A PROJECT REPORT ON

AI IMAGE ENHANCER

SUBMITTED TO MANGALORE UNIVERSITY IN PARTIAL FULFILMENT

OF THE REQUIREMENT FOR THE AWARD OF

BACHELOR OF COMPUTER APPLICATION DEGREE.

****

**SUBMITTED BY**

PRANAM HEBBAR MV (U05FM21S0051)

UNDER THE GUIDANCE OF

**Mrs. ARPITHA K C**

Department of Computer Applications

Field Marshal K. M. Cariappa College

[A Constituent College of Mangalore University]

Madikeri – 57l20l, Kodagu District, karnataka

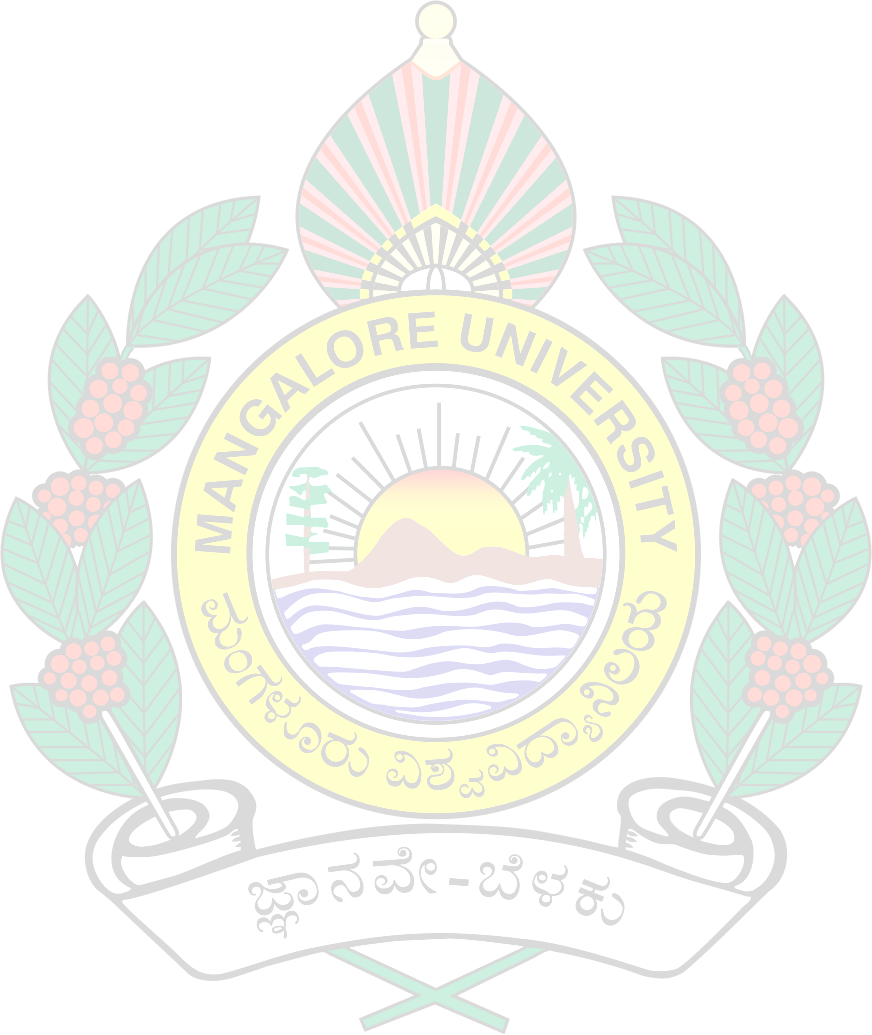
*(Re-Accredicted with ‘B++’ Grade by NAAC)*

2023-2024

**MANGALORE UNIVERSITY**

**FIELD MARSHAL K. M. CARIAPPA COLLEGE**

Madikeri – 57l20l, Kodagu District, Karnataka

(Re-Accredicted with ‘B++’ Grade by NAAC)

**CERTIFICATE**

This is to certify that the project work entitled **“**AI IMAGE ENHANCER**”** submitted in partial fulfilment of requirement for the award of Bachelor of Computer Application Degree of Mangalore University is a result of the Bonafede work carried out by

**PRANAM HEBBAR MV(U05FM21S0051)**

during the academic year 2023-24**.**

**Guide:**

**Ravishankar M N**

Assistant Professor andHead of the

Department of Computer Science,

**Mrs. ARPITHA K C**

**Lecturer**

Department of Computer Science,

**Principal**

**Field Marshal K M Cariappa College**

***----For Examination purpose----***

This Project Report has been evaluated during the Mangalore University UG Practical Examination held on

**Examiners**

1.

