# Shade Sync

## MINI PROJECT (01CE1705) REPORT

***Submitted by :***

**Aum Pitroda**

## 92000103148

**BACHELOR OF TECHNOLOGY**

***in***

## Computer Engineering



**Marwadi University, Rajkot**

**December, 2023**



**Mini Project (01CE1705) Marwadi University Faculty of Technology**

Computer Engineering Department

**2023-24**

# CERTIFICATE

This is to certify that the project report submitted along with the project entitled Shade Synchas been carried out by Aum Pitroda (92000103148) under my guidance in partial fulfillment for the degree of Bachelor of Technology in Computer Engineering, 7th Semester of Marwadi University, Rajkot during the academic year 2023-24.

Sign: Sign:

**Prof. Chetankumar Chudasama Prof. Hardik Doshi**

Internal Guide Head of the Department

# Undertaking about Originality of Work

We hereby certify that we are the sole authors of this Internship report and that neither any part of this Internship project report nor the whole of the Internship Project report has been submitted for a degree by other student to any other University or Institution.

We certify that, to the best of our knowledge, the current Internship/Project report does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations or any other material from the work of other people included in our Internship/Project report, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that we have included copyrighted material that surpasses the boundary of fair dealing within the meaning of the Indian Copyright (Amendment) Act 2012, we certify that we have obtained a written permission from the copyright owner(s) to include such material(s) in the current IDP/UDP Project report and have included copies of such copyright clearances to our appendix.

We have checked the write up of the present Internship/Project report using an anti-plagiarism database and it is within the allowable limit. In case of any complaints pertaining to plagiarism, we certify that we shall be solely responsible for the same and we understand that as per norms, University can even revoke BE degree conferred upon the student(s) submitting this Internship/Project report, in case it is found to be plagiarised.

**Team:**

|  |  |  |
| --- | --- | --- |
| **Enrolment Number** | **Name** | **Signature** |
| 92000103148 | Aum Pitroda |  |

Place: Rajkot Date:

# Index

[Acknowledgments i](#_TOC_250019)

[Abstract ii](#_TOC_250018)

1. [Introduction 1](#_TOC_250017)
2. [Technology used and Implementation Strategy](#_TOC_250013) 3
3. [Implementation Snapshot](#_TOC_250010) 5
4. [Conclusion](#_TOC_250005) 14

[References](#_TOC_250000) 15

# Acknowledgments

This mini project would not have been possible without the Guidance and the help of several Individuals who in some ways contributed their valuable time and assistance in the completion of this Internship.

I would like to thank our college **Marwadi University** for giving us this opportunity of doing Internship and gaining Industry Level Experience in the field that we are interested in. My sincere gratitude to our Head of The Department (HOD), **Prof. Hardik Doshi**, our Internal Guide, **Prof. Chetankumar Chudasama** for helping and solving our queries whenever required.

# 

# Abstract

In today's design-centric landscape, where aesthetics plays a paramount role, the "ShadeSync" project emerges as a transformative endeavor. This mini-project is dedicated to the development of a powerful and intuitive tool that simplifies and refines the art of color palette creation by synchronizing with the colors found within an input image.

ShadeSync is not just about generating color palettes; it's about forging a seamless connection between visual inspiration and design realization. This innovative solution harnesses the potential of image analysis, machine learning, and color theory to automatically extract a balanced and harmonious color palette from an image.

The project's key components include:

**1. Image Analysis:** ShadeSync employs cutting-edge image analysis techniques to identify the dominant colors, gradients, and overall color distribution within the input image.

**2. Color Harmony:** The extracted colors are rigorously analyzed for harmony, ensuring that the generated palette synchronizes with the image's aesthetics.

**3. User Interaction:** The project offers an intuitive interface that allows designers to fine-tune the generated palette, enabling a perfect synchronization with their creative vision.

Export and Integration: Designers can easily export the synchronized palette in a hex color format and download the generated palette in various web-development & creative works friendly format­­s(CSS, SCSS, JSON) making integration into design projects a breeze.

ShadeSync empowers designers, creators, and artists to effortlessly translate visual inspiration into stunning design reality. The project is positioned at the intersection of technology and design, embodying a new era where creative ideation effortlessly synchronizes with visual execution.

