Major Project (CS801PC)

on

IMAGE PLAGIARISM DETECTION USING COMPRESSED IMAGES

Submitted

in partial fulfillment of the requirements for

the award of the degree of

Bachelor of Technology

In

Computer Science and Engineering

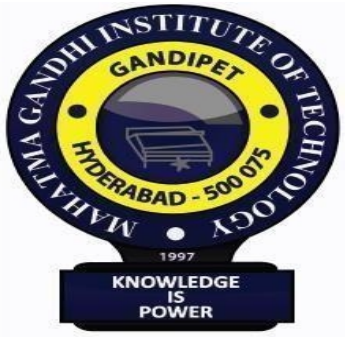
By

V.SANDEEP (18265A0512)

Under the Guidance of

Ms. M. MAMATHA

(Assistant Professor)



Dept. of Computer Science and Engineering

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY

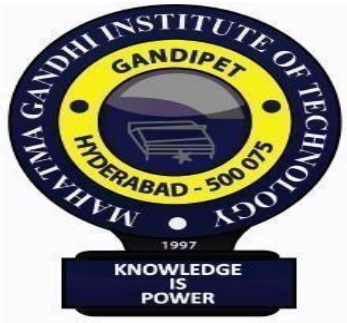
GANDIPET, HYDERABAD, TELANGANA - 500 075, INDIA

2020-2021

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

GANDIPET, HYDERABAD, TELANGANA– 500 075 (INDIA)



CERTIFICATE

This is to certify that the project entitled **IMAGE PLAGIARISM DETECTION USING COMPRESSED IMAGES** is being submitted by **V. SANDEEP** bearing **Roll No: 18265A0512** in partial fulfillment for the award of **Bachelor of Technology** in **Computer Science and Engineering** to **Jawaharlal Nehru Technological University, Hyderabad** is a record of bonafide work carried out by him/her under my guidance and supervision.

The results embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma.

Supervisor	Head of the Department
Ms. M. MAMATHA	**Dr. C.R.K. REDDY**
Assistant Professor	Professor

EXTERNAL EXAMINER

DECLARATION

This is to certify that the work reported in this project titled **IMAGE PLAGIARISM DETECTION USING COMPRESSED IMAGES** is a record of work done by me in the Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology, Hyderabad.

No part of the work is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred to in the text. The report is based on the work done entirely by me and not copied from any other source.

V.SANDEEP(18265A0512)

ACKNOWLEDGEMENT

I would like to express my sincere thanks to **Dr. K Jaya Sankar, Principal MGIT**, for providing the working facilities in college.

I wish to express our sincere thanks and gratitude to **Dr. C.R.K.Reddy, Professor and HOD**, Department of CSE, MGIT, for all the timely support and valuable suggestions during the period of project.

I am extremely thankful to **Dr. C.R.K.Reddy, Professor and HOD, Mrs. B. Prasanthi, Assistant Professor**, Department of CSE, MGIT, Major Project Coordinators for their encouragement and support throughout the project.

I am extremely thankful and indebted to my internal guide **Ms. M.MAMATHA, Assistant Professor,** Department of CSE, for her constant guidance, encouragement and moral support throughout the project.

Finally, I would also like to thank all the faculty and staff of the CSE Department who helped me directly or indirectly, for completing this project.

V.SANDEEP(18265A0512)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In an educational environment, plagiarism is a crucial task that needs to be identified, in recent years all known journals and conferences, as well as universities, request a plagiarism report from students and researchers to prove the originality of published text or scientific paper. Plagiarism detection usually checks the text content via many of the platforms which are available for productive use reliably identifying copied text or near-copies of text and these systems usually fail to detect the images, and File's plagiarism since it is originally built for text mainly. In this paper, we suggest an adaptive, scalable, and extensible, robust method for image plagiarism.

Image plagiarism is stealing of another's work and passing it off as their own work without crediting the source. Image plagiarism is based on image processing, which helps to manipulate and perform operations on image to detect plagiarism. Previously lot of work is done to detect plagiarism on text, but there is no much work done in this area. In this paper an attempt is made to detect difference between the images using image subtraction. The system is also overcoming the vulnerability of re-sizing, compression and color differentiation. The similarity and the difference between the images are displayed using histogram.

1. INTRODUCTION

1.1 PROBLEM STATEMENT

The main idea of this project is to detect the plagiarized images, Plagiarism is the practice of copying someone else's work or ideas, and passing them off as one's own original work. Not only images but, architecture, flow diagram, UML diagrams, even snapshots of test results can be plagiarized. If the author has not mentioned the credit for the original author from where he/she copied the image then it is said to be plagiarized Previously lot of work is done to detect the plagiarism, but now in this I made an attempt to detect difference between images using image subtraction.

1.2 EXISTING SYSTEM

The existing system uses Content-Based Image Retrieval (CBIR) is a kind of feature extraction method which uses contents of an image like shape, colour, texture for representation and indexing of image. This system can detect only images it failed to detect flow chart, UML diagrams.

DISADVANTAGES

This system has the vulnerability of re-sizing, compression and colour differentiation.
CBIR requires lot of work to detect whether the images are plagiarized or not.
The existing system is not efficient to detect plagiarism properly for different types of images.

1.3 PROPOSED SYSTEM

In the proposed system it uses FMM (Five modulus method) algorithm.
The FMM (Five Modulus Method) algorithm converts a random image into 8*8 block matrix. I have used FMM for compression as the intended data which is required is not lost in this method.
The proposed work mainly focusses on finding the similarity between two images. Sample image is given as the reference and it is compared with the other image which is taken from any journal and comparison is done through histogram.

ADVANTAGES

In the proposed system the image result is displayed using Histogram. It is the best way to visualize the largest intensities of an image.

This system also overcomes the vulnerability of re-sizing, compression, color differentiation.

The main advantage of image compression this will reduce the original size of image to lower size. So, this makes the image processing faster.

This system also capable to detect plagiarism in UML diagrams, flow charts, architecture and even in snap shot of test results.

1.4 SYSTEM REQUIREMENTS

1.4.1 Hardware Requirements

- System Processor : Min i3 and above
- Hard Disk : Min120 GB
- RAM : 4GB or above
- System Type : 64-bit Operating System
- Input Devices : Keyboard, Mouse

1.4.2 Software Requirements

- Operating System : Windows 8 /10 or Linux
- Tools : Django, Python shell
- Software : Python 3.7

2. LITERATURE SURVEY

S.NO	TITLE	AUTHORS	ALGORITHM/ TECHNIQUE USED	MERITS	DEMERITS/ FUTURE SCOPE
1	Image plagiarism detection using Compressed images,2019	Akshay S, Rishab kumar, Chaitanya B	In this paper the (FMM) Five modulus method is used	Result is displayed in Histogram. It also detects plagiarism in re-sized images	Extraction of images directly from PDF can be done rather than manual technique
2	Plagiarism Detection Engine for Images,2018	Gasper Pizarro V, Juan D Velasquez	It uses built in python, OpenCV and scikit-image library	It handles queries for a large number of different images	Lack in performance with color transformed images
3	Plagiarism detection of flow chart images,2017	Mohammad Javad kargar, Behnam Hadi	In this paper the ANN is used for training and testing the data	Less computational load of K-L transform	Can detect only flow chart images
4	A Tool for image plagiarism detection	Petr Hurthik, Petra Hodakova	In this paper the FTP algorithm is used.	One dimensional searching and Two-dimensional searching	Pre-processing step take lot of time, High computational time

Table 2.1 Literature survey for Image plagiarism detection using compressed images

3. DESIGN METHODOLOGY

3.1 System Architecture

The work is mainly to focus on finding the similarity between two images. Sample image is given as the reference and it is compared with the other image and comparison is done through histogram. Histogram is the best way to visualize the largest intensities of an image. It is used to find the problems which originate during image acquisition such as exposure, contrast etc. even a minute difference with the pixel is noticed by histogram.

i) Input two images.

