TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	
1	**INTRODUCTION**	1
	1.1 Introduction	2
	1.2 Purpose of the Project	3
2	**BACKGROUND WORK**	
	2.1 Background	5
3	**EXISTING SYSTEM**	
	3.1 Existing System	6
4	**PROPOSED SYSTEM**	
	4.1 Proposed Solution	7
5	**DESIGN**	
	5.1 UML Diagrams	14
6	**RESULTS AND DISCUSSION**	
	6.1 Input & Output Images	15
7	**CONCLUSION AND FUTURESCOPE**	
	7.1 Scope	16
8	**REFERENCES**	17

Abstract

PROBLEM TITLE:

Color detection using OpenCV and Pandas

ABSTRACT:

Colour detection is the process of detecting the name of any colour. For humans this is an extremely easy task but for computers, it is not straightforward. Human eyes and brains work together to translate light into colour. Light receptors that are present in our eyes transmit the signal to the brain. Our brain then recognizes the colour. Since childhood, we have mapped certain lights with their colour names.

We will be using the somewhat same strategy to detect colour names. In this colour detection Python project, we are going to build an application through which you can automatically get the name of the colour by clicking on them. So for this, we will have a data file that contains the colour name and its values. Then we will calculate the distance from each colour and find the shortest one. We will be using the computer vision library of Python that is OpenCV and Pandas. In this way we can extract colour RGB values and the colour name of a pixel.

CHAPTER 1
INTRODUCTION

1.1 Introduction

Color (American English) or **color** (Commonwealth English) is the visual perceptual property corresponding in humans to the categories called *red*, *blue*, *yellow*, etc. Color derives from the spectrum of light (distribution of light power versus wavelength) interacting in the eye with the spectral sensitivities of the light receptors. Color categories and physical specifications of color are also associated with objects or materials based on their physical properties such as light absorption, reflection, or emission spectra. By defining a color space color can be identified numerically by their coordinates.

Because perception of colour stems from the varying spectral sensitivity of different types of cone cells in the retina to different parts of the spectrum, colours may be defined and quantified by the degree to which they stimulate these cells. These physical or physiological quantifications of color, however, do not fully explain the psychophysical perception of color appearance.

The science of colour is sometimes called *chromatics*, *colorimetry*, or simply *color science*. It includes the perception of color by the human eye and brain, the origin of color in materials, color theory in art, and the physics of electromagnetic radiation in the visible range (that is, what is commonly referred to simply as *light*).

Most light sources emit light at many different wavelengths; a source's *spectrum* is a distribution giving its intensity at each wavelength. Although the spectrum of light arriving at the eye from a given direction determines the color sensation in that direction, there are many more possible spectral combinations than color sensations. In fact, one may formally define a color as a class of spectra that give rise to the same color sensation, although such classes would vary widely among different species, and to a lesser extent among individuals within the same species.

RGB VALUES:

A colour's RGB value indicates **its red, green, and blue intensity**. Each intensity value is on a scale of 0 to 255, or in hexadecimal from 00 to FF. RGB values are used in HTML, XHTML, CSS, and other web standards.

RGB refers to a system for representing the colors to be used on a computer display. RGB is a combination of Red, Green and Blue. These colors can be combined in various proportions to obtain any color in the visible spectrum.

Each level is measured by the range of decimal numbers from 0 to 255 (256 levels for each color). For example, if a color has zero Blue, it will be a mixture of Red and Green. This means we can generate 256 x 256 x 256 = 16.777.216 different colors with this model.

The colors of the visible light spectrum[1]

Color	Wavelength interval	Frequency interval
Red	~ 700–635 nm	~ 430–480 THz
Orange	~ 635–590 nm	~ 480–510 THz
Yellow	~ 590–560 nm	~ 510–540 THz
Green	~ 560–520 nm	~ 540–580 THz
Cyan	~ 520–490 nm	~ 580–610 THz
Blue	~ 490–450 nm	~ 610–670 THz
Violet	~ 450–400 nm	~ 670–750 THz

1.2 Purpose of the Project:

Colour detection is the process of detecting the name of any color.
Well, for humans this is an extremely easy task but for computers, it is not straightforward. Human eyes and brains work together to translate light into color. Light receptors that are present in our eyes transmit the signal to the brain. Our brain then recognizes the color. Since childhood, we have mapped certain lights with their color names.