The name "ShadeSync" epitomizes the project's essence - it's about seamlessly harmonizing the shades found in imagery with the creative aspirations of designers. In doing so, ShadeSync not only expedites the design process but also enhances the overall quality of the creative output. This project is poised to redefine how we approach color palettes, making the art of synchronization the cornerstone of inspired and impactful design.

# Introduction

# 

# In the realm of contemporary design, color is not merely a visual element but an emotional and experiential language. Picture a world devoid of vibrant shades, where websites lack the power to captivate, marketing campaigns fail to resonate, and living spaces remain devoid of personality. In this modern, design-centric age, color is more than an aesthetic choice; it is an intrinsic part of storytelling and expression. Against this backdrop, the "ShadeSync" project emerges, driven by the aspiration to simplify the intricate craft of color palette creation and to unite it seamlessly with the essence of images.

# The creative process in design, be it in web development, graphic design, fashion, or interior decor, invariably confronts the challenge of curating harmonious color palettes. In recognition of this universal design imperative, ShadeSync takes center stage. By harmonizing technology and design, it reimagines the act of color palette creation, placing synchronization with the essence of images at the core.

# ShadeSync emerges as a groundbreaking system that artfully and efficiently synchronizes color palettes with the vivacious hues found within images. Leveraging advanced techniques in image analysis, machine learning, and color theory, ShadeSync automatically extracts, harmonizes, and presents a unified color palette inspired by the image itself. This project report undertakes a comprehensive exploration of ShadeSync's development, methodology, and its potential to transform the creative landscape.

* 1. **Motivation**

The motivation behind ShadeSync is deeply rooted in the recognition that crafting harmonious color palettes is a common challenge encountered by designers across diverse fields. Whether one is engaged in web design, graphic artistry, fashion, or interior decoration, the endeavor to create a visually cohesive and emotionally resonant palette is universal. The motivation for ShadeSync is to provide designers and creators with a tool that simplifies this complex process, streamlining the path to a harmonious color scheme and enhancing creative outcomes.

* 1. **Problem Statement**

# Designers often grapple with the complexities of color selection and harmonization. The sheer diversity and nuances of colors found within images can overwhelm, making it difficult to distill a coherent and aesthetically pleasing palette. Traditional methods for color palette creation are time-consuming and often rely heavily on manual intervention, which can be cumbersome and subjective. The problem ShadeSync addresses is to automate and refine the process, synchronizing color palettes with the colors found within an image to save time and promote visual harmony.

* 1. **Objectives**

1. Develop an automated system capable of extracting and harmonizing a unified color palette from an input image.
2. Implement advanced image analysis, machine learning, and color theory techniques to facilitate the automatic generation of harmonious color palettes.
3. Simplify and expedite the creative process by providing designers with a tool that seamlessly synchronizes color palettes with the vibrant hues found within images.
4. Enhance the efficiency of color palette creation, enabling designers to save time and effort while promoting visual harmony in their projects.
5. Offer a practical and innovative solution for designers across various fields, including web design, graphic artistry, fashion, and interior decoration, to streamline and improve their creative outcomes.
   1. **Scope**

The scope of the "ShadeSync" project encompasses the design and development of a web-based application that facilitates the creation of harmonious color palettes based on uploaded images. The project includes the following key components:

1. **Image Processing:** Users can upload images of various formats.
2. **Color Palette Generation:** The project employs the K-Means clustering algorithm to identify and extract dominant colors from the uploaded image.
3. **User Interface:** The application features a user-friendly two-page interface. The first page allows users to upload an image and select the desired number of colors for the palette. The second page displays the uploaded image alongside the generated color palette.
4. **Download Options:** Users can download the color palette in CSS, SCSS, or JSON format.
5. **Color Format Conversion:** The project includes a function to convert RGB color values to hexadecimal format, making them compatible with CSS.
6. **User Interaction:** Users can interact with the image and color palette, allowing for customization and exploration.
   1. **Summary**

In the pages that follow, this project report embarks on an in-depth exploration of ShadeSync. We will delve into its development, methodology, and its potential to redefine color palette creation. By harnessing the power of technology and design, ShadeSync stands as a guiding light in the convergence of creative inspiration and efficient design realization, offering designers, artists, and creators a tool that bridges the gap between vision and reality.

