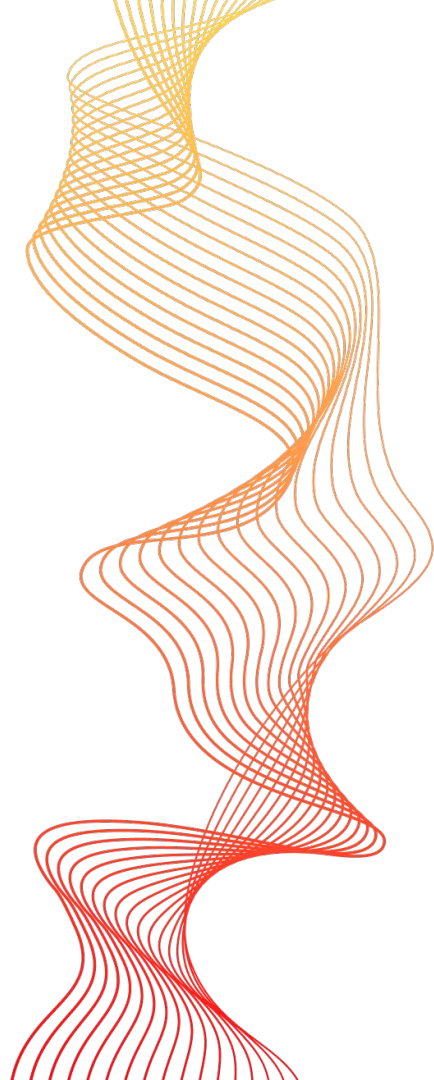




# Guide to Using RESTful APIs for Color Management

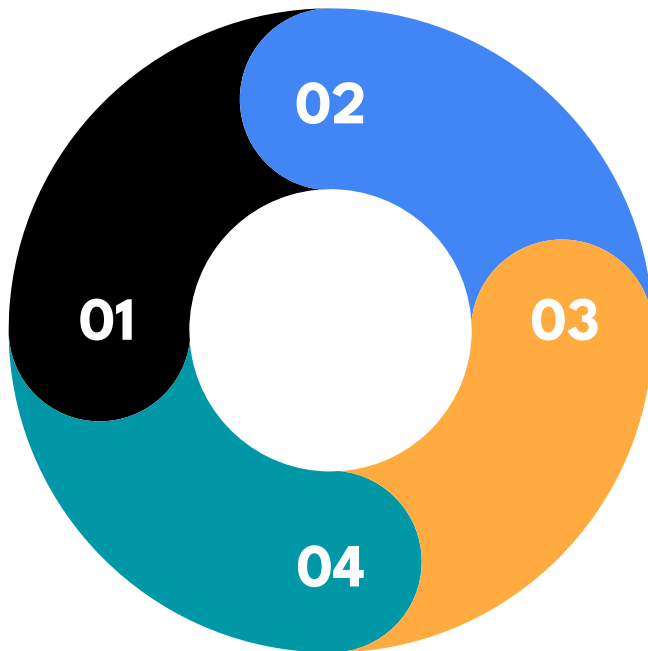
Welcome to the guide on using RESTful APIs for color management. In this guide, we will learn how to interact with APIs to perform CRUD operations (Create, Read, Update, Delete) on colors using JSON format. We will use the OpenAPI Specification to define our APIs and Flask to create the web application that will host the APIs. The guide also covers deployment on the Google Cloud Platform (GCP) using Google Cloud Firestore as the database.



# What is a RESTful API?

A RESTful API is a set of endpoints that allows us to interact with our system through HTTP requests.

Requests are represented by URLs, and the type of access to the resource depends on the HTTP method used.



For example, we use the GET method to retrieve a resource.

To make communication effective, we use the JSON format for data representation, as it is one of the most common and easily manageable formats.

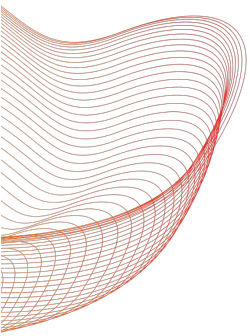


# OpenAPI Specification (Swagger)

The OpenAPI Specification is a standard that allows us to describe our APIs in a standardized way using a YAML or JSON file.

In our case, we will use YAML.