This project helps us by detecting the colour present in the image along with the RGB values of that particular point. Many graphic designers and web designers will understand how RGB values can be helpful.

Most of the machine learning and artificial intelligence fields are strongly connected to Computer Vision. This project helps very much in progressing of computer vision.

CHAPTER 2
BACKGROUND WORK

PYTHON LIBRARIES:

Three of the most popular python libraries will be used in this project. Python has a lot of libraries which in various ways. We will be using three libraries in this project.

- OpenCV
- Pandas
- NumPy

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception

he library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scenes.

Pandas is an open-source library built on top of NumPy providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. It allows for fast analysis and data cleaning and preparation.

NumPy is the fundamental package for scientific computing in Python. ... NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.

CHAPTER 3
EXISTING SYSTEM

The existing System is nothing but identifying the colors with eyes. Human eyes and brains work together to translate light into color. Light receptors that are present in our eyes transmit the signal to the brain. Our brain then recognizes the color. Since childhood, we have mapped certain lights with their color names. But we cannot identify every color and it has many disadvantages.

DISADVANTAGES:

- Could only identify a smaller number of colors.
- People with color vision deficiency will not be able to identify exactly.
- There are 1000's of colors which humans cannot remember everything.

CHAPTER 4
PROPOSED SYSTEM

USING OPENCV & PANDAS LIBRARIES:

- This System uses Python Libraries and detects the color of the image accurately along with the RGB values of that location.
- This System uses a Dataset which consists of 1000s of color names along with RGB values.

Color Detection Using Python**: Data Set**

Colors are made up of 3 primary colors; red, green, and blue. In computers, we define each color value within a range of 0 to 255. There are approximately 16.5 million different ways to represent a color. In our dataset, we need to map each colour's values with their corresponding names. But, we don't need to map all the values. We will be using a dataset that contains RGB values with their corresponding names.

COLOURS.CSV FILE

alice_blue	Alice Blue	#f0f8ff	240	248	255
alizarin_crimson	Alizarin Crimson	#e32636	227	38	54
alloy_orange	Alloy Orange	#c46210	196	98	16
almond	Almond	#efdecd	239	222	205
amaranth	Amaranth	#e52b50	229	43	80
amber	Amber	#ffbf00	255	191	0

Taking an image from the user:

We will be using argparse library to create an argument parser.

```
import argparse

ap = argparse.ArgumentParser()
ap.add_argument('-i', '--image', required=True, help="Image Path")
args = vars(ap.parse_args())
img_path = args['image']
#Reading image with opencv
Img = cv2.imread(img_path)
```

Teaching the Colors:

import *colors.csv* file to our program using *read_csv* method. Since the csv file we downloaded doesn't have column names, I will be defining them in the program. This process is known as data manipulation.

```
index=["color", "color_name", "hex", "R", "G", "B"]

csv = pd.read_csv('colors.csv', names=index, header=None)
```

The pandas library is very useful when we need to perform various operations on data files like CSV. **pd.read_csv()** reads the CSV file and loads it into the pandas Data Frame.

```
index=["color","color_name","hex","R","G","B"]
csv = pd.read_csv('colors.csv', names=index, header=None)
```

Mouse Click Function:

- A callback function which will be called when a mouse event happens.

```
cv2.namedWindow('image')
cv2.setMouseCallback('image',draw_function)
```

Calculating the RGB values of the double-clicked location:

It will calculate the rgb values of the pixel which we double click. The function parameters have the event name, (x,y) coordinates of the mouse position, etc. In the function, we check if the event is double-clicked then we calculate and set the r,g,b values along with x,y positions of the mouse.

```
def draw_function(event, x,y,flags,param):
    if event == cv2.EVENT_LBUTTONDBLCLK:
        global b,g,r,xpos,ypos, clicked
        clicked = True
        xpos = x
        ypos = y
        b,g,r = img[y,x]
        b = int(b)
        g = int(g)
        r = int(r)
```

Calculate distance to get color name

We have the r,g and b values.