# Technology Used and Implementation Strategy

**2.1 Technology Stack:**

* **Python:** The core programming language for implementing the project.
* **Flask:** A lightweight web framework used for building the web application.
* **Pillow (PIL):** A Python Imaging Library used to open, process, and analyze images.
* **NumPy:** A Python library for numerical operations, particularly for array and matrix manipulations.
* **JavaScript, HTML, CSS:** For building the front-end user interface and interactivity.
* **K-Means Clustering Algorithm:** Used to group similar colors and generate color palettes.

**2.2 Implementation Strategy:**

**2.2.1 Front-end Development:**

* **HTML/CSS*:*** The user interface is designed using HTML and styled with CSS to create a visually appealing and user-friendly interface. It includes two pages: one for uploading an image and selecting the number of colors, and another for displaying the image and the generated color palette.
* **JavaScript:** JavaScript is used to handle user interactions, such as image upload and color palette download requests.

**2.2.2 Image Processing:**

**Pillow (PIL):** The project uses Pillow to open and manipulate the uploaded image. The image is resized to a smaller dimension (200x400 pixels) for efficiency, converted to the RGB color mode, and analyzed to extract color data.

**2.2.3 Color Palette Generation:**

**K-Means Clustering:** The heart of the project lies in the K-Means clustering algorithm. This algorithm groups similar colors together to generate a coherent color palette. It is applied to the image's color data to identify dominant colors.

**2.2.4 Back-end (Flask):**

* **Flask Web Framework:** Flask is employed to create a simple yet powerful web application. It handles user requests, file uploads, form submissions, and serves the web pages.
* **Jinja2 Templates:** Jinja2 is used to render dynamic content, enabling the presentation of the uploaded image, generated color palette, and download links.

**2.2.5 User Experience:**

* Users are presented with a straightforward two-page interface. On the first page, they can upload an image and specify the number of colors they want in the palette. After submission, the Flask back-end processes the image, applies the K-Means algorithm, and redirects the user to the second page.
* On the second page, the uploaded image is displayed along with the generated color palette. Users can interact with the image and choose to download the color palette in CSS, SCSS, or JSON format.

**2.2.6 Color Format Conversion:**

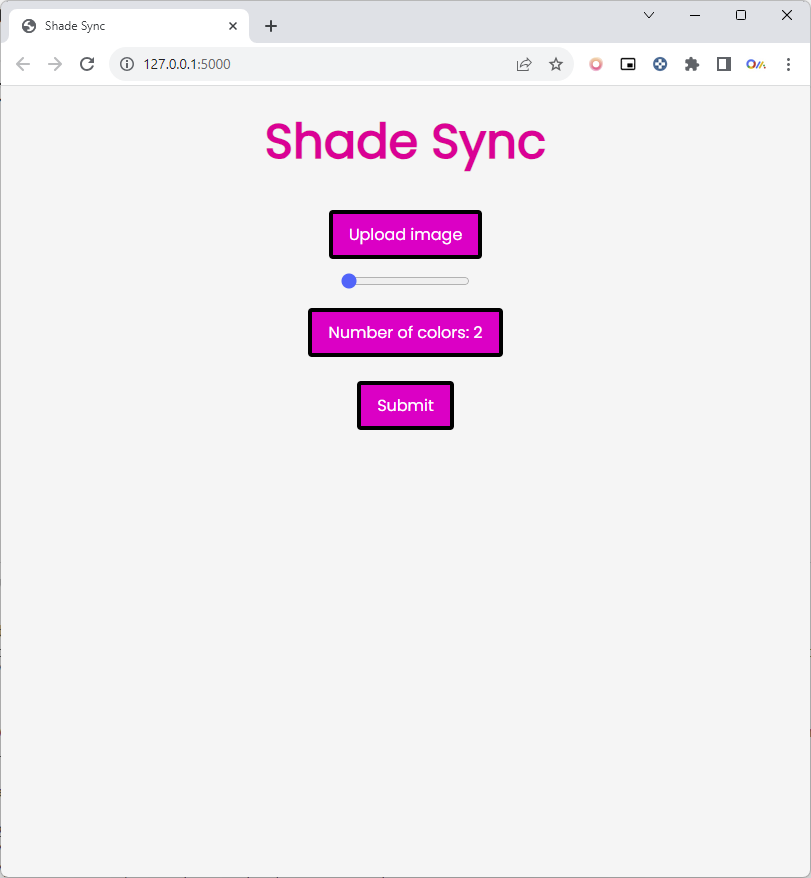
* The project includes a function to convert RGB color values to hexadecimal format for CSS compatibility.

