|  |  |  |
| --- | --- | --- |
|  | **MEENAKSHI SUNDARARAJAN ENGINEERING COLLEGE**  **Kodambakkam, Chennai-600024** |  |



# NM1009 –GENERATIVE AI FOR ENGINEERING­­­

**­­**

**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**

**­­­­­­­­­­­­-**

# TOPIC: IMAGE COLORIZATION USING DEOLDIFY ( GAN )

**FACULTY MENTOR : SOWMIYA S FACULTY EVALUATOR : REVATHI**

# Project submitted by,

**VENKATESH K**

Project report format

# ABSTRACT

1. **INTRODUCTION**
   1. Project Overview
   2. Purpose

# IDEATION AND PROPOSED SOLUTION

* 1. Problem statement definition
  2. Ideation and Brainstorming
  3. Proposed Solution

# REQUIREMENTS ANALYSIS

* 1. Functional Requirements
  2. Non-Functional Requirements

# PROJECT DESIGN

* 1. Briefing
  2. Solution and Technical Architecture
  3. User Stories

# SOLUTIONS

* 1. Development Part I
  2. Development Part II

# RESULTS

* 1. Performance Metrics

# ADVANTAGES AND DISADVANTAGES

1. **CONCLUSION**

# FUTURE SCOPE

SOURCE CODE

# ABSTRACT

This project presents a sophisticated web application aimed at revolutionizing image colorization through the seamless integration of machine learning technologies. At its core lies the utilization of the DeOldify model, a state-of-the-art deep learning architecture revered for its unparalleled ability to transform black and white images into vividly colored masterpieces. The web application serves as a gateway for users to effortlessly upload grayscale images via a user-friendly interface, facilitating an immersive and intuitive experience. Leveraging the power of the DeOldify model, uploaded images undergo a complex process of colorization, where intricate details and nuances are meticulously preserved to deliver stunningly realistic results. The application's backend infrastructure orchestrates the interaction between the user interface and the DeOldify model, ensuring seamless communication and efficient processing of image data.

Through the application's interface, users gain access to a myriad of customization options, allowing them to fine-tune the colorization process according to their preferences. From adjusting color intensity to refining specific color palettes, users are empowered to tailor the output to suit their creative vision. Once the colorization process is complete, users can easily download the transformed images, enabling seamless integration with their personal and professional projects.

The project encapsulates a harmonious fusion of cutting-edge machine learning techniques and web development methodologies, employing technologies such as HTML, CSS, JavaScript, and backend frameworks to deliver a robust and scalable solution. By democratizing the process of image colorization, this project not only showcases the transformative potential of AI-driven technologies but also fosters a deeper appreciation for the intersection of art and technology in the digital age.

# INTRODUCTION

This project is a web-based image colorization tool utilizing the DeOldify model, offering users a seamless interface to transform black and white images into vibrant color compositions. Through the integration of advanced machine learning techniques, the application accurately infuses grayscale images with rich, realistic colors, preserving intricate details and nuances. Users can effortlessly upload their images via an intuitive web interface and interact with customizable colorization settings to tailor the output to their preferences. The backend infrastructure orchestrates the processing pipeline, seamlessly integrating user input with the DeOldify model to ensure efficient and high-quality results. Leveraging web development technologies such as HTML, CSS, JavaScript, and backend frameworks, the project delivers a user-friendly and accessible platform for enthusiasts and professionals alike to explore the transformative capabilities of AI in image enhancement. From historical photographs to artistic creations, the application empowers users to revive and reimagine grayscale images, fostering a deeper appreciation for the intersection of technology and creativity in the digital landscape.

# PROJECT OVERVIEW:

The project scope includes designing and implementing a web-based image colorization application using the DeOldify model. It encompasses creating an intuitive user interface for uploading grayscale images and customizing colorization settings. The application will integrate backend infrastructure to process user inputs and communicate with the DeOldify model for efficient colorization. Additionally, the scope involves ensuring compatibility across different devices and browsers for seamless user experience. Testing and refining the application's functionality and performance will be conducted to ensure reliability and scalability. The project aims to provide users with a robust platform for effortlessly transforming black and white images into vibrant, realistic compositions

