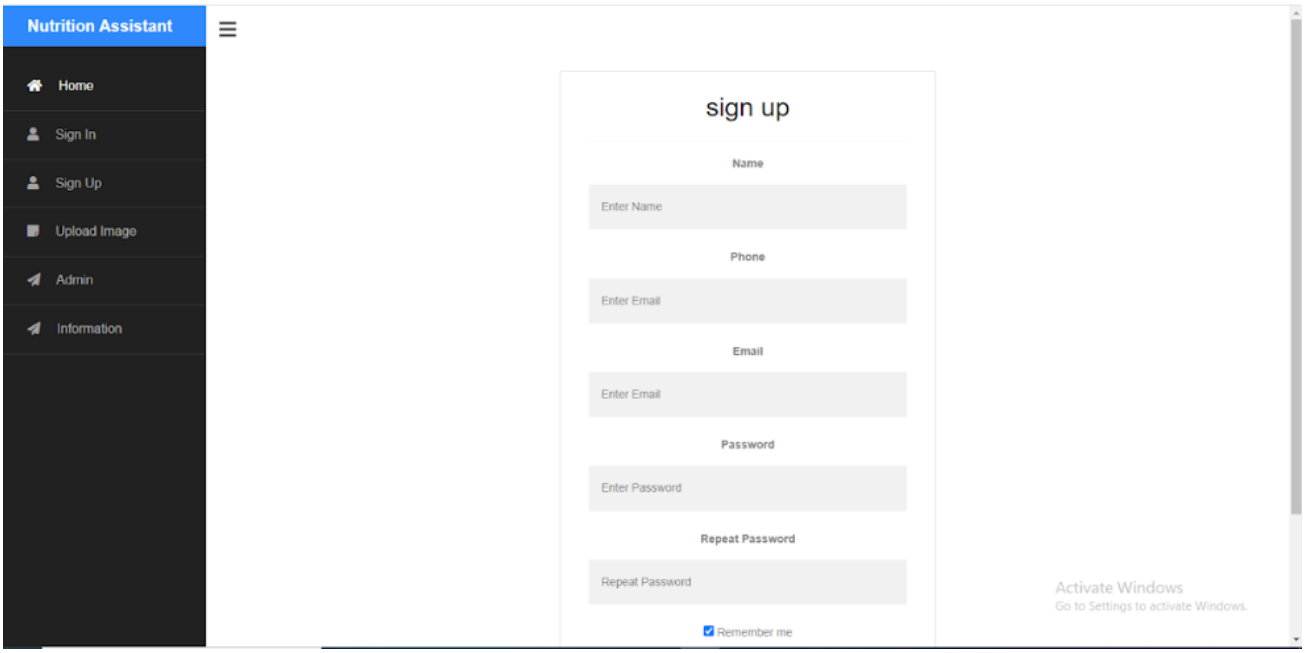
X-axis - Days
Y-axis - Story Points

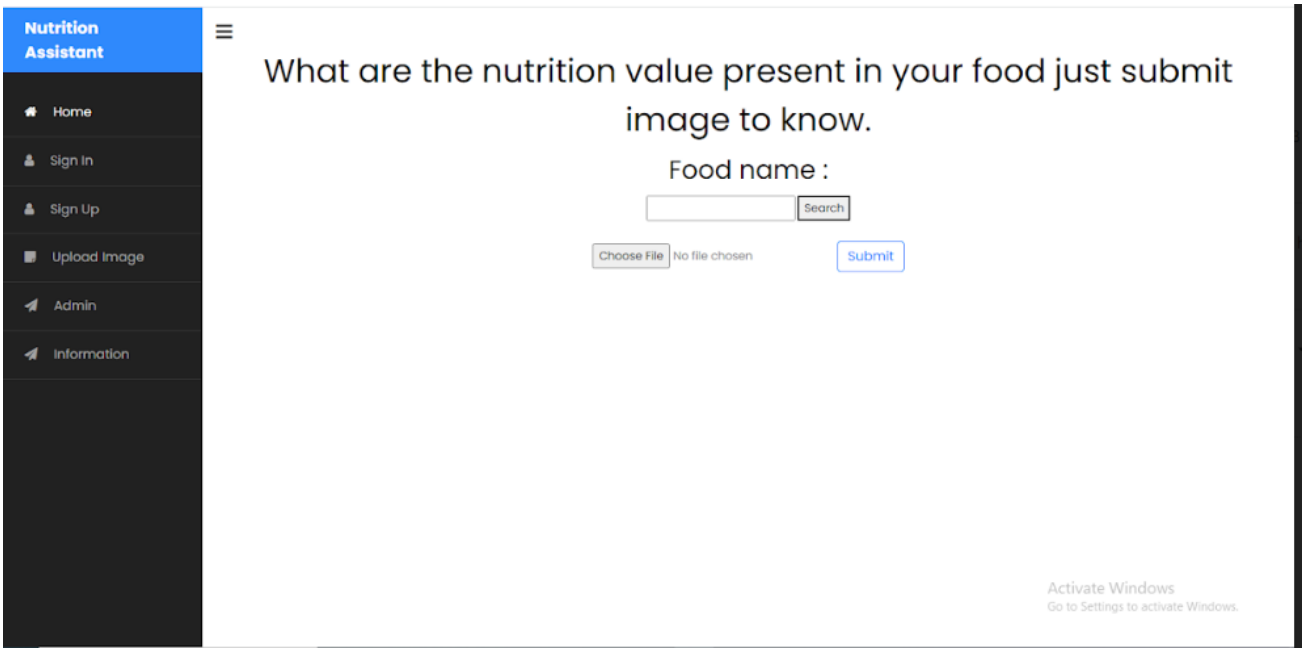


7.RESULTS


Nutrition-Assistant
Home
signin
signup
Nutrients
Health is
LOVE
"Diet is the essential key to all successful healing." - Joseph Raynauld Raymond
Activate Windows
Go to Settings to activate Windows.


Nutrition Assistant
Home
Sign In
Sign Up
Upload Image
Admin
Information
Email
Enter Email
Password
Enter Password
Sign in
Remember me
Cancel
Forgot password?
Activate Windows
Go to Settings to activate Windows.


Nutrition Assistant
Home
Sign In
Sign Up
Upload Image
Admin
Information
sign up
Name
Enter Name
Phone
Enter Email
Email
Enter Email
Password
Enter Password
Repeat Password
Repeat Password
Remember me
Activate Windows
Go to Settings to activate Windows.


Nutrition Assistant
Home
Sign In
Sign Up
Upload Image
Admin
Information
What are the nutrition value present in your food just submit image to know.
Food name :
Search
Choose File
No file chosen
Submit
Activate Windows
Go to Settings to activate Windows.


Dashboard
Welcome Gowtham,
Log Out
Control panal
Delete All History
History
Mon, 14 Nov 2022 08:33:51 View Delete
Nutrition Facts
Calories 355.63cal
Daily Value
Total Fat 23.29g
Saturated Fat 10.44g
Polyunsaturated Fat 3.85g
Monounsaturated Fat 9.0g
Cholesterol 111.36mg
Sodium 272.14mg
Potassium 228.13mg
Sugar 0.1g
Protein 27.17g
Carbohydrates 0.73g
Vitamin A 452.47IU
Vitamin C 1.74mg
Vitamin D 0.4µg
Vitamin B5 1.11mg
Calcium 224.03mg
Clear Image
load Image
No file chosen
Upload

8.ADVANTAGES & DISADVANTAGES

Advantages:-

1. This device is user friendly.
2. Its only required the image of the food .
3. To know the different type of nutrients present in food .
4. And also know that how much composition of the nutrients are present.
5. Output of the screen is easy understandable.