ii) Pre-processing includes Compression, Grey Scaling, Binarization

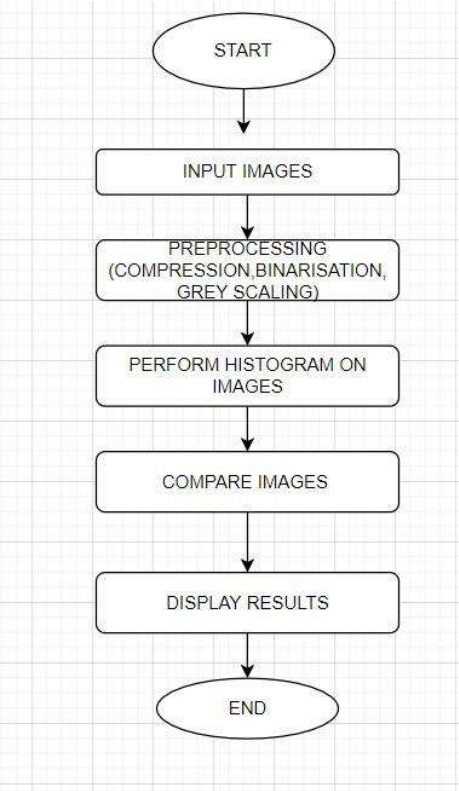


Fig 3.1 Flow chart

Fig 3.1 Tells about the flow chart of Image plagiarism detection using compressed images

The main advantage of image **compression** is to minimize the original size of the image to lower size. Now a days the image size is very huge so to reduce the computational time FMM algorithm is used. The FMM (Five Modulus Method) algorithm converts a random image into 8*8 block matrix. The block matrix is divided by 5 which reduces the size of the image. We have used FMM for compression as the intended data which is required is not lost in this

method. It also reduces the size of the image from higher intensity to lower intensity which helps to compare images more rapidly.

FMM firstly converts the input image into **gray scaled** image and then compresses using threshold value.

To overcome the vulnerability of images with different size, we are re-sizing the images to a fixed size. To get the preferred size of the image the re-sizing is done. The pixel information is changed when the images are re-sized to lower size or upper size. The value of pixel gets added when the images are enlarged.

Binarization is the process of converting images into black-0 and white-1 as pixel values. The pixels of the image are replaced by 1 and 0 as per the threshold value. Comparison of the images is done pixel by pixel; the difference determines the similarity between the images. Binary images also help in reducing the memory storage than normal images. It is also used to overcome the weakness of images with colored and non-colored.

We also use image subtraction method to find out the difference between the pixel. So the images are converted into binary values. The difference between the images specifies whether the images are same or not. Each pixel value of a image is subtracted with the pixel value of other image.

The algorithm describes about the comparison between the images. In the initial stage the two images are loaded into the work space. Later the images are re-sized in-order to get the same size for both the images so that the result will be accurate when compared. The image is also compressed in order to improve the accuracy while comparing. Image is automatically saved in a folder after compression.

To over-come the weakness of having images with different colors, the image is converted into grayscale image. In the next step the similar features of the images are detected and the comparison is done using image subtraction method where each pixel value of an image is compared with pixel value of another image and finally the result will be display by Histogram Histogram is used for displaying the results of both the images and to check whether the images are plagiarized or not.

3.2 UML DIAGRAMS

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system. UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modeling workflows as well as a wider range of features to improve readability and efficiency.

3.2.1 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

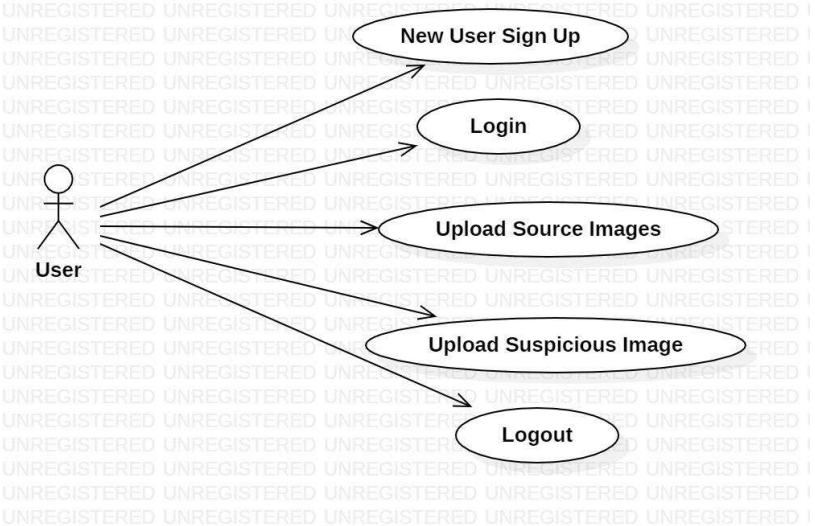


Fig 3.2 Use case diagram for Image plagiarism detection using compressed images

Fig 3.2 Shows the Use case of user module and helps in understanding how the user is interacted with the system

3.2.2 CLASS DIAGRAM

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

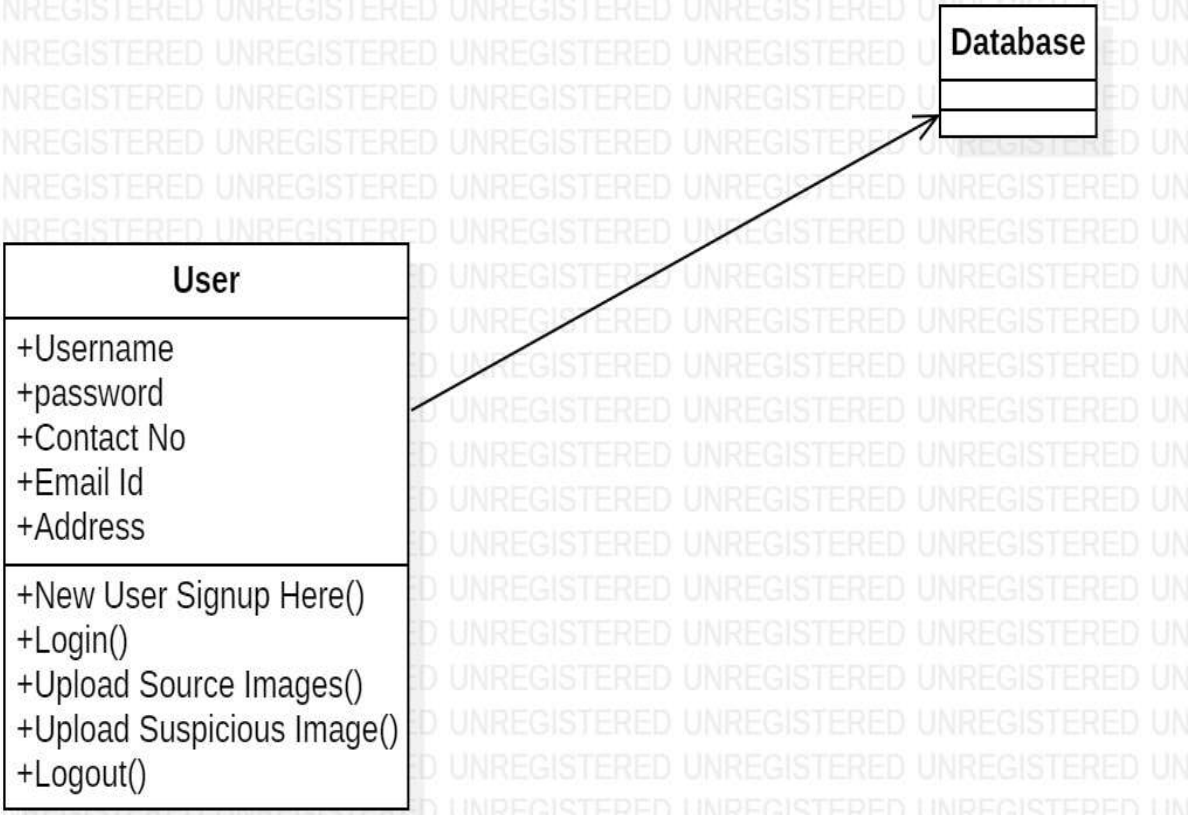


Fig 3.3 Class diagram for Image plagiarism detection using compressed images

Fig 3.3 Shows about the class diagram of user module and what information is stored in the database