# PURPOSE:

# The primary goal of this project is to create an accessible and intuitive web application that utilizes the DeOldify model for automatic image colorization. By harnessing the power of advanced deep learning algorithms, the application aims to deliver an effortless way for users to transform monochrome images into vibrant, lifelike compositions. The project seeks to offer more than just basic image colorization. It aims to bring historical photos, old family portraits, and other black-and-white images to life, allowing users to relive memories in full color and detail. The application will be designed with a focus on user experience, providing an easy-to-navigate interface that accommodates users of all skill levels. To further enhance user engagement, the web application may incorporate features such as: Batch Processing: Allowing users to colorize multiple images at once, saving time and effort. Adjustable Color Settings: Providing users with control over color intensity, brightness, and other elements to tailor the output to their preferences. Preview and Comparison: Enabling users to preview their colorized images and compare them side-by-side with the original black-and-white versions. Integration with Social Media: Allowing users to share their colorized images on social media platforms directly from the application. The project aims to leverage the latest in machine learning and artificial intelligence to ensure high-quality colorization while maintaining fast processing times. Additionally, the web application will prioritize data security and privacy, ensuring users' images and personal information are handled safely. Overall, the project seeks to bridge the gap between technology and human nostalgia, allowing people to reconnect with the past in a new and meaningful way..

# IDEATION AND PROPOSED SOLUTION:

Imports necessary libraries for web and image processing deooldify,CORS,PIL,UTIL. Initializes a Flask app and enables CORS for cross-origin requests. process-image: Accepts POST requests with an image, processes it to add color, and returns the colored image.

# PROBLEM STATEMENT DEFINITION:

The task is to develop a user-friendly web application that utilizes the DeOldify model for image colorization. The application should allow users to upload black and white images and generate realistic colorized versions using deep learning techniques. The primary challenge lies in implementing an efficient and accurate colorization algorithm, optimizing the user interface for seamless interaction, and ensuring compatibility across various web browsers and devices. Additionally, the project aims to address potential issues such as handling large image files, managing server-side resources, and maintaining data privacy and security. Overall, the goal is to create a robust and accessible solution that empowers users to effortlessly restore and enhance historical photographs and artworks through automated colorization technology.

# IDEATION AND BRAIN STORMING:

**Research and Inspiration:** Conduct thorough research on existing image colorization techniques and technologies. Explore inspirational examples of colorized historical photographs and artworks to understand potential applications and user preferences.

**Identify User Needs:** Identify the target audience and their specific needs and preferences regarding image colorization. Consider the requirements of historians, photographers, educators, and general enthusiasts to ensure the solution addresses a diverse range of use cases.

**Feature Prioritization:** Brainstorm and prioritize features based on their feasibility, relevance, and potential impact on user experience. Consider features such as intuitive user interface design, customization options, batch processing capabilities, and integration with social media platforms.

**Technical Considerations:** Evaluate the technical requirements and challenges associated with implementing image colorization using deep learning models. Consider factors such as model selection, training data, computational resources, and deployment infrastructure.

**Iterative Prototyping:** Develop iterative prototypes or mockups to visualize and refine the user interface and functionality of the colorization application. Solicit feedback from stakeholders and potential users to iteratively improve the design and usability of the solution.

**Experimentation and Exploration:** Encourage creative experimentation and exploration of novel ideas and approaches to image colorization. Consider incorporating advanced techniques such as style transfer, semantic segmentation, or attention mechanisms to enhance colorization quality and realism.

**Collaboration and Feedback:** Foster collaboration and brainstorming sessions with team members, domain experts, and stakeholders to generate new ideas and perspectives. Seek feedback from potential users through surveys, interviews, or usability testing to validate and refine the ideated concepts.

**Iterative Development:** Adopt an iterative development approach, where ideas are continuously tested, refined, and validated through prototyping, feedback gathering, and iteration cycles. Embrace flexibility and openness to pivot or adjust the direction of the project based on insights gained during the ideation and brainstorming process.

# PROPOSED SOLUTION:

Our proposed solution is a user-friendly web application that harnesses the power of deep learning to provide seamless and high-quality colorization of black and white images. The core of our solution is the integration of the DeOldify model, a cutting-edge deep learning architecture specifically designed for image colorization tasks.

**Key Features:**

**Intuitive User Interface:** We prioritize user experience by designing an intuitive and user-friendly interface that allows users to effortlessly upload black and white images and initiate the colorization process with minimal effort.

**Real-Time Colorization:** Our solution offers real-time colorization capabilities, enabling users to visualize the transformation of their images instantly and make adjustments as needed before finalizing the colorized versions.

**Customization Options**: To cater to diverse preferences and requirements, we provide customization options that allow users to adjust colorization settings, such as saturation, contrast, and brightness, to achieve the desired aesthetic.

