CHAPTER 1

1.1 Sodh Yatra

The Project basically focuses on image recognition based on its content i.e. to bifurcate images and store them in the relevant folder.

Image is recognized based on its colour, type, shape (natural, automobiles, watches, animals, books) etc.

There are often certain images whose type is not known to the user, and thus the need for recognizing the image increases.

1.2 Introduction

This Project involves the concept of Image Recognition i.e. the main aim is to bifurcate the user's photos based on the content and store them in the relevant folder.

CHAPTER 2: Requirement Gathering

2.1 Functional Requirements

- The System must be able to:

1. Break down an image into a set of features.
2. Flag images upon uploading that are identical or very similar to existing images in the dataset.
3. Find images that have a similar set of features with the searched image.
4. Return whether the image is an NSFW content or not.

2.2 Non- Functional Requirements

1. Response time: When querying for similar images it is extremely important that the results are returned in real time and that the implementation scales to a large collection of images.
2. Robustness: The system must be invariant to input image rotation, intensity variation, size of the input image.
3. Web Interface: The web interface elements will be easy to understand and the user will be provided with the tags of the related images using which, user can search similar images.
4. Reliability: The reliability of the web interface has also a crucial importance. In case the user needs help, web interface should be 100% percent available at least 165 hours per week.
5. Performance: Web server should be able to handle multiple device and user connection.
6. Safety: In case of malfunction, system should shutdown itself and reboot in order to prevent unpredicted results.
7. Security: System should store data on database securely and set access permissions to these data carefully.

2.3 Hardware Requirements

1. User Side: As the Project is a service of providing the analysis of the input, it can be run on any machine irrespective of the specifications of the machine it is being run on. Any machine capable of browsing as well as uploading image and capable of connecting to internet and can surf the web can use the service.
2. Developer Side: RAM: 4GB, Graphics Card: 1GB, Processor: 2.3GHz

2.4 Software Requirements

1. User Side: Web-Browser
2. Developer Side: Python IDLE, Open-CV, TensorFlow.

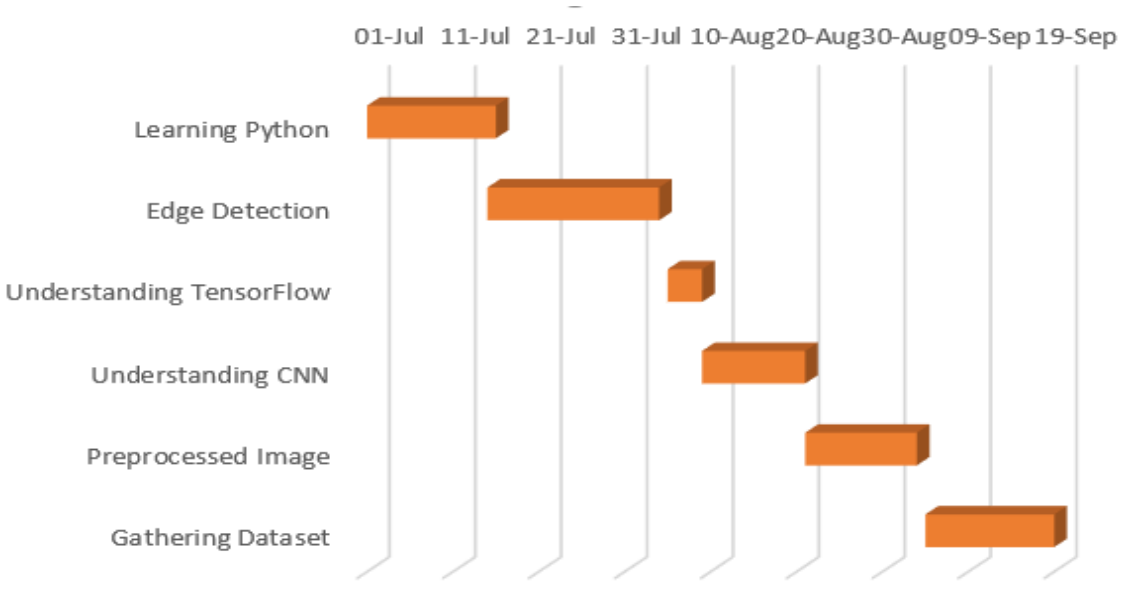
CHAPTER 3: Feasibility Study

3.1 Technical Study

- The Concepts Related to the Project are:

1. Converting Images in the dataset from RGB→Gray using the function cv2.cvtColor(img, cv2.COLOR_BGR2GRAY).
2. Enhancing the Gray image obtained in the first step using the enhancing functions using OpenCV.
3. Performing Function to detect Edges from the enhanced image using edging function using OpenCV.
4. Collecting attributed of the images obtained in the 3rd step.