3.2.3 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

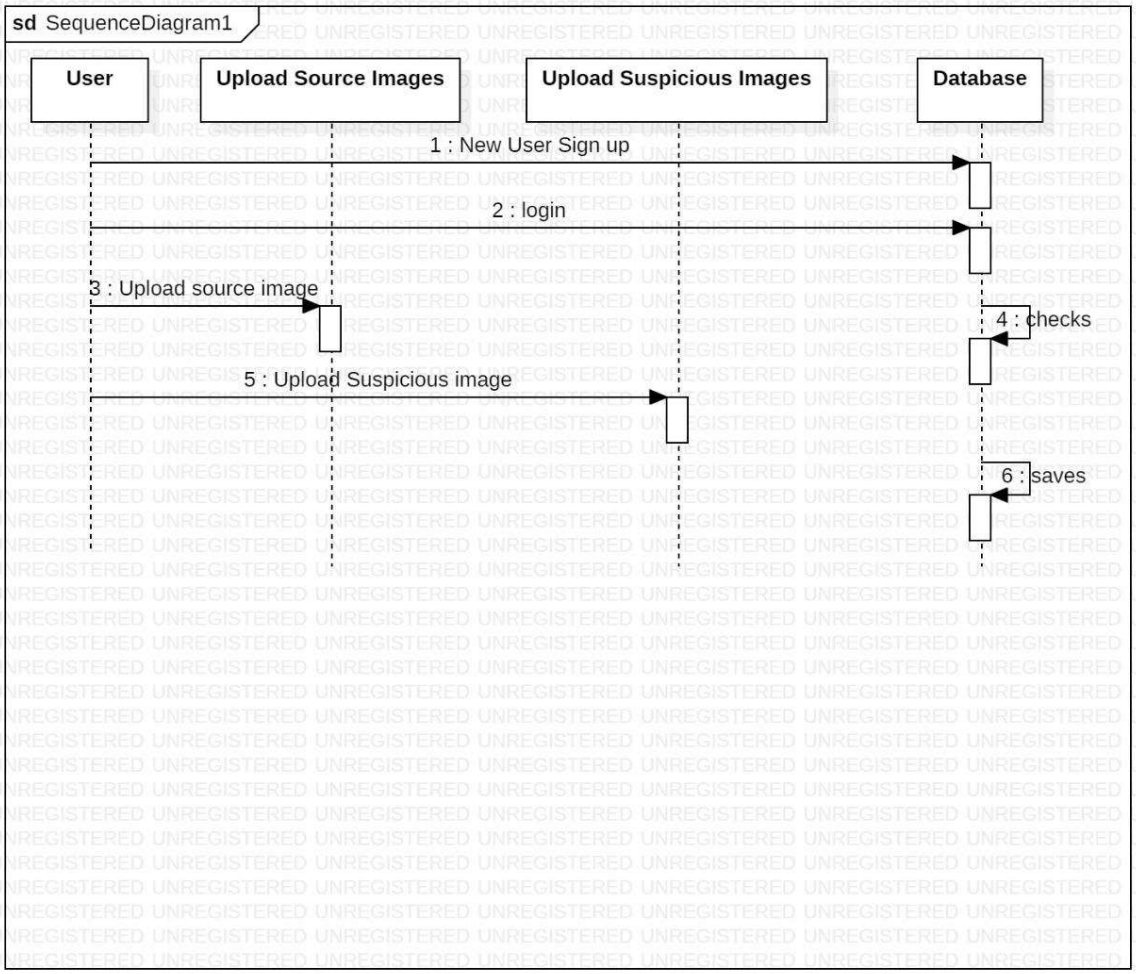


Fig 3.4 Sequence diagram for Image plagiarism detection using compressed images

Fig 3.4 Tells about the Sequence diagram of the user Module how the steps are done sequentially can be understand easily by seeing the sequence diagram

3.2.4 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

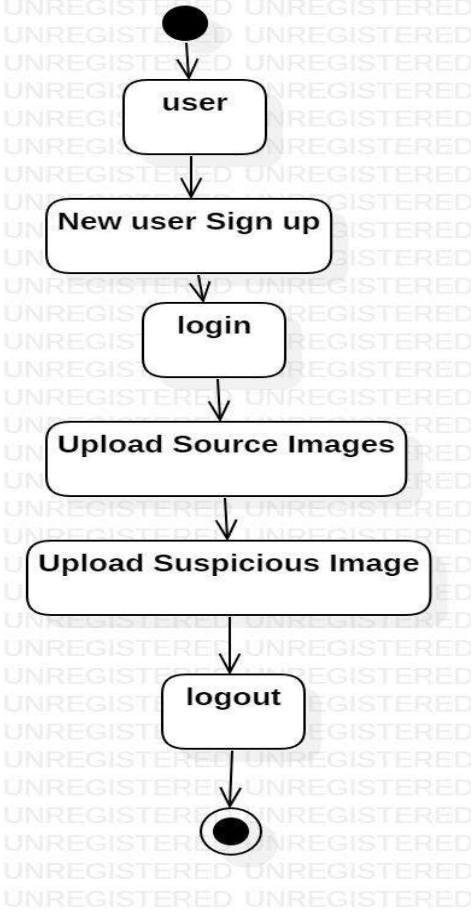


Fig 3.5 Activity diagram for Image plagiarism detection using compressed images

Fig 3.5 Tells about the flow control of the user module this helps in understanding the stepwise activities of user module

3.3 TOOLS USED

3.3.1 Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source. We have used Django because of its advantages such as Ridiculously fast. Django was designed to help developers take applications from concept to completion as quickly as possible, reassuringly secure. Django takes security seriously and helps developers avoid many common security mistakes, Exceedingly scalable. Some of the busiest sites on the Web leverage Django's ability to quickly and flexibly scale.

3.3.2 MYSQL

MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database.

MySQL is a Relational Database Management System (RDBMS) software that provides many things, which are as follows:

- It allows us to implement database operations on tables, rows, columns, and indexes.
- It defines the database relationship in the form of tables (collection of rows and columns), also known as relations.
- It provides the Referential Integrity between rows or columns of various tables.
- It allows us to updates the table indexes automatically.
- It uses many SQL queries and combines useful information from multiple tables for the end-users.
- To store the user details and also to organize source files and images we have used MySQL.

3.3.3 LIBRARIES

OpenCV

OpenCV (Open-Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision. OpenCV is used to manipulate images and perform basic preprocessing before passing in images to the model for training or predictions. OpenCV promotes OpenVisionCapsules, which is a portable format, compatible with all other formats.

Matplotlib

Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. As such, it offers a viable open-source alternative to MATLAB. Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications.

NUMPY

It is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices along with a large collection of high-level mathematical functions to operate on these arrays.

3.4 MODULES

The modules present in this project are listed and explained below

3.4.1 New user Signup

Firstly, user will register in to application. Once registered user can login into application with username and password. Signing up gives the developer the advantage of receiving feedbacks and knowing the number of users that are using the application.

3.4.2 Login

User will login into Application through username and password**.**

3.4.3 Upload Source Image

In this module all the images from the database histogram will be calculated and the histogram values are stored in array and whenever we upload new test image then both histograms of the uploaded image and database image will be compared to check plagiarism.

3.4.4 Upload Suspicious Image

Once the source image is uploaded into database folder it is then loaded in the web application and then the suspicious image is uploaded. Now, the histogram values of the matching source images get compared and threshold value is calculated for resized image and checked for plagiarism.

3.5 FMM ALGORITHM

FMM algorithm means FIVE MODULUS METHOD

In the image plagiarism in order to compress the image with low amount of lossy compression we use the five-modulus method.

The main advantage of image compression is to minimize the original size of the image to lower size. Now a days the image size is very huge so to reduce the computational time FMM algorithm is used. The FMM (Five Modulus Method) algorithm converts a random image into 8*8 block matrix. The block matrix is divided by 5 which reduces the size of the image. We have used FMM for compression as the intended data which is required is not lost in this method. It also reduces the size of the image from higher intensity to lower intensity which helps to compare images more rapidly. FMM firstly converts the input image into gray scaled image and then compresses using threshold value.

ALGORITHM:

```
if A(i,j) Mod 5 = 4
A(i,j)=A(i,j)+1
Else if A(i,j) Mod 5 = 3
A(i,j)=A(i,j)+2
Else if A(i,j) Mod 5 = 2
A(i,j)=A(i,j)-2
Else if A(i,j) Mod 5 = 1
A(i,j)=A(i,j)-1
for i in range(rows):
for j in range(cols):
k = img[i,j]
k = k / 5
img[i,j] = k
temp = img.ravel()
temp = np.min(img)
```

221	232	231	242	246	247	251	250
220	227	231	236	242	241	250	251
221	215	221	232	240	247	251	251
217	216	216	225	237	241	245	247
216	221	217	222	231	235	242	247
220	216	222	215	227	231	242	247
216	216	211	216	222	227	237	247
217	216	211	216	217	222	237	235

Table 3.1 Original 8 x 8 block matrix

Table 3.1 It is an exactly 8×8 block has been taken from any of the red, yellow, or green arrays in any arbitrary image.