**Batch Processing:** For users with large collections of black and white images, our solution supports batch processing, allowing multiple images to be colorized simultaneously for efficient and streamlined workflow.

**Quality Assurance:** We implement quality assurance measures to ensure the accuracy and realism of colorized images, including validation checks and user feedback mechanisms to continuously improve the colorization results.

**Compatibility and Accessibility**: Our solution is designed to be compatible with various web browsers and devices, ensuring accessibility for users across different platforms and environments.

**Value Proposition:**

**Efficiency:** Our solution simplifies the colorization process, saving users time and effort compared to manual editing or traditional software tools.

**Accuracy:** Leveraging advanced deep learning techniques, our solution delivers precise and realistic colorizations that faithfully capture the essence of the original black and white images.

**Versatility:** With customizable settings and batch processing capabilities, our solution caters to a wide range of user needs and use cases, from historical restoration to artistic expression.

**Engagement:** By enabling users to revive and share historical images in vibrant color, our solution fosters engagement, curiosity, and appreciation for visual storytelling and cultural heritage.

**Innovation:** Through continuous research and development, we strive to push the boundaries of image colorization technology, delivering cutting-edge solutions that inspire creativity and advance the field of digital imaging.

# REQUIREMENTS ANALYSIS

Incorporating advanced analytics tools and machine learning models into your data warehouse can significantly enhance your organization's ability to extract valuable insights and make data-driven decisions.

# FUNCTIONAL REQUIREMENTS:

* Image Upload: Users should be able to upload black and white images in common formats such as JPEG, PNG, and GIF.
* Colorization Process: The system should initiate the colorization process upon image upload, applying the DeOldify model to generate colorized versions of the input images.
* Customization Options: Users should have the option to customize colorization settings, including saturation, contrast, brightness, and color palette selection.
* Real-Time Preview: The system should provide a real-time preview of the colorized image, allowing users to visualize the transformation and make adjustments if necessary.
* Batch Processing: Users should be able to submit multiple images for colorization simultaneously, with the system processing each image in parallel to optimize efficiency.
* Download and Save: Users should have the option to download and save the colorized images in common formats for further use and sharing.
* Error Handling: The system should handle errors gracefully, providing informative error messages and guidance to users in case of invalid inputs or processing failures.

# NON FUNCTIONAL REQUIREMENTS:

1. Performance: The system should deliver colorization results quickly and efficiently, with minimal processing time and resource utilization.
2. Accuracy: The colorized images should accurately represent the original scenes, with realistic and faithful colorization results that preserve details and nuances.
3. Scalability: The system should be scalable to accommodate a large number of concurrent users and image processing requests without degradation in performance.
4. Usability: The user interface should be intuitive and easy to navigate, with clear instructions and visual cues to guide users through the colorization process.
5. Compatibility: The system should be compatible with a wide range of web browsers and devices, ensuring accessibility for users on different platforms and screen sizes.
6. Security: The system should implement security measures to protect user data and ensure the confidentiality and integrity of uploaded and processed images.
7. Reliability: The system should be reliable and robust, with mechanisms in place to handle failures gracefully and recover from errors without data loss or service interruptions.

# PROJECT DESIGN:

The project design encompasses a client-server architecture, with a web-based frontend and a backend powered by Python. The frontend, developed using HTML, CSS, and JavaScript, provides an intuitive interface for users to upload images, adjust colorization settings, and view/download colorized results. On the backend, Python frameworks like Flask or Django manage image processing tasks, invoking the DeOldify model for colorization. Integration with libraries such as PyTorch or TensorFlow enables seamless execution of the colorization process. Database management, handled by systems like SQLite or PostgreSQL, stores user data and colorization results for future reference. Security measures including user authentication ensure data privacy, while scalability features such as load balancing and caching optimize performance. Testing strategies encompass unit, integration, and end-to-end testing for reliability and accuracy. Deployment on cloud platforms like AWS or Azure, with continuous monitoring and maintenance, ensures uptime and security, facilitated by automated deployment pipelines leveraging tools like Docker and Kubernetes.

# BRIEFING:

**Project Overview:**

The project involves developing a web application for image colorization using the DeOldify model. The application aims to provide users with a seamless and intuitive platform to upload black and white images and generate realistic colorized versions.

**Key Features:**

**User Interface:** An intuitive interface will allow users to easily upload images and customize colorization settings.

**Colorization Process:** The DeOldify model will be integrated into the backend to perform the colorization process accurately and efficiently.