3.2 Timeline Chart

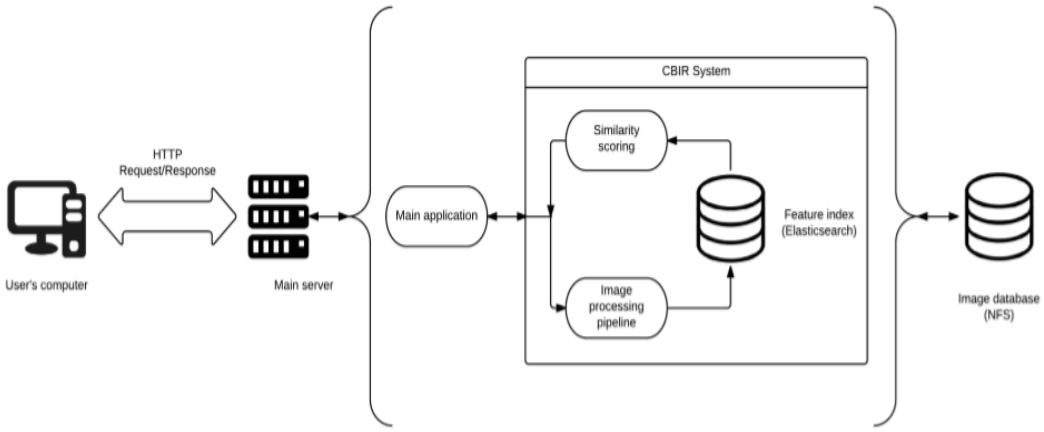


3.3 Economic Study

- The project is designed using free and open-source API's and does not make use of any chargeable software or hardware for its development. Thus, the project is completely free to design and even to use once it is completely developed.

