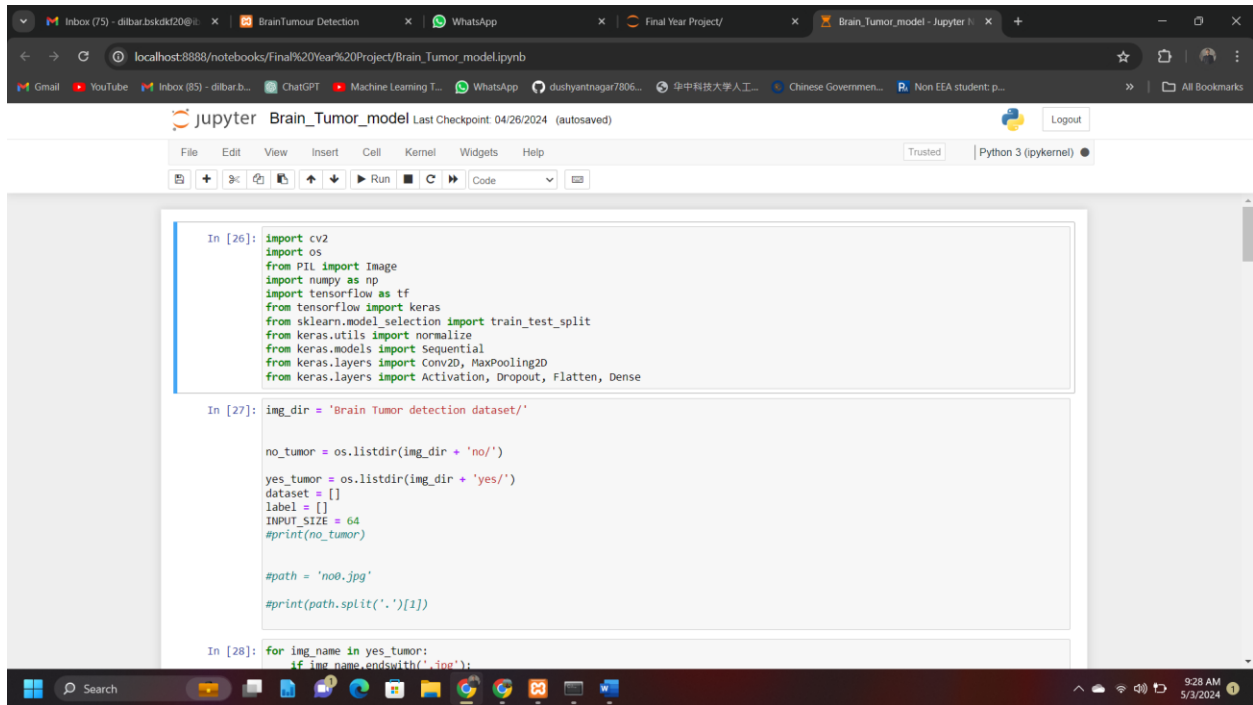


Code



```
In [26]: import cv2
import os
from PIL import Image
import numpy as np
import tensorflow as tf
from tensorflow import keras
from sklearn.model_selection import train_test_split
from keras.utils import normalize
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Activation, Dropout, Flatten, Dense

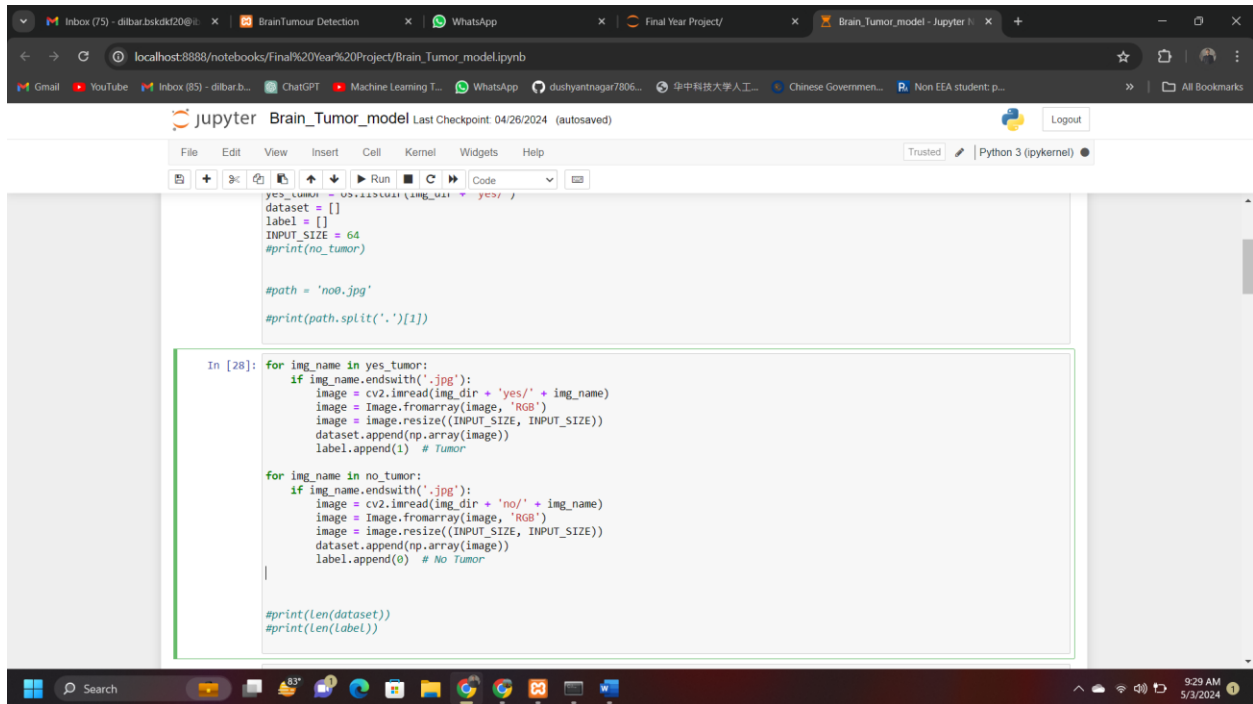
In [27]: img_dir = 'Brain Tumor detection dataset/'

no_tumor = os.listdir(img_dir + 'no/')

yes_tumor = os.listdir(img_dir + 'yes/')
dataset = []
label = []
INPUT_SIZE = 64
#print(no_tumor)

#path = 'no0.jpg'
#print(path.split('.')[1])

In [28]: for img_name in yes_tumor:
if img_name.endswith('.jpg'):
```



```
yes_tumor = os.listdir(img_dir + 'yes/')
dataset = []
label = []
INPUT_SIZE = 64
#print(no_tumor)

#path = 'no0.jpg'
#print(path.split('.')[1])

In [28]: for img_name in yes_tumor:
if img_name.endswith('.jpg'):
image = cv2.imread(img_dir + 'yes/' + img_name)
image = Image.fromarray(image, 'RGB')
image = image.resize((INPUT_SIZE, INPUT_SIZE))
dataset.append(np.array(image))
