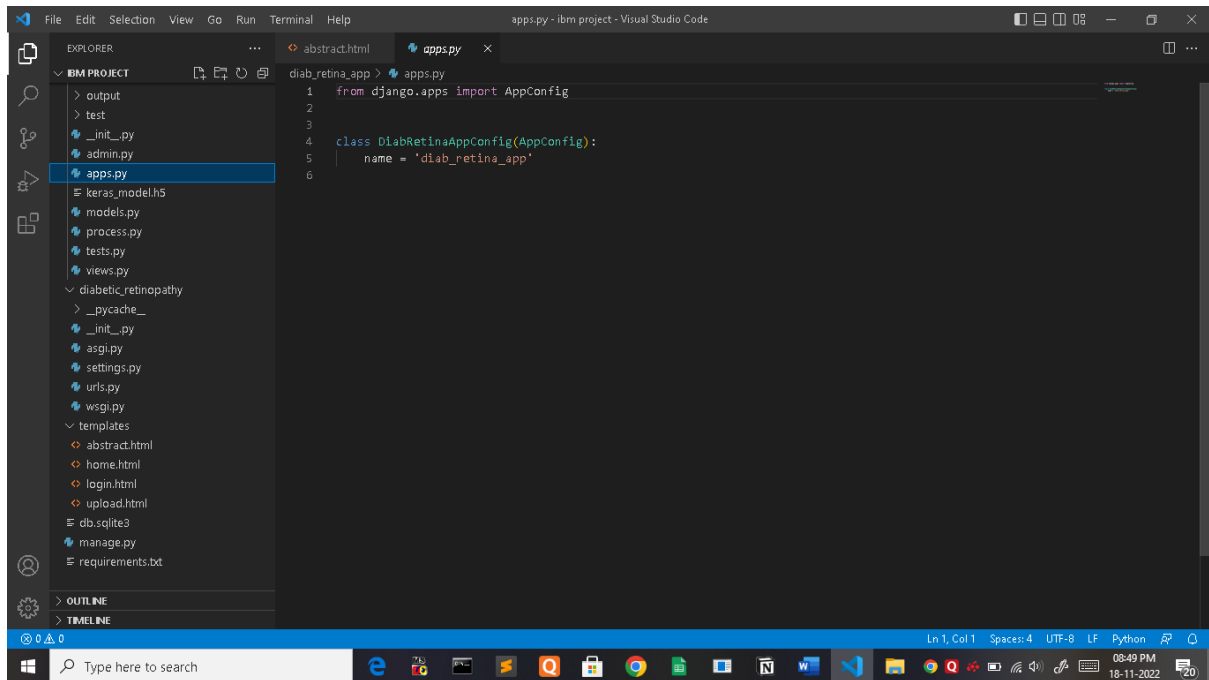


APPLICATION (PYTHON CODE)