**2.2.7 Color Palette Presentation:**

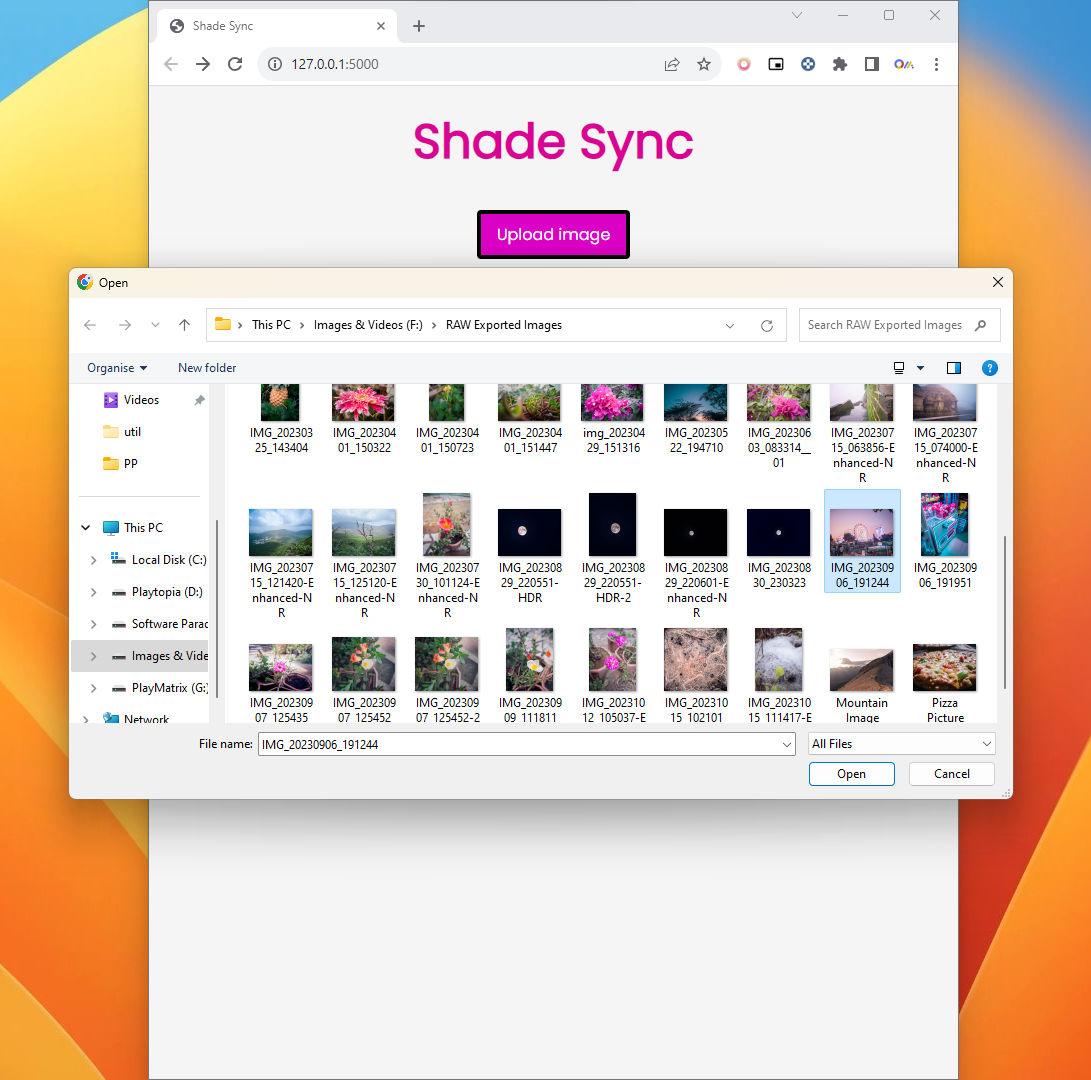
* The color palette is presented to the user as a horizontal list of color swatches below the displayed image.

