AI - POWERED NUTRITION ANALYZER FOR FITNESS ENTHUSIASTS

1.INTRODUCTION

Food is essential for human life and has been the concern of many healthcare conventions. Nowadays new dietary assessment and nutrition analysis tools enable more opportunities to help people understand their daily eating habits, exploring nutrition patterns and maintain a healthy diet. Nutritional analysis is the process of determining the nutritional content of food. It is a vital part of analytical chemistry that provides information about the chemical composition, processing, quality control and contamination of food.

1.1 PROJECT OVERVIEW:

The project is to building a model which is used for classifying the fruit depends on the different characteristics like color, shape, texture etc. Here the user can capture the images of different fruits and then the image will be sent the trained model. The model analyses the image and detect the nutrition based on the fruits like (Sugar, FibeR, Protein, Calories, etc.).

1.2 PURPOSE:

Nutritional Analysis detects the exact nutritional value of any given food item. It determines the percentage of macro and micro nutrients present in that food item as well as the presence of inhibitors, toxic chemicals, or any other new component.It is also important in nutrition mapping where a variety of food items are regularly being tested and included in the standardized book of Nutritive Value of Indian Foods by the Indian Council of Medical Research. Presences of inhibitors, toxic chemicals in various foods are tested in food nutrition analysis. Inhibitors like hydrate, palate decrease the bio availability of nutrients, and toxic chemicals like poisoning, Trypsin inhibitors, pathogens, etc. cause mild to severe ailments in the human body.

2 LITERATURE SURVEY:

2.1 Deep Food: Food Image Analysis and Dietary Assessment via Deep Model.

This system will analyze the nutritional ingredients based on the recognition results and generate a dietary assessment report by calculating the amount of calories, fat, carbohydrate and protein.

ALGORITHMS USED:

- Region-based Convolution Neural Network
- Convolution Neural Network
- Non-maximum suppression
- Bounding Box Regression

CHALLENGES:

Three main challenges in real food image recognition and analysis are addressed as follows:

1. Region of Interest

2. The Delay of Food Recognition

3. Insufficient Information of Nutrition Content for dietary assessment

2.2 A New Deep Learning-based Food Recognition System for Dietary Assessment on An Edge Computing Service Infrastructure

A design of food recognition system employing edge computing-based service computing paradigm to overcome some inherent problems of traditional mobile cloud computing paradigm, such as unacceptable system latency and low battery life of mobile devices.

ALGORITHMS USED:

- K-means clustering algorithms
- Convolution Neural Network
- Bounding Box Regression
- Deep learning

CHALLENGES:

Using this simple cropping-based approach will not work well if the food is scattered on different parts of the image.

2.3 Precision Nutrient Management Using Artificial Intelligence Based on Digital Data Collection Framework

Nutritional intake is fundamental to human growth and health, and the intake of different types of nutrients and micro nutrients can affect health. The content of the diet affects the occurrence of disease, with the incidence of many diseases increasing each year while the age group at which they occur is gradually decreasing.

ALGORITHM USED:

- Okapi BM25
-
- TF-IDF
- Liechtenstein
- Jacquard
- Synonyms

CHALLENGES:

This model has very little error and can significantly improve the efficiency of the analysis.

2.4 Calculating Nutrition Facts with Computer Vision

People are becoming more health-conscious than before. However, there is alack of knowledge about different fitness and wellness aspects of food. Thus, come up with Foodify.ai-a deep learning-based application that detects food from the image and provides information of food such as protein, vitamins, calories, minerals, carbs, etc.

ALGORITHM USED:

- Deep learning
- Machine learning
- Image processing

CHALLENGES:

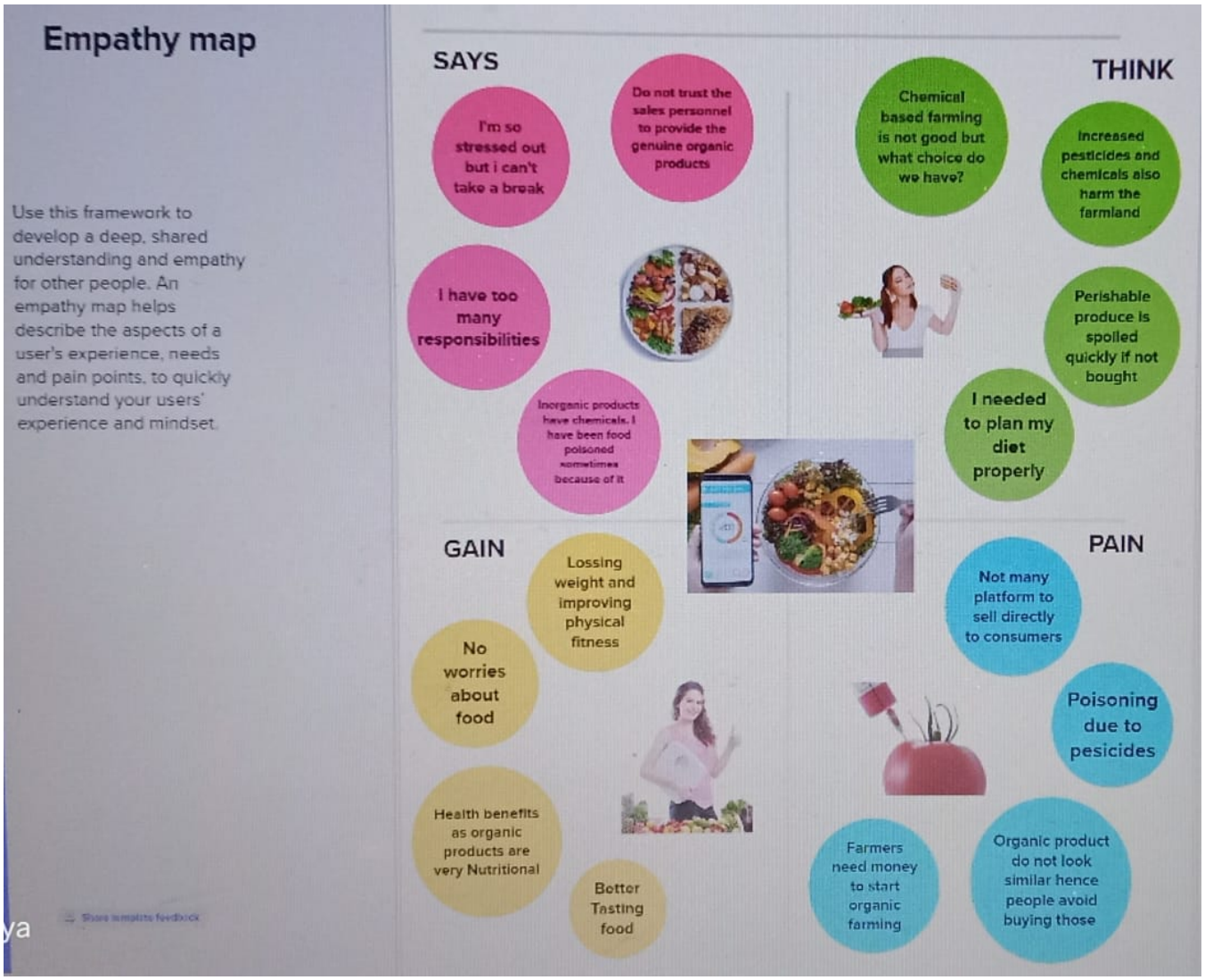
1. This is to collect images to create a huge dataset.

This is related to training the deep learning model. It is an extremely computationally expensive and time-consuming task to train the model again and again. This can be solved by using cloud-based services.

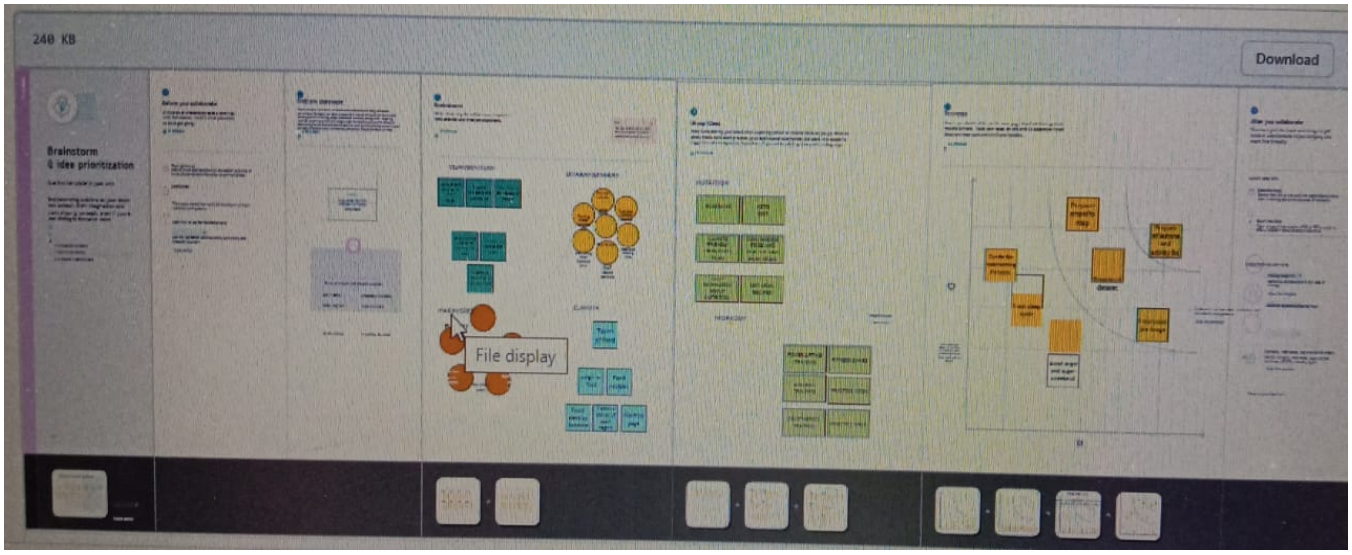
3 IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