Disadvantages:-

1. This device is not able to predict the multiple image as input.
2. The internet is only necessary for opening the web application.(After converting the mobile app internet is not necessary for opening .)

9.CONCLUSION

During this assignment we were able to take a closer look at our daily eating habits. From here we can now improve our application so that we can help clients to eat and grow healthier as a person and athlete.
I can truly say that I learnt a lot from this assignment. I was able to point out changes I needed to make and how to move forward and make it work in my life.
I am now more educated on the powers of food and how they control our body. I hope that people will use our application to lead a healthy life. When choosing the right foods for yourself you should be focused on what is the healthiest choice. Eating healthy and feeling good go hand in hand, eating better will automatically give you a better functioning body. Eating healthy means eating a variety of foods that meet your daily requirements.
We would recommend our application to anyone who is interested in eating healthy. Not only is it easy to use, but it is a great way to evaluate what you are eating and understand the vitamins and minerals that you need.

10.FUTURE SCOPE

The device will also assist you determine the quantity and degree of flavour of the food. Future goals include increasing the accuracy of our machine learning model and expanding the types of food categories so that we can better meet user needs. We are also increasing dataset of categories of images and nutrition to better efficiency to get output. Our research essentially identifies simply the nutrients, but our team members raise the bar for our project so that we also understand the ingredients and the amount of nutrients in a particular cuisine.