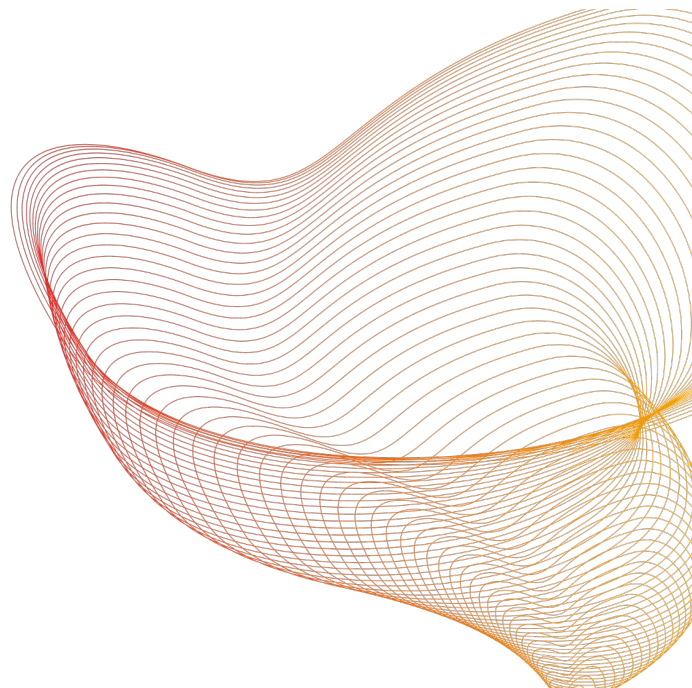
This allows us to document the APIs, define endpoints, HTTP methods, parameters, responses, and data types.





# Creating the Base Application

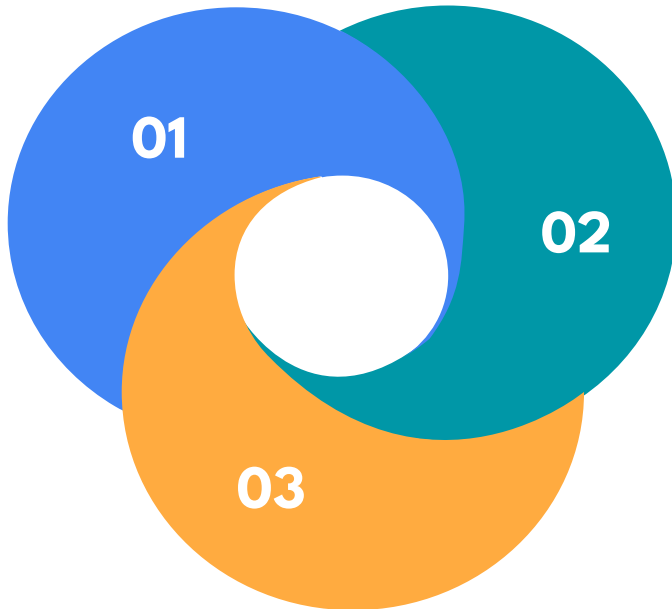
- We will start by creating a base application using Flask.
- To do this, we will create a virtual environment, install Flask, and create the main application file.



# Creating the DAO Object for Color Management

Next, we will create a Data Access Object (DAO) to manage colors and their persistence.


Later on, we will switch to using Google Cloud Firestore as the database.



We will start with a local version using a JSON database.