220	230	230	240	245	245	250	250
220	225	230	235	240	245	250	250
220	215	220	230	240	245	250	250
215	215	215	225	235	240	245	245
215	220	215	220	230	235	240	245
220	215	220	215	225	230	240	245
215	215	210	215	220	225	235	245
215	215	210	215	215	220	235	235

Table 3.2 Transformed block matrix using FMM

Table 3.2 shows the values After FFM is applied, i.e., transform each pixel into multiple of 5, the new 8×8 block can be seen in the above table

44	46	46	48	49	49	50	50
44	45	46	47	48	49	50	50
44	43	44	46	48	49	50	50
43	43	43	45	47	48	49	49
43	44	43	44	46	47	48	49
44	43	44	43	45	46	48	49
43	43	42	43	44	45	47	49
43	43	42	43	43	44	47	47

TABLE 3.3 Dividing FMM block by 5

Table 3.3 shows the values after diving the 8 x 8 block matrix (table 3.2) by 5

2	4	4	6	7	7	8	8
2	3	4	5	6	7	8	8
2	1	2	4	6	7	8	8
1	1	1	3	5	6	7	7
1	2	1	2	4	5	6	7
2	1	2	1	3	4	6	7
1	1	0	1	2	3	5	7
1	1	0	1	1	2	5	5

TABLE 3.4 After subtracting minimum value

Table 3.4 shows the values of the table after subtracting the minimum value from table 3.3 i.e., minimum is 42

4. TESTING AND RESULTS

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property function as a unit. The test data should be chosen such that it passed through all possible conditions. The following is the description of the testing strategies, which were carried out during the testing period.

4.1 TEST CASES

- REGISTRATION TEST CASE

Test Case	Input	Test case Description	Expected Output	Actual Output	Status
1	Invalid user id and password	User registration	Unable to Register	Displays message to choose different username	Pass
2	Valid User id and password	User registration	Registration Successful	user registered successfully	Pass

Table 4.1: Test case for user registration

Table 4.1 shows that, user has to register in order to login to the account if the details entered already exist, then it displays a message to choose a different username. If the user enters the valid details, then registration will be successful.

- LOGIN TEST CASE

Test Case	Input	Test case Description	Expected Output	Actual Output	Status
3	Invalid user id and password	Account login	Login failed	User id or Password is incorrect	Pass
4	Valid User id and password	Account login	Login Successful	User logs in successfully	Pass

Table 4.2 : Test case for user login

Table 4.2 shows that, invalid user id and password has been given as input so it gives output as user id and password is incorrect. When we enter the correct user id and password it shows login successful.

4.2 RESULTS

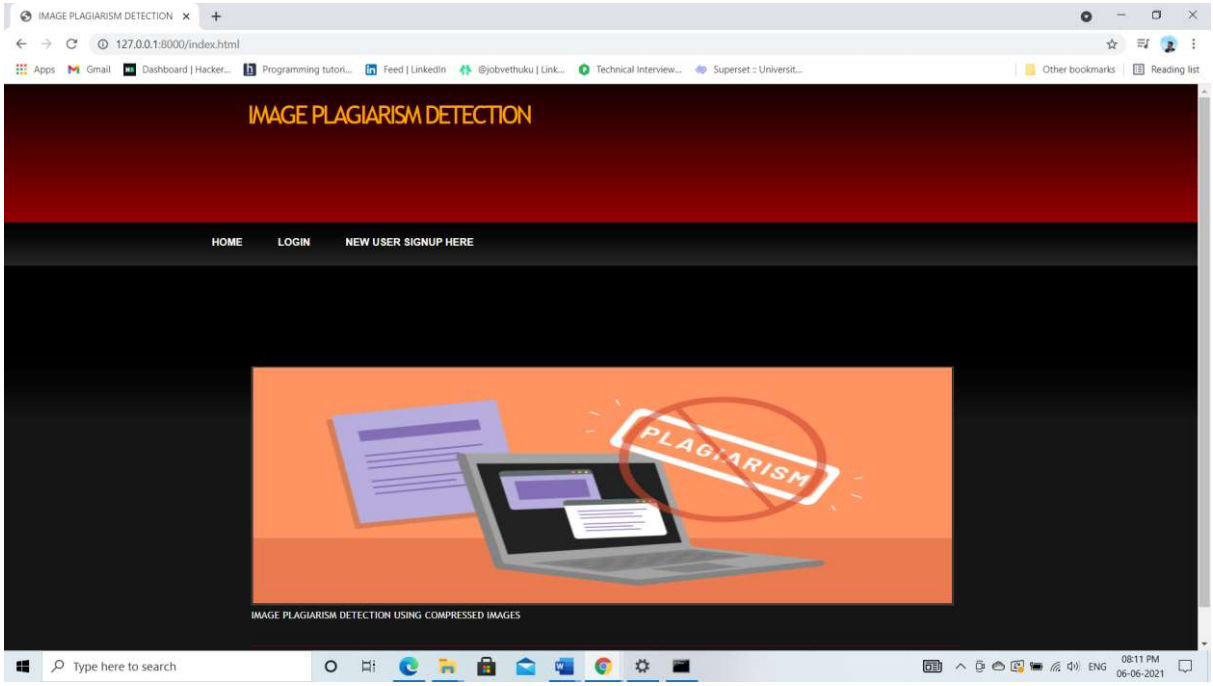


Fig 4.1 Home page

Fig 4.1 Shows the home page of Image plagiarism detection using compressed images

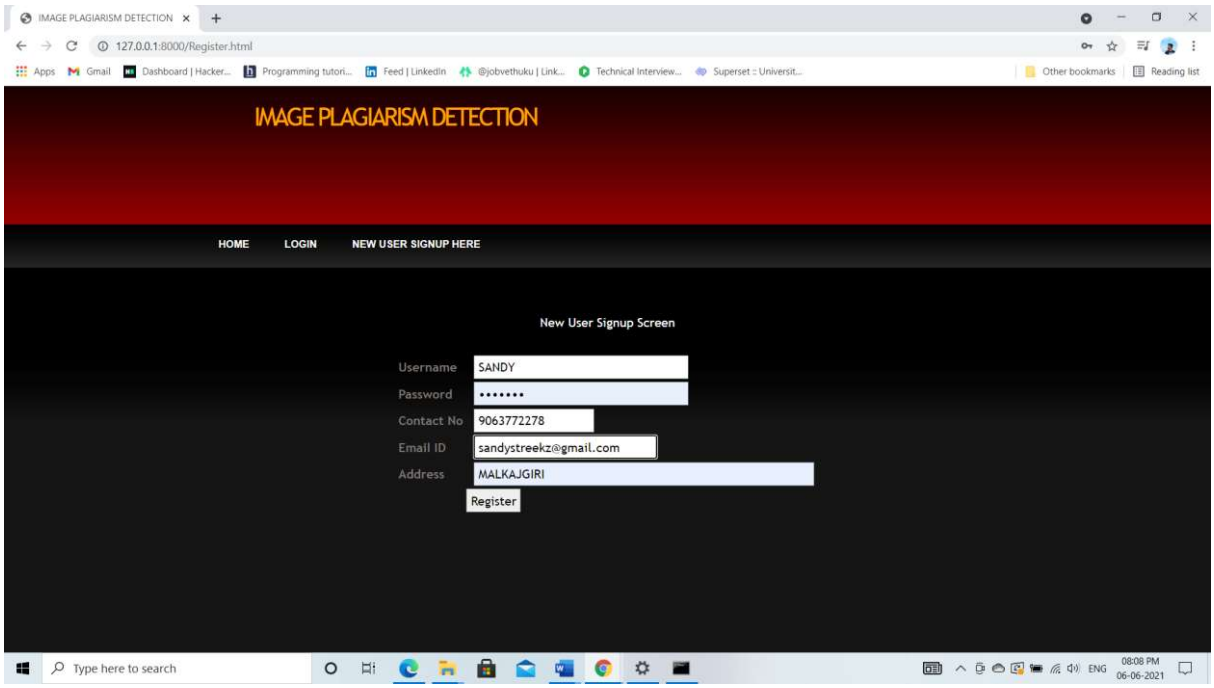


Fig 4.2 Sign-up page

Fig 4.2 Shows the user registration process and details need to enter by the user for Image plagiarism detection using compressed images