11.APPENDIX

13.1 Source Code

```
 from crypt import methods
from re import RegexFlag
from clarifai_grpc.grpc.api import service_pb2, resources_pb2
from clarifai_grpc.grpc.api.status import status_code_pb2
from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel
from clarifai_grpc.grpc.api import service_pb2_grpc
from flask import Flask, json, jsonify, render_template, url_for, request, redirect,
session, flash
from turtle import st
from markupsafe import escape
import bcrypt
import requests
import dns.resolver
from bson import ObjectId
import sendgrid
from sendgrid.helpers.mail import Mail, Email, To, Content
import ibm_db
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-8fc0-
d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=31321;SECURITY=SSL;
SSLServerCertificate=DigiCertGlobalRootCA
(3).crt;UID=vbz29727;PWD=zgvT4AKDdg9I90uZ",'','')
session = {}

dns.resolver.default_resolver = dns.resolver.Resolver(configure=False)
dns.resolver.default_resolver.nameservers = ['8.8.8.8']


class MyEncoder(json.JSONEncoder):

    def default(self, obj):
        if isinstance(obj, ObjectId):
            return str(obj)
        return super(MyEncoder, self).default(obj)

app = Flask(__name__)



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

@app.route('/signup')
def signup():
  return render_template('sign up.html')

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

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

    name = request.form['name']
    email = request.form['email']
    phone = request.form['phone']
    password = request.form['password']
    retypepassword = request.form['retypepassword']


    sql = "SELECT * FROM students WHERE name =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,name)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)

    if account:
      return render_template('admin.html', msg="You are already a member, please
login using your details")
    else:
      insert_sql = "INSERT INTO students VALUES (?,?,?,?,?)"
      prep_stmt = ibm_db.prepare(conn, insert_sql)
      ibm_db.bind_param(prep_stmt, 1, name)
      ibm_db.bind_param(prep_stmt, 2, email)
      ibm_db.bind_param(prep_stmt, 3, phone)
      ibm_db.bind_param(prep_stmt, 4, password)
      ibm_db.bind_param(prep_stmt, 5, retypepassword)
      ibm_db.execute(prep_stmt)

    return render_template('home.html', msg="Student Data saved successfuly..")



@app.route('/signin',methods=['GET','POST'])
def signin():
    message = ''
    if request.method == 'POST':
        # get the data from the form
        email = request.form['email']
        password = request.form['password']
        # if nothing is entered in the form
        if not email or not password:
            message = 'Please fill all the fields!'
            return render_template('sign in.html', message=message)
        # check if the username and password are valid
        sql = "SELECT * FROM students WHERE email = '" + email + "' AND password =
'" + password + "'"
        stmt = ibm_db.exec_immediate(conn, sql)
        result = ibm_db.fetch_assoc(stmt)
        # print("result", result)
        if result:
            # message = 'You have successfully logged in!'
            session['email'] = result['EMAIL']
            session['password'] = result['PASSWORD']
            return redirect('/window')
        else:
            message = 'The email or password is incorrect!'
```

```
    return render_template('sign in.html', message=message)


@app.route('/window')
def window():
    # Calorie Ninja
    url = "https://calorieninjas.p.rapidapi.com/v1/nutrition"

    headers = {
        "X-RapidAPI-Key": "aa95b88b45mshe4394a422ce8c48p13a698jsn9d8eb019e144",
        "X-RapidAPI-Host": "calorieninjas.p.rapidapi.com"
    }

    if request.method == 'POST':
        foodname = request.form['foodname']

        querystring = {"query": foodname}
        response = requests.request(
            "GET", url, headers=headers, params=querystring)

        return response.text

    return render_template('window.html')

@app.route('/window', methods=['POST', 'GET'])
def clarifai():
    if request.files.get('image'):
        image = request.files['image'].stream.read()
        stub = service_pb2_grpc.V2Stub(ClarifaiChannel.get_grpc_channel())

        CLARIFAI_API_KEY = "04fe7a95051541789ba44a08eaa5722e"
        APPLICATION_ID = "Nutrition_Assistant1"

        # Authenticate

        # image = '/home/bala/Desktop/Images/foodsample.jpeg'

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

        with open(image, "rb") as f:
            file_bytes = f.read()

        request = service_pb2.PostModelOutputsRequest(
            model_id='9504135848be0dd2c39bdab0002f78e9',
            inputs=[
                resources_pb2.Input(
                    data=resources_pb2.Data(
                        image=resources_pb2.Image(
                            base64=file_bytes
                        )
                    )
                )
            ])
        response = stub.PostModelOutputs(request, metadata=metadata)

        if response.status.code != status_code_pb2.SUCCESS:
```