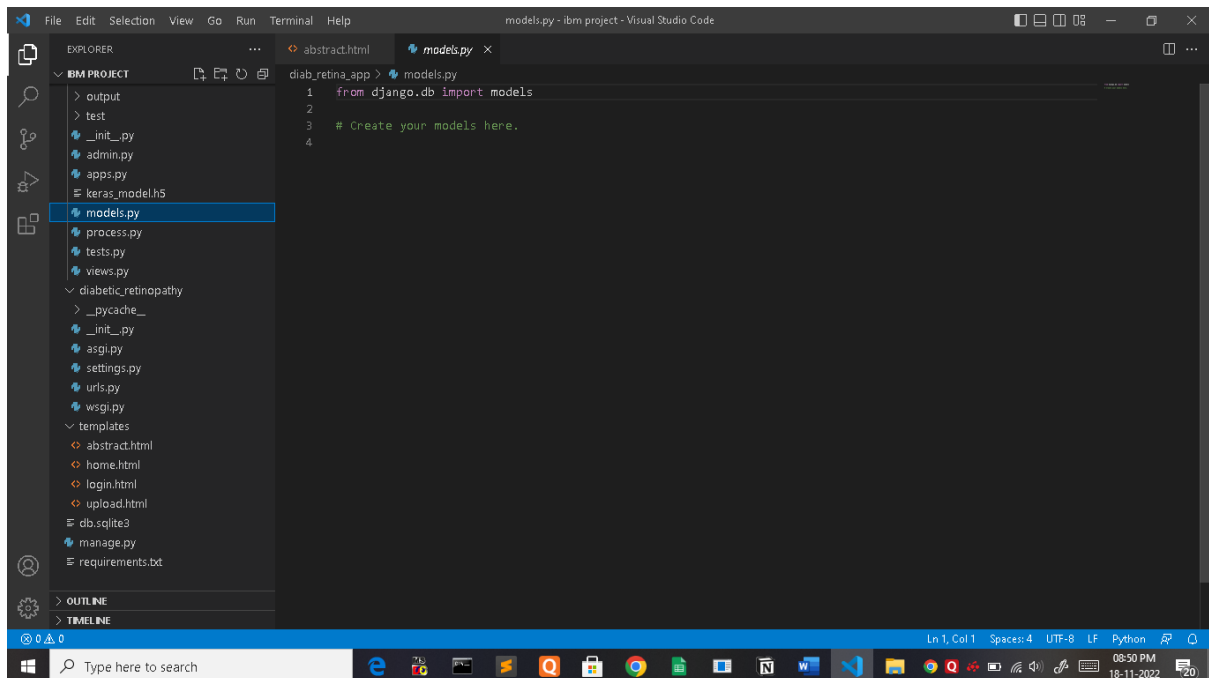


The screenshot shows the Visual Studio Code editor with the Django project 'diab_retina_app' open. The Explorer sidebar on the left shows the project structure, with 'apps.py' selected under the 'diabetic_retinopathy' app. The main editor window displays the content of 'apps.py'.

```
1 from django.apps import AppConfig
2
3
4 class DiabRetinaAppConfig(AppConfig):
5     name = 'diab_retina_app'
6
```

The status bar at the bottom indicates the file is at Line 1, Column 1, using UTF-8 encoding and LF line endings, with the Python interpreter selected.

MODEL(PYTHON CODE)