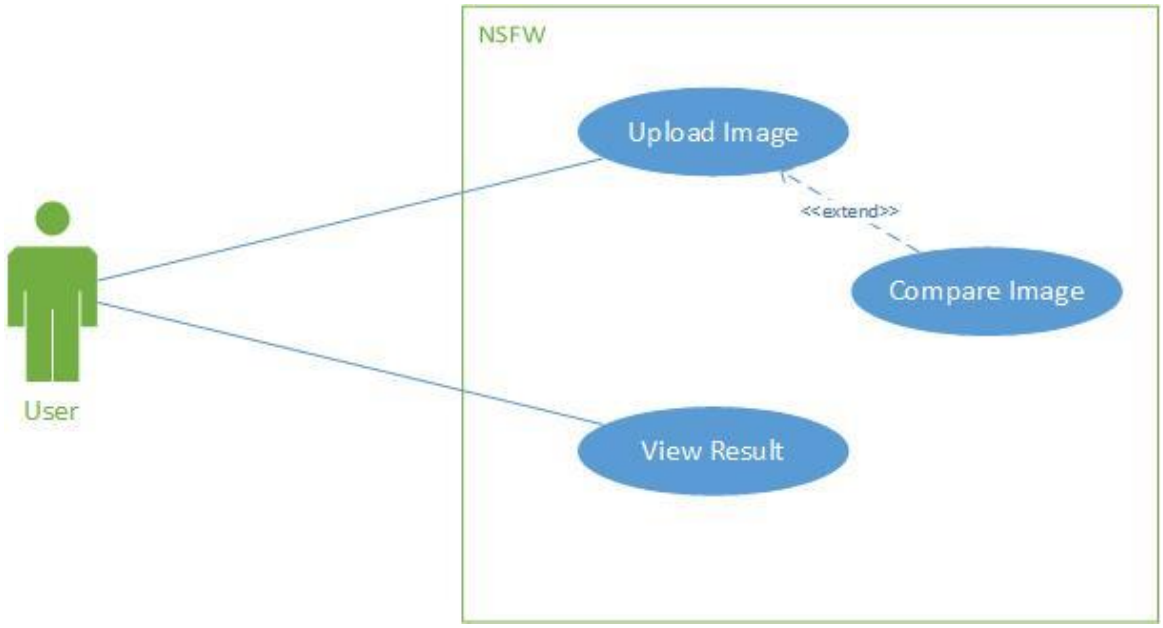
CHAPTER 4: System Design

4.1 Functional Requirements/System Architecture



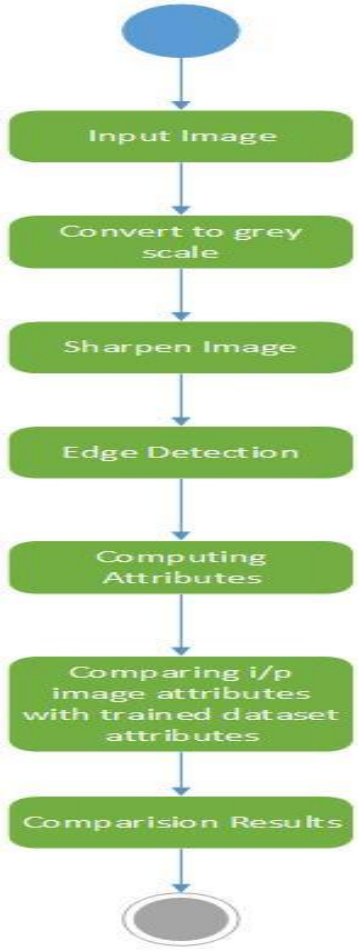
CHAPTER 5: System Diagram

5.1 USE CASE



5.2 CLASS DIAGRAM

5.3 ALGORITHM FLOW



CHAPTER 6: Software Description

6.1 Python

Python widely used general-purpose, high-level programming language. It is a multi-paradigm programming language. It's support includes object-oriented, imperative, functional programming and procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library. Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. Type constraints are not checked at compile time; rather, operations on an object may fail, signifying that the given object is not of a suitable type. Despite being dynamically typed, Python is strongly typed, forbidding operations that are not well-defined (for example, adding a number to a string) rather than silently attempting to make sense of them. The reasons why Python was chosen as the programming language can be summed up as following:

• The current web stack is build on Python making it the default language.

• It is free and Open Source. There is no need for licensing.

• It is a general-purpose language, i.e., it can be used for scientific computing, enterprise software, web design, back-end, front-end, and everything in between.

• It has great selection of libraries. Libraries like Flask or Bottle make it easy to create a web server with just a few lines of code and signal processing packages add Matlab like functionality to the language. NumPy provides efficient operations on homogeneous type of data. SciPy used for mathematical and scientific applications, Matplotlib is used to print pretty graphs, and IPython emulates the Matlab desktop environment.

• It is fast enough. Bottlenecks and performance sensitive modules can be addressed by writing C modules with Python binding, or the execution can be parallelized.

6.2 Open-CV

OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation. Wrappers in other languages such as C#, Perl, Haskell and Ruby have been developed to encourage adoption by a wider audience.

All the new developments and algorithms in OpenCV are now developed in the C++ interface.

OpenCV supports the Deep Learning frameworks TensorFlow, Torch/ PyTorch and Caffe.

6.3 Tensorflow

TensorFlow is an open source software library for machine learning across a range of tasks, and developed by Google to meet their needs for systems capable of building and training neural networks to detect and decipher patterns and correlations, analogous to the learning and reasoning which humans use. It is currently used for both research and production at Google products often replacing the role of its closed-source predecessor, DistBelief. TensorFlow was originally developed by the Google Brain team for internal Google use before being released under the Apache 2.0 open source license on November 9, 2015.

The name TensorFlow derives from the operations which neural networks perform on multidimensional data arrays, referred to as "tensors".

6.4 Convolutional Neural Networks

Design: A CNN consists of an input and an output layer, as well as multiple hidden layers. The hidden layers are either convolutional, pooling or fully connected

Convolutional: Convolutional layers apply a convolution operation to the input, passing the result to the next layer. Each convolutional neuron processes data only for its receptive field. Tiling allows CNNs to tolerate translation of the input image. The convolution operation reduces the number of free parameters and improves generalization.

CHAPTER 7: Hardware and Component Description

7.1 User Side

- Any machine capable of connecting to internet and has web browsing capability.

