RECIPE RECOGNITION WITH DEEP LEARNING

A UG Phase- II Project report submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

MUPPALA KEERTHANA	**18UK1A0534**
BOURISHETTY SWETHA	**18UK1A0556**
KALERU NARESH	**18UK1A0538**
BHUKYA SANTOSH	**18UK1A0507**

Under the guidance of

Mr.CH.SHIVA SAI PRASAD

Assistant professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

(Affiliated to JNTU Hyderabad & Approved by AICTE, New Delhi)
Bollikunta, Warangal – 506005

CERTIFICATE

This is to certify that the UG phase-II project report entitled "**RECIPE RECOGNITION WITH DEEP LEARNING"** is being submitted by **MUPPALA KEERTHANA (18UK1A0534), BOURISHETTY SWETHA (18UK1A0556), KALERU NARESH (18UK1A0538), BHUKYA SANTHOSH (18UK1A0507)** in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2021- 2022

Project Guide	**Head of The Department**
Mr.CH.SHIVA SAI PRASAD	**DR.R. NAVEEN KUMAR**
(Assistant professor)	(professor)

EXTERNAL

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. P. Prasad Rao**, Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG phase-II project in the institute.

We extend our heartfelt thanks to **Dr. R. Naveen Kumar**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG phase-II project.

We express heartfelt thanks to the UG phase-II Project Coordinator, **Dr. J. Srikanth**, Assistant Professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG phase-II project.

We express heartfelt thanks to the guide, **Mr. CH. SHIVA SAI PRASAD**, Assistant Professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG phase-II project.

Finally, we express our sincere thanks and gratitude to our family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

MUPPALA KEERTHANA (18UK1A0534)
BOURISHETTY SWETHA (18UK1A0556)
KALERU NARESH (18UK1A0538)
BHUKYA SANTHOSH (18UK1A0507)

ABSTRACT

Food is an essential component of our individual and social life. Eating habits have a direct impact on our health and wellbeing, while ingredients, flavour and cooking recipes shape specific cuisines that are part of our personal and collective cultural identities. But there are also interesting applications of automatic food recognition to self-service restaurants and dining halls. For instance, accurate detection and segmentation of the different food items in a food tray can be used for monitoring food intake and nutritional information, and automatic billing to avoid the cashier bottleneck in self-service restaurants. This work deals with the problem of automated recognition of a photographed cooking dish and the subsequent output of the appropriate recipe.

In this project, we focus on applications of automatic food recognition and identify the recipe in food by using convolutional neural networks. And this model will classify images into food categories and to output a matching recipe.

TABLE OF CONTENTS:

LIST OF FIGURES **page no.**

1.INTRODUCTION

1.1 OVERVIEW:

Food is an essential component of our individual and social life. Eating habits have a direct impact on our health and wellbeing, while ingredients, flavour and cooking recipes shape specific cuisines that are part of our personal and collective cultural identities. But there are also interesting applications of automatic food recognition to self-service restaurants and dining halls. For instance, accurate detection and segmentation of the different food items in a food tray can be used for monitoring food intake and nutritional information, and automatic billing to avoid the cashier bottleneck in self-service restaurants. This work deals with the problem of automated recognition of a photographed cooking dish and the subsequent output of the appropriate recipe.

In this project, we focus on applications of automatic food recognition and identify the recipe in food by using convolutional neural networks. And this model will classify images into food categories and to output a matching recipe.

1.2 OBJECTIVE OF PROJECT:

By the end of this project, you will:

- know fundamental concepts and techniques of the Artificial Neural Network and Convolution Neural Networks
- Gain a broad understanding of image data.
- Work with Sequential type of modelling
- Work with Keras and TensorFlow capabilities
- Work with image processing techniques
- know how to build a web application using the Flask framework.

1.3 PURPOSE:

By Using the Recipe Recognition with Deep Learning:

- We know the concepts and can work on Deep Learning Architecture.
- Applications of automatic food recognition includes self-service restaurants and dining halls.
- Accurate detection and segmentation of the different food items in a food tray can be used for monitoring food intake and nutritional information, and automatic billing in restaurants.

2. PROBLEM STATEMENT:

- Our project deals with the problem of automated recognition of a photographed cooking dish and the subsequent output of the appropriate recipe.
- In this project, we focus on applications of automatic food recognition and identify the recipe in food by using convolutional neural networks.
- Our Deep Learning model will classify images into food categories and to output a matching recipe.

3.LITERATURE SURVEY:

3.1 EXISTING SYSTEM:

- The distinction between the difficulty of the chosen problem and previous supervised classification problems is that there are large overlaps in food dishes, as dishes of different categories.
- combination of object recognition using Convolutional Neural Networks (short CNN) over the input record of images.
- Implementations of techniques and concepts of Convolutional Neural Networks helps to find the recipe more likely.

3.2 PROPOSED SYSYEM:

We are creating a web application where the user selects the image which is to be classified. The image is fed into the model that is trained and the output will be displayed on the webpage.
Our Convolutional Neural Network model is for accurate recognition of a photographed cooking dish and the subsequent output of the appropriate recipe.

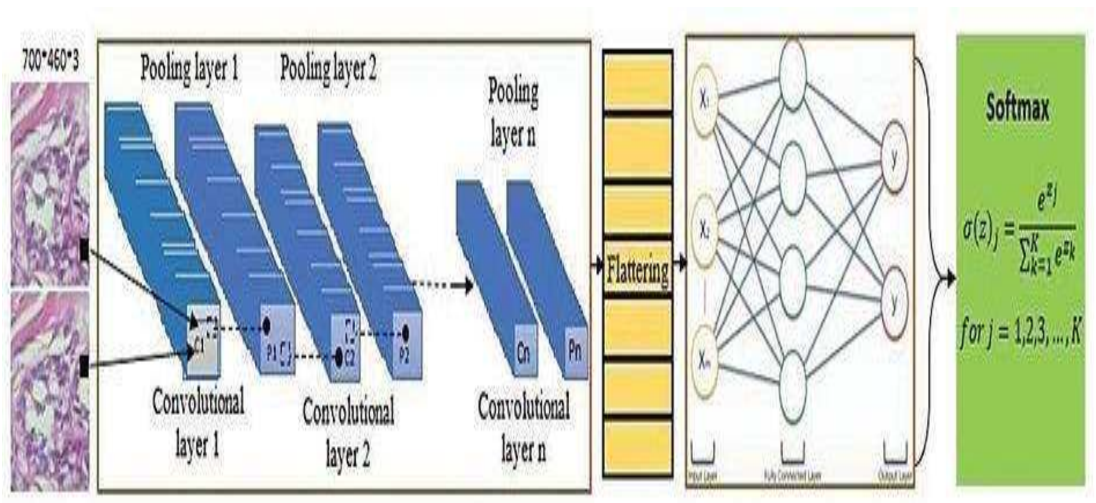


Figure 1: Training and Testing using Histopathological images with CNN

These filters scan the image by a sliding window on the image, while learning the recurrent patterns which arise in any area of the image. The interval between filters is known as the stride. The convolution is extended to overlapping windows if the stride hyper parameter is smaller than the filter dimension. Convolutional layers bring out the features of images with precise positions. If the positions change, even a small amount for any reason, the feature maps will be different. To overcome this problem, the down sampling process must be done at the output of every convolutional layer.
With convolutional layers, down sampling can be done by changing the convolution’s phase across the image.

DEEP LEARNING:

Deep learning neural networks, or artificial neural networks, attempts to mimic the human brain through a combination of data inputs, weights, and bias. These elements work together to accurately recognize, classify, and describe objects within the data.

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called visible layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.

Another process called backpropagation uses algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model. Together, forward propagation and backpropagation allow a neural network to make predictions and correct for any errors accordingly. Over time, the algorithm becomes gradually more accurate.