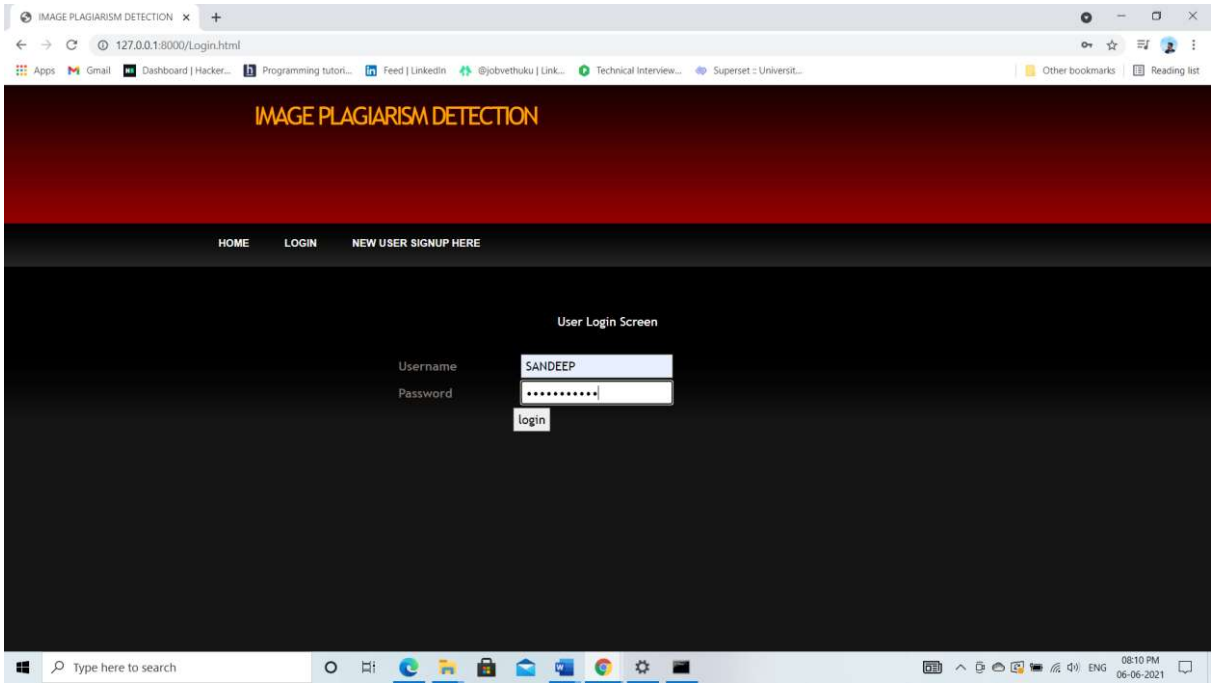


Fig 4.3 Login Page

Fig 4.3 Shows the user login page where user needs to enter Username and password and click on Login button

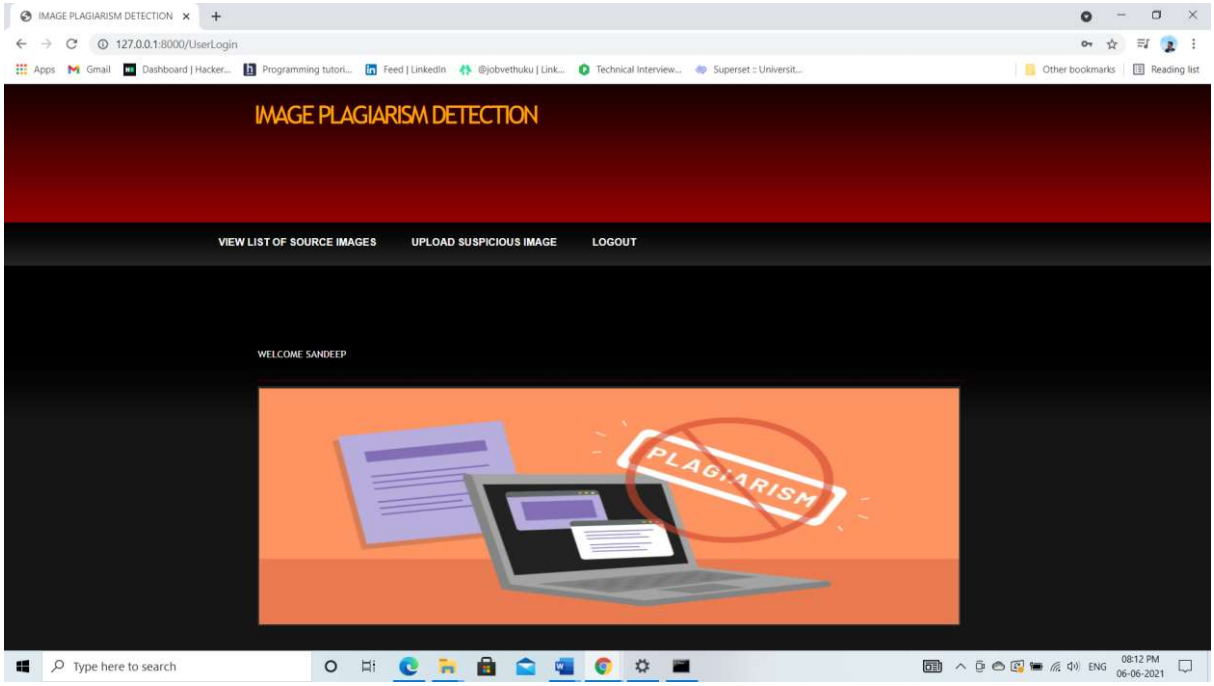


Fig 4.4 User Home Page

Fig 4.4 Shows the home page of the user after successful login for Image plagiarism detection using compressed images

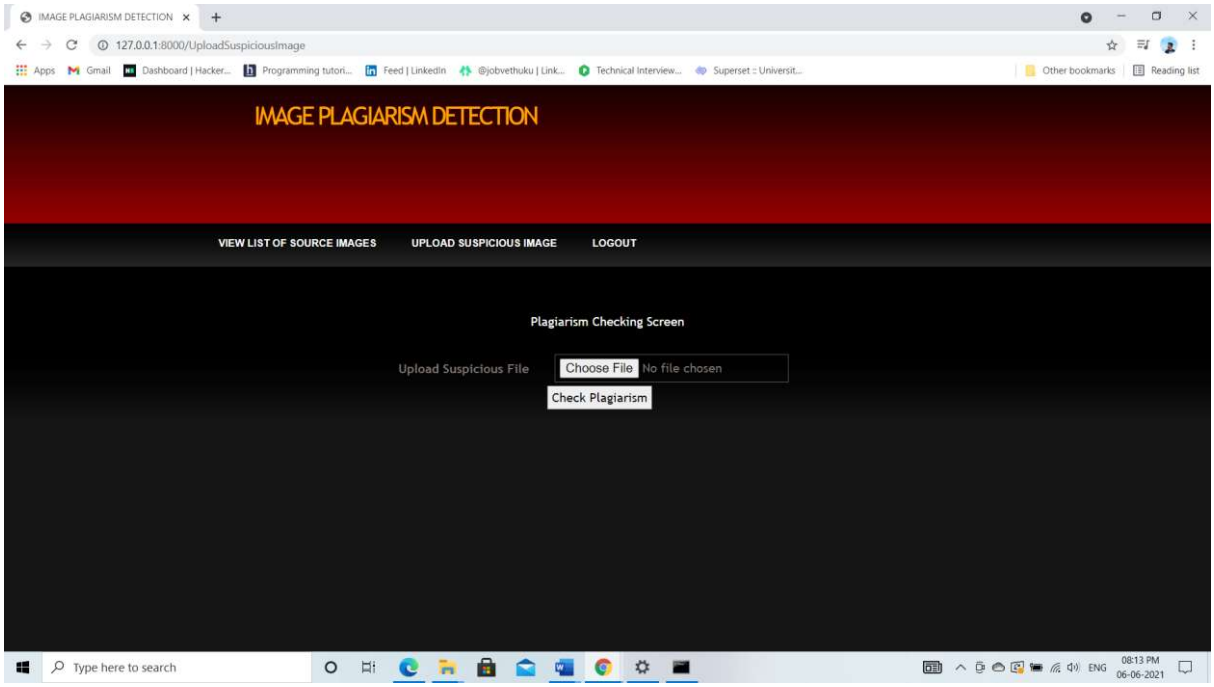


Fig 4.5 Upload suspicious image page

Fig 4.5 shows the page for uploading the suspicious image by clicking on “Choose file”