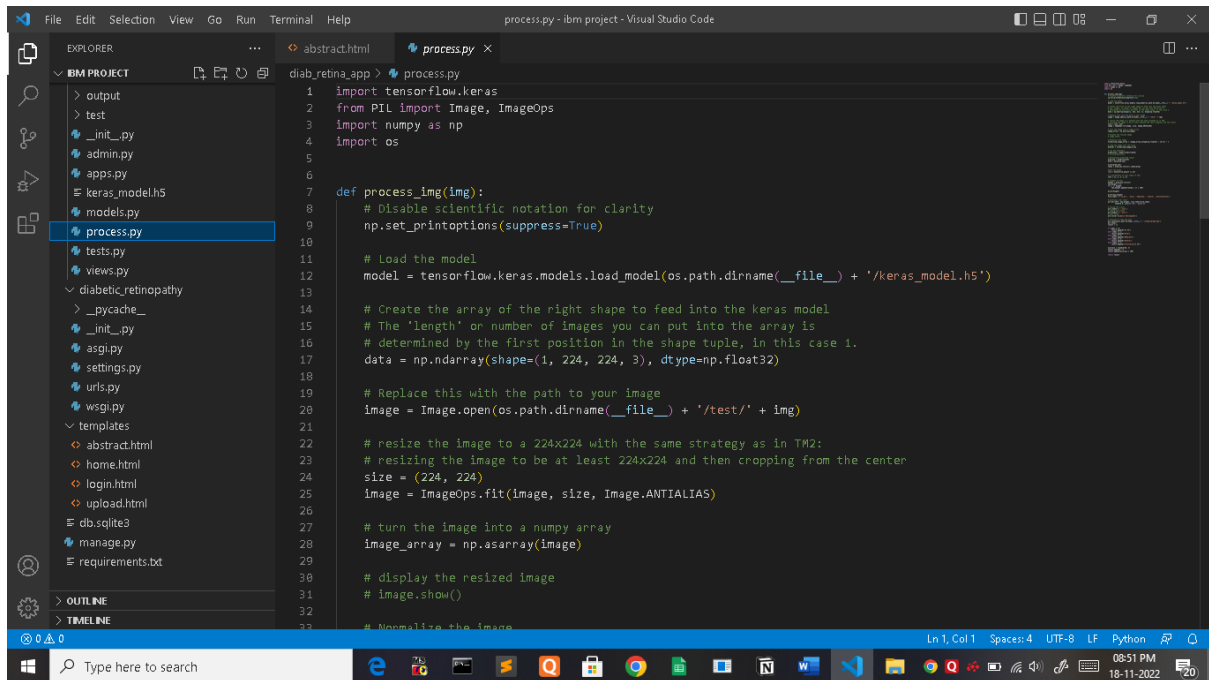


The screenshot shows the Visual Studio Code editor with the Django project 'diab_retina_app' open. The Explorer sidebar on the left shows the project structure, with 'models.py' selected under the 'diabetic_retinopathy' app. The main editor window displays the content of 'models.py'.

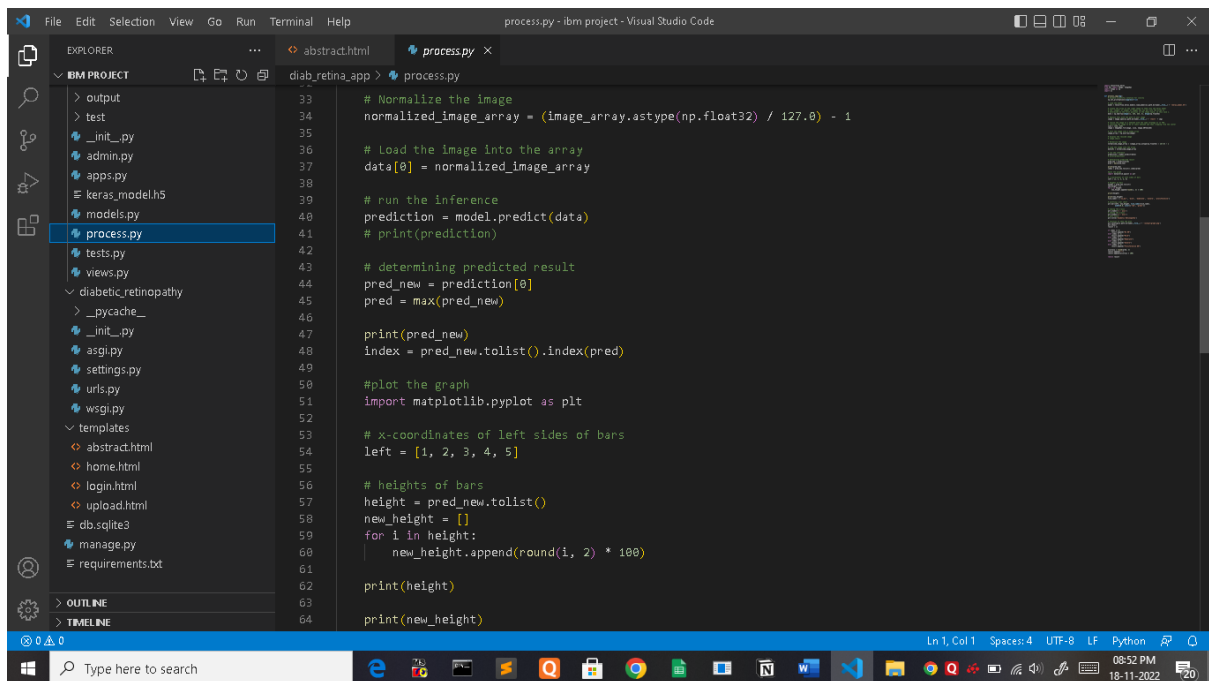
```
1 from django.db import models
2
3 # Create your models here.
4
```

The status bar at the bottom indicates the file is at Line 1, Column 1, using UTF-8 encoding and LF line endings, with the Python interpreter selected.