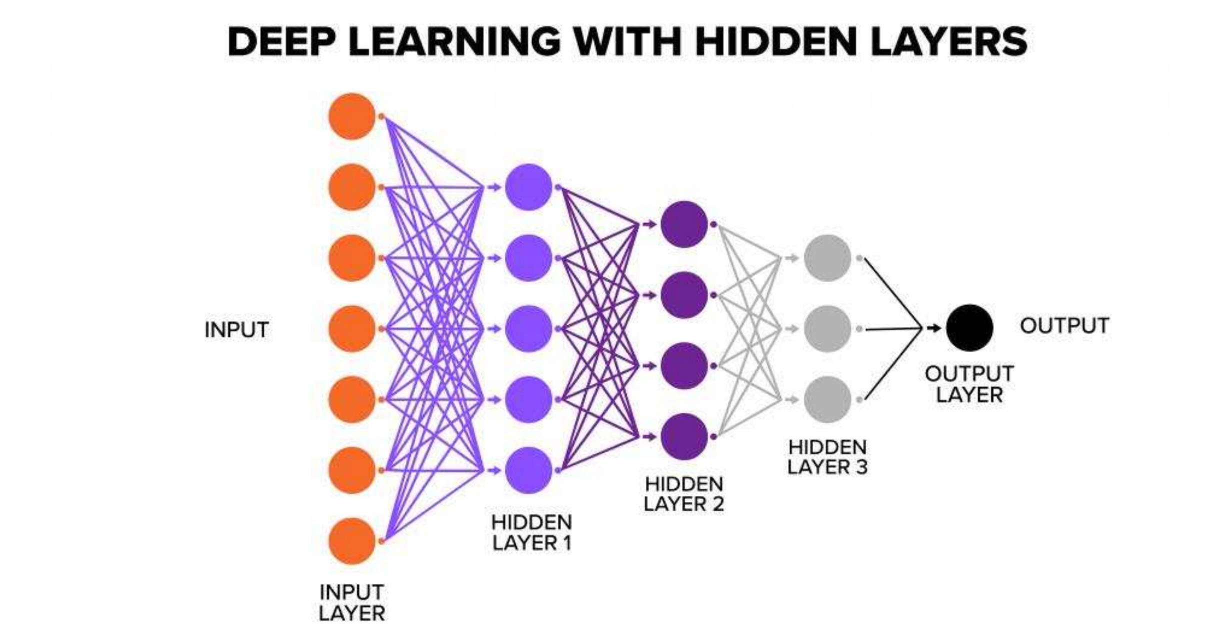
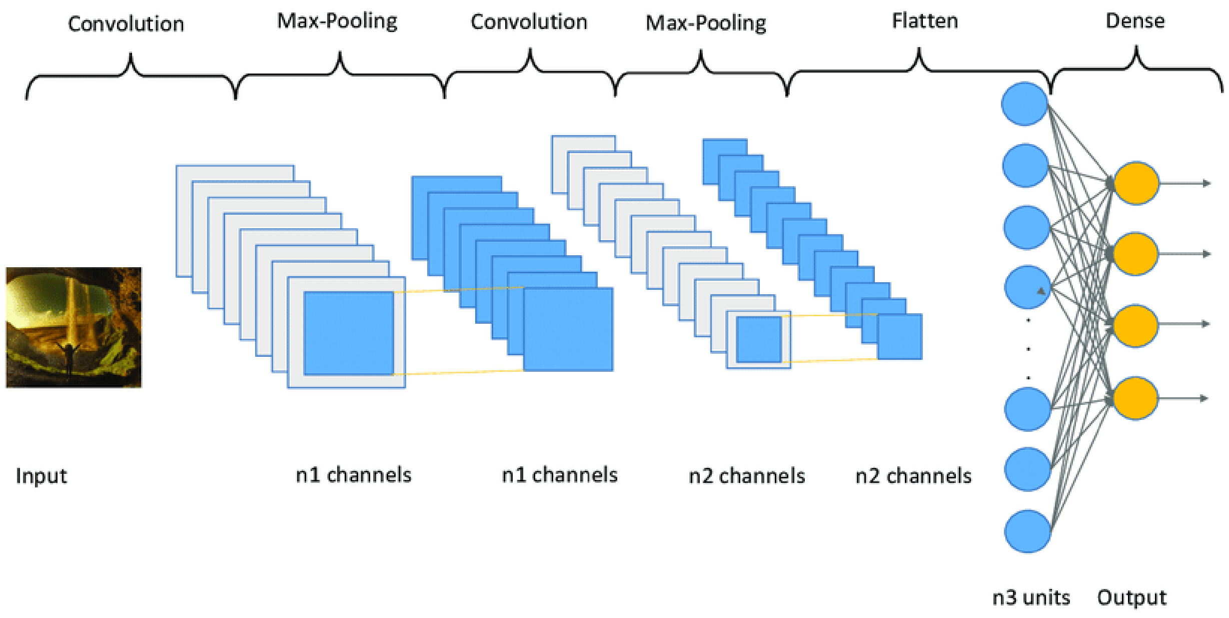


Fig 2: Deep Learning with Hidden Layers

CONVOLUTIONAL NEURAL NETWORKS:

A convolutional neural network is a specific kind of neural network with multiple layers. It processes data that has a grid-like arrangement then extracts important features.

Convolutional neural networks are based on neuroscience findings. They are made of layers of artificial neurons called nodes. These nodes are functions that calculate the weighted sum of the inputs and return an activation map. This is the convolution part of the neural network.



Each node in a layer is defined by its weight values. When you give a layer some data, like an image, it takes the pixel values and picks out some of the visual features.

Usually with images, a CNN will initially find the edges of the picture. Then this slight definition of the image will get passed to the next layer. Then that layer will start detecting things like corners and colour groups. Then that image definition will get passed to the next layer and the cycle continues until a prediction is made.

4.THEORITICAL ANALYSIS:

4.1 BLOCK DIAGRAM:

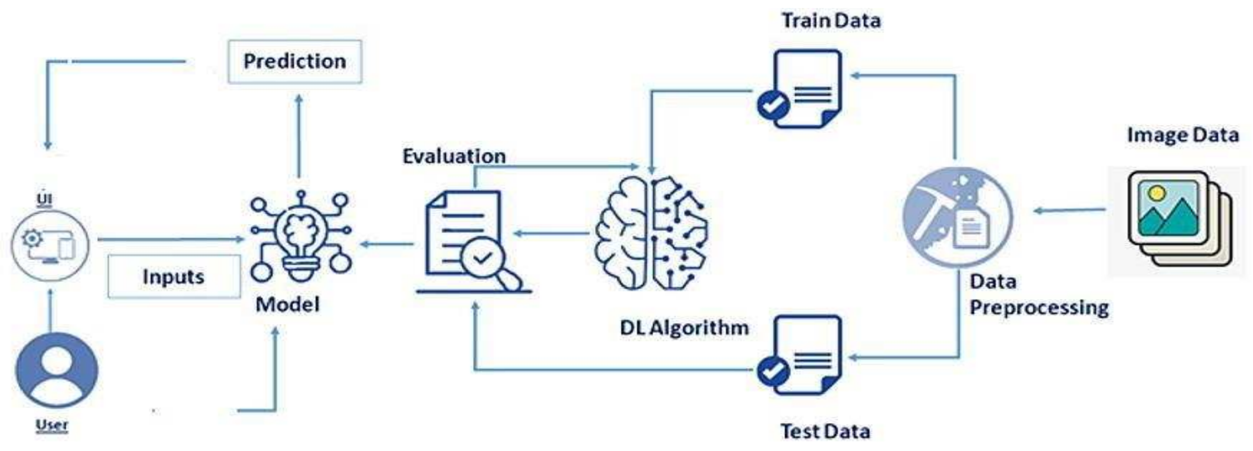
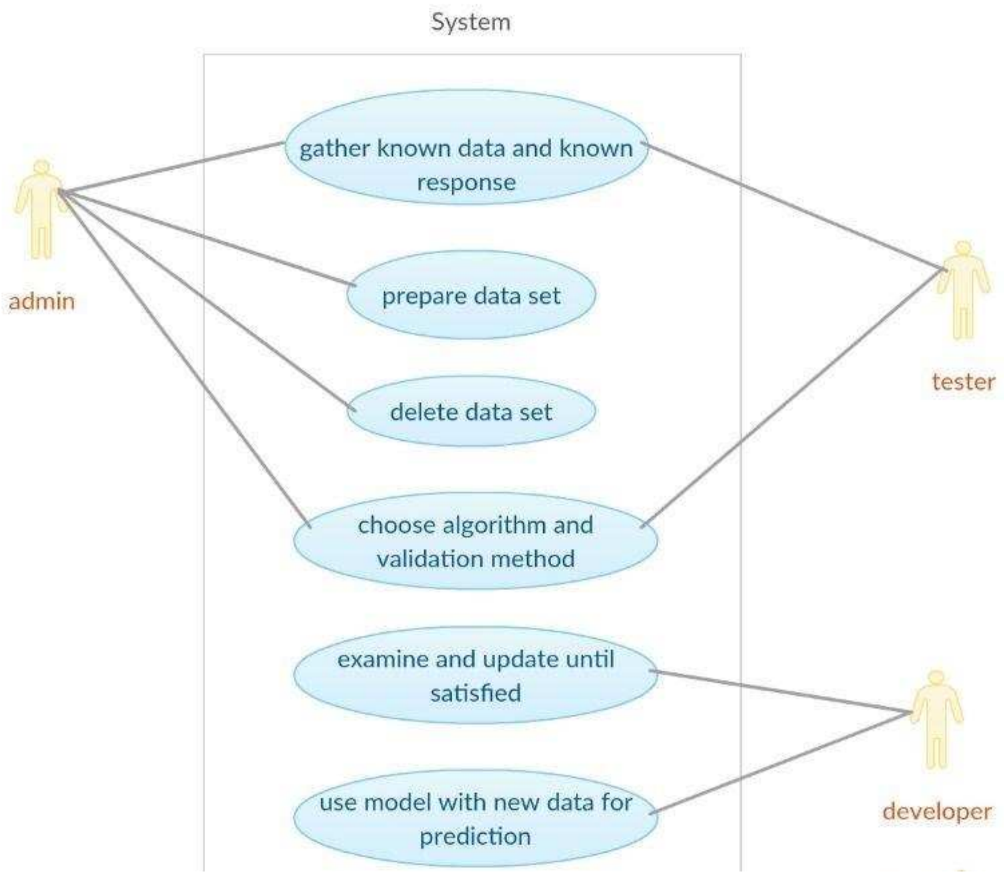


Figure 2: Project architecture

Fig 3: project Architecture

4.2 USE CASE DIAGRAM:



System
gather known data and known response
admin
prepare data set
tester
delete data set
choose algorithm and validation method
examine and update until satisfied
use model with new data for prediction
developer

Fig 4. Use Case diagram for model building

4.3 SOFTWARE SPECIFICATIONS:

REQUIREMENTS	**SPECIFICATIONS**
Anaconda Navigator	You must have anaconda installed in your device prior to begin.
Spyder, Jupyter Notebook, Flask Framework, deep learning software tools like keras, tensorflow	1. One should have Spyder and Jupyter notebook. 2. One should install flask framework through anaconda prompt for running their web application 3. We need to build the model using jupyter notebook with all the imported packages.
Web browser	For all Web browsers, the following must be enabled: ● cookies ● HTML ● Java script

4.4 HARDWARE SPECIFICATIONS:

REQUIREMENTS	SPECIFICATIONS
Operating system	Microsoft Windows UNIX Linux
Processor	Minimum: 2 CPU cores for one user. For each deployment, a sizing exercise is highly recommended.
RAM	Minimum 8 GB.
Operating system specifications	File descriptor limit set to 8192 on UNIX and Linux
Disk space	A minimum of 7 GB of free space is required to install the software.

5.EXPERIMENTAL ANALYSIS:

In this project, we focus on applications of automatic food recognition and identify the recipe in food by using convolutional neural networks. And this model will classify images into food categories and to output a matching recipe.

- User interacts with the UI (User Interface) to upload the image as input
- Uploaded image is analysed by the model which is integrated
- Once model analyses the uploaded image, the model predicts the food item and the recipe is showcased on the UI.