To get the color name, we calculate a distance(d) which tells us how close we are to color and choose the one having minimum distance.

Distance =

d = abs(Red – ithRedColor) + (Green – ithGreenColor) + (Blue – ithBlueColor)

Display image on the window:

Finally, when a mouse-click happens, it will update the color name and RGB values on the screen. By using **cv2.imshow(),** we draw the image on the window. When the user double clicks the window, we draw a rectangle and get the color name to draw text on the window using **cv2.rectangle** and **cv2.putText()** functions.

SOURCE CODE:

```
import cv2
import pandas as pd
num=int(input("1:temple\n2:F\n3:cmr\n"))
img_path  =r'C:/Users/ginugaripavan/OneDrive/Desktop/hindu-temple-gopuram,2618304.jpg'
```

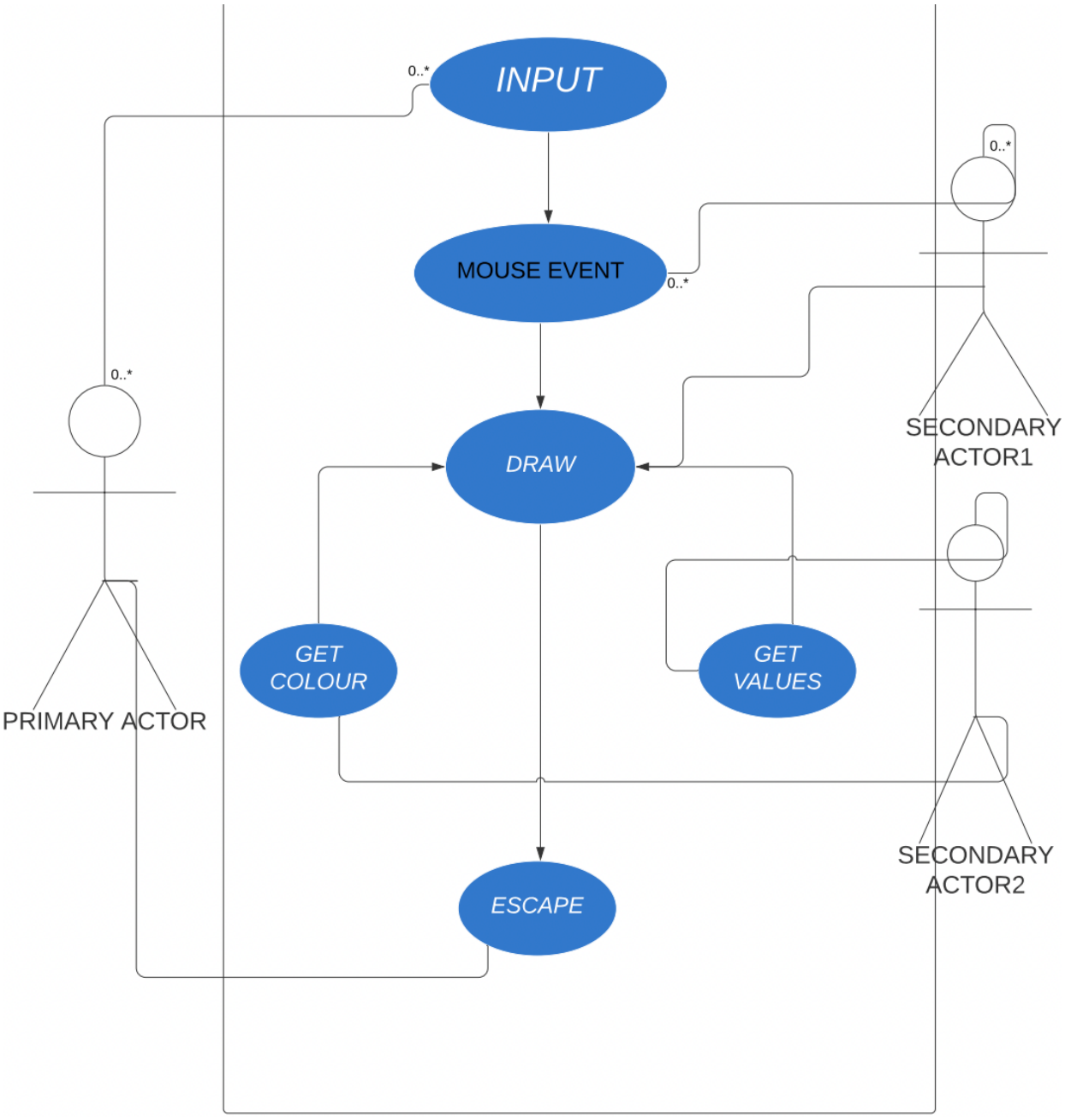
```
img = cv2.imread(img_path)

# declaring global variables (are used later on)
clicked = False
r = g = b = x_pos = y_pos = 0

# Reading csv file with pandas and giving names to each column
index = ["color", "color_name", "hex", "R", "G", "B"]
csv = pd.read_csv('colors.csv', names=index, header=None)


# function to calculate minimum distance from all colors and get the most matching color
def get_color_name(R, G, B):
    minimum = 10000
    for i in range(len(csv)):
        d = abs(R - int(csv.loc[i, "R"])) + abs(G - int(csv.loc[i, "G"])) + abs(B - int(csv.loc[i, "B"]))
        if d <= minimum:
            minimum = d
            cname = csv.loc[i, "color_name"]
```

```
    return cname


# function to get x,y coordinates of mouse double click
def draw_function(event, x, y, flags, param):
    if event == cv2.EVENT_LBUTTONDBLCLK:
        global b, g, r, x_pos, y_pos, clicked
        clicked = True
        x_pos = x
        y_pos = y
        b, g, r = img[y, x]
        b = int(b)
        g = int(g)
        r = int(r)



cv2.namedWindow('image')
cv2.setMouseCallback('image', draw_function)

while True:
```

```
    cv2.imshow("image", img)
    if clicked:
        cv2.rectangle(img, (20, 20), (750, 60), (b, g, r), -1)
        text = get_color_name(r, g, b) + ' R=' + str(r) + ' G=' + str(g)
+ ' B=' + str(b)
cv2.putText(img,text,start,font(0-
7),fontScale,color,thickness,lineType)
        cv2.putText(img, text, (50, 50), 2, 0.8, (255, 255, 255), 2,
cv2.LINE_AA)
        if r + g + b >= 600:
            cv2.putText(img, text, (50, 50), 2, 0.8, (0, 0, 0), 2,
cv2.LINE_AA)
        clicked = False
    if cv2.waitKey(20) & 0xFF == 27:
        break
cv2.destroyAllWindows()
```