PROCESS(PYTHON CODE)



```
1 import tensorflow.keras
2 from PIL import Image, ImageOps
3 import numpy as np
4 import os
5
6
7 def process_img(img):
8     # Disable scientific notation for clarity
9     np.set_printoptions(suppress=True)
10
11     # Load the model
12     model = tensorflow.keras.models.load_model(os.path.dirname(__file__) + '/keras_model.h5')
13
14     # Create the array of the right shape to feed into the keras model
15     # The 'length' or number of images you can put into the array is
16     # determined by the first position in the shape tuple, in this case 1.
17     data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
18
19     # Replace this with the path to your image
20     image = Image.open(os.path.dirname(__file__) + '/test/' + img)
21
22     # resize the image to a 224x224 with the same strategy as in TM2:
23     # resizing the image to be at least 224x224 and then cropping from the center
24     size = (224, 224)
25     image = ImageOps.fit(image, size, Image.ANTIALIAS)
26
27     # turn the image into a numpy array
28     image_array = np.asarray(image)
29
30     # display the resized image
31     # image.show()
32
33     # Normalize the image
```



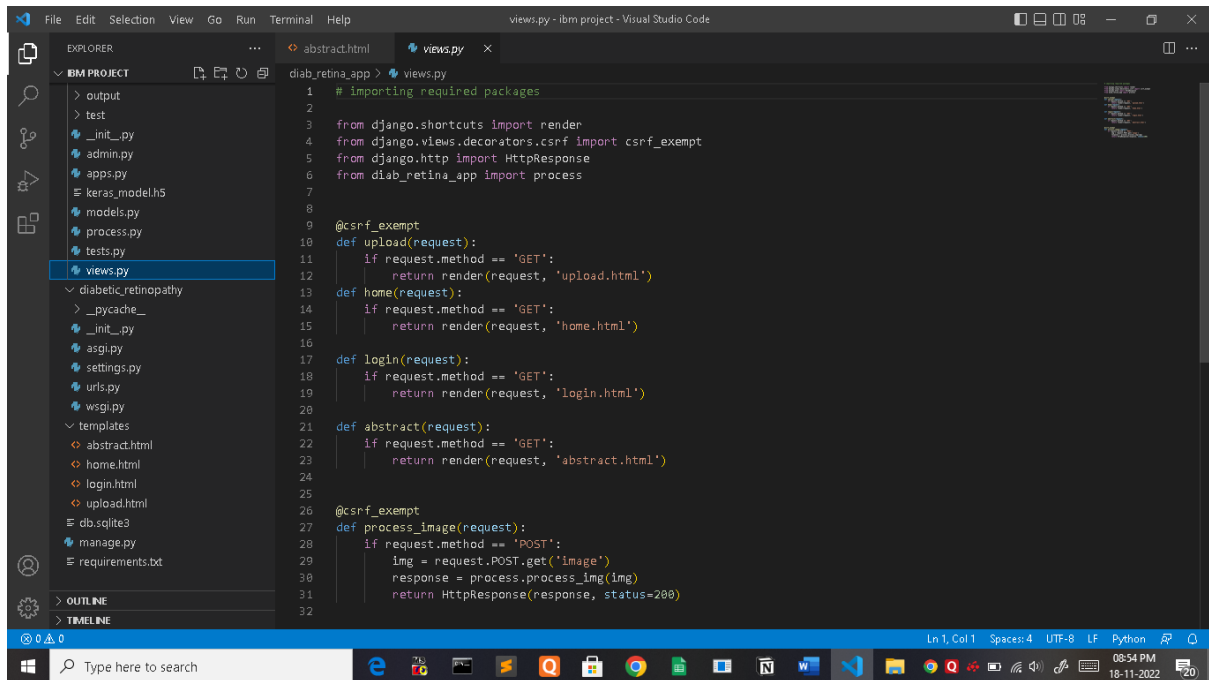
```
33 # Normalize the image
34 normalized_image_array = (image_array.astype(np.float32) / 127.0) - 1
35
36 # Load the image into the array
37 data[0] = normalized_image_array
38
39 # run the inference
40 prediction = model.predict(data)
41 # print(prediction)
42
43 # determining predicted result
44 pred_new = prediction[0]
45 pred = max(pred_new)
46
47 print(pred_new)
48 index = pred_new.tolist().index(pred)
49
50 #plot the graph
51 import matplotlib.pyplot as plt
52
53 # x-coordinates of left sides of bars
54 left = [1, 2, 3, 4, 5]
55
56 # heights of bars
57 height = pred_new.tolist()
58 new_height = []
59 for i in height:
60     new_height.append(round(i, 2) * 100)
61
62 print(height)
63
64 print(new_height)
```

```
process.py - ibm project - Visual Studio Code
diab_retina_app > process.py
64 print(new_height)
65 tick_label = ['no_dir', 'mild', 'moderate', 'severe', 'proliferative']
66
67 # plotting a bar chart
68 plt.bar(left, new_height, tick_label=tick_label,
69        width=0.8, color=['red', 'green'])
70
71 # naming the x-axis
72 plt.xlabel('x - axis')
73 # naming the y-axis
74 plt.ylabel('y - axis')
75 # plot title
76 plt.title('Diabetic Retinopathy')
77
78 # function to show the plot
79 plt.savefig(os.path.dirname(__file__) + '/output/graph.png')
80 plt.show()
81 result = []
82
83 if index == 0:
84     result.append("No DR")
85 elif index == 1:
86     result.append("Mild")
87 elif index == 2:
88     result.append("Moderate")
89 elif index == 3:
90     result.append("Severe")
91 elif index == 4:
92     result.append("Proliferative DR")
93
94 accuracy = round(pred, 2)
95 result.append("-")
```