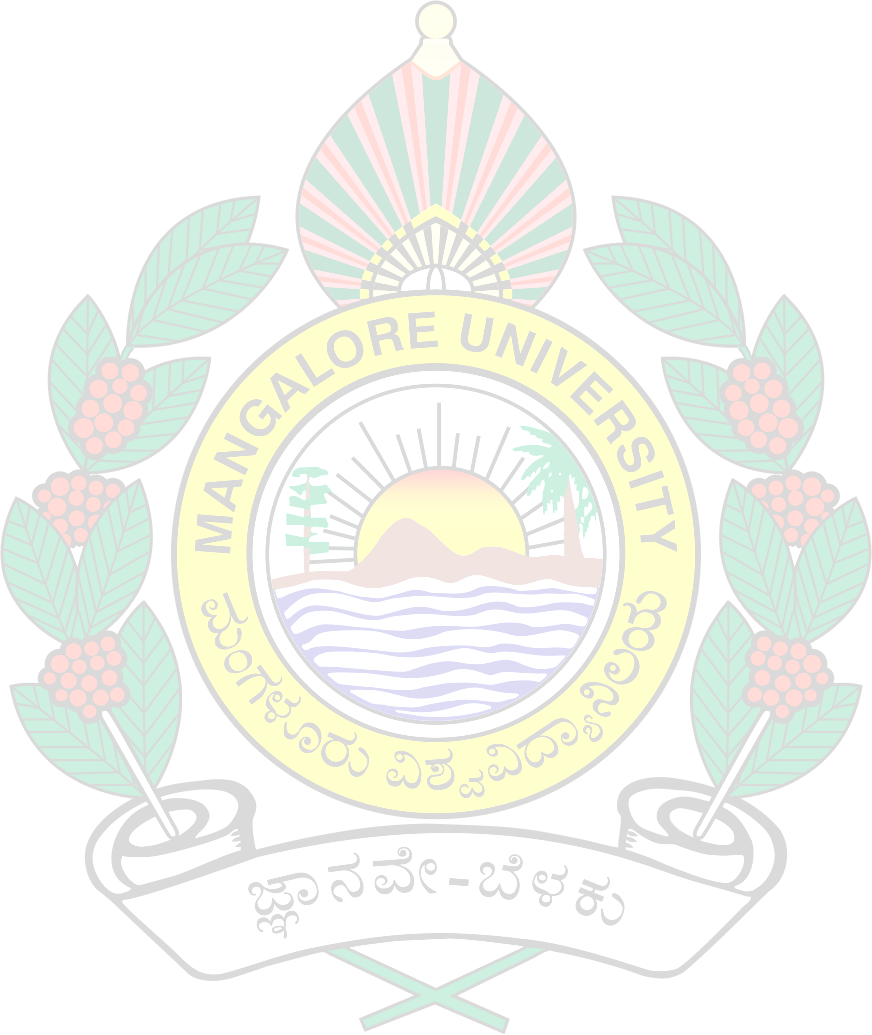
2.

**ACKNOWLEDGEMENT**

We would like to express our gratitude to the following individuals and

organizations for their valuable contributions and support throughout the development of

this project:

We extend our heartfelt thanks to Maj. Prof. Raghav B, the Principal of FMKMC college

for granting permission to carry out our project.

We would like to thank Field Marshall K.M Cariappa college for providing us with the

necessary resources and facilities to carry out this project successfully. Their support has

been instrumental in our endeavours.