Al-Powered Nutrition Analyzer For Fitness Enthusiasts



3.2 IDEATION AND BRAINSTORMING

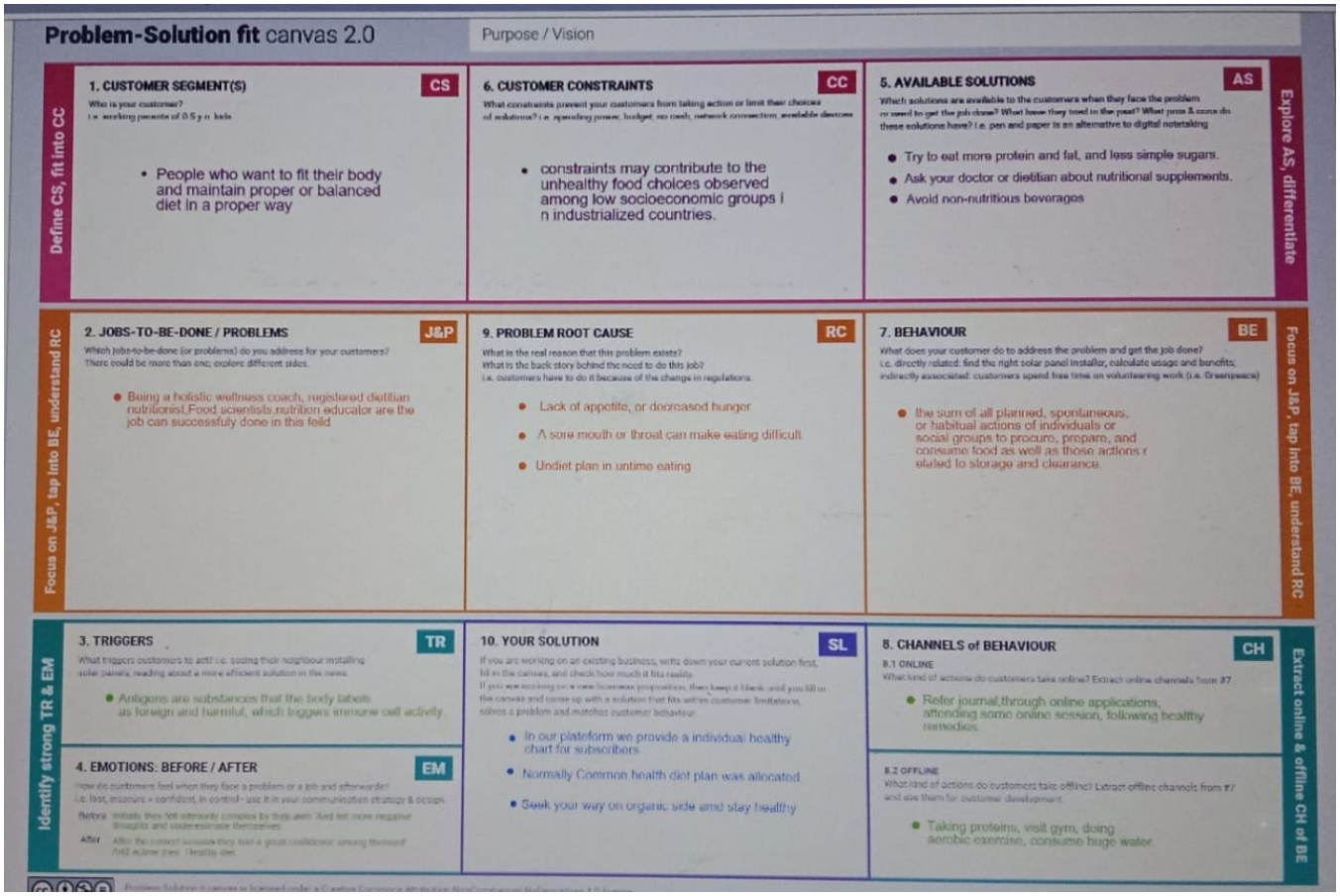


3.3 PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To create an AI Based Nutrition Analyzer forDieticians and Fitness enthusiasts so that food can be classified based on the nutrients present
2.	Idea / Solution description	To develop a CNN model to classify and detect types of food and nutrients present in it to maintain a proper diet.
3.	Novelty / Uniqueness	A lightweight model of CNN is to be used along with image processing to identify food and nutrients present in it along with the calories that are present in the food. So that food identification takes less time and is easier to do.
4.	Social Impact / Customer Satisfaction	The proposed solution will enable people around the world to be able to keep track of their calorie intake and be able to set goals for themselves that they'd want to hit

		within a certain period of time.
5.	Business Model (Revenue Model)	By using this system, the users can predict and analyze the picture of the fruits and foods. In which it results to the visualizing the description of the foods taken as input.