TEST(PYTHON CODE)

```
tests.py - ibm project - Visual Studio Code
diab_retina_app > tests.py
1 from django.test import TestCase
2
3 # Create your tests here.
4
```

VIEWS(PYTHON CODE)



The screenshot displays the Visual Studio Code interface with a project named 'ibm project'. The Explorer sidebar on the left shows the project structure, including files like output, test, __init__.py, admin.py, apps.py, keras_model.hs, models.py, process.py, tests.py, views.py, diabetic_retinopathy, __pycache__, __init__.py, asgi.py, settings.py, urls.py, wsgi.py, templates, abstract.html, home.html, login.html, upload.html, db.sqlite3, manage.py, and requirements.txt. The views.py file is selected and open in the main editor. The code in views.py is as follows:

```
1 # importing required packages
2
3 from django.shortcuts import render
4 from django.views.decorators.csrf import csrf_exempt
5 from django.http import HttpResponseRedirect
6 from diab_retina_app import process
7
8
9 @csrf_exempt
10 def upload(request):
11     if request.method == 'GET':
12         return render(request, 'upload.html')
13
14 def home(request):
15     if request.method == 'GET':
16         return render(request, 'home.html')
17
18 def login(request):
19     if request.method == 'GET':
20         return render(request, 'login.html')
21
22 def abstract(request):
23     if request.method == 'GET':
24         return render(request, 'abstract.html')
25
26
27 @csrf_exempt
28 def process_image(request):
29     if request.method == 'POST':
30         img = request.POST.get('image')
31         response = process.process_img(img)
32         return HttpResponseRedirect(response, status=200)
```

The status bar at the bottom indicates the current cursor position is at Line 1, Column 1, with 4 spaces, in UTF-8 encoding, using LF line endings, and the file is a Python script. The system clock shows 08:54 PM on 18-11-2022.