7.2 Developer Side

- 4 GB RAM
- 2 GB 940M Nvidia Graphic Card
- 2.3 GHz Processor

CHAPTER 8: Implementation

Model Architecture

activation_26 (Activation)	(None, 64, 26, 26)	0
max_pooling2d_11 (MaxPooling	(None, 64, 13, 13)	0
dropout_13 (Dropout)	(None, 64, 13, 13)	0
conv2d_23 (Conv2D)	(None, 128, 11, 11)	73856
activation_27 (Activation)	(None, 128, 11, 11)	0
conv2d_24 (Conv2D)	(None, 128, 9, 9)	147584
activation_28 (Activation)	(None, 128, 9, 9)	0
max_pooling2d_12 (MaxPooling	(None, 128, 4, 4)	0
dropout_14 (Dropout)	(None, 128, 4, 4)	0
flatten_3 (Flatten)	(None, 2048)	0
dense_5 (Dense)	(None, 64)	131136
activation_29 (Activation)	(None, 64)	0
dropout_15 (Dropout)	(None, 64)	0
dense_6 (Dense)	(None, 3)	195
activation_30 (Activation)	(None, 3)	0

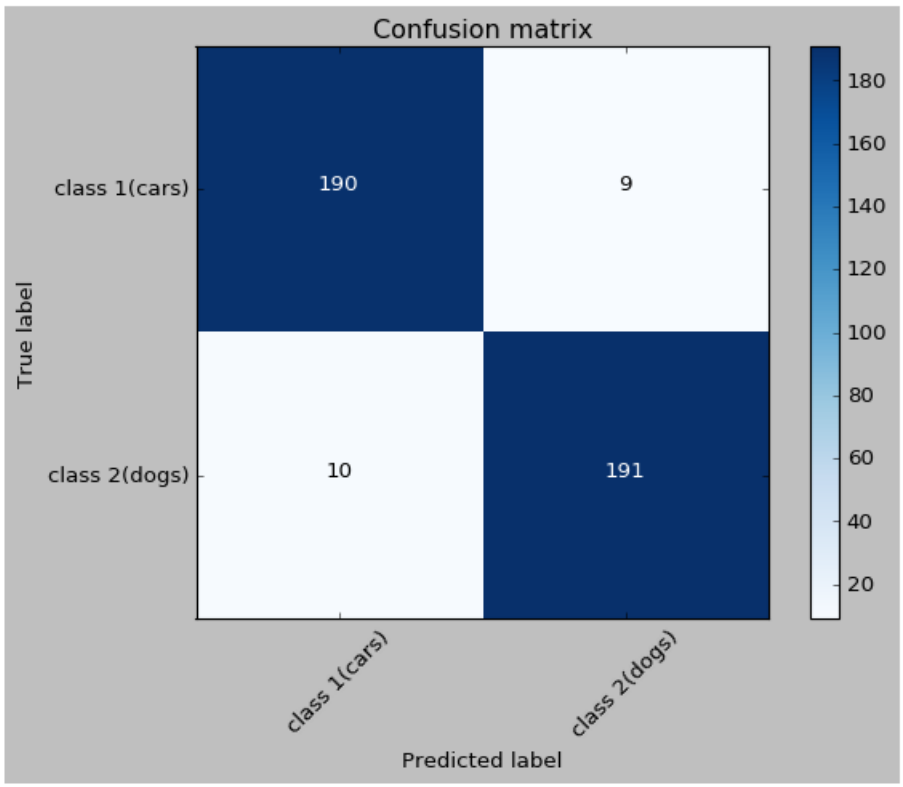
Total params: 424,851
Trainable params: 424,851
Non-trainable params: 0

Layer (type)	Output Shape	Param #
conv2d_17 (Conv2D)	(None, 16, 128, 128)	448
activation_21 (Activation)	(None, 16, 128, 128)	0
conv2d_18 (Conv2D)	(None, 16, 126, 126)	2320
activation_22 (Activation)	(None, 16, 126, 126)	0
max_pooling2d_9 (MaxPooling2	(None, 16, 63, 63)	0
dropout_11 (Dropout)	(None, 16, 63, 63)	0
conv2d_19 (Conv2D)	(None, 32, 63, 63)	4640
activation_23 (Activation)	(None, 32, 63, 63)	0
conv2d_20 (Conv2D)	(None, 32, 61, 61)	9248
activation_24 (Activation)	(None, 32, 61, 61)	0
max_pooling2d_10 (MaxPooling	(None, 32, 30, 30)	0
dropout_12 (Dropout)	(None, 32, 30, 30)	0
conv2d_21 (Conv2D)	(None, 64, 28, 28)	18496
activation_25 (Activation)	(None, 64, 28, 28)	0
conv2d_22 (Conv2D)	(None, 64, 26, 26)	36928

```
Epoch 12/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.1020 - acc: 0.9619 -
val_loss: 0.1345 - val_acc: 0.9525
Epoch 13/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.1030 - acc: 0.9600 -
val_loss: 0.1442 - val_acc: 0.9525
Epoch 14/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.1069 - acc: 0.9600 -
val_loss: 0.1533 - val_acc: 0.9500
Epoch 15/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.0950 - acc: 0.9594 -
val_loss: 0.1316 - val_acc: 0.9500
Epoch 16/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.0889 - acc: 0.9631 -
val_loss: 0.1825 - val_acc: 0.9450
Epoch 17/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.0853 - acc: 0.9675 -
val_loss: 0.2100 - val_acc: 0.9400
Epoch 18/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.0838 - acc: 0.9675 -
val_loss: 0.1900 - val_acc: 0.9375
Epoch 19/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.0820 - acc: 0.9663 -
val_loss: 0.1552 - val_acc: 0.9425
Epoch 20/20
1600/1600 [==============================] - 8s 5ms/step - loss: 0.0613 - acc: 0.9762 -
val_loss: 0.1496 - val_acc: 0.9525
(128, 128, 3)
(1, 3, 128, 128)
('4', array([[ 0.01262941,  0.98737061]], dtype=float32))
5---dog
```