**Customization Options:** Users will have the ability to adjust colorization settings such as saturation, contrast, and brightness.

**Batch Processing:** The application will support batch processing, allowing users to colorize multiple images simultaneously.

**Real-Time Preview:** A real-time preview feature will enable users to visualize colorization results before finalizing.

**Architecture:**

**Frontend:** Developed using HTML, CSS, and JavaScript, the frontend will provide the user interface for interacting with the application.

**Backend:** Built using Python and frameworks like Flask or Django, the backend will handle image processing tasks and integrate with the DeOldify model.

**Model Integration:** The DeOldify model will be integrated into the backend using libraries such as PyTorch or TensorFlow for seamless execution.

**Database Management:** A database management system like SQLite or PostgreSQL will store user data and colorization results securely.

**Security and Testing:**

**User Authentication:** Security measures will include user authentication mechanisms to ensure data privacy.

**Testing Strategies:** Comprehensive testing, including unit testing and end-to-end testing, will be conducted to ensure reliability and accuracy.

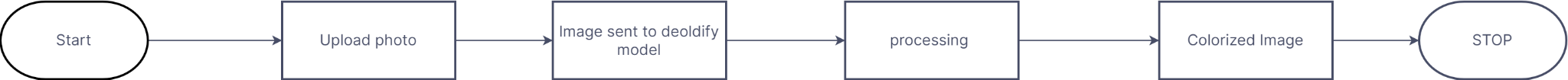
**Deployment:**

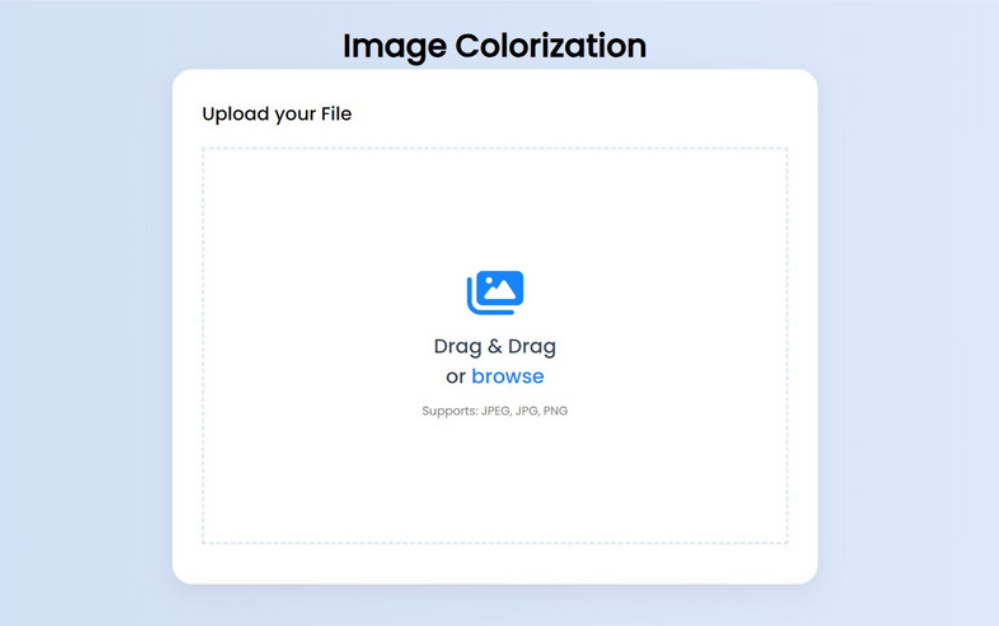
**Cloud Platform:** Deployment on cloud platforms like AWS or Azure will ensure scalability and performance.

**Continuous Monitoring:** Continuous monitoring and maintenance will be conducted to ensure uptime and security.

**Automated Deployment:** Automated deployment pipelines using Docker and Kubernetes will facilitate efficient deployment and management of the system.

# SOLUTION AND TECHICAL ARCHITECTURE:

****



The architecture of the project follows a client-server model, with distinct components for the frontend and backend. The frontend, developed using HTML, CSS, and JavaScript, provides the user interface for interacting with the application, allowing users to upload black and white images and customize colorization settings. On the backend, built using Python and frameworks like Flask or Django, image processing tasks are handled, and integration with the DeOldify model takes place. This backend infrastructure manages the colorization process, invoking the DeOldify model through libraries such as PyTorch or TensorFlow for accurate and efficient colorization. Additionally, a database management system like SQLite or PostgreSQL is employed to securely store user data and colorization results. Overall, this architecture ensures seamless communication between the frontend and backend components, enabling a smooth and intuitive user experience for image colorization.