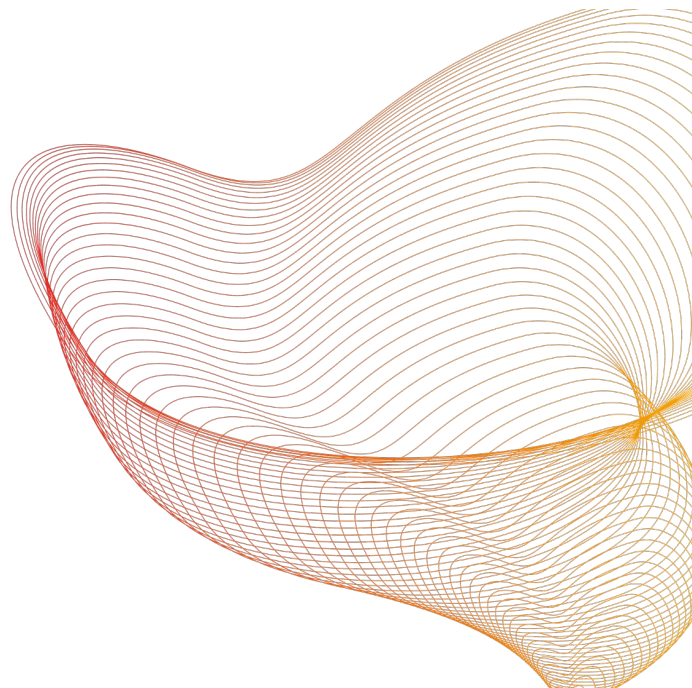



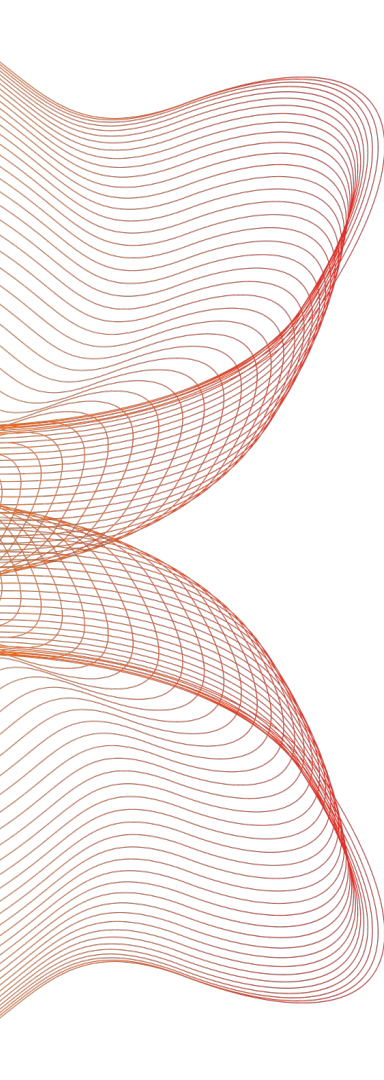
# Using WTForms for Data Validation

- 01** In this case, we will create a form for adding colors.
  - 02** We will use WTForms, a Python module that helps us handle forms and data validation similar to Django Forms.
- 

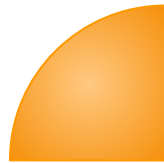
# Using Flask-RESTful for API Development

- 01** We will define endpoints for CRUD operations on colors.
- 02** Next, we will use Flask-RESTful to develop the APIs.
- 03** Flask-RESTful is an extension for Flask that simplifies the development of RESTful APIs.





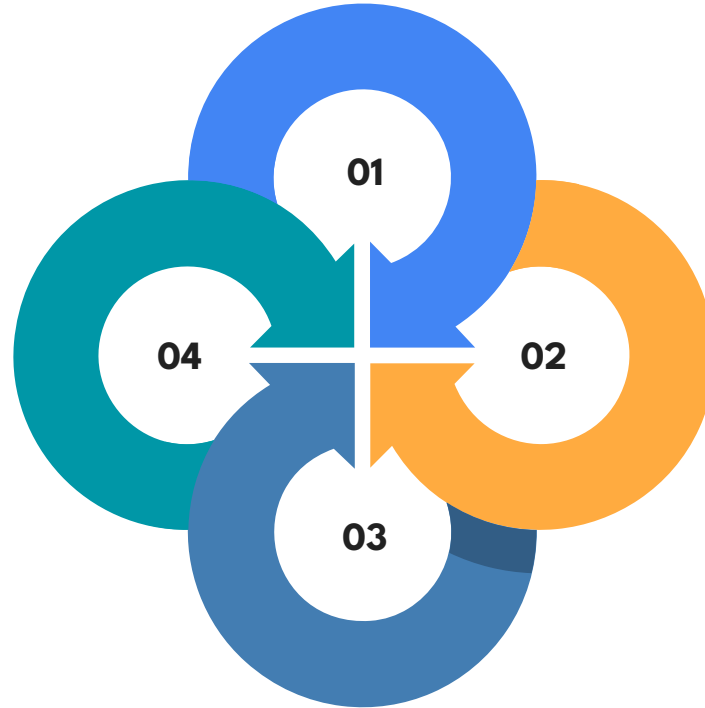
## Deployment on Google Cloud Platform (GCP)

- Next, let's deploy our application on GCP.
  - We will use Google Cloud Firestore as the database and Gunicorn as the server to handle incoming requests.
- 



# Conclusion

Thank you for reading!



Guide created by Vincenzo  
Pio Cassino.

Congratulations! You have completed the guide on using RESTful APIs for color management.

Now you have the knowledge to interact with APIs, perform CRUD operations on colors using JSON format, and deploy your application on the Google Cloud Platform.



Thank you for your time 😊