Confusion Matrix

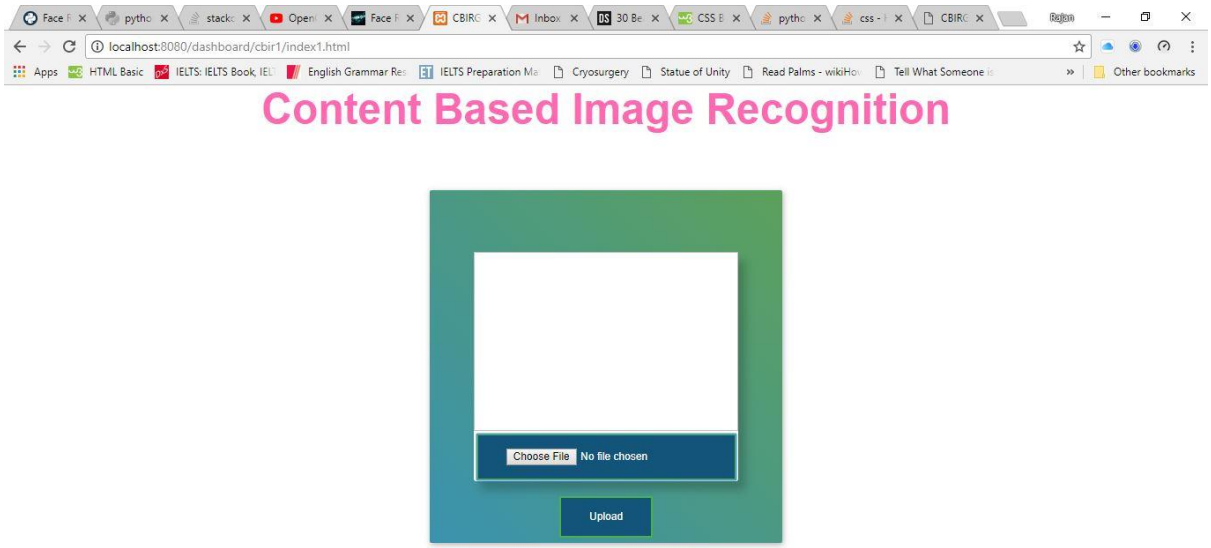
```
 [ 10 191]]
8--Confusion matrix, without normalization
9 [[190   9]
 [ 10 191]]
```