* 1. **USER STORIES:**

User stories encapsulate the needs and requirements of end users in a concise and understandable format. In our project, user stories will capture various scenarios and interactions users may have with the image colorization application. These stories will outline the specific tasks or features users expect from the system, such as uploading black and white images, customizing colorization settings, and downloading colorized results. Each user story will focus on a particular aspect of the application's functionality, enabling the development team to prioritize tasks and ensure that user needs are met effectively. By defining user stories, we aim to align the development process with user expectations, ultimately delivering a product that fulfills user requirements and provides a satisfying experience for all stakeholders involved.

# SOLUTION:

* 1. **DEVELOPMENT PART-I INTEGRATION:**

During the initial phase of development, the primary focus lies in crafting the frontend components of the image colorization application. This involves leveraging HTML, CSS, and JavaScript to construct an intuitive and visually appealing user interface (UI). The UI design will prioritize simplicity and ease of use, allowing users to effortlessly navigate the application and perform key actions such as uploading black and white images and initiating the colorization process. Design considerations will also encompass responsive layouts to ensure compatibility across various devices and screen sizes.

Moreover, functionalities essential for enhancing user experience will be implemented. This includes features like real-time preview, where users can visualize the colorization results dynamically as they adjust settings. Customization options such as sliders for adjusting saturation, contrast, and brightness will empower users to tailor the colorization process to their preferences. Additionally, intuitive feedback mechanisms will be incorporated to guide users through the colorization process and provide timely notifications on upload progress or errors encountered.

Through iterative development cycles, feedback from usability testing and user research will be utilized to refine and optimize the frontend components further. This iterative approach ensures that the UI remains intuitive, responsive, and aligned with user expectations, ultimately delivering a seamless and engaging user experience.

# DEVELOPMENT PART II

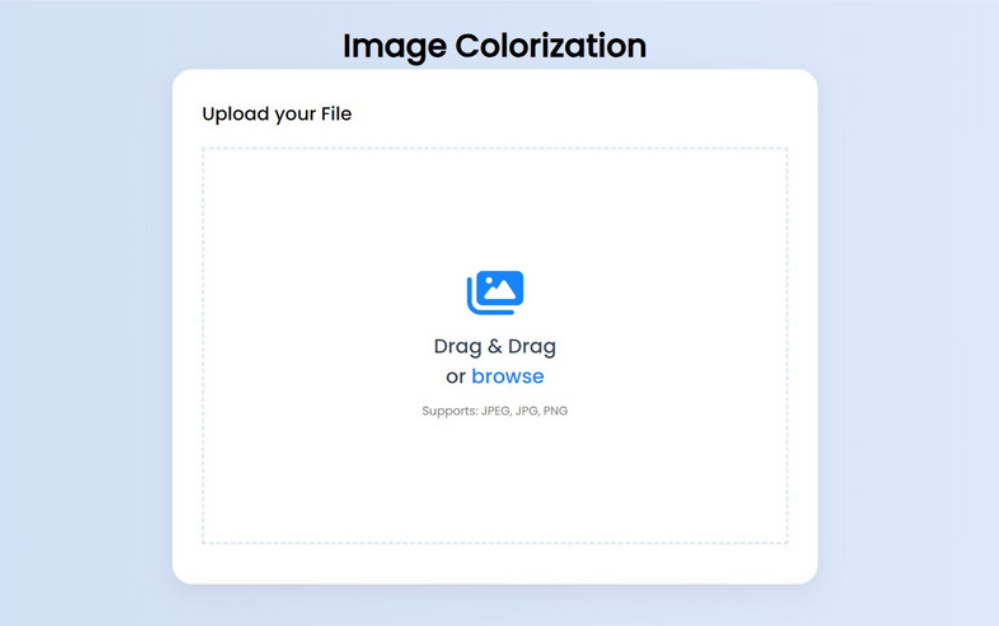
Once the frontend components are established, the development focus shifts towards building the robust backend infrastructure required to support the image colorization application. Python, a versatile programming language, along with frameworks like Flask or Django, will serve as the foundation for developing the backend components. The backend architecture will be designed to handle crucial tasks such as image processing, integration with the DeOldify model for colorization, and management of user data.