6.	Scalability of the Solution	The lightweight algorithm can be used with different kinds of mobile apps like exercise apps etc.

3.4 SOLUTION FIT TEMPLATE:



4 REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Login	Login through Google Login through Email
FR-4	Choose package	Selection of desired package
FR-5	Generate the daily plan	Daily plans will be generated by dietitian
FR-6	Manage progress report	Gathering information from database and generating report
FR-7	Query	The user can ask for changes in plan

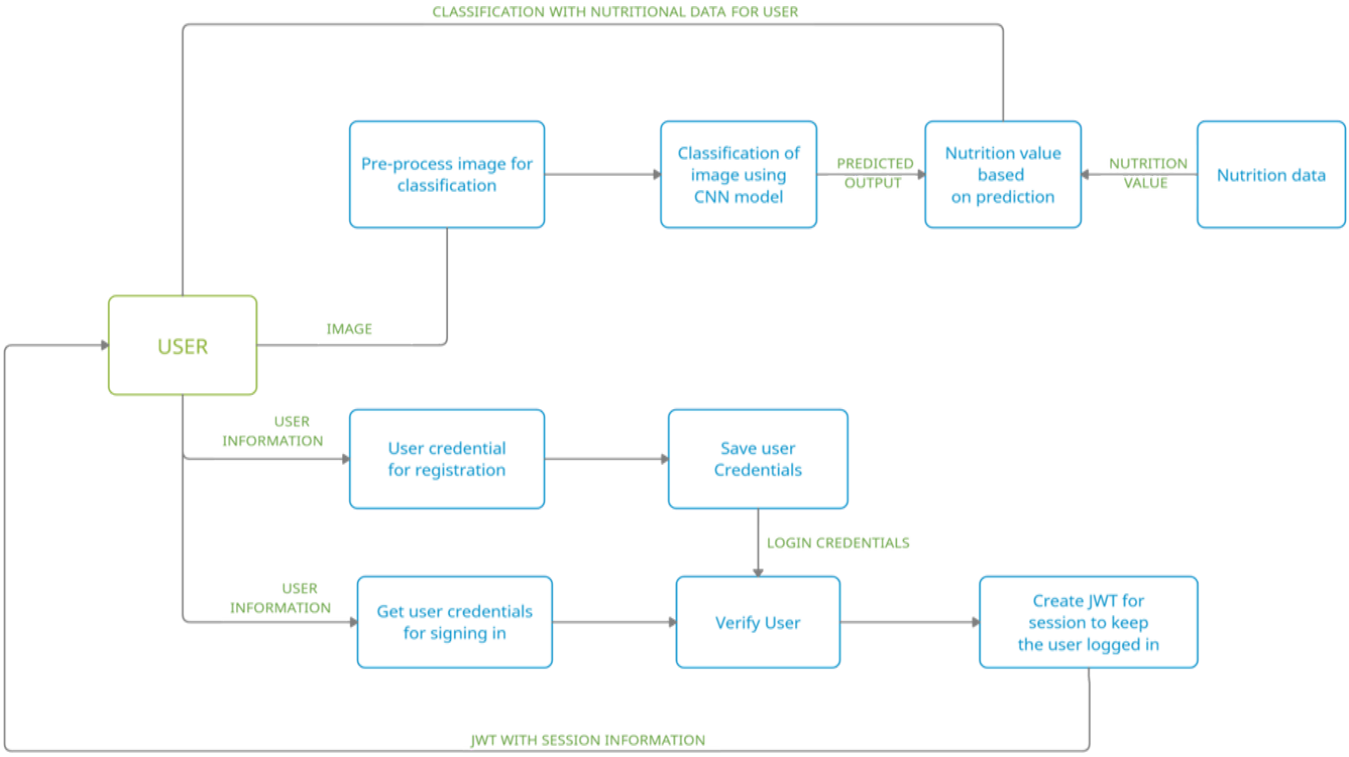
Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Easy to use with interactive User Interface
NFR-2	Security	User can access only their personal information and not that of other users.
NFR-3	Reliability	The average time of failure shall be 7 days.
NFR-4	Performance	The results has to be shown within 10 sec
NFR-5	Availability	The dietitian shall be available to users 24 hours a day, 7 days a week.
NFR-6	Scalability	Supports various food items

