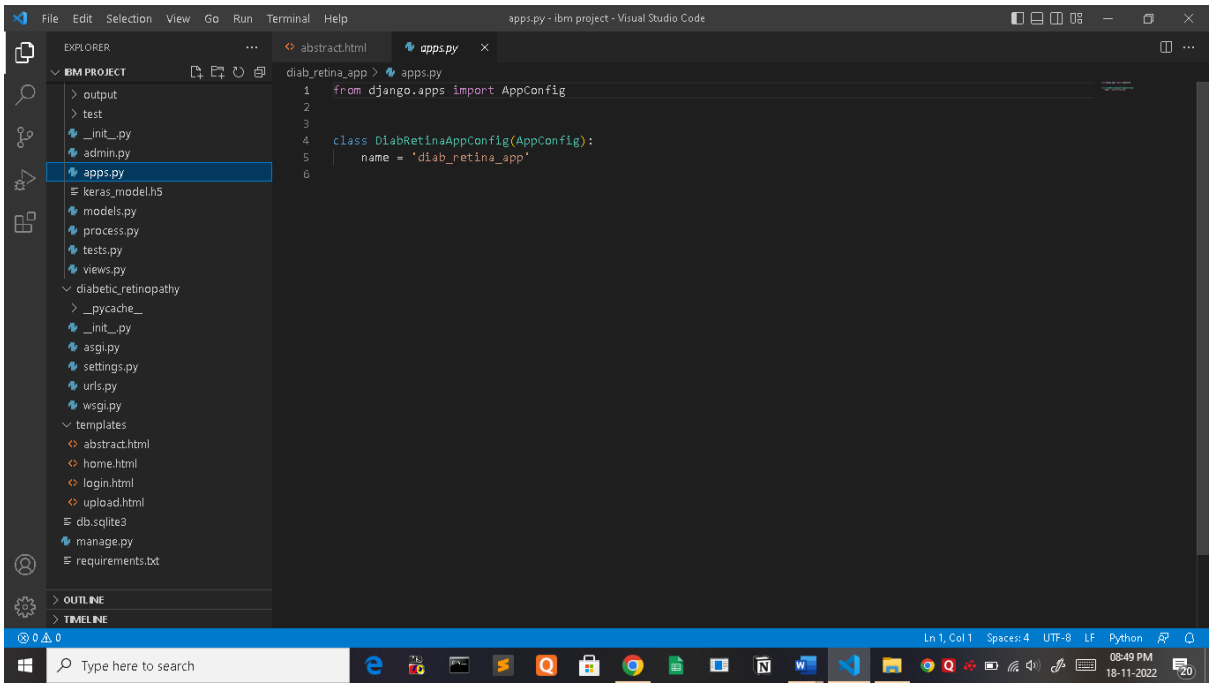
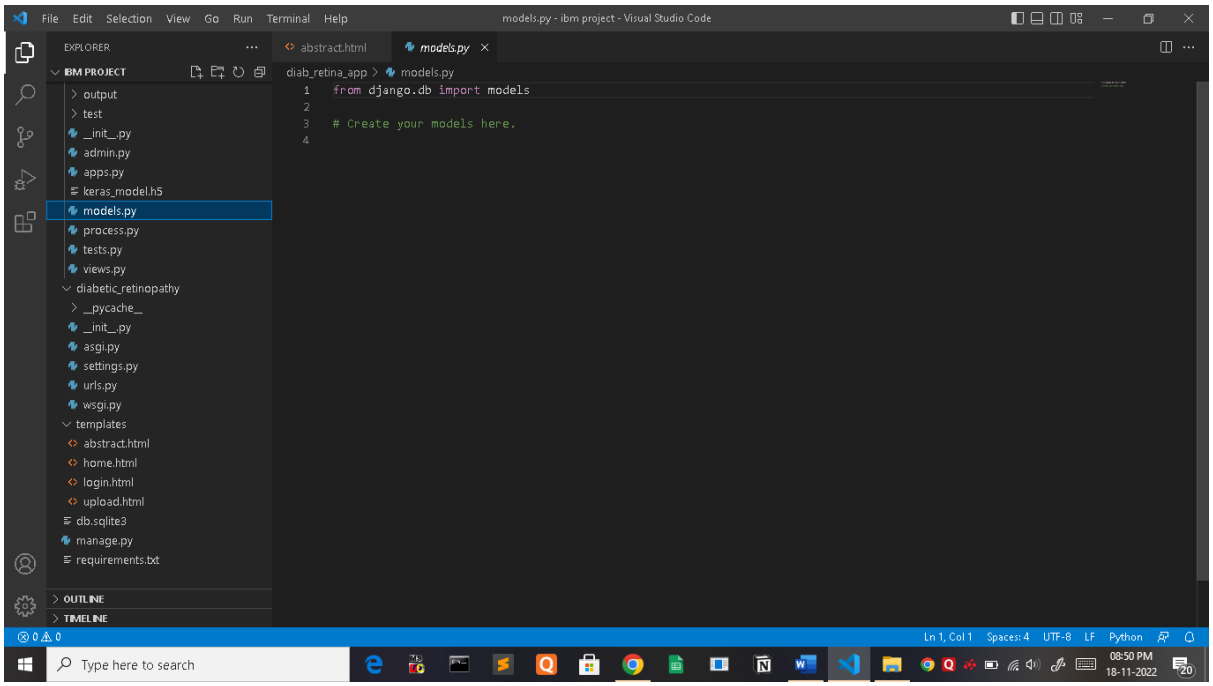
APPLICATION (PYTHON CODE)



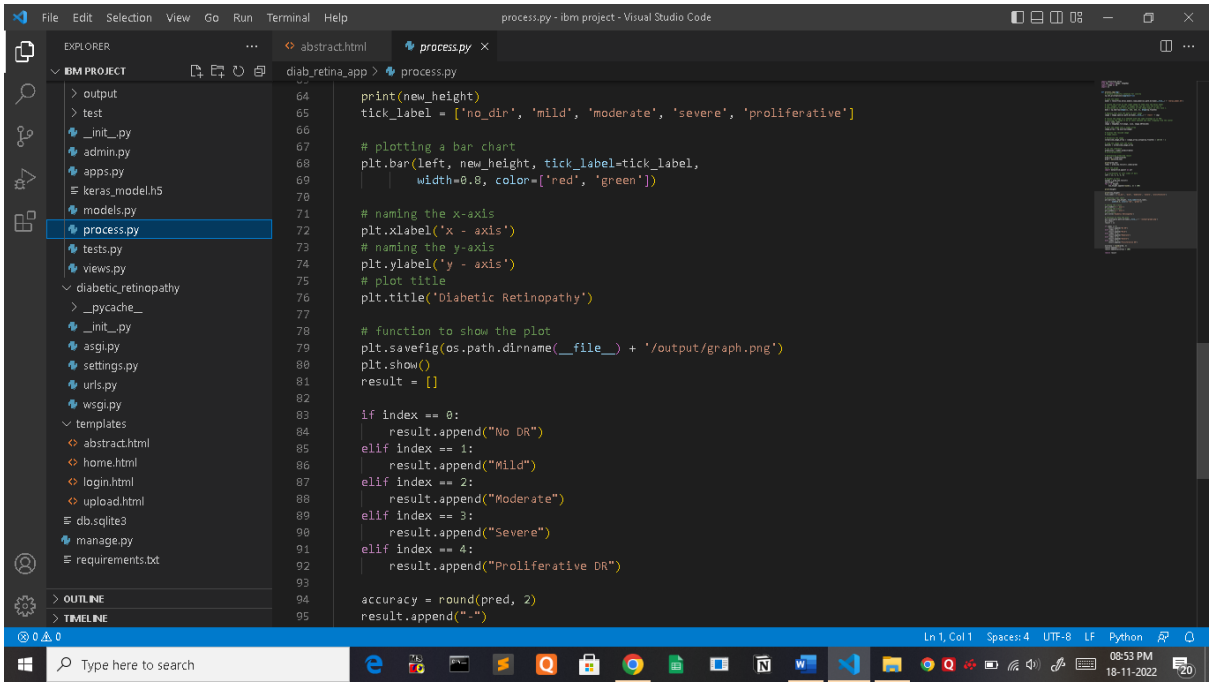
MODEL(PYTHON CODE)



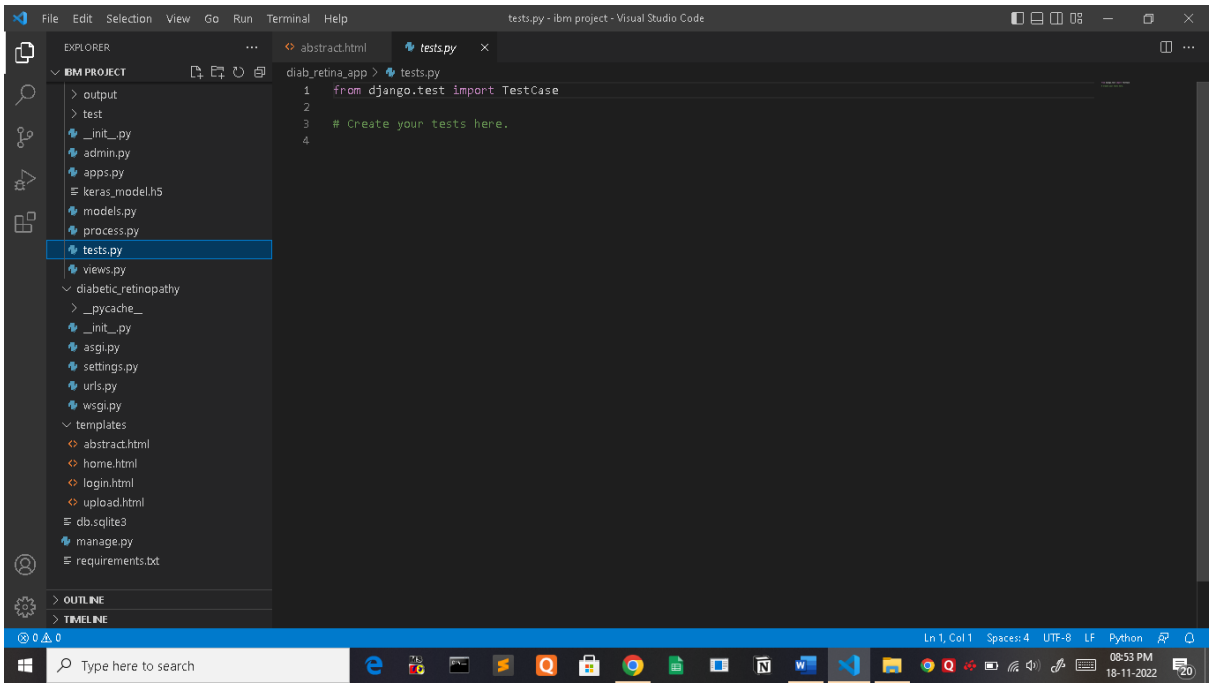
PROCESS(PYTHON CODE)

```python
import tensorflow.keras
from PIL import Image, ImageOps
import numpy as np
import os


def process_img(img):
    # Disable scientific notation for clarity
    np.set_printoptions(suppress=True)

    # Load the model
    model = tensorflow.keras.models.load_model(os.path.dirname(__file__) + '/keras_model.h5')

    # Create the array of the right shape to feed into the keras model
    # The 'length' or number of images you can put into the array is