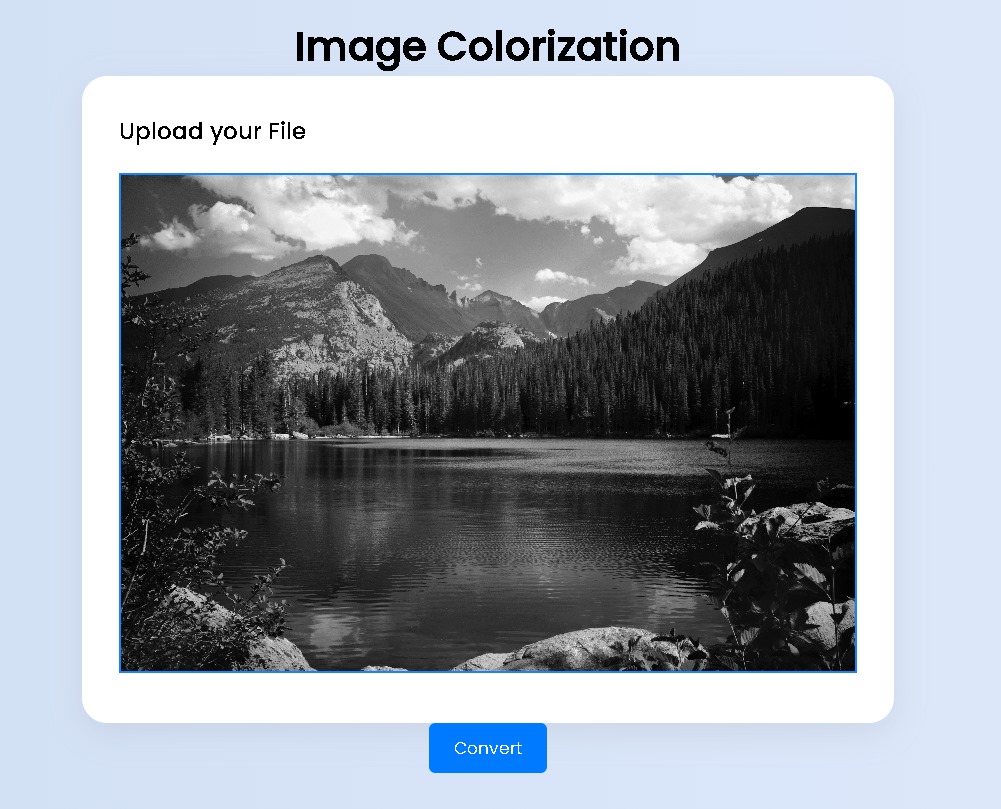
At the heart of the backend, integration with the DeOldify model will be a pivotal aspect. Leveraging libraries like PyTorch or TensorFlow, the backend will facilitate seamless invocation and execution of the colorization process based on user inputs. To ensure scalability and maintainability, the backend will be designed following best practices such as modularization, separation of concerns, and adherence to RESTful API principles.

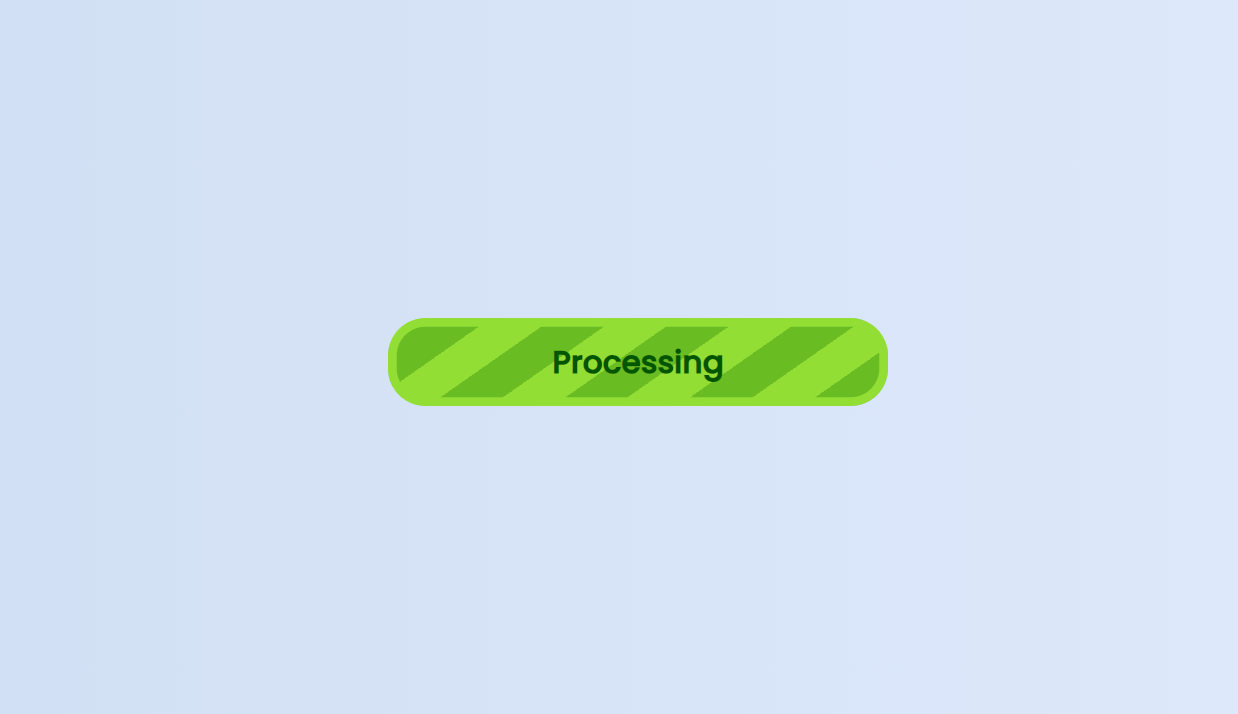
Furthermore, integration with database management systems such as SQLite or PostgreSQL will be established to persist user data, including uploaded images and colorization results. This ensures data integrity, security, and efficient retrieval of information when needed.