# Implementation Snapshot

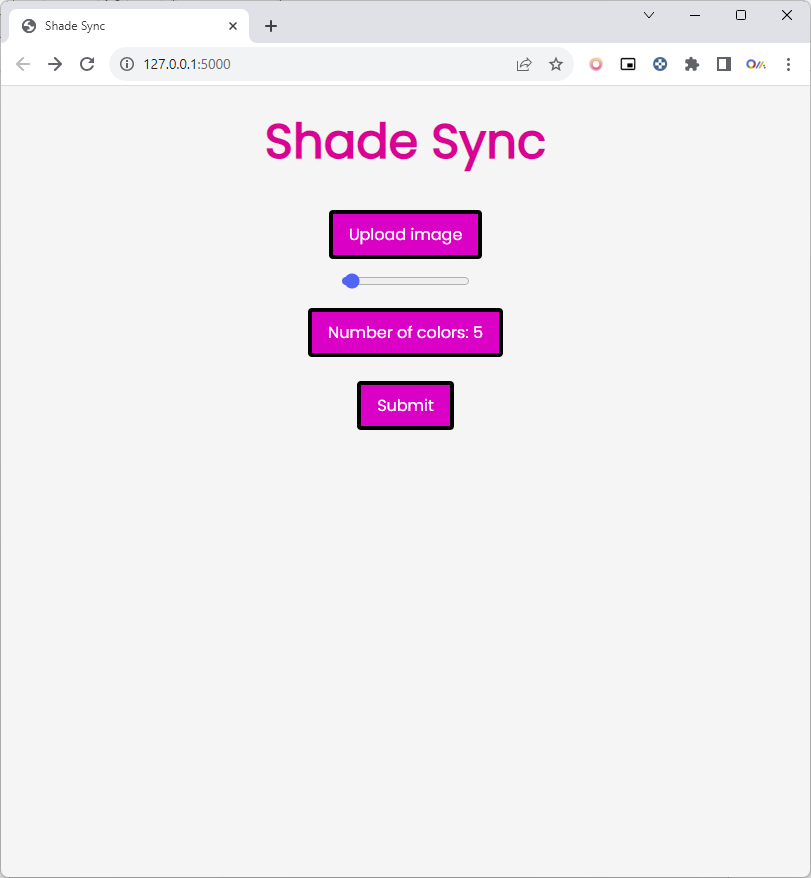
* 1. **Main Window**



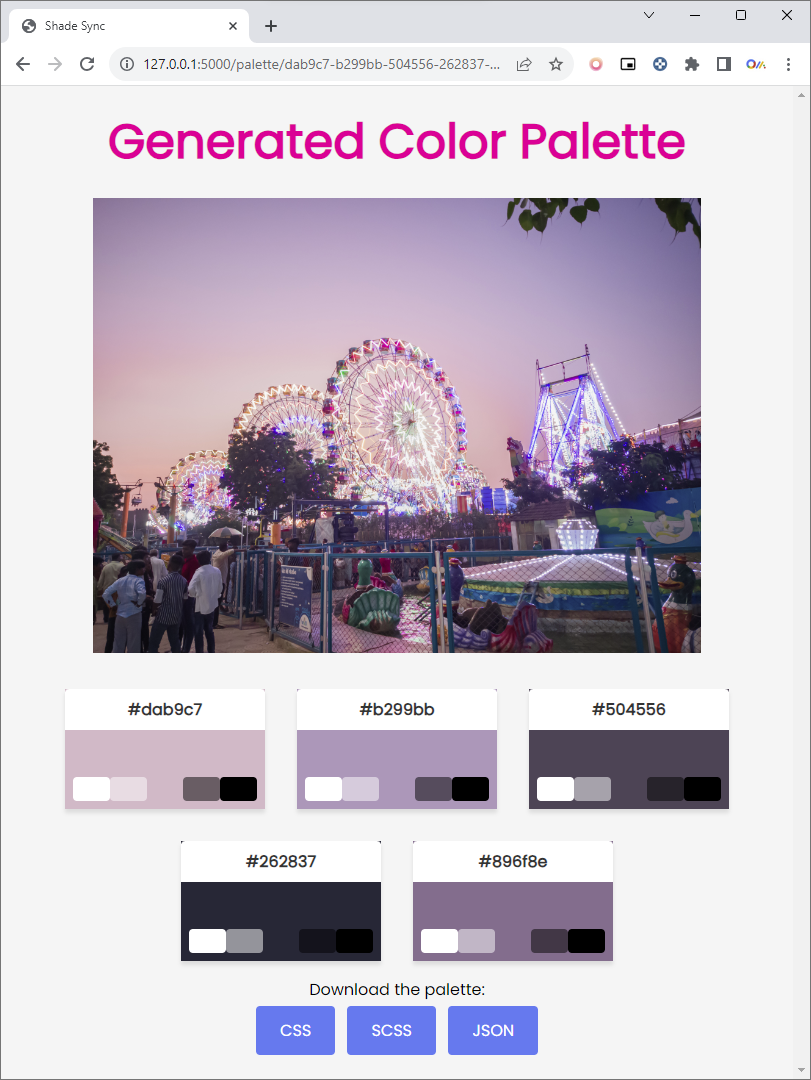
* + 1. Upload Image button for uploading the image to server for which color palette needs to be generated