```
            raise Exception("Request failed, status code: " +
                            str(response.status.code))

        for concept in response.outputs[0].data.concepts:
            print('%12s: %.2f' % (concept.name, concept.value))

    return render_template('window.html')

@app.route('/admin')
def admin():
  students = []
  sql = "SELECT * FROM Students"
  stmt = ibm_db.exec_immediate(conn, sql)
  dictionary = ibm_db.fetch_both(stmt)
  while dictionary != False:
    # print ("The Name is : ",  dictionary)
    students.append(dictionary)
    dictionary = ibm_db.fetch_both(stmt)

  if students:
    return render_template("admin.html", students = students)

@app.route('/delete/<name>')
def delete(name):
  sql = f"SELECT * FROM Students WHERE name='{escape(name)}'"
  print(sql)
  stmt = ibm_db.exec_immediate(conn, sql)
  student = ibm_db.fetch_row(stmt)
  print ("The Name is : ",  student)
  if student:
    sql = f"DELETE FROM Students WHERE name='{escape(name)}'"
    print(sql)
    stmt = ibm_db.exec_immediate(conn, sql)

    students = []
    sql = "SELECT * FROM Students"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
      students.append(dictionary)
      dictionary = ibm_db.fetch_both(stmt)
    if students:
      return render_template("admin.html", students = students, msg="Delete
successfully")



  # # while student != False:
  # #   print ("The Name is : ",  student)

  # print(student)
  return "success..."

# @app.route('/posts/edit/<int:id>', methods=['GET', 'POST'])
# def edit(id):
```

```
#     post = BlogPost.query.get_or_404(id)

#     if request.method == 'POST':
#         post.title = request.form['title']
#         post.author = request.form['author']
#         post.content = request.form['content']
#         db.session.commit()
#         return redirect('/posts')
#     else:
#         return render_template('edit.html', post=post)
# sendgrid
def send_mail(email):
    sg = sendgrid.SendGridAPIClient(SENDGRID_API_KEY)
    from_email = Email(os.environ.get('EMAIL_ID'))
    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()
    sg.client.mail.send.post(request_body=mail_json)


def custom_send_mail(email, data):
    sg = sendgrid.SendGridAPIClient(SENDGRID_API_KEY)
    from_email = Email(os.environ.get('EMAIL_ID'))
    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",
                      f"'{data}'")
    mail = Mail(from_email, to_email, subject, content)

    # Get a JSON-ready representation of the Mail object
    mail_json = mail.get()
    sg.client.mail.send.post(request_body=mail_json)


def generateOTP():
    digits = os.environ.get('DIGIT')
    OTP = ""
    for i in range(6):
        OTP += digits[math.floor(random.random() * 10)]
    return OTP


def get_history():
    history = []
    sql = f"SELECT * FROM PERSON WHERE email = '{session['email']}'"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary:
        history.append(dictionary)
```

```
        dictionary = ibm_db.fetch_both(stmt)
    return history


def get_history_person(email):
    history = []
    sql = f"SELECT * FROM PERSON WHERE email = '{email}'"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary:
        history.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)
    return history


def get_history_person_time(time):
    history = []
    sql = f"SELECT * FROM PERSON WHERE time = '{time}'"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary:
        history.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)
    return history


def get_user():
    user = []
    sql = f"SELECT * FROM USER"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary:
        user.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)
    return user


backend = default_backend()


@app.route('/dashboard', methods=['GET', 'POST'])
def upload_file():
    history = []
    # sql = "SELECT * FROM Students"
    sql = f"SELECT * FROM PERSON WHERE email = '{session['email']}'"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary:
        history.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)
    if request.method == 'POST':
        # check if the post request has the file part
        if 'logout' in request.form:
            session["loggedIn"] = None
            session['name'] = None
            session['email'] = None
```

```
            return render_template('index.html', error="Successfully created")
        if 'file' not in request.files:
            # flash('No file part')
            return redirect(request.url)
        file = request.files['file']
        # If the user does not select a file, the browser submits an
        # empty file without a filename.

        if file.filename == '':
            return render_template('dashboard.html', msg="File not found",
history=history)
        baseimage = file.read()
        if file and allowed_file(file.filename):
            requests = service_pb2.PostModelOutputsRequest(
                model_id="food-item-recognition",
                user_app_id=resources_pb2.UserAppIDSet(app_id=YOUR_APPLICATION_ID),
                inputs=[
                    resources_pb2.Input(