I am extremely thankful to Mr.Ravishankar.M.N , Head of the department of Computer Science

for his words of inspiration that has motivated us during the time of project.

We extend our heartfelt thanks to our project guide Mrs. Arpitha.K.C for her guidance,

expertise, and continuous encouragement throughout the project. Her insights and feedback

have been invaluable in shaping our work.

I extend my heartfelt thanks to all lecturers of Computer Science Department for their help

and suggestions.

We acknowledge any other individuals or groups whose support and assistance may not

have been explicitly mentioned but have nonetheless contributed to the success of this

project.

We sincerely appreciate the collaborative efforts of everyone involved in making this project

possible.

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sl.no.** | **Content** | **Pg.no.** |
| 1. | **INTRODUCTION** | 01-06 |
| 2. | **LITERATURE SURVEY** | 07-11 |
| 3. | **SOFTWARE REQUIREMENT SPECIFICATION** | 12-14 |
| 4. | **SYSTEM DESIGN DESCRIPTION** | 15-19 |
| 5. | **PROGRAM CODE** | 20-25 |
| 6. | **USER INTERFACE** | 26-27 |
| 7. | **TESTING** | 28 |
| 8. | **FUTURE SCOPE AND FURTHER ENHANCEMENT** | 29 |
| 9. | **CONCLUSION** | 30 |
| 10. | **REFERENCES** | 31 |

**AI IMAGE ENHANCER**

**1. INTRODUCTION:**

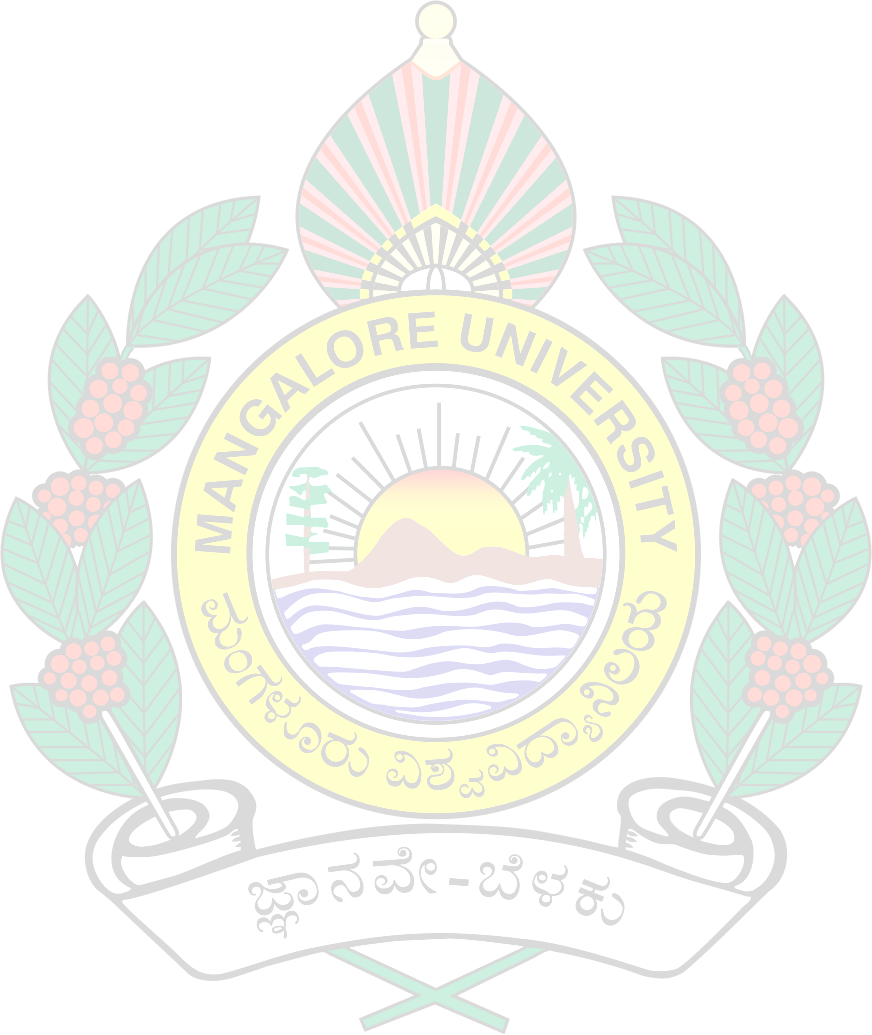
Welcome to **Imagenary**, our cutting-edge AI image enhancer designed to bring your digital images to life! In this documentation, we present an innovative platform that features state-of-the-art image restoration and generative fill capabilities, all powered by the robust Cloudinary API.