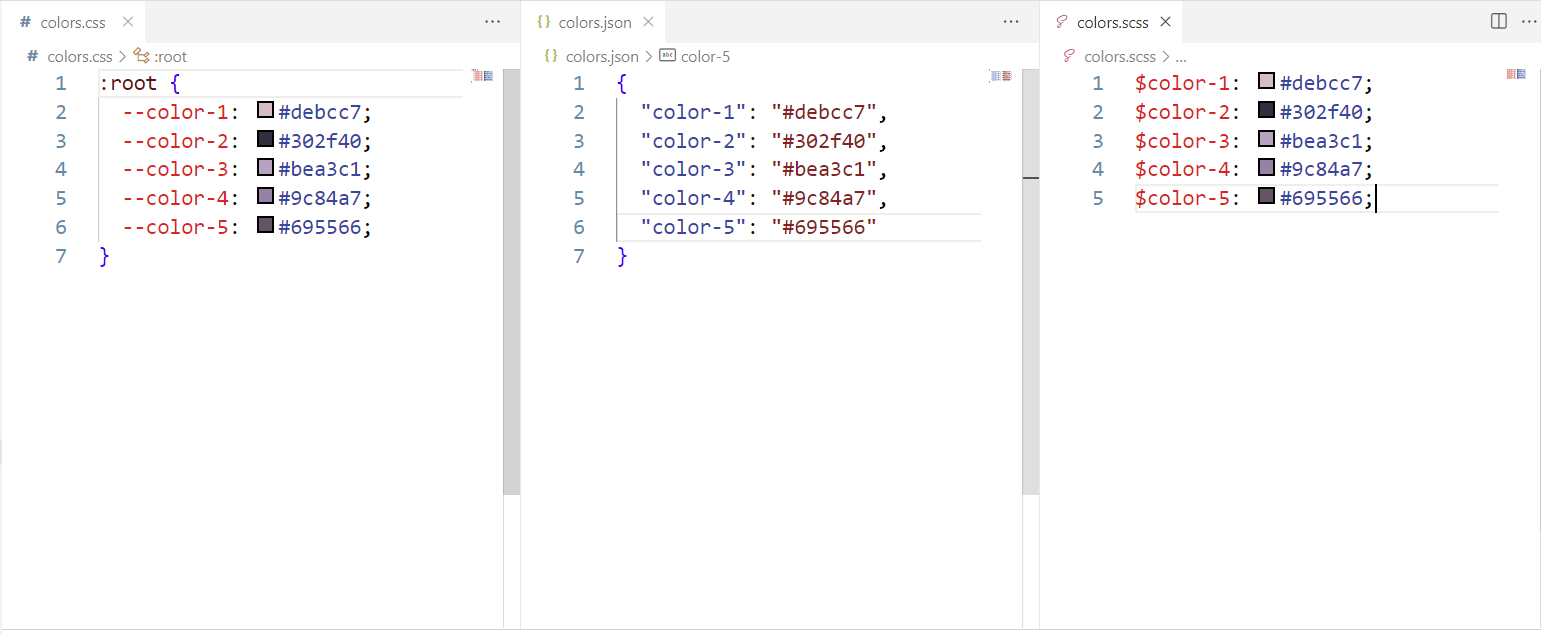
* + 1. Selection of No. of Colors to Generate Palette
    2. Click on “Submit” button



* 1. **Generated Color Palette for uploaded image**



* + 1. Downloading Color Palette in “CSS”, “SCSS”, “JSON” format



* 1. **K-Means Algorithm Implementation**

from math import sqrt

import random

from PIL import Image

import numpy as np

def get\_points(image\_path):

img = Image.open(image\_path)

img.thumbnail((200, 400))

img = img.convert("RGB")

w, h = img.size

points = []

for count, color in img.getcolors(w \* h):

for \_ in range(count):

points.append(Point(color))

return points

def euclidean(p, q):

n\_dim = len(p.coordinates)

return sqrt(sum([

(p.coordinates[i] - q.coordinates[i]) \*\* 2 for i in range(n\_dim)