    # determined by the first position in the shape tuple, in this case 1.
    data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)

    # Replace this with the path to your image
    image = Image.open(os.path.dirname(__file__) + '/test/' + img)

    # resize the image to a 224x224 with the same strategy as in TM2:
    # resizing the image to be at least 224x224 and then cropping from the center
    size = (224, 224)
    image = ImageOps.fit(image, size, Image.ANTIALIAS)

    # turn the image into a numpy array
    image_array = np.asarray(image)

    # display the resized image
    # image.show()

    # Normalize the image
```

```python
    # Normalize the image
    normalized_image_array = (image_array.astype(np.float32) / 127.0) - 1

    # Load the image into the array
    data[0] = normalized_image_array

    # run the inference
    prediction = model.predict(data)
    # print(prediction)

    # determining predicted result
    pred_new = prediction[0]
    pred = max(pred_new)

    print(pred_new)
    index = pred_new.tolist().index(pred)

    #plot the graph
    import matplotlib.pyplot as plt

    # x-coordinates of left sides of bars
    left = [1, 2, 3, 4, 5]

    # heights of bars
    height = pred_new.tolist()
    new_height = []
    for i in height:
        new_height.append(round(i, 2) * 100)

    print(height)

    print(new_height)
```



TEST(PYTHON CODE)



VIEWS(PYTHON CODE)