5.1 DATA COLLECTION:

ML or DL depends heavily on data, without data, it is impossible for a machine to learn. It is the most crucial aspect that makes algorithm training possible. It is the actual data set used to train the model for performing various actions.

- Collect the dataset or create the dataset

5.2 DATA PREPROCESSING:

Image Pre-processing includes the following main tasks:

The ImageDataGenerator accepts the original data, randomly transforms it, and returns only the new, transformed data.

- Import the ImageDataGenerator library
- Configure ImageDataGenerator class
- Apply ImageDataGenerator functionality to Trainset and Test set

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.
- 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.

5.3 MODEL BUILDING:

The neural network model is to be built by adding different network layers like convolution, pooling, flattening, dropout, and neural layers.

In this milestone, we start building our model by:

- Import the model building Libraries
- Initializing the model
- Adding Input Layer
- Adding Hidden Layer
- Adding Output Layer
- Configure the Learning Process
- Training the model
- Save the Model
- Test the Model

5.4 APPLICATION BUILDING:

After the model is built, we will be integrating it into a web application so that users can interact with the model.

- Create an HTML file
- Build Python Code

6. FLOW CHART:

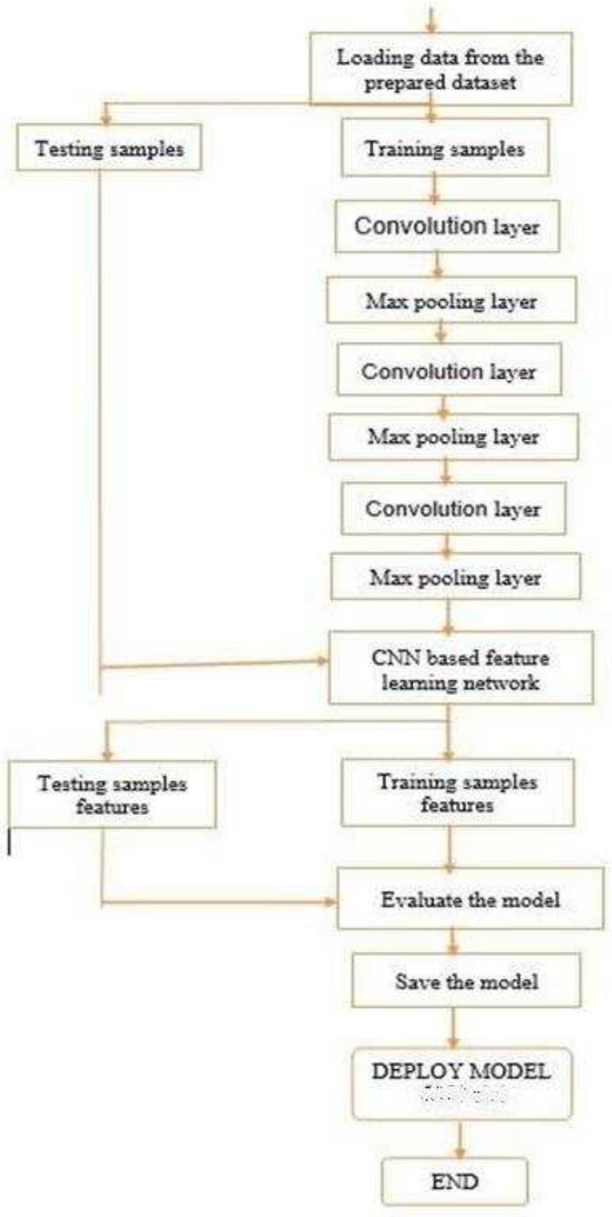
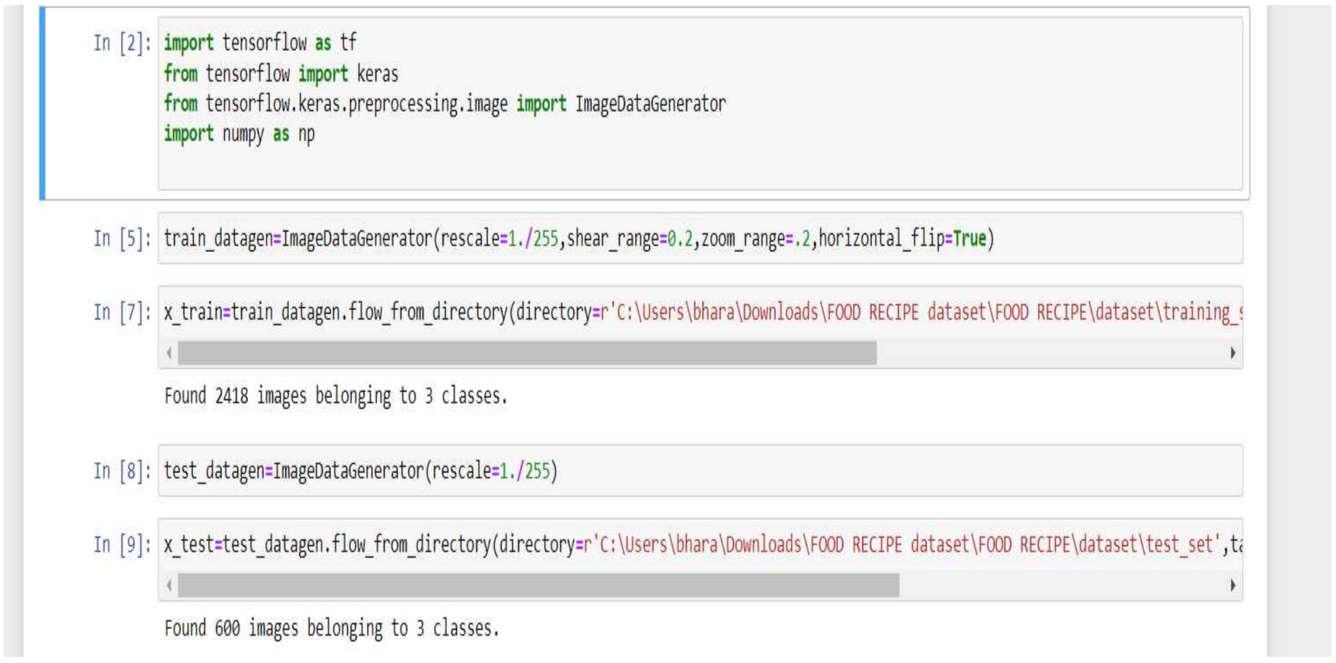