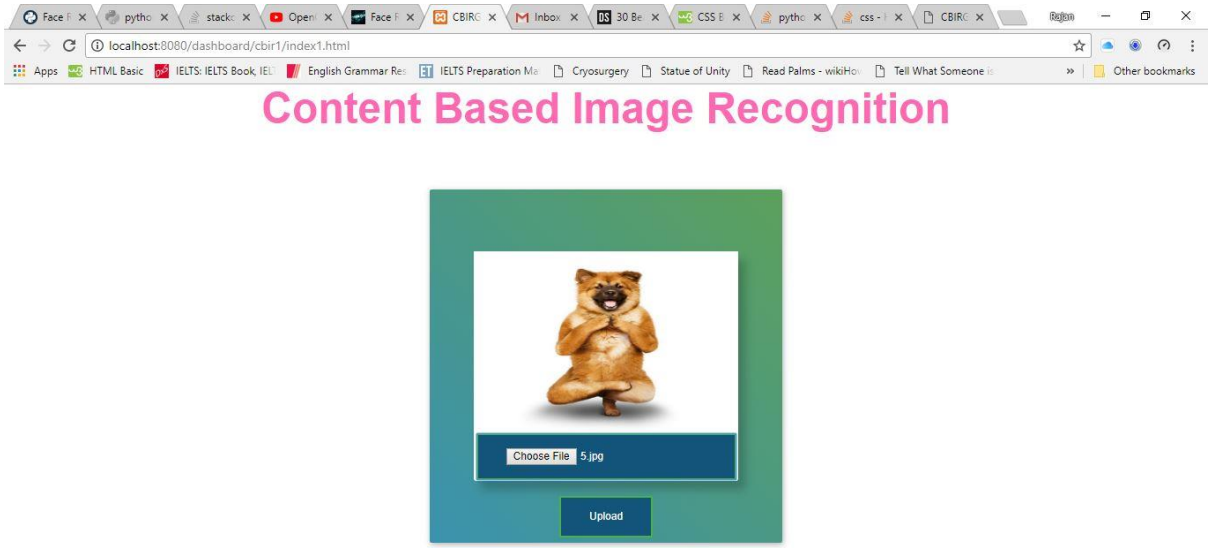

```
In [8]:
```

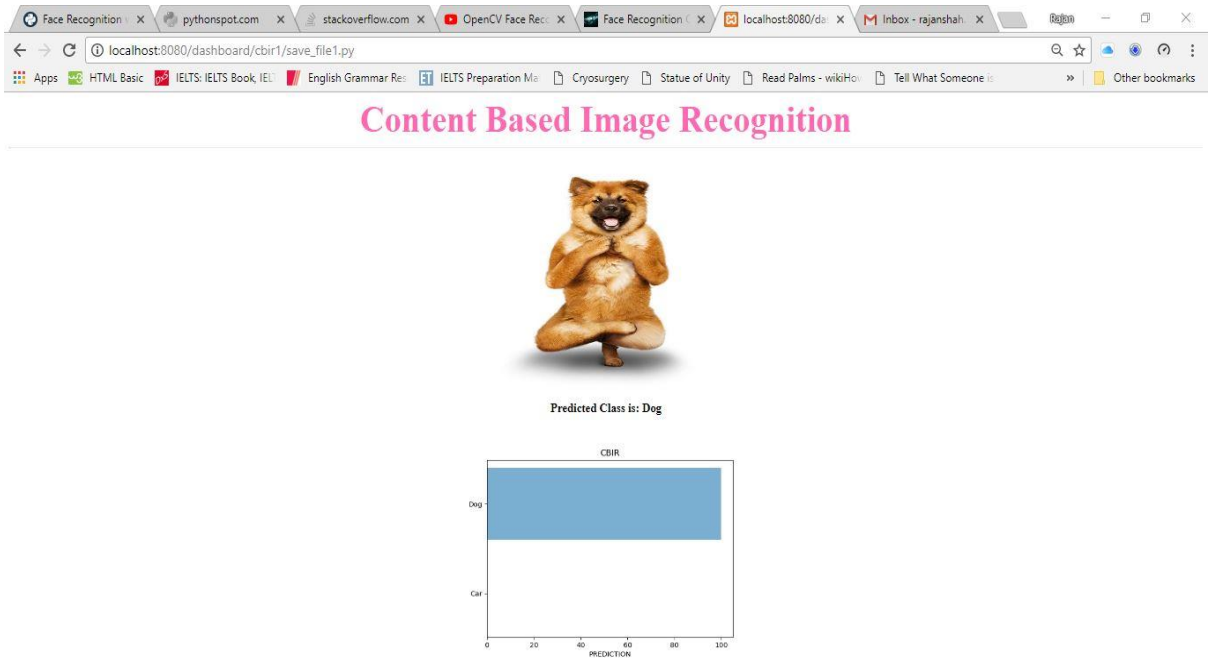
Feature Maps:

Web UI:


localhost:8080/dashboard/cbir1/index1.html
Content Based Image Recognition
Choose File
No file chosen
Upload

CHAPTER 9: Testing





CHAPTER 10: Conclusion

Thus, we have implemented our project of CONTENT BASED IMAGE RECOGNITION for 2 classes viz: CARS and DOGS using Sequential Model built using Keras: A Machine Learning Framework for Python language.