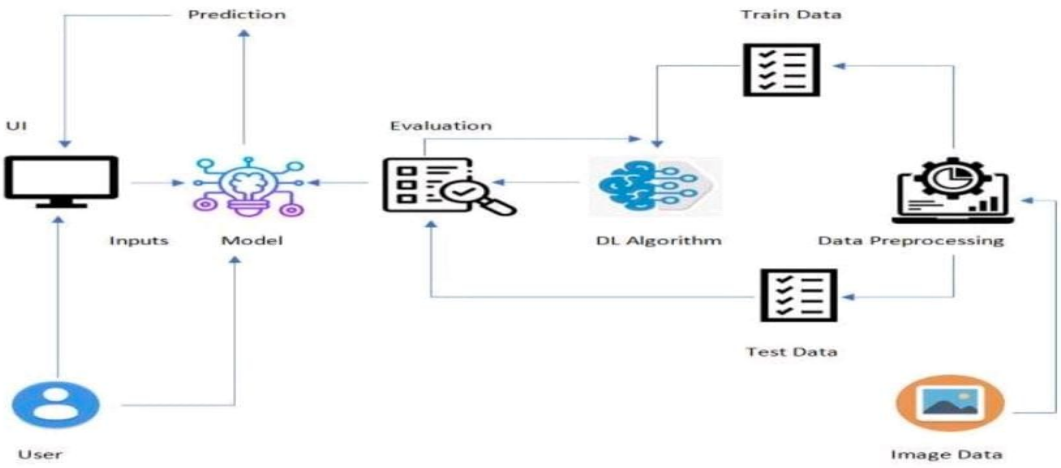
5 PROJECT DESIGN :

5.1 DATAFLOW DLAGRAM:

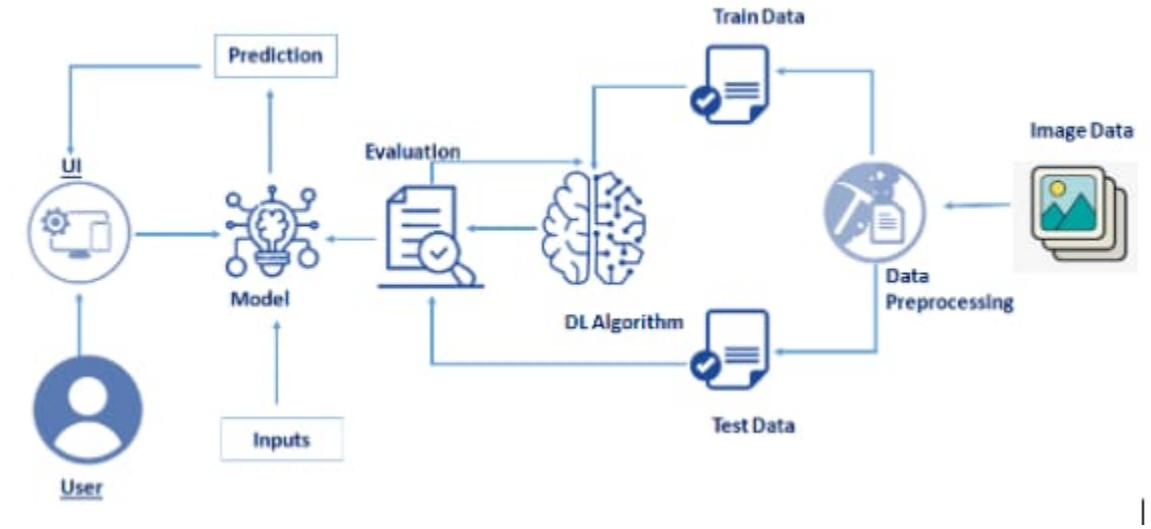


5.2 SOLUTION AND TECHNICAL ARCHITECTURE:

SOLUTION ARCHITECTURE:



TECHNICAL ARCHITECTURE:



6 PROJECT PLANNING & SCHEDULING

6.1 Spirit Delivery Plan:

Every project manager should consider the delivery strategy of the project deliverables as a strategic component. Every project's objective is to deliver a product that fulfill a certain need. The word "purpose" can be used to refer to a wide range of objectives, including those for a chair, a building, a translation, etc. Delivery planning is one of the activities used in Project Spirit to finish the project and display the projected timeline. This delivery plan aids in comprehending the team members' workflow and project procedure. Each individual module is given to a team member so they can showcase their efforts and contributions to the project's

development.

Milestone:

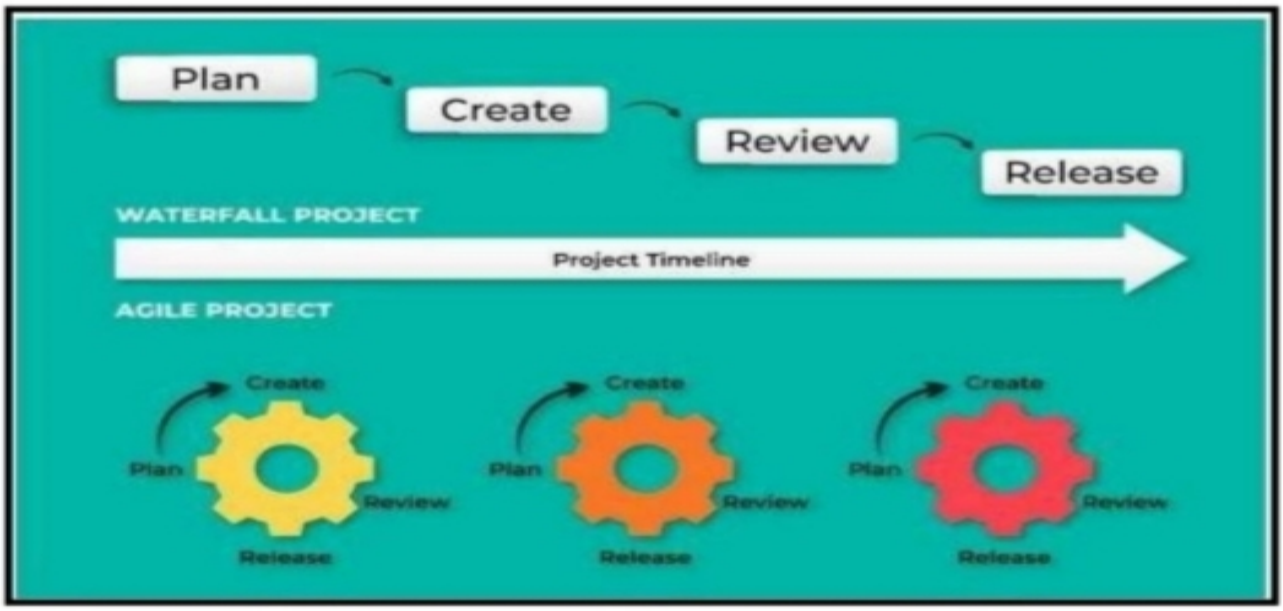
Thanks to modern technology, artificial intelligence (Al) model performance is improving. The development of a model that is used to categorize fruit is dependent on various traits, including color, shape, texture, etc. Here, users can take pictures of various fruits, which are subsequently-uploaded to a trained algorithm for analysis. The algorithm examines the image and determines the nutritious content of fruits, such as sugar, fiber, protein, calories, etc.).

Activity List:

In Project Management Planning is an Important task to scheduling the phrase of the project to the Team Member. In this Activity can shows the various activity are allocated and Done by the Team Members! In Project we can Split into the Four Step of Phrases are

Phrase 1: Information Collection and Requirement Analysis.Phrase 2: Project Planning and Developing Modules.