]))

class Point:

def \_\_init\_\_(self, coordinates):

self.coordinates = coordinates

class Cluster:

def \_\_init\_\_(self, center, points):

self.center = center

self.points = points

class KMeans:

def \_\_init\_\_(self, n\_clusters, min\_diff=3):

self.n\_clusters = n\_clusters

self.min\_diff = min\_diff

init: 'kmeans++'

def calculate\_center(self, points):

n\_dim = len(points[0].coordinates)

vals = [0.0 for i in range(n\_dim)]

for p in points:

for i in range(n\_dim):

vals[i] += p.coordinates[i]

coords = [(v / len(points)) for v in vals]

return Point(coords)

def assign\_points(self, clusters, points):

point\_lists = [[] for i in range(self.n\_clusters)]

for p in points:

smallest\_distance = float('inf')

for i in range(self.n\_clusters):

distance = euclidean(p, clusters[i].center)

if distance < smallest\_distance:

smallest\_distance = distance

idx = i

point\_lists[idx].append(p)

return point\_lists

def fit(self, points):

clusters = [Cluster(center=p, points=[p])

for p in random.sample(points, self.n\_clusters)]

while True:

point\_lists = self.assign\_points(clusters, points)

diff = 0

for i in range(self.n\_clusters):

if not point\_lists[i]:

continue

old = clusters[i]

center = self.calculate\_center(point\_lists[i])

new = Cluster(center, point\_lists[i])

clusters[i] = new

diff = max(diff, euclidean(old.center, new.center))

if diff < self.min\_diff:

break

return clusters

def rgb\_to\_hex(rgb):

return '%s' % ''.join(('%02x' % p for p in rgb))

def get\_colors(filename, n\_colors=8):

jpg = Image.open(filename)

points = get\_points(filename)

clusters = KMeans(n\_clusters=n\_colors).fit(points)

clusters.sort(key=lambda c: len(c.points), reverse=True)

rgbs = [map(int, c.center.coordinates) for c in clusters]

color\_list = list(map(rgb\_to\_hex, rgbs))

return color\_list

# 4. Conclusion

The "ShadeSync" project stands as a testament to the harmonious convergence of innovative technology and creative design, aimed at simplifying the intricate craft of color palette creation. This project offers a transformative solution to streamline and elevate the process of generating visually appealing color schemes based on uploaded images.

In the course of its development, "ShadeSync" has harnessed a robust technology stack, including Python, Flask, Pillow (PIL), NumPy, and the K-Means clustering algorithm. The utilization of these technologies empowers the project to analyze images, extract color data, and seamlessly generate cohesive color palettes.

The implementation strategy meticulously guides users through a two-page web interface. The first page allows for image upload and the selection of the desired number of colors for the palette. Behind the scenes, Flask and Pillow efficiently process the image, while the K-Means algorithm groups similar colors to create a harmonious palette. The second page reveals the uploaded image alongside the generated color palette, allowing users to interact with the visuals and download the palette in various formats.

"ShadeSync" is not merely a technical marvel but a user-centric tool designed to enhance the efficiency and creativity of designers and creators across various fields. It serves as a bridge between inspiration and realization, enabling users to effortlessly distill the essence of an image into a harmonious color palette.

In conclusion, "ShadeSync" represents an innovative fusion of technology and artistry, empowering designers and creators to bring their visions to life with seamless, harmonized color palettes. As technology continues to evolve, "ShadeSync" stands as a beacon, illustrating the transformative potential of digital solutions in the realm of design and aesthetics.

# References

**Technology References:**

1. Python Official Website. (https://www.python.org/)
2. Flask Documentation. (https://flask.palletsprojects.com/)
3. Pillow (PIL) Documentation. (https://pillow.readthedocs.io/en/stable/)
4. NumPy Documentation. (https://numpy.org/doc/)
5. W3Schools. (https://www.w3schools.com/)

**Algorithm Reference:**

1. Anil K. Jain, Data Clustering: 50 Years Beyond K-Means. (https://cseweb.ucsd.edu/~aloise/dm03-jain.pdf)
2. SciKit-Learn Documentation - Clustering. (https://scikitlearn.org/stable/modules/clustering.html)