Imagenary is a testament to the transformative potential of AI and cloud-based technologies, offering users an intuitive and powerful tool for enhancing and revitalizing their images. Our project seamlessly integrates advanced AI techniques with a user-friendly interface, making high-quality image enhancement accessible to everyone.

Built using JavaScript (React), CSS (Tailwind), HTML5, Imagenary leverages the Cloudinary API to perform complex image transformations with ease and precision. This integration allows us to offer powerful enhancement tools similar to those found in premium platforms, but without the associated costs.

Our commitment to user experience goes beyond providing powerful tools; it encompasses accessibility, responsiveness, and a seamless workflow. Imagenary is designed to cater to a wide range of users, from photographers and digital artists to businesses and hobbyists.

In this documentation, we explore the development journey of Imagenary, from its conceptualization to its implementation. We delve into the technological framework that supports the application, providing insights into our design decisions, implementation strategies, and optimization techniques. Additionally, we outline our vision for future enhancements, aiming for continuous improvement and innovation..

We invite you to explore Imagenary and discover the incredible possibilities it offers for image enhancement. Whether you're looking to restore cherished memories or create stunning visual content, our project is here to inspire and empower your creative endeavors.

**OVERVIEW:**

Creating an AI Image Enhancer project is an exciting venture! Here's a basic overview to get you started:

* Image Restoration: Restores old, faded, or damaged images by enhancing details, correcting imperfections, and revitalizing overall image quality.
* Technology Stack: Built with HTML for structure, Next.js for optimized server-side rendering and static site generation, and Tailwind CSS (shadcn) for streamlined styling.
* Cloudinary API Integration: Utilizes the Cloudinary API to perform complex image transformations efficiently, leveraging cloud-based processing for high-speed results.
* User-Friendly Interface: Designed to be intuitive and easy to use, making advanced image enhancement accessible to users of all skill levels.
* Customization Options: Offers instant previews of changes, allowing users to see the results of enhancements in real-time.
* Cost-Effective Solution: Delivers high-quality image enhancement tools similar to premium platforms but without the associated costs, making it accessible to a wider audience.
* Future Vision: Envisions continuous improvement and innovation, with plans for future enhancements to expand capabilities and further improve user experience.

**AIMS AND OBJECTIVES:**

* Provide Intuitive User Experience : Design a user-friendly interface that allows users to effortlessly upload, enhance, and download their images.
* Implement Advanced Image Restoration: Develop robust algorithms for restoring old, faded, or damaged images, enhancing details and correcting imperfections.
* Integrate Generative Fill: Offer intelligent generative fill capabilities that seamlessly complete missing parts of images for a polished result.
* Ensure Real-Time Processing: Provide instant feedback and real-time previews of enhancements, allowing users to see changes immediately.
* Customize Enhancement Options: Allow users to tailor the enhancement settings to their specific needs, offering flexibility and control over the process.

**ADVANTAGES:**

* User-Friendly Interface: Offers an intuitive and easy-to-navigate interface, making advanced image enhancement accessible to users of all skill levels.
* Cost-Effective: Provides powerful image enhancement tools similar to premium platforms without the associated costs, making it an affordable solution for users.
* High-Quality Enhancements: Delivers impressive image restoration and generative fill capabilities, significantly improving the visual quality of digital images.
* Real-Time Processing: Offers instant previews and real-time feedback, allowing users to see the results of their enhancements immediately.
* Customization Options: Allows users to customize settings and tailor the enhancement process to their specific needs and preferences.