Phrase 3: Implementing the High Accuracy Deep Learning Algorithm to Perform. Phrase 4: Deploying the Model on Cloud and Testing the Model and UI Performance



7 CODING AN SOLUTIONING:

7.1 Import The ImageDataGenerator Library

Image data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in

the dataset. The Karas deep learning neural network library provides the capability to fit models using image data augmentation via the ImageDataGenerator class. Let us import the ImageDataGenerator class from Karas.

```
from keras.preprocessing.image  import    ImageDataGenerator
```

7.2 Configure ImageDataGenerator Class

ImageDataGenerator class is instantiated and the configuration for the types of data augmentation.

There are five main types of data augmentation techniques for image data; specifically:

Image shifts via the width_shift_range and height_shift_range arguments. The image flips via the horizontal_flip and vertical flip arguments. Image rotations via the rotation_range

argument Image brightness via the brightness_range argument.
Image zoom via the zoom range argument.An instance of the
ImageDataGenerator class can be constructed for train and test.

```
train_datagen= ImageDataGenerator (rescale=1./255, shear_range=0,shear_range=0.2,
 zoom range= 0.2, horizontal_ flip-True)
test_datagen=ImageDataGenerator (rescale=1./255)
```

7.3 Apply Image DataGenerator Functionality To Train set And Test set

Let us apply ImageDataGenerator functionality to Train set and Test set by using the following code
For Training set using flow from directory function.
This function will return batches of images from the subdirectories'apples', 'banana','orange', 'pineapple', 'watermelon' together with labels 0 to 4('apples': 0, 'banana': 1,'orange': 2, 'pineapple: 3, 'watermelon': 4)

```
x_train train_datagen.flow_from_directory(
    r'/content/drive/MyDrive/TRAIN SET,
target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse'
)
x_test=test_datagen.flow_from_directory(
r'/content/drive/MyDrive/TRAIN SET, target_size=(64,64),
batch_size=5,color_mode='rgb',class_mode='sparse'
)
```

```
print (x_train.class_indices)
print (x_test.class_indices)
from collections import Counter as c
c(x_train .labels)
```

7.4 Initializing The Model

Keras has 2 ways to define a neural network:

. Sequential

. Function API

The Sequential class is used to define linear initialization of network layers which then, collectively, constitute a model. In our example below, we will use the Sequential constructor to create a model, which will then have layers added to it using the add() method.

```
model = models.Sequential ()
```

7.5 Compiling the model and Fitting the model

```
#Compiling the model
model.compile (optimizer Adam",
                loss tf.keras.losses. SparseCategorical Cross-entropy
(from log its=True),
```

```
                metrics=['accuracy'])
#Fitting the model
 history model.fit(train_images, train_labels, epochs=10,
validation_data=(test images, test_labels))
```

7.6 Test The Model

Evaluation is a process during the development of the model to check whether the model is the best fit for the given problem and corresponding data. Load the saved model using load_model

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model= load_model ('nutrition.h5')
```

8 ADVANTAGES AND DISADVANTAGES:

ADVANTAGES:

Yet people consume foods, not nutrients, so it is helpful to view food or a meal as more than just a set of nutrients that impact our health. Some weight-loss diets have assigned a negative connotation to certain nutrients, such as low-fat or low-carbohydrate diets. This can create a view that a specific nutrient is bad, regardless of the role it plays when foods containing that nutrient are consumed as part of a healthy, balanced diet. This model helps in analyzing a nutrition in the food.

DISADVANTAGES:

Like anything, there are always drawbacks. In some cases the predicting algorithm may gives the wrong output.

9 CONCLUSION:

The good nutrition is fundamental for children's current and future health, as well as their development and learning. The benefits of developing healthy dietary and lifestyle patterns from an early age onwards can positively impact on people's nutrition and health throughout their adult lives, and enhance the productivity of individuals and nations. Nutrition education is an important element in an overall strategy aimed at improving food security and preventing all forms of malnutrition.

Most countries in the region implement school health and nutrition programmer, including school feeding, de-worming, vitamin and mineral supplementation, etc. Innovative, creative and effective school nutrition education programmed exist in some countries in the region. However, these are often small-scale and implemented as pilot projects, focus on children with special needs and prioritize the transfer of knowledge over the promotion of active learning and the creation of appropriate attitudes, life skills and behaviors.

10 APPENDIX

MODEL BUILDING:

```
from keras.preprocessing.image  import    ImageDataGenerator
train_datagen= ImageDataGenerator (rescale=1./255, shear_range=0,shear_range=0.2,
 zoom range= 0.2, horizontal_ flip-True)
test_datagen=ImageDataGenerator (rescale=1./255)
x_train train_datagen.flow_from_directory(
    r'/content/drive/MyDrive/TRAIN SET,
target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse'
```

```
)
x_test=test_datagen.flow_from_directory( r'/content/drive/MyDrive/TRAIN SET, target_size=(64,64), batch_size=5,color_mode='rgb',class_mode='sparse' )
print (x_train.class_indices)
print (x_test.class_indices)
from collections import Counter as c
c(x_train .labels)

import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense,Flatten
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Dropout
from keras.preprocessing.image import ImageDataGenerator

#initialize the model
model=Sequential()

from google.colab import drive
drive.mount('/content/drive')
Mounted at /content/drive

#image data augmentation
train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)

#performing data augmentation to test the data
x_train=train_datagen.flow_from_directory(r'/content/drive/MyDrive/TRAIN_SET',target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse')
```

Found 1 images belonging to 5 classes.

```
#check no of classes
print(x_train.class_indices)
{'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}

classifier = Sequential()

#adding CNN layers
classifier.add(Conv2D(32, (3, 3), input_shape=(64,64,3),activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2,2)))
classifier.add(Conv2D(32, (3,3),activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2,2)))
classifier.add(Flatten())

#summary of model
classifier.summary()

#Compiling the model
model.compile (optimizer Adam",
                loss tf.keras.losses. SparseCategorical Cross-entropy
(from log its=True),
                metrics=['accuracy'])
#Fitting the model
 history model.fit(train_images, train_labels, epochs=10,
validation_data=(test images, test_labels))
#Saving our model
model= load_model ('nutrition.h5')
#Prediciting our results
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image model-
```

load_model('nutrition.h5')
ing-image.load_ing("/content/drive/lyDrive/1_100.jpg',target_size=(70,7
0}}
img
x=image.img_to_array (img)
x= np.expand_dima (x, axis)
index=('AFFLES, BAHANA, ORANGE PINEAPPLE, WATERMELON]
resultant: (index[0])
result

Flask Application And Loading Our Model By Using Load_model Method

```
from flask import Flask,render_template,request
import os

import numpy as np

import requests
from tensorflow.keras.models import load_model #to load our trained model
from tensorflow.keras.preprocessing import image
import requests

app = Flask(__name__,template_folder='template') #initializing a flask app
# Loading the model
model=load_model('nutrition.h5')
print("Loaded model from disk")

@ app.route('/')# route to display the home page
def home():
    print("Loaded model from disk")
    return render_template('Home.html') #rendering the home page

@ app.route('/image1', methods=['GET', 'POST']) # routes to the index html
def image1():
    return render_template("Image.html")


@ app.route('/predict' ,methods=['GET','POST']) # route to show the
predictions in a Web UI
def lanuch():
    if request.method=='POST':
        f=request.files['file'] # requesting the file
        f
        basepath=os.path.dirname('__file__') #storing the file directory
        print(basepath)
        filepath=os.path.join(basepath,"test",f.filename)
```

```
        #storing the file in uploads folder
        f.save(filepath) #saving the file

        img=image.load_img(filepath,target_size=(32,32)) #load and reshaping
the image
        x=image.img_to_array(img) #converting image to an array
        x=np.expand_dims(x,axis=0) #changing the dimensions of the image

        pred=np.argmax(model.predict(x), axis=1)
        print("prediction",pred) #printing the prediction
        index=['APPLE','BANANA','ORANGE','BANANA','WATERMELON','WATERMELON','W
ATERMELON','APPLE','BANANA','WATERMELON']

        result=str(index[pred[0]])
        print(result)
        x=result
        result=nutrition(result)
        print(result)

        return render_template("0.html",showcase=(result),showcase1=(x))
def nutrition(index):

    import requests

    url = "https://calorieninjas.p.rapidapi.com/v1/nutrition"

    querystring = {"query":index}

    headers = {
    "X-RapidAPI-Key": "85887549f4msh51e7315b280a87ep1f43e0jsn585c940f2ea6",
    "X-RapidAPI-Host": "calorieninjas.p.rapidapi.com"
     }

    response = requests.request("GET", url, headers=headers,
params=querystring)

    print(response.text)
    return response.json()['items']
if __name__ == "__main__":
    # running the app
    app.run(debug=False)
```

Github Link:

https://github.com/IBM-EPBL/IBM-Project-29562-1660127155

Demo video Link:

https://youtu.be/9MdLgs7onWo