Fig 5: Flow Chart

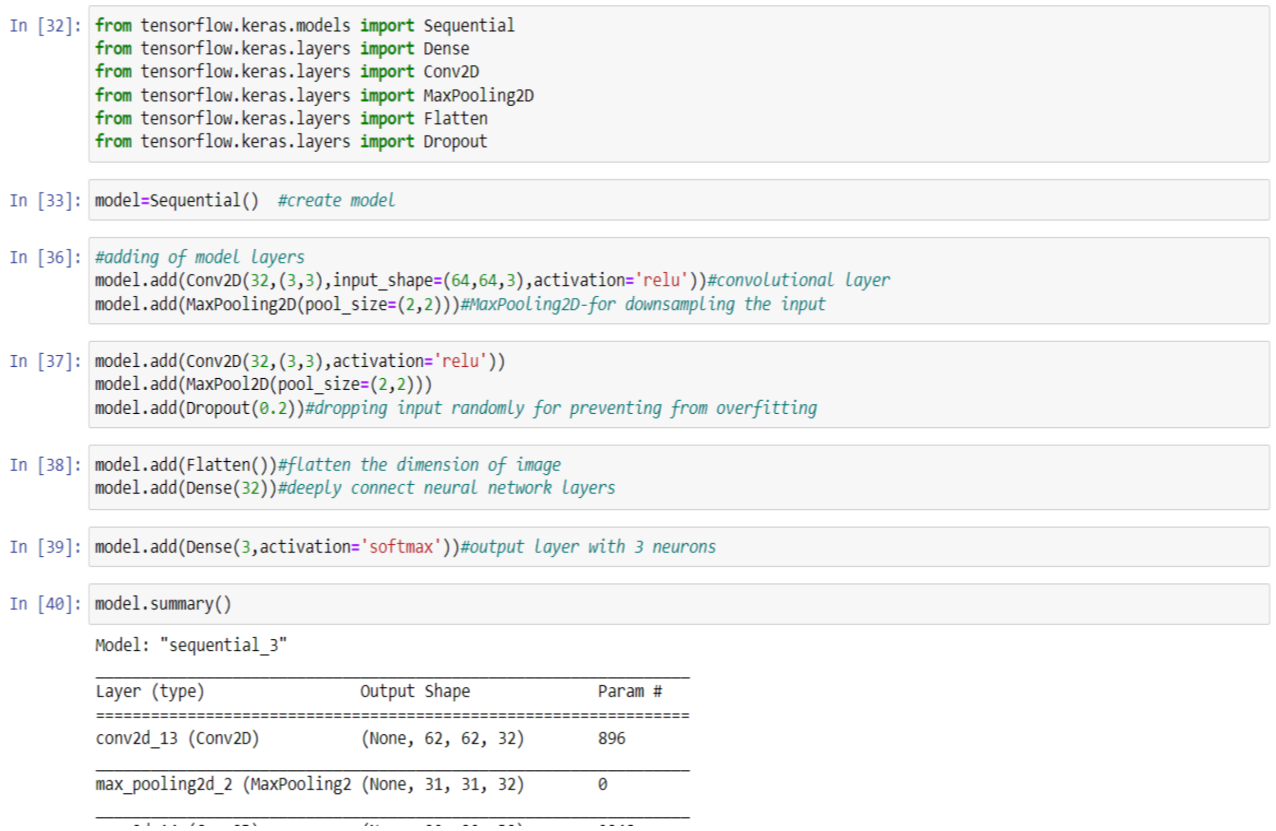
7.CODE SNIPPETS

7.1 MODEL CODE:



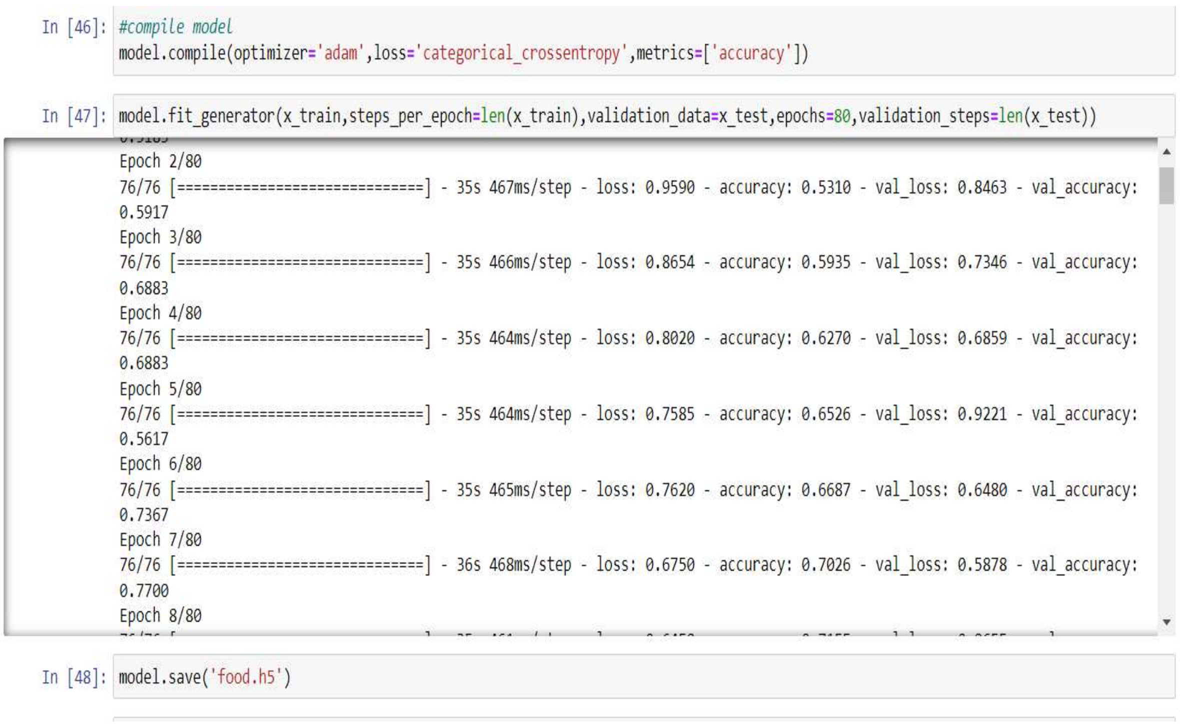
```
In [2]:  import tensorflow as tf
         from tensorflow import keras
         from tensorflow.keras.preprocessing.image import ImageDataGenerator
         import numpy as np

In [5]:  train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=.2,horizontal_flip=True)

In [7]:  x_train=train_datagen.flow_from_directory(directory=r'C:\Users\bhara\Downloads\FOOD RECIPE dataset\FOOD RECIPE\dataset\training_s

         Found 2418 images belonging to 3 classes.

In [8]:  test_datagen=ImageDataGenerator(rescale=1./255)

In [9]:  x_test=test_datagen.flow_from_directory(directory=r'C:\Users\bhara\Downloads\FOOD RECIPE dataset\FOOD RECIPE\dataset\test_set',ta

         Found 600 images belonging to 3 classes.
```

Fig 6: code for importing libraries and data pre-processing



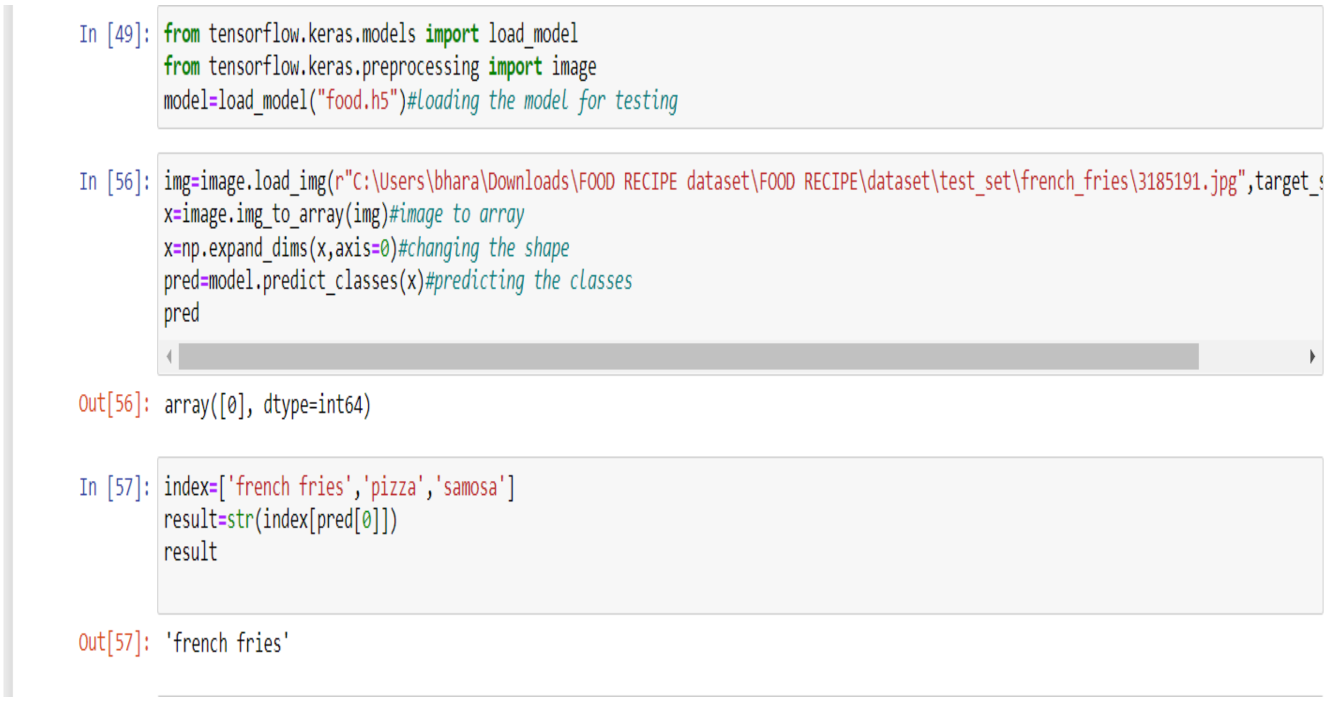
```
In [32]: from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Dense
         from tensorflow.keras.layers import Conv2D
         from tensorflow.keras.layers import MaxPooling2D
         from tensorflow.keras.layers import Flatten
         from tensorflow.keras.layers import Dropout

In [33]: model=Sequential()  #create model

In [36]: #adding of model layers
         model.add(Conv2D(32,(3,3),input_shape=(64,64,3),activation='relu'))#convolutional layer
         model.add(MaxPooling2D(pool_size=(2,2)))#MaxPooling2D-for downsampling the input

In [37]: model.add(Conv2D(32,(3,3),activation='relu'))
         model.add(MaxPool2D(pool_size=(2,2)))
         model.add(Dropout(0.2))#dropping input randomly for preventing from overfitting

In [38]: model.add(Flatten())#flatten the dimension of image
         model.add(Dense(32))#deeply connect neural network layers

In [39]: model.add(Dense(3,activation='softmax'))#output layer with 3 neurons

In [40]: model.summary()

         Model: "sequential_3"
         _________________________________________________________________
         Layer (type)                 Output Shape              Param #
         =================================================================
         conv2d_13 (Conv2D)           (None, 62, 62, 32)        896
         _________________________________________________________________
         max_pooling2d_2 (MaxPooling2 (None, 31, 31, 32)        0
         _________________________________________________________________
```

Fig 7: code for Model Building



```
In [46]: #compile model
         model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

In [47]: model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,epochs=80,validation_steps=len(x_test))

         Epoch 2/80
         76/76 [==============================] - 35s 467ms/step - loss: 0.9590 - accuracy: 0.5310 - val_loss: 0.8463 - val_accuracy:
         0.5917
         Epoch 3/80
         76/76 [==============================] - 35s 466ms/step - loss: 0.8654 - accuracy: 0.5935 - val_loss: 0.7346 - val_accuracy:
         0.6883
         Epoch 4/80
         76/76 [==============================] - 35s 464ms/step - loss: 0.8020 - accuracy: 0.6270 - val_loss: 0.6859 - val_accuracy:
         0.6883
         Epoch 5/80
         76/76 [==============================] - 35s 464ms/step - loss: 0.7585 - accuracy: 0.6526 - val_loss: 0.9221 - val_accuracy:
         0.5617
         Epoch 6/80
         76/76 [==============================] - 35s 465ms/step - loss: 0.7620 - accuracy: 0.6687 - val_loss: 0.6480 - val_accuracy:
         0.7367
         Epoch 7/80
         76/76 [==============================] - 36s 468ms/step - loss: 0.6750 - accuracy: 0.7026 - val_loss: 0.5878 - val_accuracy:
         0.7700
         Epoch 8/80

In [48]: model.save('food.h5')
```