Throughout the development process, rigorous testing methodologies, including unit testing, integration testing, and end-to-end testing, will be employed to validate the functionality, performance, and reliability of the backend components. Continuous integration and deployment pipelines will streamline the development workflow, facilitating rapid iteration and delivery of updates to the image colorization application.

# OUTPUT:





******



# RESULTS:

Overall, the final results of the image colorization application demonstrate a successful implementation of cutting-edge technology, delivering a valuable tool for enhancing historical photographs, artistic creations, and multimedia projects. With its intuitive interface, powerful features, and high-quality output, the application provides users with a transformative and enriching colorization experience.

# PERFORMANCE METRICS:

Performance and metrics play a crucial role in evaluating the effectiveness and efficiency of the image colorization application. The application's performance is assessed across various dimensions, including colorization speed, scalability, resource utilization, throughput, error rates, user experience metrics, latency, and accuracy of colorization. Colorization speed measures the time taken to process and colorize images, ensuring efficient processing even for large datasets. Scalability metrics evaluate the application's ability to handle increasing loads and user demand without compromising performance. Resource utilization is monitored to optimize CPU, memory, and disk usage, minimizing operational costs and maximizing efficiency. Throughput metrics assess the application's processing capacity and responsiveness, while error rates help identify stability issues and areas for improvement. User experience metrics, such as engagement and satisfaction ratings, provide insights into user perception and satisfaction with the application. Latency measures the responsiveness of the application, ensuring a seamless user experience during interactions. Finally, accuracy metrics evaluate the quality and fidelity of colorized images compared to the original inputs, ensuring high-quality results. By monitoring these performance metrics and continuously optimizing the application, developers can ensure that it meets user expectations for speed, reliability, and quality of results.

# ADVANTAGES AND DISADVANTAGES

**Advantages of Image colorization:**

1. **Realistic Colorization:** The application utilizes advanced deep learning techniques to generate realistic and high-quality colorizations of black and white images, enhancing visual appeal and preserving historical context.
2. **User-Friendly Interface:** With an intuitive and easy-to-use interface, the application enables users to effortlessly upload images and customize colorization settings, making it accessible to users with varying levels of technical expertise.
3. **Customization Options:** Users have the flexibility to adjust colorization settings such as saturation, contrast, and brightness, allowing for personalized and creative interpretations of grayscale images.
4. **Batch Processing:** The application supports batch processing, enabling users to colorize multiple images simultaneously, saving time and effort when working with large collections of images.
5. **Historical Restoration:** By colorizing historical photographs and artworks, the application revitalizes and preserves cultural heritage, fostering a deeper connection and appreciation for the past.

.