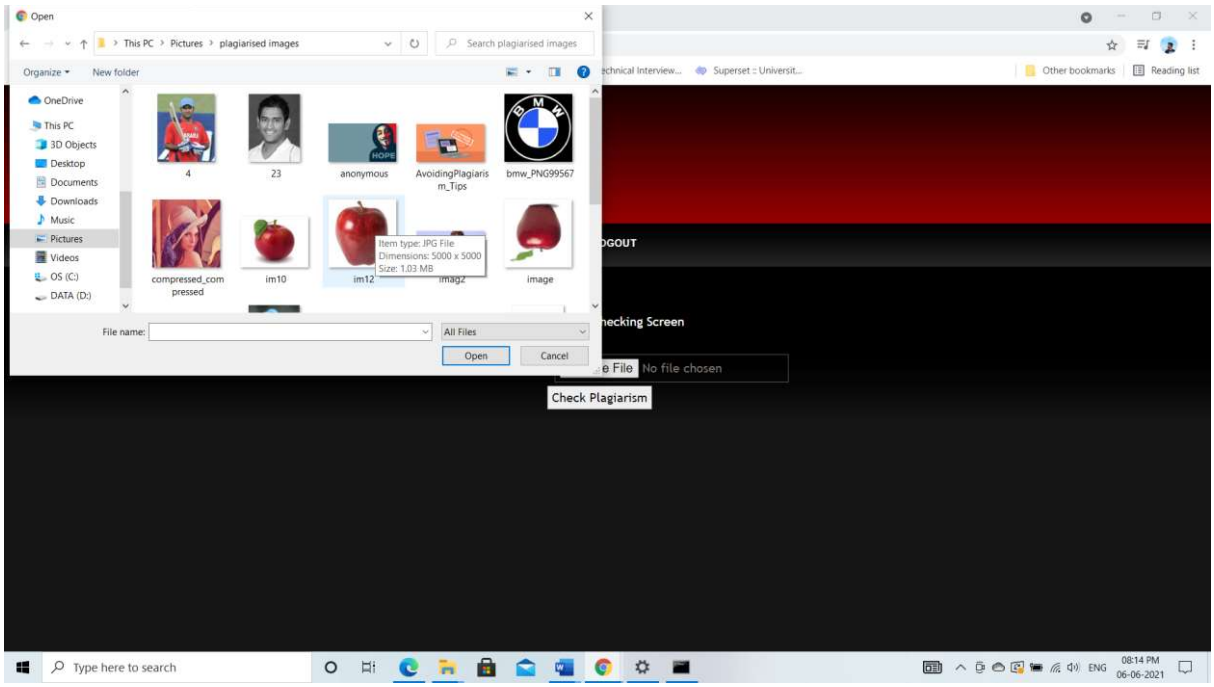


Fig 4.6 Select suspicious image

Fig 4.6 Shows that user can browse any image from the local disk and check plagiarism with the Source images

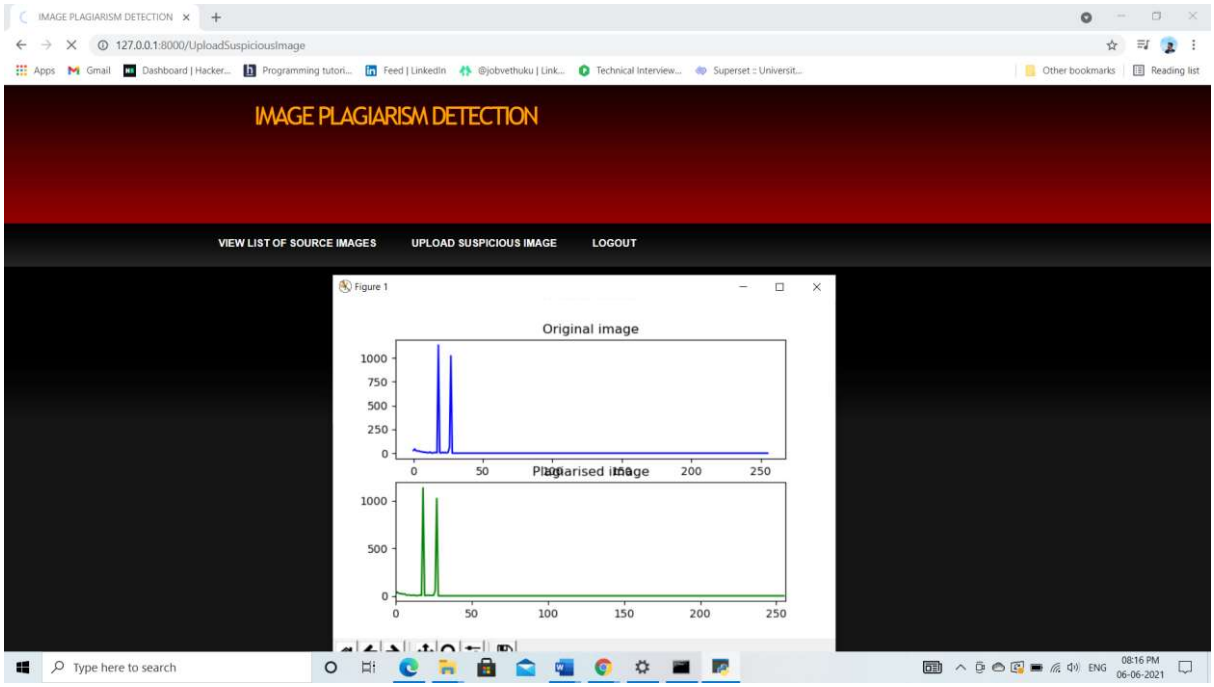


Fig 4.7 Histogram of Source image and Suspicious image

Fig 4.7 Shows the histogram of Original and Suspicious image and both the results are compared to check whether the image is plagiarized or not

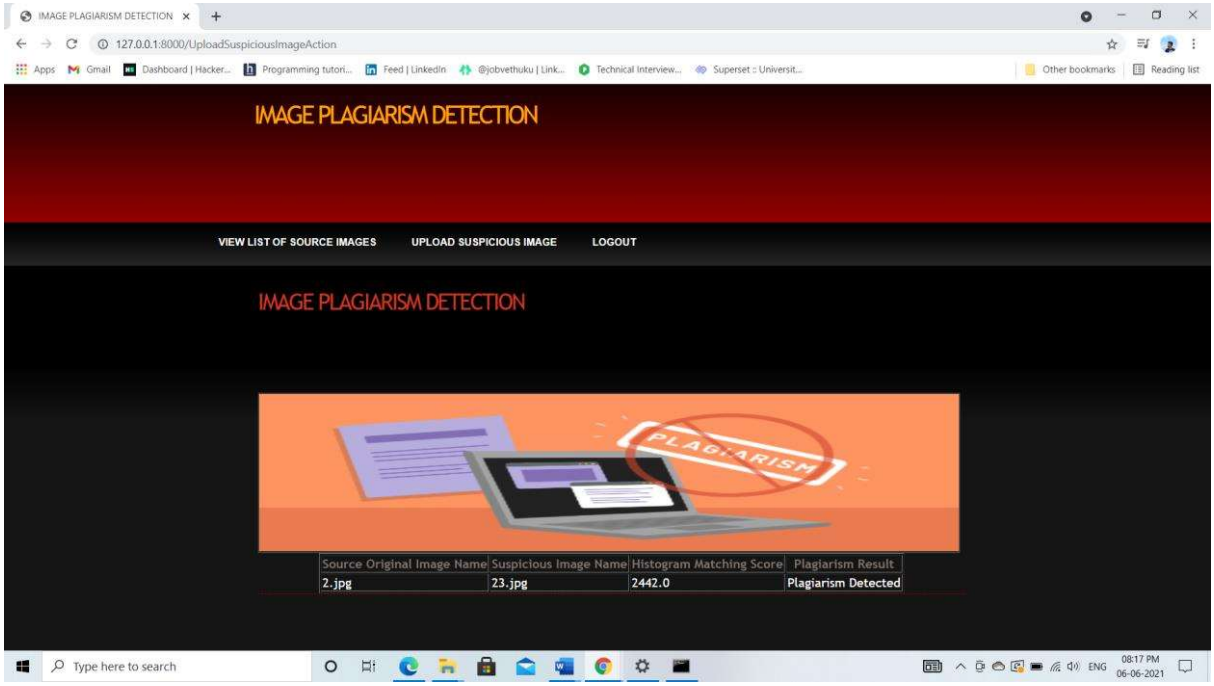


Fig 4.8 Result page

Fig 4.8 Shows the Histogram matching score of the uploaded suspicious image with Source image

5. CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in Python & MySQL. It also provides knowledge about the latest technology used in developing web enabled applications and client server technology that will be in great demand in future. This will provide better opportunities and guidance in future in developing projects independently. Tools used to develop this project are Django, Python 3.7, MySQL, SQLYog, HTML and CSS. Image plagiarism can be detected even when the image color is changed or even if the image is rotated. There were no errors observed.

5.2 FUTURE SCOPE

Image plagiarism accuracy is about 80 percent which can be improved further by implementing it with deep learning techniques and extracting images from documents can be done in future scope

BIBLIOGRAPHY

[1] B. Hadi and M. J. Kargar, "Plagiarism detection of flowchart images in the texts," 2017 3th International Conference on Web Research (ICWR), 2017, pp. 128-132, doi: 10.1109/ICWR.2017.7959317.

[2] P. Ovhal, "Detecting plagiarism in images," 2015 International Conference on Information Processing (ICIP), 2015, pp. 85-89, doi: 10.1109/INFOP.2015.7489356.

[3] Firas A. Jassim and Hind E. Qassim," Five Modulus Method for Image Compression", An International Journal (SIPIJ), Vol:3, No:5,2012.

[4] Wang Wen, Wang Yanb and Li Bingbing , "Research on Plagiarism IdentificationofDigitalImages",IEEE,2010.

[5] Harpreet Kaur and Neelofar Sohi," A Study for Applications of Histogram in Image Enhancement", The International Journal of Engineering and Science (IJES), Vol:6, Issue:6, PP:59-63,2017.

[6] Jithin S Kuruvila, Midhun Lal V L, Rejin Roy, Tomin Baby, Sangeetha Jamal and Sherly K K, "Flowchart Plagiarism Detection System: An Image Processing Approach", 7th International Conference on Advances in Computing Communications,2017.

APPENDIX

Views.py

```
#Importing_Libraries
from django.shortcuts import render
from django.template import RequestContext
from django.contrib import messages
import pymysql
from django.http import HttpResponse
from django.conf import settings
from django.core.files.storage import FileSystemStorage
import matplotlib.pyplot as plt
import cv2
import numpy as np
import os



image_files = []
image_data = []

def index(request):
    if request.method == 'GET':
        return render(request, 'index.html', {})

def Register(request):
    if request.method == 'GET':
        return render(request, 'Register.html', {})

def Login(request):
    if request.method == 'GET':
        return render(request, 'Login.html', {})
```

```
def UploadSuspiciousImage(request):
    if request.method == 'GET':
        return render(request, 'UploadSuspiciousImage.html', {})
```

#FMM algorithm

```
def FMM(name):#five modulus algorithm
    img = cv2.imread(name)
    img = cv2.resize(img,(50,50))
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    rows,cols = img.shape
    for i in range(rows):
        for j in range(cols):
            k = img[i,j]
            if (k % 5) == 4:
                img[i,j] = k + 1
            elif (k % 5) == 3:
                img[i,j] = k + 2
            elif (k % 5) == 2:
                img[i,j] = k - 2
            elif (k % 5) == 1:
                img[i,j] = k - 1
    for i in range(rows):
        for j in range(cols):
            k = img[i,j]
            k = k / 5
            img[i,j] = k
    temp = img.ravel()
    temp = np.min(img)
    for i in range(rows):
        for j in range(cols):
            if img[i,j] > 0:
                img[i,j] = img[i,j] - temp
```

#Plotting histogram

```
hist = cv2.calcHist([img], [0], None, [256], [0, 256])
    return hist
```

#Suspicious_Image_processing

```
def UploadSuspiciousImageAction(request):
    if request.method == 'POST' and request.FILES['t1']:
        output = ''
        myfile = request.FILES['t1']
        fs = FileSystemStorage()
        name = str(myfile)
        filename = fs.save(name, myfile)
        hist = FMM(name)
        os.remove(name)
        similarity = 0
        file = 'No Match Found'
        hist1 = 0
        for i in range(len(image_files)):
            metric_val = cv2.compareHist(hist, image_data[i], cv2.HISTCMP_INTERSECT)
            if metric_val > similarity:
                similarity = metric_val
                file = image_files[i]
                hist1 = image_data[i]
        output = '<table border=1 align=center><tr><th>Source Original Image
Name</th><th>Suspicious Image Name</th><th>Histogram Matching
Score</th><th>Plagiarism Result</th></tr>'
        result = 'No Plagiarism Detected'
        print(str(name)+" "+str(similarity))
        if similarity >= 2000:
            result = 'Plagiarism Detected'
        output+='<tr><td><font size="" color="white">'+file+'</td><td><font size=""
color="white">'+name+'</td>'
```

```
        output+='<td><font size="" color="white">'+str(similarity)+'</td><td><font size="" color="white">'+result+'</td></tr>'
        context= {'data':output}
        fig, ax = plt.subplots(2,1)
        ax[0].plot(hist1, color = 'b')
        ax[1].plot(hist, color = 'g')
        plt.xlim([0, 256])
        ax[0].set_title('Original image')
        ax[1].set_title('Suspicious image')
        plt.show()
        return render(request, 'SuspiciousImageResult.html', context)
```

#Source_image_processing

```
def UploadSourceImage(request):
    if request.method == 'GET':
        if len(image_files) == 0:
            for root, dirs, directory in os.walk('images'):
                for j in range(len(directory)):
                    hist = FMM(root+"/"+directory[j])
                    image_data.append(hist)
                    image_files.append(directory[j])
        output = '<table border=1 align=center><tr><th>Source Image File Name</th><th>Histogram Values</th></tr>'
        for i in range(len(image_files)):
            output+='<tr><td><font size="" color="white">'+image_files[i]+'</td><td><font size="" color="white">'+str(image_data[i])+"</td></tr>"
        context= {'data':output}
        return render(request, 'UploadSourceImage.html', context)
def UserLogin(request):
    if request.method == 'POST':
      username = request.POST.get('username', False)
      password = request.POST.get('password', False)
      index = 0
```

```
    con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root',
database = 'plagiarism',charset='utf8')
    with con:
        cur = con.cursor()
        cur.execute("select * FROM users")
        rows = cur.fetchall()
        for row in rows:
            if row[0] == username and password == row[1]:
                index = 1
                break
    if index == 1:
     file = open('session.txt','w')
     file.write(username)
     file.close()
     context= {'data':'welcome '+username}
     return render(request, 'UserScreen.html', context)
    else:
     context= {'data':'login failed'}
     return render(request, 'Login.html', context)

def Signup(request):
    if request.method == 'POST':
     username = request.POST.get('username', False)
     password = request.POST.get('password', False)
     contact = request.POST.get('contact', False)
     email = request.POST.get('email', False)
     address = request.POST.get('address', False)
     db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password
= 'root', database = 'plagiarism',charset='utf8')
     db_cursor = db_connection.cursor()
     student_sql_query = "INSERT INTO
users(username,password,contact_no,email,address)
VALUES('"+username+"','"+password+"','"+contact+"','"+email+"','"+address+"')"
     db_cursor.execute(student_sql_query)
```

```
    db_connection.commit()
    print(db_cursor.rowcount, "Record Inserted")
    if db_cursor.rowcount == 1:
     context= {'data':'Signup Process Completed'}
     return render(request, 'Register.html', context)
    else:
     context= {'data':'Error in signup process'}
     return render(request, 'Register.html', context)
```