                        data=resources_pb2.Data(image=resources_pb2.Image(base64=bas
eimage))
                    )
                ],
            )
            response = stub.PostModelOutputs(requests, metadata=metadata)

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

            calcium = 0
            vitaminb5 = 0
            protein = 0
            vitamind = 0
            vitamina = 0
            vitaminb2 = 0
            carbohydrates = 0
            fiber = 0
            fat = 0
            sodium = 0
            vitaminc = 0
            calories = 0
            vitaminb1 = 0
            folicacid = 0
            sugar = 0
            vitamink = 0
            cholesterol = 0
            potassium = 0
            monounsaturatedfat = 0
            polyunsaturatedfat = 0
            saturatedfat = 0
            totalfat = 0
            calciumu = 'g'
            vitaminb5u = 'g'
            proteinu = 'g'
            vitamindu = 'g'
            vitaminau = 'g'
```

```
            carbohydratesu = 'g'
            sodiumu = 'g'
            vitamincu = 'g'
            caloriesu = 'cal'
            sugaru = 'g'
            cholesterolu = 'g'
            potassiumu = 'g'
            monounsaturatedfatu = 'g'
            polyunsaturatedfatu = 'g'
            saturatedfatu = 'g'

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

                        for z in range(0, len(nutri)):
                            temp = nutri[z]
                            if temp['name'] == 'Calcium':
                                calcium += round(temp['amount'], 2)
                                calciumu = temp['unit']
                            elif temp['name'] == 'Vitamin B5':
                                vitaminb5 += round(temp['amount'], 2)
                                vitaminb5u = temp['unit']
                            elif temp['name'] == 'Protein':
                                protein += round(temp['amount'], 2)
                                proteinu = temp['unit']
                            elif temp['name'] == 'Vitamin D':
                                vitamind += round(temp['amount'], 2)
                                vitamindu = temp['unit']
                            elif temp['name'] == 'Vitamin A':
                                vitamina += round(temp['amount'], 2)
                                vitaminau = temp['unit']
                            elif temp['name'] == 'Vitamin B2':
                                vitaminb2 += round(temp['amount'], 2)
                                vitaminb2u = temp['unit']
                            elif temp['name'] == 'Carbohydrates':
                                carbohydrates += round(temp['amount'], 2)
                                carbohydratesu = temp['unit']
                            elif temp['name'] == 'Fiber':
                                fiber += round(temp['amount'], 2)
                                fiberu = temp['unit']
                            elif temp['name'] == 'Vitamin C':
                                vitaminc += round(temp['amount'], 2)
                                vitamincu = temp['unit']
                            elif temp['name'] == 'Calories':
                                calories += round(temp['amount'], 2)
                                caloriesu = 'cal'
                            elif temp['name'] == 'Vitamin B1':
```

```
                                vitaminb1 += round(temp['amount'], 2)
                                vitaminb1u = temp['unit']
                            elif temp['name'] == 'Folic Acid':
                                folicacid += round(temp['amount'], 2)
                                folicacidu = temp['unit']
                            elif temp['name'] == 'Sugar':
                                sugar += round(temp['amount'], 2)
                                sugaru = temp['unit']
                            elif temp['name'] == 'Vitamin K':
                                vitamink += round(temp['amount'], 2)
                                vitaminku = temp['unit']
                            elif temp['name'] == 'Cholesterol':
                                cholesterol += round(temp['amount'], 2)
                                cholesterolu = temp['unit']
                            elif temp['name'] == 'Mono Unsaturated Fat':
                                monounsaturatedfat += round(temp['amount'], 2)
                                monounsaturatedfatu = temp['unit']
                            elif temp['name'] == 'Poly Unsaturated Fat':
                                polyunsaturatedfat += round(temp['amount'], 2)
                                polyunsaturatedfatu = temp['unit']
                            elif temp['name'] == 'Saturated Fat':
                                saturatedfat += round(temp['amount'], 2)
                                saturatedfatu = temp['unit']
                            elif temp['name'] == 'Fat':
                                fat += round(temp['amount'], 2)
                                fatu = temp['unit']
                            elif temp['name'] == 'Sodium':