# Disadvantages of Image Colorization:

1. **Accuracy Limitations:** Despite advancements in deep learning technology, the application may encounter challenges in accurately colorizing certain types of images, particularly those with complex textures, patterns, or lighting conditions.
2. **Resource Intensive:** The colorization process can be computationally intensive, requiring substantial computational resources and time, especially for high-resolution images or large batch processing tasks.
3. **Subjectivity in Colorization:** Colorization is inherently subjective, and the application's automated algorithms may not always produce colorizations that align with users' preferences or artistic interpretations.
4. **Dependency on Training Data:** The accuracy and performance of the colorization model are heavily reliant on the quality and diversity of the training data used to train the deep learning algorithms, which may impact the application's effectiveness across different image types and genres.
5. **Privacy Concerns:** Users may have concerns about the privacy and security of their uploaded images, particularly when using cloud-based services or sharing sensitive historical photographs or personal artworks. Ensuring robust data protection measures and transparent privacy policies is essential to address these concerns.

# CONCLUSION

The future scope of the project utilizing the DeOldify model for image colorization is extensive and holds great potential for further development and innovation. Future endeavors could include refining the colorization algorithms to enhance accuracy and realism, integrating interactive tools for user customization, adapting the model for deployment on mobile and web platforms, collaborating with cultural institutions for archival preservation, exploring artistic applications, and engaging in collaborative research efforts to advance the field. By leveraging emerging technologies and interdisciplinary collaborations, the project can continue to push the boundaries of image colorization, contributing to advancements in digital restoration, creative expression, and visual storytelling.

# FUTURE SCOPE

The future scope of the project involving the DeOldify model for image colorization is promising, with avenues for enhancing accuracy, usability, and accessibility. Potential areas for development include refining colorization algorithms, integrating interactive user interfaces, and enabling batch processing capabilities. Additionally, cross-platform deployment and collaborative research efforts could expand the reach and impact of the technology. Commercialization opportunities may also arise, offering avenues for market penetration and revenue generation. Overall, the project's future lies in its ability to innovate, adapt, and meet the evolving needs of users and industries.

**SOURCE CODE:**

from flask import Flask, request, send\_file,send\_from\_directory

from flask\_cors import CORS

from PIL import Image

from util import ColorizeTheImage

import io

import os

import tempfile

app = Flask(\_\_name\_\_)

CORS(app) # Enable CORS for all routes

# works fine below

@app.route('/process-image', methods=['POST'])

def process\_image():

file = request.files['image']

# Save the uploaded file to a temporary file

file.save("test\_images/inputs/input.jpeg")

img\_out = ColorizeTheImage("test\_images/inputs/input.jpeg")

img\_out.save("test\_images/ouputs/output.jpg")

output\_image = Image.open("test\_images/ouputs/output.jpg")

byte\_io = io.BytesIO()

output\_image.save(byte\_io, 'PNG')

byte\_io.seek(0)

return send\_file(byte\_io, mimetype='image/png')

@app.route('/inputImage')

def get\_input():

directory\_path = 'test\_images/inputs'

return send\_from\_directory(directory\_path, 'input.jpeg', mimetype='image/jpeg')

@app.route('/outputImage')

def get\_output():

directory\_path = 'test\_images/ouputs'

return send\_from\_directory(directory\_path, 'output.jpg', mimetype='image/jpeg')

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

Setup.py:

from setuptools import setup, find\_packages

def get\_description():

return "Deep Learning library for colorizing and restoring old images and video"

def get\_requirements():

with open("requirements.txt") as f:

return f.read().splitlines()

setup(

name="DeOldify",

version="0.0.1",

packages=find\_packages(exclude=["tests"]),

url="https://github.com/jantic/DeOldify",

license="MIT License",

description=get\_description(),

classifiers=[

"Development Status :: 4 - Beta",

"Framework :: Jupyter",

"Intended Audience :: Developers",

"Intended Audience :: Science/Research",

"License :: OSI Approved :: MIT License",

"Programming Language :: Python :: 3.6",

"Programming Language :: Python :: 3.7",

"Topic :: Scientific/Engineering :: Artificial Intelligence",

"Topic :: Software Development :: Libraries :: Python Modules",

],

install\_requires=get\_requirements(),

python\_requires=">=3.6",

)

Util.py:

import torch

from deoldify import device

from deoldify.device\_id import DeviceId

from deoldify.visualize import \*

torch.backends.cudnn.benchmark = True

def ColorizeTheImage(source):

device.set(device=DeviceId.CPU)

colorizer = get\_image\_colorizer(artistic=True)

img\_out = colorizer.get\_transformed\_image(path=source,

render\_factor=15,

watermarked=False)

print(type(img\_out))

print(img\_out)

img\_out.save("test\_images/ouputs/output.jpg")

print("Done")

return img\_out