URLS.py

```
from django.urls import path
from.import views
urlpatterns = [path("index.html", views.index, name="index"),
            path("Register.html", views.Register, name="Register"),
            path("Signup", views.Signup, name="Signup"),
            path("Login.html", views.Login, name="Login"),
            path("UserLogin", views.UserLogin, name="UserLogin"),
            path("UploadSource", views.UploadSource, name="UploadSource"),
            path("UploadSourceImage", views.UploadSourceImage,
name="UploadSourceImage"),
            path("UploadSuspiciousFile", views.UploadSuspiciousFile,
, name="UploadSuspiciousImage"),
            path("UploadSuspiciousImageAction", views.UploadSuspiciousImageAction,
name="UploadSuspiciousImageAction"),
```

Apps.py

```
from django.apps import AppConfig
class PlagiarismappConfig(AppConfig):
   name = 'PlagiarismApp'
```

#Manage.py

```
#!/usr/bin/env python
import os
import sys
if __name__ == '__main__':
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'Plagiarism.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)
```

#DEFAULT.CSS

```
body {
    margin: 0;
    padding: 0;
    background: #151515 url(images/img01.jpg) repeat-x left top;
    text-align: justify;
    font: 13px "Trebuchet MS", Arial, Helvetica, sans-serif;
    color: #7F7772;
}
form {
    margin: 0;
    padding: 0;
}
    fieldset {
    margin: 0;
    padding: 0;
```

```
    border: none;
}
input, textarea {
    padding: 5px;
    border: 1px solid #626262;
    font: normal 1em "Trebuchet MS", Arial, Helvetica, sans-serif;
}
h1, h1 a, h2, h2 a, h3, h3 a {
    margin: 0;
    text-decoration: none;
    font-weight: normal;
    color: #CF3822;
}
h1 {
    letter-spacing: -3px;
    font-size: 2.6em;
}
h2 {
    letter-spacing: -2px;
    font-size: 2em;
}
h3 {
    margin-bottom: 2em;
    font-size: 1em;
    font-weight: bold;
}
blockquote {
    margin: 0 0 0 1.5em;
    padding-left: 1em;
    border-left: 5px solid #DDDDDD;
}
a {
    color: #FFFFFF;
    text-decoration: none;}
```

```
a:hover {
    text-decoration: none;
    color: #FF5134;
}
img {
    border: none;
}
img.left {
    float: left;
    margin: 8px 20px 0px 0px;
    border: 2px solid #434343;
}
#registersuccess{
    margin-top: 10px;
    text-align: center;
    text-transform: uppercase;
    font-family: 'Arvo', serif;
    font-size: 15px;
}

/* Header */
#wrapper {
}
#header {
    width: 900px;
    height: 170px;
    margin: 0 auto;
    background: url(images/img02.jpg) no-repeat right top;
}
/* Menu */
#menu {
    width: 1020px;
    height: 60px;
```

```
    margin: 0 auto;
}
#menu ul {
    margin: 0;
    padding: 0;
    list-style: none;
}
#menu li {
    display: inline;
    text-transform: uppercase;
}
#menu a {
    display: block;
    float: left;
    background: url(images/img03.gif) no-repeat left 75%;
    padding: 18px 30px 0 15px;
    text-decoration: none;
    font-family: Arial, Helvetica, sans-serif;
    font-size: 1em;
    font-weight: bold;
    color: #FFFFFF;
}
#menu a:hover {
    color: #FF5134;
}
#menu .current_page_item {
}
#menu .current_page_item a {
    color: #FFFFFF;
}
/* Page */
#page {
    width: 890px;
    margin: 0 auto;
```

```
    padding: 30px 0;
}
/* Content */
#content {
    float: left;
    width: 900px;
}
.post {
    margin-bottom: 50px;
    border-bottom: 1px #A90000 dashed;
}
.post .title {
}
.post .title h2 {
    font-size: 2.2em;
    color: #CF3822;
}
.post .title p {
    margin: 0;
    line-height: normal;
    color: #BABABA;
}
.post .title p a {
    color: #880A0B;
}
.post .entry {
    padding-top: 30px;
}
.post .links {
    width: 410px;
    height: 29px;
    margin: 0;
    padding: 6px 0 0 20px;
}
```

```
.post .links a {
    padding: 0 23px;
    text-decoration: none;
    font-weight: bold;
    color: #CF3822;
}
.post .links a:hover {
    text-decoration: underline;
    color: #CF3822;
}
.post .links .more {
    background: url(images/img07.gif) no-repeat;
}
.post .links .comments {
    background: url(images/img08.gif) no-repeat;
}
/* Footer */
#footer {
    clear: both;
    width: 950px;
    margin: 0 auto;
    padding: 30px 0;
    border-top: 2px solid #2B2B2B;
}

#footer p {
    margin: 0 0 5px 0;
    text-align: center;
    line-height: normal;
    font-size: .9em;
}
#footer a {
    text-decoration: none;
}
```

#DATABASE

```
create database plagiarism;
use plagiarism;

create table users(username varchar(50),
password varchar(50),
contact_no  varchar(12),
email  varchar(50),
address  varchar(50));
```

INDEX.HTML

```
{% load static %}
<html>
<head>
<title>IMAGE PLAGIARISM DETECTION</title>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<link href="{% static 'default.css' %}" rel="stylesheet" type="text/css" media="screen"
/>
</head>
<body>
<div id="wrapper">
  <div id="header">
    <div id="logo">
      <h1><font color="orange" size=6>IMAGE PLAGIARISM
DETECTION</font></h1>
        <marquee><font color="pink" size=4>IMAGE PLAGIARISM
DETECTION</font></marquee>
    </div> </div>
  </div>
  <div id="menu">
     <ul>
      <li><a href="{% url 'index' %}">Home</a></li>
      <li><a href="{% url 'Login' %}">Login</a></li>
```

```
        <li><a href="{% url 'Register' %}">New User Signup Here</a></li> </ul>
      </div>
      <div id="page">
       <div id="content">
         <div class="post">
          <div class="title">
            <h2>IMAGE PLAGIARISM DETECTION </h2>
           </div>
          <div class="entry">
                  <br/><br/><br/>
    {{ data }}
          <br>
            <p><img src="{% static 'images/images.jpg' %}" alt="" width="890" height="200"
    class="left" /><p></p><font size="4" color="white">
            <center><br/>About Plagiarism Detection </center></font></p>
            <p><font size="" color="white">Image Plagiarism using compressed images</p>
            </div>
          </div>
      </div>
  </div>
</body> </html>
```

USERSCREEN.HTML

```
{% load static %}
<html>
<head>
<title>TEXT and IMAGE Plagiarism Detection</title>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<link href="{% static 'default.css' %}" rel="stylesheet" type="text/css" media="screen" />
</head>
<body>
<div id="wrapper">
  <div id="header">
    <div id="logo">
```

```
    <h1>
<font color="orange" size=6>TEXT and IMAGE Plagiarism Detection</font></h1>
            <marquee>
<font color="pink" size=4>TEXT and IMAGE Plagiarism Detection</font>
</marquee>
   </div>
  </div>
 </div>
 <div id="menu">
  <ul>
   <li>
<a href="{% url 'UploadSource' %}">View List of Source Files</a></li>
   <li>
<a href="{% url 'UploadSuspiciousFile' %}">Upload Suspicious File</a></li>
            <li><a href="{% url 'UploadSourceImage' %}">View List of Sourc
Images</a></li>
            <li>
<a href="{% url 'UploadSuspiciousImage' %}">Upload Suspicious Image</a></li>
            <li>
<a href="{% url 'index' %}">Logout</a></li>
     </ul>
 </div>
 <div id="page">
  <div id="content">
   <div class="post">
    <div class="title">
     <h2>TEXT and IMAGE Plagiarism Detection</h2>
    </div>
    <div class="entry">
                <br/><br/><br/>
<font size="" color="white">{{ data|safe }}
          <br>
         <p>
<img src="{% static 'images/images.jpg' %}" alt="" width="890" height="200" class="left"
```

```
/><p>
</p>
<font size="4" color="white">
          </div>
        </div>
      </div> </div> </body>
</html>
```