label.append(1) # Tumor

for img_name in no_tumor:
if img_name.endswith('.jpg'):
image = cv2.imread(img_dir + 'no/' + img_name)
image = Image.fromarray(image, 'RGB')
image = image.resize((INPUT_SIZE, INPUT_SIZE))
dataset.append(np.array(image))
label.append(0) # No Tumor

#print(len(dataset))
#print(len(label))
```

```

# Check the number of samples in x_train and y_train
print('Number of samples in x_train:', len(x_train))
print('Number of samples in y_train:', len(y_train))
print('Number of samples in x_test:', len(x_test))
print('Number of samples in y_test:', len(y_test))

Number of samples in x_train: 2400
Number of samples in y_train: 2400
Number of samples in x_test: 600
Number of samples in y_test: 600

In [31]: #Building a Model

model = Sequential()

model.add(Conv2D(32, (3,3), input_shape=(INPUT_SIZE, INPUT_SIZE, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(32, (3,3), kernel_initializer='he_uniform'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64, (3,3), kernel_initializer='he_uniform'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(1))

```

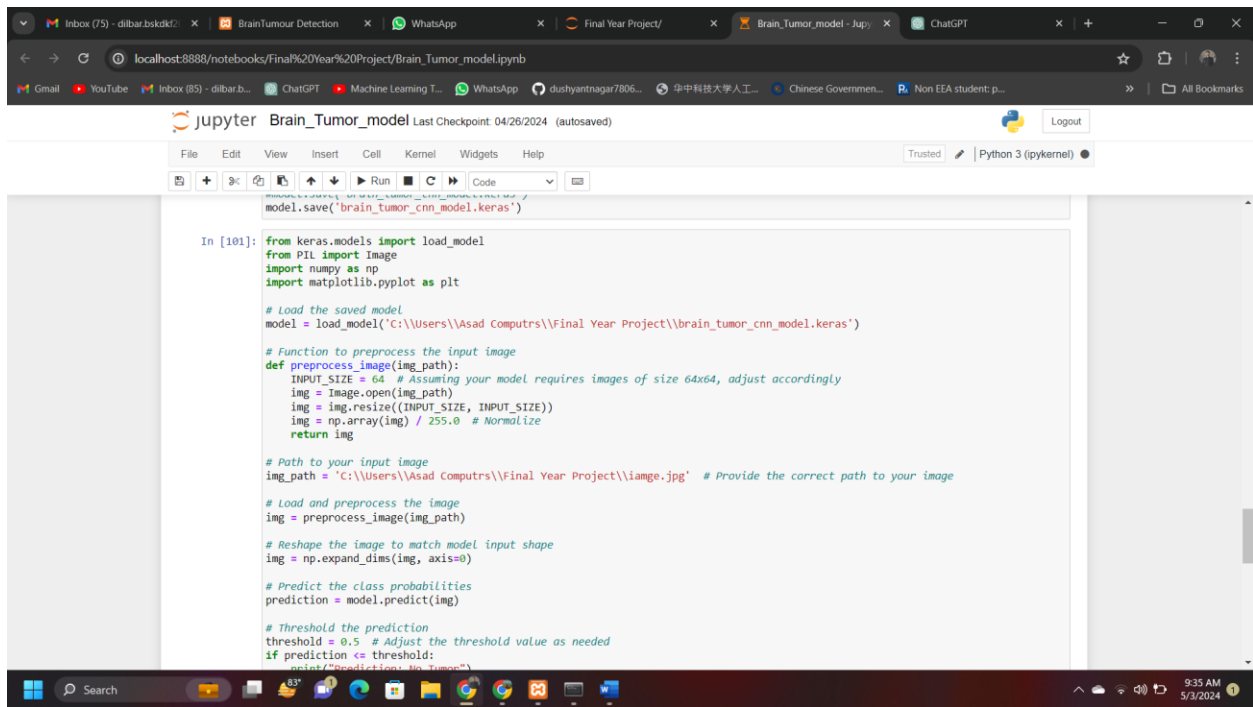
```

model.compile(loss = 'binary_crossentropy', optimizer='adam', metrics=['accuracy'])

In [33]: model.fit(x_train, y_train, batch_size=32, epochs=10, validation_data=(x_test, y_test), shuffle=False)

Epoch 1/10
75/75 [=====] - 7s 84ms/step - loss: 0.6425 - accuracy: 0.6100 - val_loss: 0.5938 - val_accuracy: 0.66
83
Epoch 2/10
75/75 [=====] - 6s 79ms/step - loss: 0.5762 - accuracy: 0.6787 - val_loss: 0.5567 - val_accuracy: 0.67
00
Epoch 3/10
75/75 [=====] - 6s 80ms/step - loss: 0.5110 - accuracy: 0.7133 - val_loss: 0.5041 - val_accuracy: 0.74
00
Epoch 4/10
75/75 [=====] - 6s 80ms/step - loss: 0.4779 - accuracy: 0.7379 - val_loss: 0.5002 - val_accuracy: 0.72
17
Epoch 5/10
75/75 [=====] - 6s 82ms/step - loss: 0.4411 - accuracy: 0.7471 - val_loss: 0.4552 - val_accuracy: 0.72
67
Epoch 6/10
75/75 [=====] - 6s 84ms/step - loss: 0.4085 - accuracy: 0.7667 - val_loss: 0.4387 - val_accuracy: 0.74
83
Epoch 7/10
75/75 [=====] - 7s 88ms/step - loss: 0.3848 - accuracy: 0.7792 - val_loss: 0.4554 - val_accuracy: 0.75
50