                                sodium += round(temp['amount'], 2)
                                sodiumu = temp['unit']
                            elif temp['name'] == 'Potassium':
                                potassium += round(temp['amount'], 2)
                                potassiumu = temp['unit']
                            else:
                                pass

            totalfat += saturatedfat + polyunsaturatedfat + monounsaturatedfat
            data = [round(calories, 2), round(totalfat, 2), round(saturatedfat, 2),
round(polyunsaturatedfat, 2),
                    round(monounsaturatedfat, 2), round(cholesterol, 2),
round(sodium, 2), round(potassium, 2),
                    round(sugar, 2), round(protein, 2), round(carbohydrates, 2),
round(vitamina, 2), round(vitaminc, 2),
                    round(vitamind, 2), round(vitaminb5, 2), round(calcium, 2)]

            unit = [caloriesu, "g", saturatedfatu, polyunsaturatedfatu,
monounsaturatedfatu, cholesterolu, sodiumu,
                    potassiumu, sugaru, proteinu, carbohydratesu, vitaminau,
vitamincu, vitamindu, vitaminb5u, calciumu]

            to_string =
"{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{}".format(data[0], data[1], data[2],
data[3],
                                                                                dat
a[4],
                                                                                dat
a[5], data[6], data[7], data[8],
```

```
                                                                                dat
a[9],
                                                                                dat
a[10], data[11], data[12], data[13],
                                                                                dat
a[14], data[15])

            to_unit =
"{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{}".format(unit[0], unit[1], unit[2],
unit[3],
                                                                              unit[
4], unit[5], unit[6], unit[7],
                                                                              unit[
8], unit[9], unit[10], unit[11],
                                                                              unit[
12], unit[13], unit[14], unit[15])

            current_time = datetime.now(timezone("Asia/Kolkata")).strftime('%Y-%m-%d
%H:%M:%S')

            complete_value = to_string + ',' + to_unit
            val_arr = complete_value.split(',')

            to_units = "Calories : {}{}" \
                       "Total Fat : {}{}" \
                       "Saturated Fat : {}{}" \
                       "Polyunsaturated Fat : {}{}" \
                       "Monounsaturated Fat : {}{}" \
                       "Cholesterol : {}{}" \
                       "Sodium : {}{}" \
                       "Potassium : {}{}" \
                       "Sugar : {}{}" \
                       "Protein : {}{}" \
                       "Carbohydrates : {}{}" \
                       "Vitamin A : {}{}" \
                       "Vitamin C : {}{}" \
                       "Vitamin D : {}{}" \
                       "Vitamin B5 : {}{}" \
                       "Calcium : {}{}".format(data[0], unit[1], data[1], unit[1],
data[2], unit[2], data[3], unit[3],
                                               data[4], unit[4], data[5], unit[5],
data[6], unit[6], data[7], unit[7],
                                               data[8], unit[8], data[9], unit[9],
data[10], unit[10], data[11],
                                               unit[11], data[12], unit[12],
data[13], unit[13], data[14], unit[14],
                                               data[15], unit[15])

            custom_send_mail(session['email'], to_units)

            try:
                insert_sql = "INSERT INTO PERSON VALUES (?,?,?,?)"
                prep_stmt = ibm_db.prepare(conn, insert_sql)
                ibm_db.bind_param(prep_stmt, 1, session['name'])
                ibm_db.bind_param(prep_stmt, 2, session['email'])
                ibm_db.bind_param(prep_stmt, 3, complete_value)
```

```
                ibm_db.bind_param(prep_stmt, 4, current_time)
                ibm_db.execute(prep_stmt)
                return render_template('dashboard.html', user=session['name'],
email=session['email'], data=val_arr,
                                       history=history)
            except ibm_db.stmt_error:
                print(ibm_db.stmt_error())
                return render_template('dashboard.html', msg='Something wnt wrong',
user=session['name'],
                                       email=session['email'], data=val_arr,
history=history)

        return render_template('dashboard.html', history=history)
    if session['name'] is None:
        return render_template('index.html')
    return render_template('dashboard.html', user=session['name'],
email=session['email'], history=history)
```

11.2 GitHub & Project Demo Link

GitHub Link

https://github.com/IBM-EPBL/IBM-Project-36381-1660294695

Project Demo Video Link

https://drive.google.com/file/d/1g5UZX1unE-yH3YaD7fMtLXBX_UaoQqhO/view?usp=sharing