CHAPTER 5
DESIGN

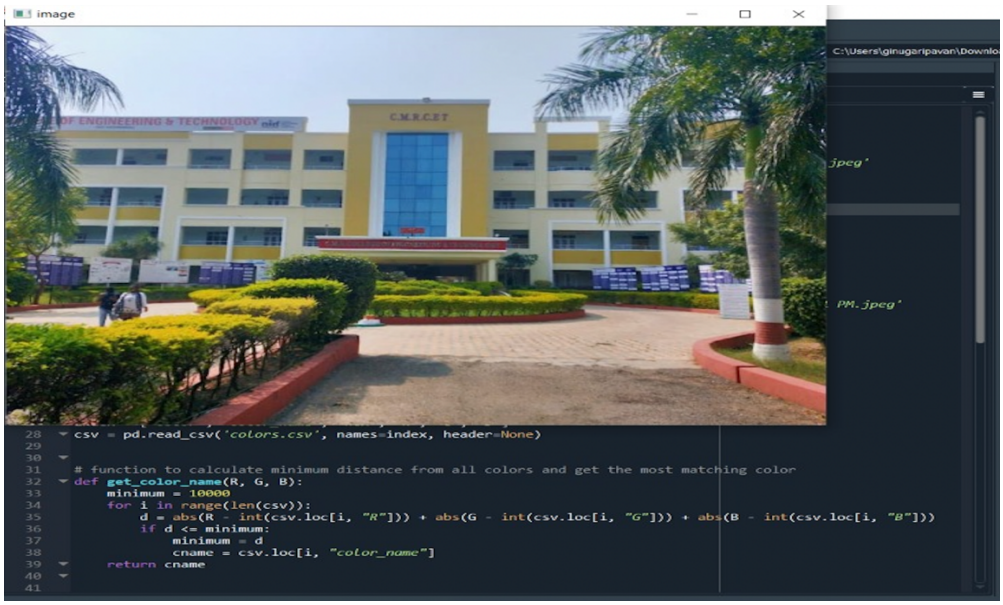
5.1 UML DIAGRAMS:

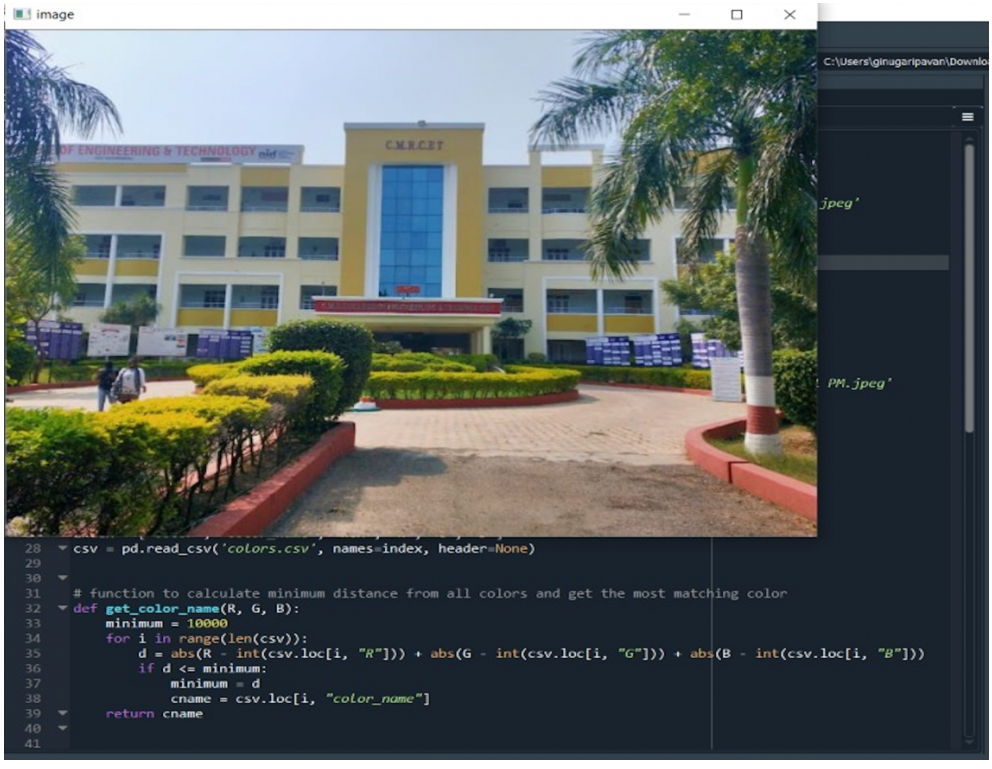


UML DIAGRAM FOR COLOUR DETECTION

CHAPTER 6
RESULT

- After giving the image as input, the color name and RGB values of the double-clicked point will be displayed on the screen.





CHAPTER 7
SCOPE

- In this design, we are implementing a color recognizer using python i.e., color detection. This basic application takes an image as input and returns the color name and RGB values of it, which will be very helpful.
- Many graphic designers and web designers will get to understand RGB values and use them in computer vision.
- We can improve the accuracy of the color detection by using color correction i.e., by using pantone's color cards.
- We can make this useful to people who suffer with color vision deficiency and make them recognize colors efficiently.

CHAPTER 8
REFERENCES

- **https://en.wikipedia.org/wiki/Color**
- **https://en.wikipedia.org/wiki/OpenCV**
- **https://en.wikipedia.org/wiki/Pandas**