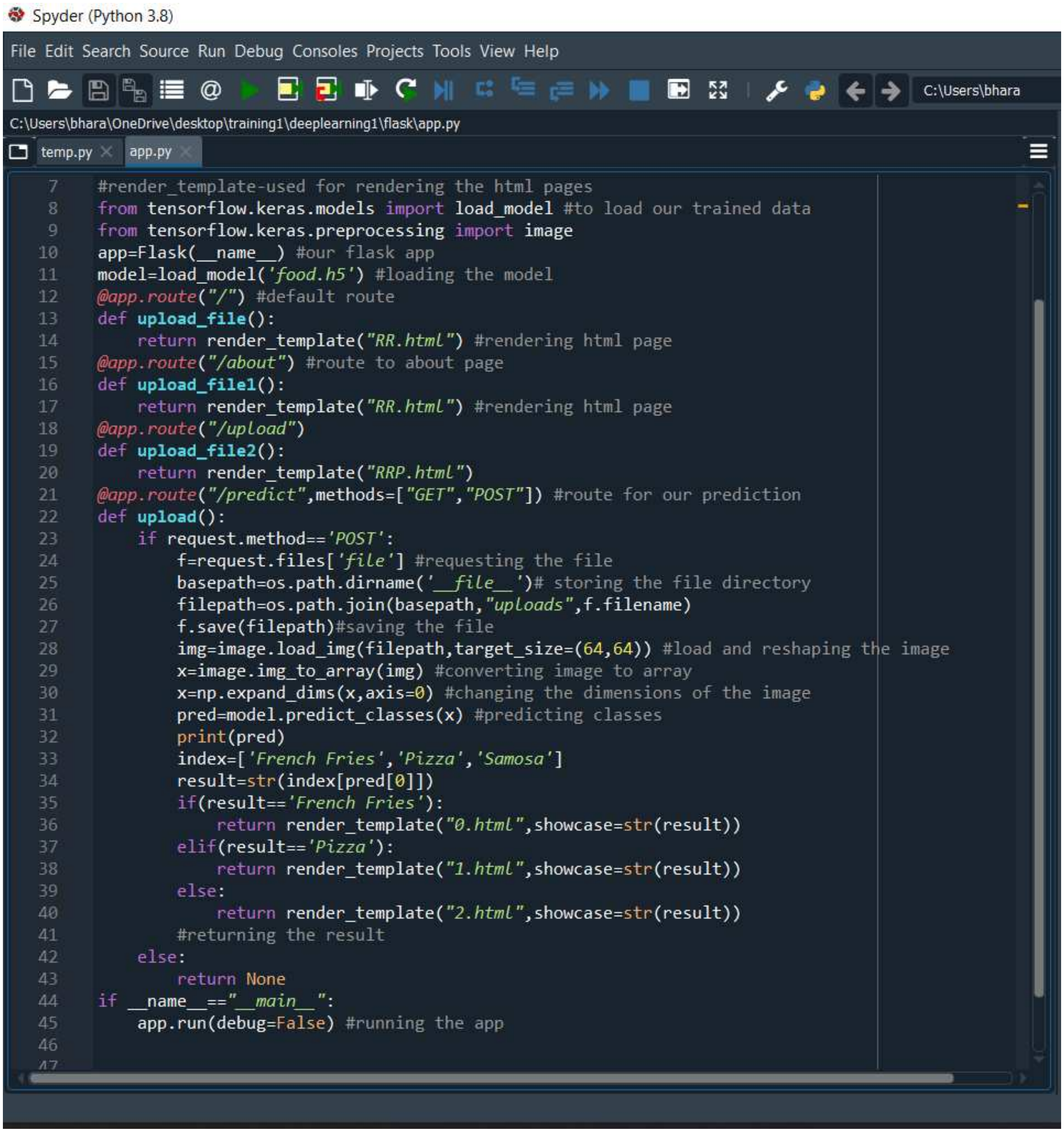
Fig 8: code for training the model



```
In [49]: from tensorflow.keras.models import load_model
         from tensorflow.keras.preprocessing import image
         model=load_model("food.h5")#loading the model for testing

In [56]: img=image.load_img(r"C:\Users\bhara\Downloads\FOOD RECIPE dataset\FOOD RECIPE\dataset\test_set\french_fries\3185191.jpg",target_
         x=image.img_to_array(img)#image to array
         x=np.expand_dims(x,axis=0)#changing the shape
         pred=model.predict_classes(x)#predicting the classes
         pred

Out[56]: array([0], dtype=int64)

In [57]: index=['french fries','pizza','samosa']
         result=str(index[pred[0]])
         result

Out[57]: 'french fries'
```

Fig 9: code for testing the model

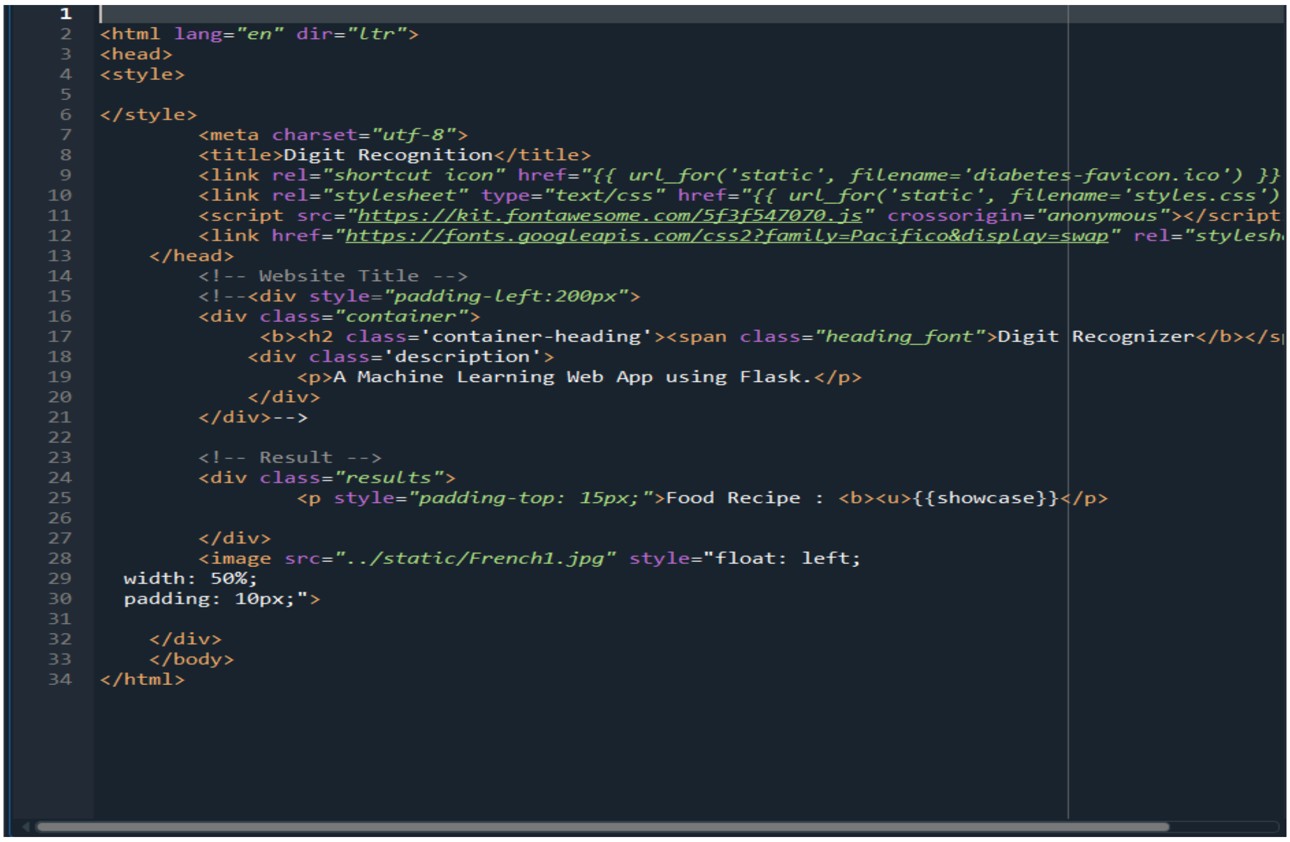
7.2 app.py CODE:


Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\bhara
C:\Users\bhara\OneDrive\desktop\training1\deeplearning1\flask\app.py
temp.py
app.py

```python
#render_template-used for rendering the html pages
from tensorflow.keras.models import load_model #to load our trained data
from tensorflow.keras.preprocessing import image
app=Flask(__name__) #our flask app
model=load_model('food.h5') #loading the model
@app.route("/") #default route
def upload_file():
    return render_template("RR.html") #rendering html page
@app.route("/about") #route to about page
def upload_file1():
    return render_template("RR.html") #rendering html page
@app.route("/upload")
def upload_file2():
    return render_template("RRP.html")
@app.route("/predict",methods=["GET","POST"]) #route for our prediction
def upload():
    if request.method=='POST':
        f=request.files['file'] #requesting the file
        basepath=os.path.dirname('__file__')# storing the file directory
        filepath=os.path.join(basepath,"uploads",f.filename)
        f.save(filepath)#saving the file
        img=image.load_img(filepath,target_size=(64,64)) #load and reshaping the image
        x=image.img_to_array(img) #converting image to array
        x=np.expand_dims(x,axis=0) #changing the dimensions of the image
        pred=model.predict_classes(x) #predicting classes
        print(pred)
        index=['French Fries','Pizza','Samosa']
        result=str(index[pred[0]])
        if(result=='French Fries'):
            return render_template("0.html",showcase=str(result))
        elif(result=='Pizza'):
            return render_template("1.html",showcase=str(result))
        else:
            return render_template("2.html",showcase=str(result))
        #returning the result
    else:
        return None
if __name__=="__main__":
    app.run(debug=False) #running the app
```

