Build Python Code (Part 1)

Team ID: PNT2022TMID28890
Project Name: A Novel Method For Handwritten Digit Recognition System
Team Leader:Karthikeyan.V
Team Member:Karthikeyan.J, Periyasamy.A, Sabarimanikandan.M

Let us build the flask file 'app.py' which is a web framework written in python for server-side scripting. Let's see step by step procedure for building the backend application.

- App starts running when the "__name__" constructor is called in main.
- render_template is used to return HTML file.
- "GET" method is used to take input from the user.
- "POST" method is used to display the output to the user.

Import Libraries:

```
from flask import Flask, render_template, request# Flask-It is our framework which we are going to use to
run/serve our application.
#request-for accessing file which was uploaded by the user on our application.

from PIL import Image #used for manipulating image uploaded by the user.
import numpy as np #used for numerical analysis
from tensorflow.keras.models import load_model#to load our model trained with MNIST data
import tensorflow as tf#to run our model.
```

Libraries required for the app to run are to be imported.

Routing to the html Page

```
@app.route('/') #default route
def upload_file():
    return render_template('main.html') #rendering html page
@app.route('/about') #Main page route
def upload_file1():
    return render_template('main.html') #rendering html page
@app.route('/upload') #main page route
def upload_file2():
    return render_template('index6.html')
```

We are routing the app to the HTML templates which we want to render.
Firstly we are rendering the main.html template and from there we are navigating to our prediction page that is index6.html

Returning the prediction on UI:

```
@app.route('/predict', methods = ['POST']) #route for our prediction
def upload_image_file():
    if request.method == 'POST':
        img = Image.open(request.files['file'].stream).convert("L")# convert image to monochrome
        img = img.resize((28,28))# resizing of input image
        im2arr = np.array(img)#converting to image
        im2arr = im2arr.reshape(1,28,28,1) #reshaping according to our requirement
        y_pred = model.predict_classes(im2arr) #predicting the results
        print(y_pred) #printing our result in prompt
        #return 'Predicted Number: ' + str(y_pred) #returning our output
```