Epoch 8/10
75/75 [=====] - 6s 85ms/step - loss: 0.3580 - accuracy: 0.8004 - val_loss: 0.4492 - val_accuracy: 0.78
00
Epoch 9/10
75/75 [=====] - 6s 86ms/step - loss: 0.3303 - accuracy: 0.8112 - val_loss: 0.4404 - val_accuracy: 0.75
83
Epoch 10/10
75/75 [=====] - 6s 85ms/step - loss: 0.3153 - accuracy: 0.8292 - val_loss: 0.4651 - val_accuracy: 0.77

```



```
model.save('brain_tumor_cnn_model.keras')

In [101]: from keras.models import load_model
          from PIL import Image
          import numpy as np
          import matplotlib.pyplot as plt

          # Load the saved model
          model = load_model('c:\Users\Asad Computers\Final Year Project\brain_tumor_cnn_model.keras')

          # Function to preprocess the input image
          def preprocess_image(img_path):
              INPUT_SIZE = 64 # Assuming your model requires images of size 64x64, adjust accordingly
              img = Image.open(img_path)
              img = img.resize((INPUT_SIZE, INPUT_SIZE))
              img = np.array(img) / 255.0 # Normalize
              return img

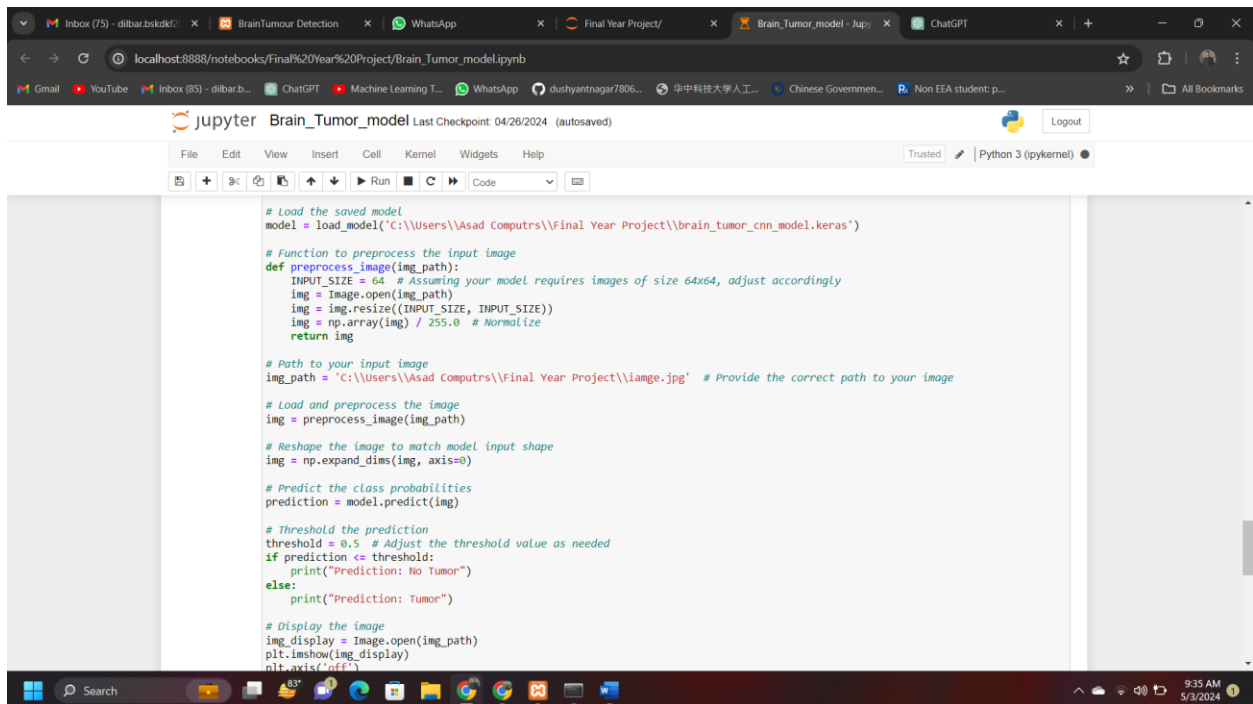
          # Path to your input image
          img_path = 'c:\Users\Asad Computers\Final Year Project\image.jpg' # Provide the correct path to your image

          # Load and preprocess the image
          img = preprocess_image(img_path)

          # Reshape the image to match model input shape
          img = np.expand_dims(img, axis=0)

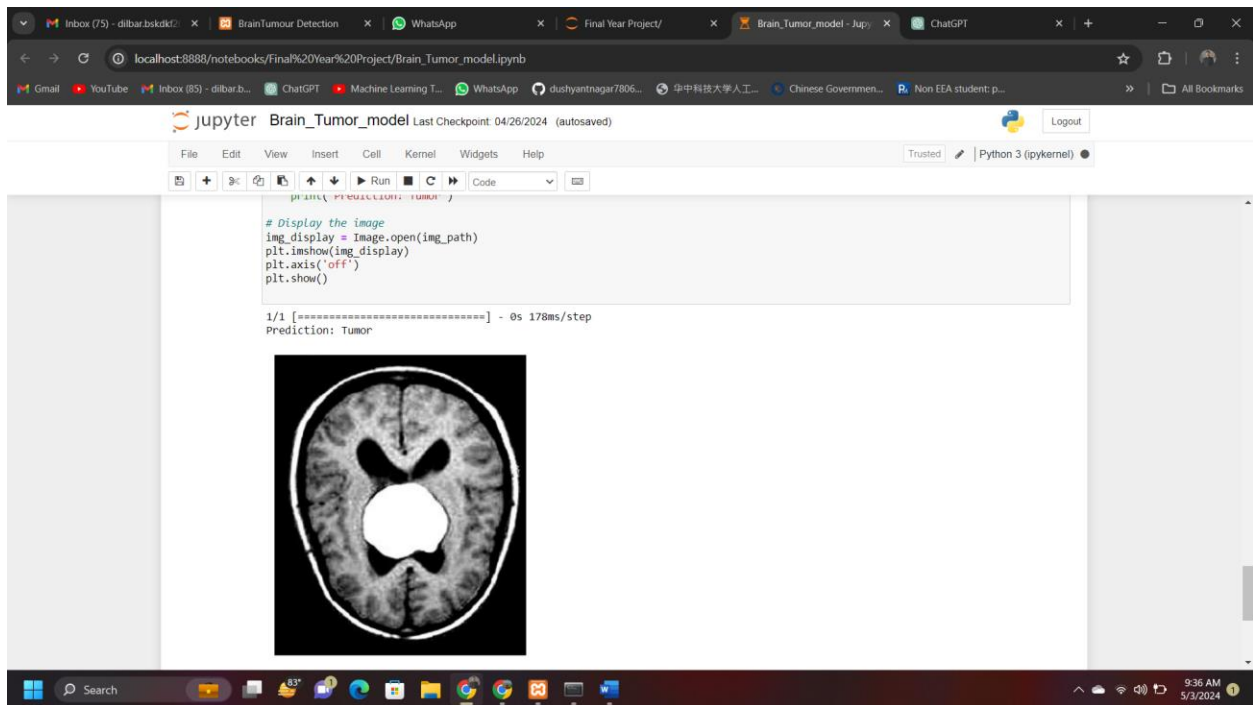
          # Predict the class probabilities
          prediction = model.predict(img)

          # Threshold the prediction
          threshold = 0.5 # Adjust the threshold value as needed
          if prediction <= threshold:
              print("Prediction: No Tumor")
```



```
          # Threshold the prediction
          threshold = 0.5 # Adjust the threshold value as needed
          if prediction <= threshold:
              print("Prediction: No Tumor")
          else:
              print("Prediction: Tumor")

          # Display the image
          img_display = Image.open(img_path)
          plt.imshow(img_display)
          plt.axis('off')
```



Website