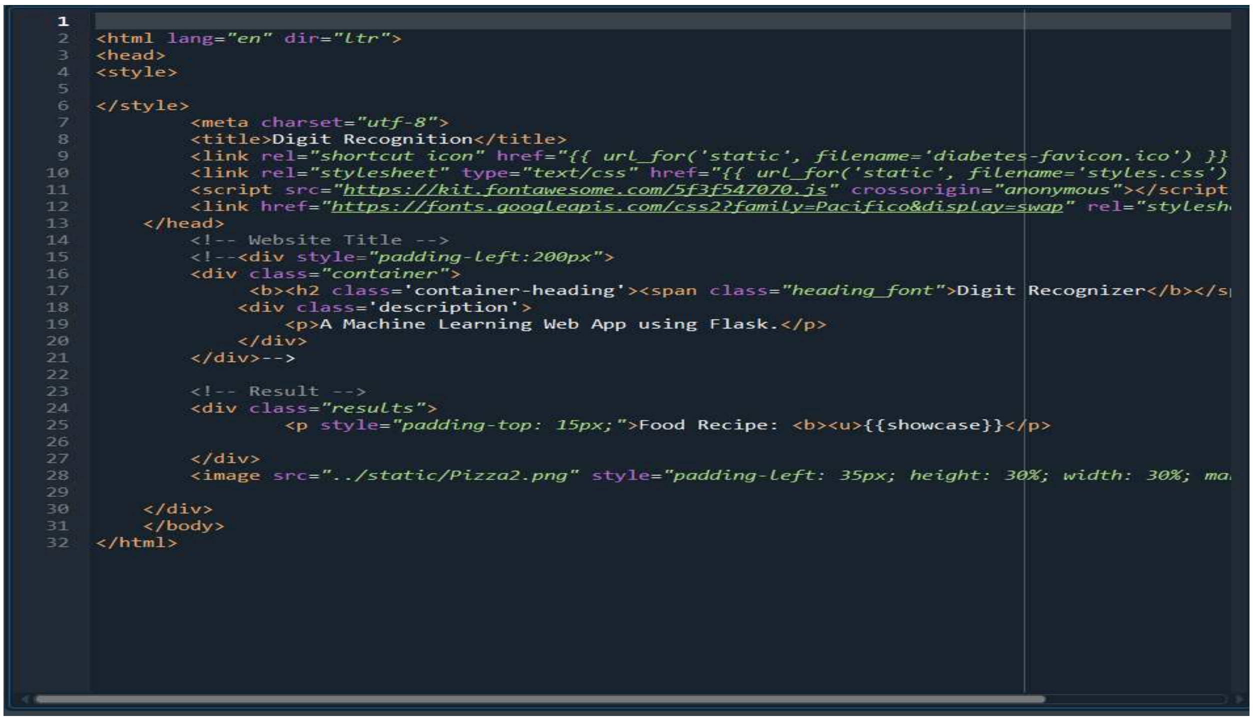
Fig 10: app.py code

7.3 HTML CODE:



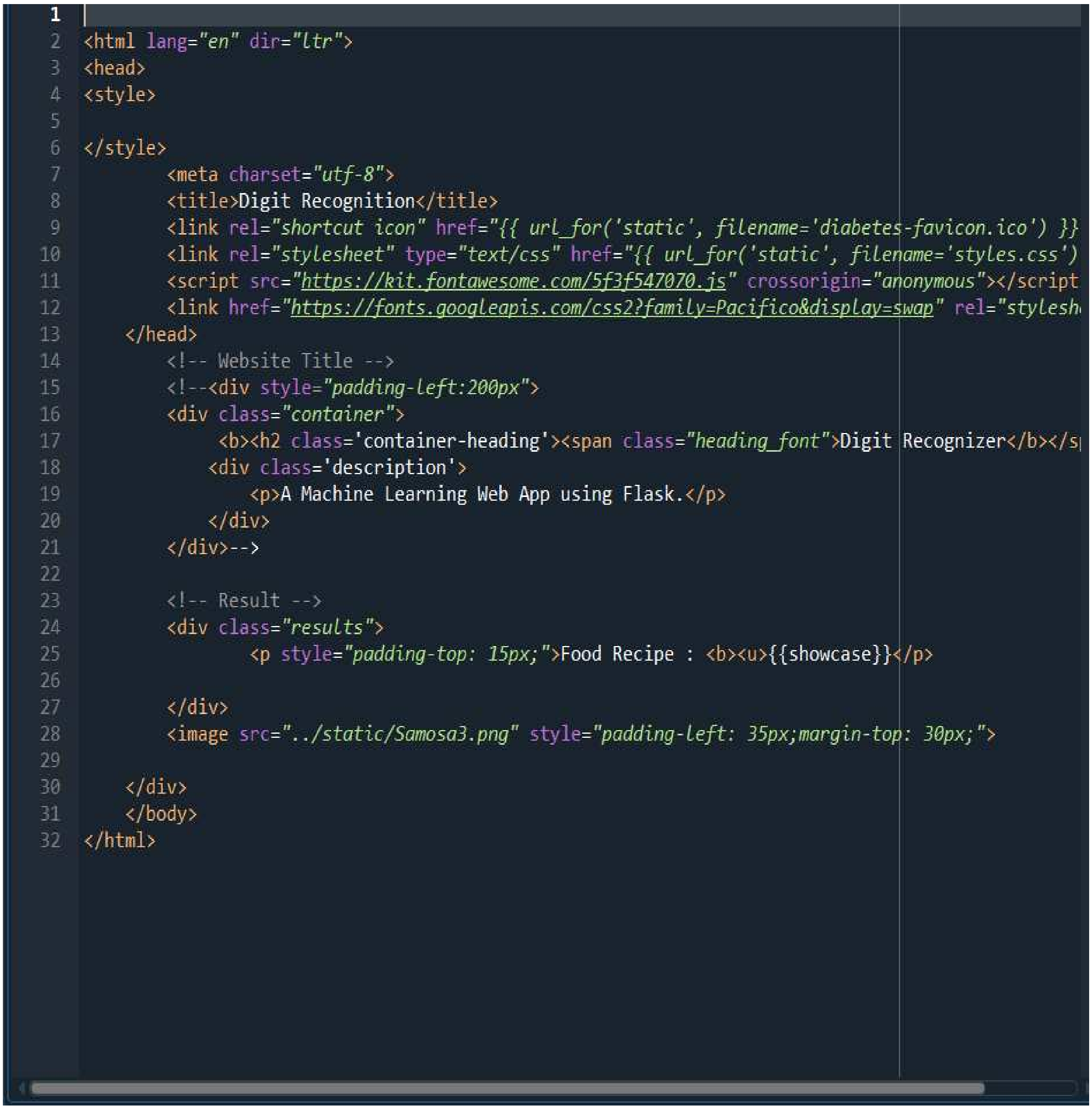
```

<html lang="en" dir="ltr">
<head>
<style>

</style>
        <meta charset="utf-8">
        <title>Digit Recognition</title>
        <link rel="shortcut icon" href="{{ url_for('static', filename='diabetes-favicon.ico') }}
        <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css')
        <script src="https://kit.fontawesome.com/5f3f547070.js" crossorigin="anonymous"></script
        <link href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap" rel="styleshe
    </head>
        <!-- Website Title -->
        <!--<div style="padding-left:200px">
        <div class="container">
             <b><h2 class='container-heading'><span class="heading_font">Digit Recognizer</b></s
            <div class='description'>
                <p>A Machine Learning Web App using Flask.</p>
            </div>
        </div>-->

        <!-- Result -->
        <div class="results">
                <p style="padding-top: 15px;">Food Recipe : <b><u>{{showcase}}</p>

        </div>
        <image src="../static/French1.jpg" style="float: left;
  width: 50%;
  padding: 10px;">

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

Fig 11: base.html



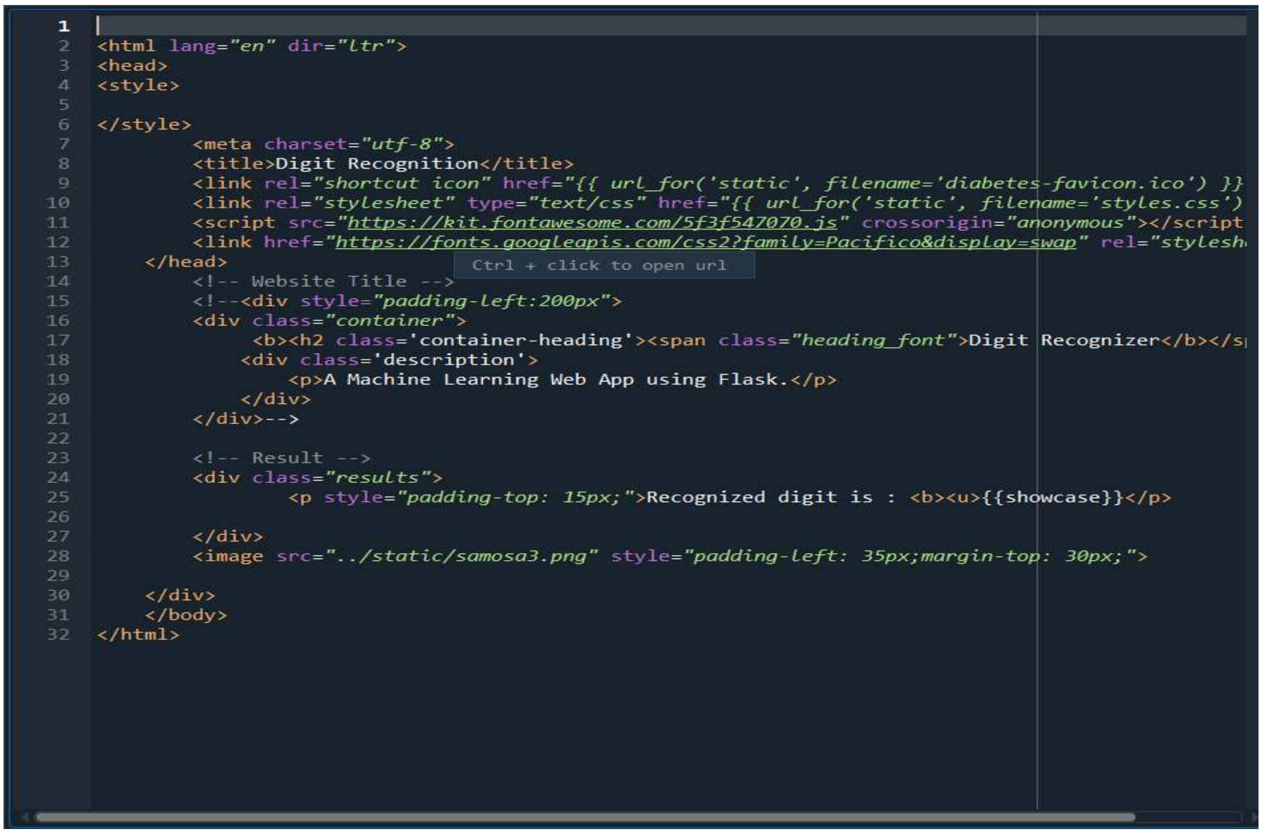
```

<html lang="en" dir="ltr">
<head>
<style>

</style>
        <meta charset="utf-8">
        <title>Digit Recognition</title>
        <link rel="shortcut icon" href="{{ url_for('static', filename='diabetes-favicon.ico') }}
        <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css')
        <script src="https://kit.fontawesome.com/5f3f547070.js" crossorigin="anonymous"></script
        <link href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap" rel="styleshe
    </head>
        <!-- Website Title -->
        <!--<div style="padding-left:200px">
        <div class="container">
             <b><h2 class='container-heading'><span class="heading_font">Digit Recognizer</b></s
            <div class='description'>
                <p>A Machine Learning Web App using Flask.</p>
            </div>
        </div>-->

        <!-- Result -->
        <div class="results">
                <p style="padding-top: 15px;">Food Recipe: <b><u>{{showcase}}</p>

        </div>
        <image src="../static/Pizza2.png" style="padding-left: 35px; height: 30%; width: 30%; ma

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

Fig 12: index1.html



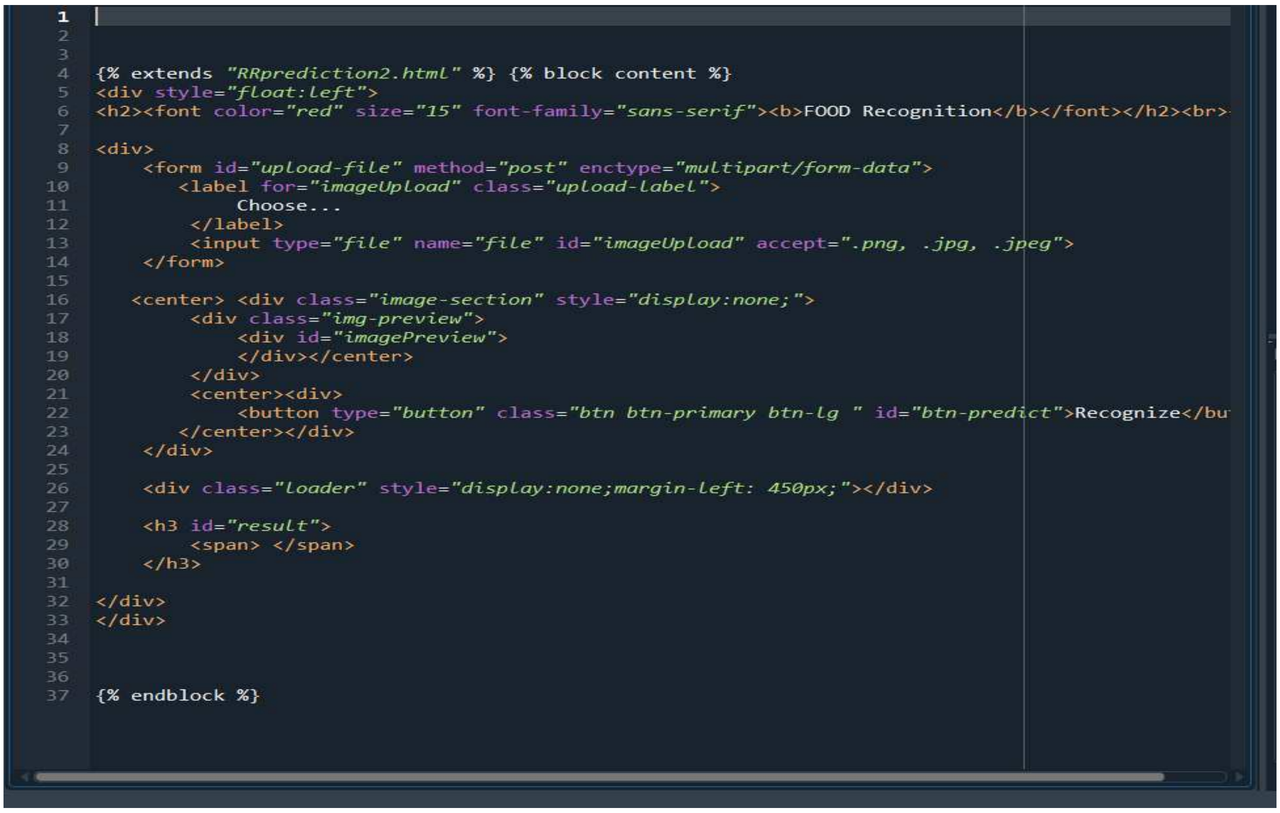
```html

<html lang="en" dir="ltr">
<head>
<style>

</style>
        <meta charset="utf-8">
        <title>Digit Recognition</title>
        <link rel="shortcut icon" href="{{ url_for('static', filename='diabetes-favicon.ico') }}
        <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css')
        <script src="https://kit.fontawesome.com/5f3f547070.js" crossorigin="anonymous"></script
        <link href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap" rel="styleshe
    </head>
        <!-- Website Title -->
        <!--<div style="padding-left:200px">
        <div class="container">
             <b><h2 class='container-heading'><span class="heading_font">Digit Recognizer</b></s
            <div class='description'>
                <p>A Machine Learning Web App using Flask.</p>
            </div>
        </div>-->

        <!-- Result -->
        <div class="results">
                <p style="padding-top: 15px;">Food Recipe : <b><u>{{showcase}}</p>

        </div>
        <image src="../static/Samosa3.png" style="padding-left: 35px;margin-top: 30px;">

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

Fig 13: index2.html



```html

<html lang="en" dir="ltr">
<head>
<style>

</style>
        <meta charset="utf-8">
        <title>Digit Recognition</title>
        <link rel="shortcut icon" href="{{ url_for('static', filename='diabetes-favicon.ico') }}
        <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css')
        <script src="https://kit.fontawesome.com/5f3f547070.js" crossorigin="anonymous"></script
        <link href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap" rel="styleshe
    </head>
        <!-- Website Title -->
        <!--<div style="padding-left:200px">
        <div class="container">
             <b><h2 class='container-heading'><span class="heading_font">Digit Recognizer</b></s
            <div class='description'>
                <p>A Machine Learning Web App using Flask.</p>
            </div>
        </div>-->

        <!-- Result -->
        <div class="results">
                <p style="padding-top: 15px;">Recognized digit is : <b><u>{{showcase}}</p>

        </div>
        <image src="../static/samosa3.png" style="padding-left: 35px;margin-top: 30px;">

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

Fig 14: index3.html

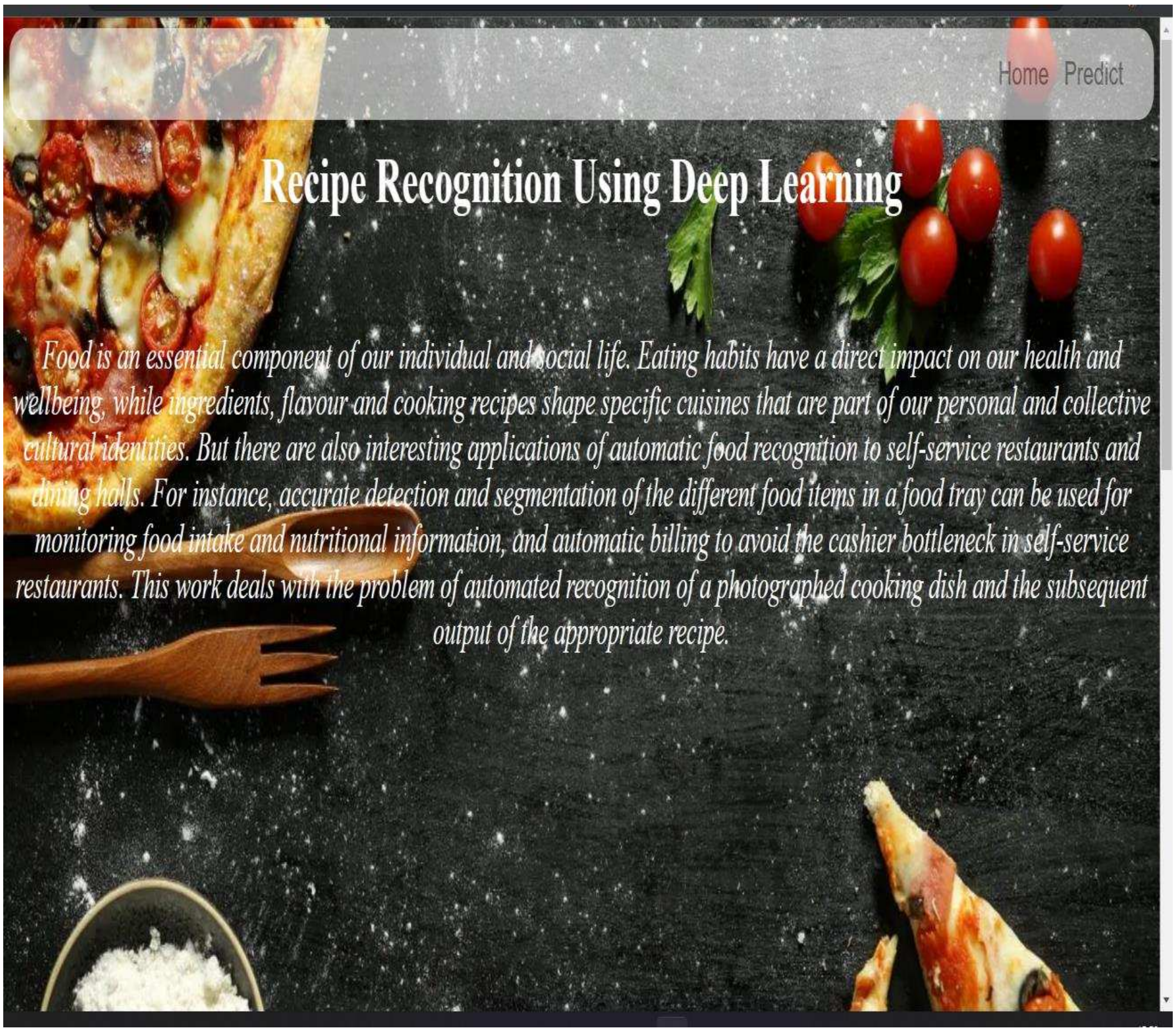


```html



{% extends "RRprediction2.html" %} {% block content %}
<div style="float:left">
<h2><font color="red" size="15" font-family="sans-serif"><b>FOOD Recognition</b></font></h2><br>

<div>
    <form id="upload-file" method="post" enctype="multipart/form-data">
       <label for="imageUpload" class="upload-label">
            Choose...
        </label>
        <input type="file" name="file" id="imageUpload" accept=".png, .jpg, .jpeg">
    </form>

   <center> <div class="image-section" style="display:none;">
        <div class="img-preview">
            <div id="imagePreview">
            </div></center>
        </div>
        <center><div>
            <button type="button" class="btn btn-primary btn-lg " id="btn-predict">Recognize</bu
       </center></div>
    </div>

    <div class="loader" style="display:none;margin-left: 450px;"></div>

    <h3 id="result">
        <span> </span>
    </h3>

</div>
</div>



{% endblock %}
```

Fig 15: prediction.html

8.CONCLUSION:

The Following are the steps listed above are performed by our team, and herewith we attach snaps of our web page we attached.



Recipe Recognition Using Deep Learning

Food is an essential component of our individual and social life. Eating habits have a direct impact on our health and wellbeing, while ingredients, flavour and cooking recipes shape specific cuisines that are part of our personal and collective cultural identities. But there are also interesting applications of automatic food recognition to self-service restaurants and dining halls. For instance, accurate detection and segmentation of the different food items in a food tray can be used for monitoring food intake and nutritional information, and automatic billing to avoid the cashier bottleneck in self-service restaurants. This work deals with the problem of automated recognition of a photographed cooking dish and the subsequent output of the appropriate recipe.

Fig 16: Application Home Page

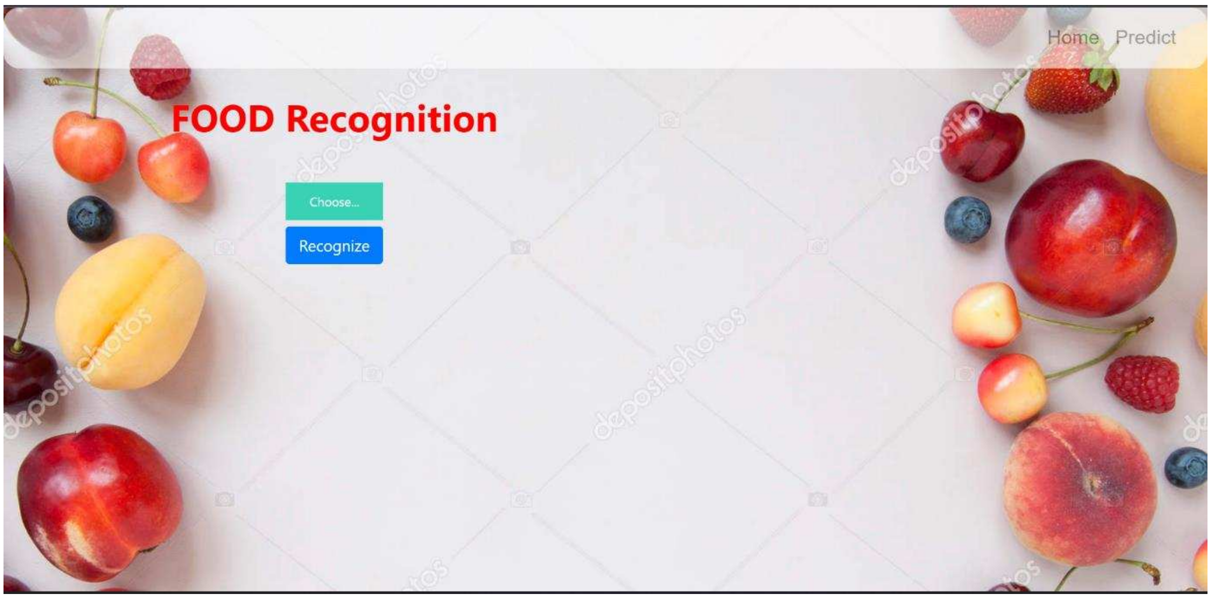
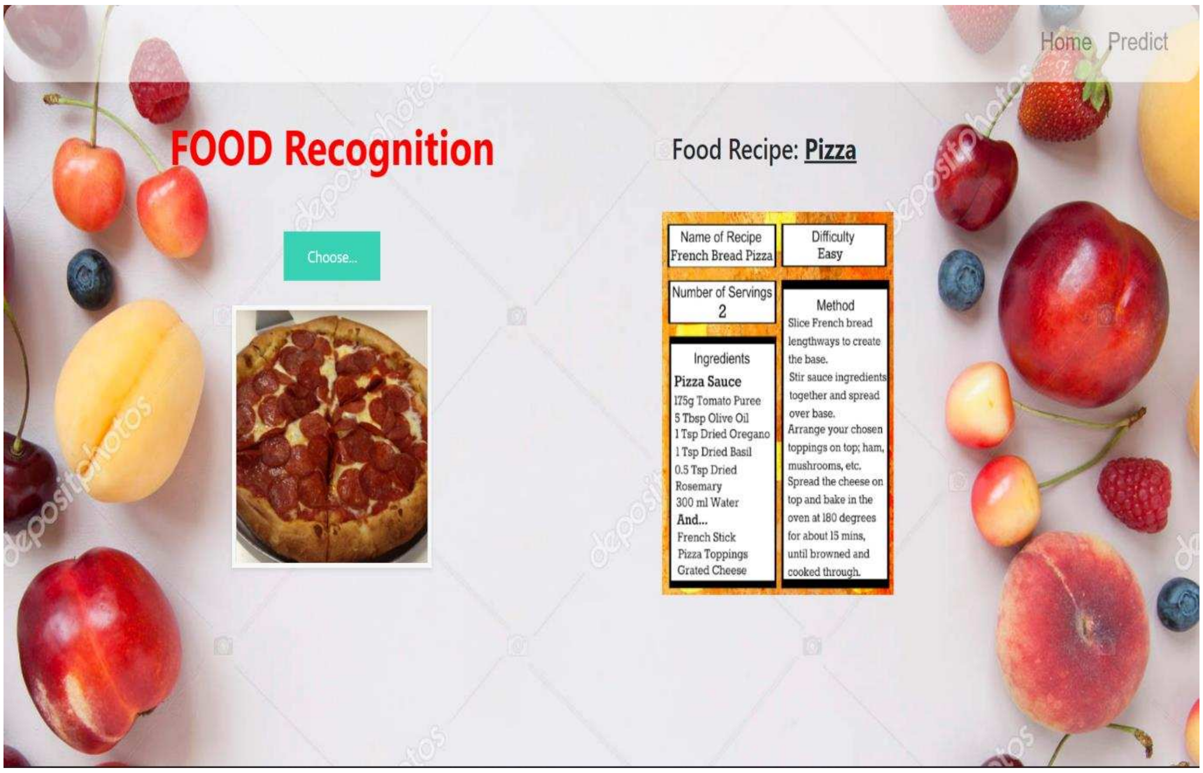


Fig 17: Application Prediction page



Food Recipe: Pizza

Name of Recipe
French Bread Pizza

Difficulty
Easy

Number of Servings
2

Ingredients
Pizza Sauce
175g Tomato Puree
5 Tbsp Olive Oil
1 Tsp Dried Oregano
1 Tsp Dried Basil
0.5 Tsp Dried Rosemary
300 ml Water
And...
French Stick
Pizza Toppings
Grated Cheese

Method
Slice French bread lengthways to create the base.
Stir sauce ingredients together and spread over base.
Arrange your chosen toppings on top; ham, mushrooms, etc.
Spread the cheese on top and bake in the oven at 180 degrees for about 15 mins, until browned and cooked through.

Fig 18: output

9.APPLICATIONS

- Applications of automatic food recognition to self-service restaurants and dining halls.
- For instance, accurate detection and segmentation of the different food items in a food tray can be used for monitoring food intake and nutritional information, and automatic billing to avoid the cashier bottleneck in self-service restaurants.

10.ADVANTAGES

- Food image classification is an emerging research field due to its increasing benefits in health and medical sectors.
- Automated food recognition tools will help in developing diet monitoring systems, calories estimation

11.DISADVANTAGES

- Sometimes people may not be able to eat what they like the most.
- Taking too much out of our diet can be unhealthy.

12.FUTURE SCOPE

- in the future automated food recognition tools will help in developing diet monitoring systems, calories estimation.
- accurate detection and segmentation of the different food items in a food tray can be used for monitoring food intake and nutritional information, and automatic billing to avoid the cashier bottleneck in self-service restaurants.

13.BIBLOGRAPHY

- https://smartinternz.com/Student/guided_projects
- https://www.kaggle.com/paultimothymooney/breast-histopathology-images

HELP FILE

PROJECT EXECUTION:

STEP-1: Go to Start, search and launch **ANACONDA NAVIGATOR**.

STEP-2: After launching of **ANACONDA NAVIGATOR**, launch **JUPYTER NOTEBOOK**, open "food recipe" IPYNB file then run all the cells, a .h5 file will be generated.

STEP-3: Create a Folder named **FLASK** on the **DESKTOP**. Extract the food.h5 file into this Flask Folder.

STEP-4: Extract all the html files (index.html) and python file(app.py) into the FLASK Folder.

STEP-5: Then go back to ANACONDA NAVIGATOR and the launch the SPYDER.

STEP-6: After launching Spyder, give the path of FLASK FOLDER which you have created on the DESKTOP.

STEP-7: Open all the app.py and html files present in the Flask Folder.

STEP-8: After running of the app.py, open ANACONDA PROMPT and follow the below steps:
cd File Path→click enter python app.py→click enter (We could see running of files).

STEP-9: Then open BROWSER, at the URL area type "localhost:5000".

STEP-10: Home page of the project will be displayed.

STEP-11: In the home page of the project click on choose file, then select an image file from the dataset.

STEP-12: Now click on "click to predict" button below the uploaded file, the Recipe of the uploaded food image is displayed on screen.