**SYSTEM ANALYSIS:**

Imagenary is an AI-powered image enhancement platform designed to offer advanced features such as image restoration and generative fill. It leverages modern web technologies and cloud-based APIs to deliver efficient and high-quality image processing.

**1. Requirement Gathering:**

- Feature Identification:: Identify and prioritize key features such as image restoration, generative fill, real-time previews, customization options, and user-friendly interface based on user feedback and market demand.

- Technical Requirements: Define technical requirements such as integration with Cloudinary API, backend architecture using FastAPI, frontend development with Next.js and Tailwind CSS, compatibility with modern web browsers, and scalability for handling concurrent users and image processing tasks.

- Security and Privacy Considerations:: Gather requirements for implementing robust security measures, data encryption, secure user authentication, and compliance with data privacy regulations to protect user data and images.

**2. System Architecture:**

- Client Interface:: The client interface, built with HTML, Next.js, and Tailwind CSS, provides users with a responsive and intuitive platform to interact with Imagenary.

- Backend Services:: The backend services,used by cloudinary servers to handle core functionalities such as image processing, user authentication, and data management.

- Cloudinary Integration:: Imagenary integrates with the Cloudinary API for advanced image processing capabilities, including image restoration and generative fill, leveraging cloud-based resources for efficient processing.

**3. Data Collection and Preprocessing:**

- Quality Control and Validation:: Implement quality control measures to ensure dataset integrity, consistency, and accuracy. Validate annotations and labels to eliminate errors and discrepancies that could impact model training and evaluation.

- Define preprocessing steps: Image resizing, normalization, noise reduction, and augmentation techniques to enhance the quality and diversity of the dataset.

**4. Feature Extraction:**

- Choose appropriate features: Extract features from the pre-processed images that are relevant for image enhancement, such as restoration and generative fill.

- Explore different feature representation techniques: For example,AI features like core image enhancements or learned features from deep neural networks.

**5. Classification:**

- Image Category Classification:: Classify images into predefined categories such as landscapes, portraits, animals, objects, etc., based on their content and characteristics.

- Quality Classification: Classify images based on their quality, such as high-resolution, low-resolution, blurred, or noisy images. This can help in identifying images that may require specific enhancement techniques.

**6. Integration and Testing:**

- Integrate the different components into a cohesive system.

- Conduct unit testing for individual modules and integration testing to ensure that the system functions correctly as a whole.

- Perform end-to-end testing with real-world data to validate the system's performance under various conditions.

**7. Deployment and Maintenance:**

- Decide on the deployment strategy: Whether the system will be deployed as a standalone application, integrated into existing software, or deployed as a web service.

- Plan for system maintenance and updates: Regularly update the model with new data and retrain it to adapt to changes in user needs or environmental conditions.

**8. Documentation and User Support:**

- Document the system architecture, algorithms used, and implementation details for future reference.

- Provide user documentation and support materials to help users understand how to use the system effectively.

**CHALLENGES FACED BY HAND SIGN ALPHABET DETECTION:**

* Integration with Cloudinary API: Seamlessly integrating the Cloudinary API to handle advanced image processing tasks requires thorough understanding and effective implementation of the API capabilities.
* Performance Optimization: Ensuring fast and efficient processing of images, particularly large files, can be challenging. Optimizing algorithms and leveraging cloud computing resources is essential to maintain performance.
* Scalability: Designing the system to handle a large number of concurrent users and image processing tasks without degradation in performance is a significant challenge.
* Real-Time Processing: Providing real-time feedback and previews for image enhancements demands robust and responsive backend infrastructure.
* User Interface Design: Creating an intuitive and user-friendly interface that accommodates a wide range of user skill levels while providing access to advanced features is a complex design challenge.

**LITERATURE SURVEY**

## INTRODUCTION:

## A literature survey, also known as a literature review, is an essential component of academic research where existing scholarly works relevant to a particular topic or research question are systematically analyzed, summarized, and synthesized. It involves reviewing a wide range of sources such as books, journal articles, conference papers, and other academic publications to gain a comprehensive understanding of the current state of knowledge in the field. Literature surveys serve multiple purposes, including identifying key concepts, theories, and methodologies, evaluating existing research findings, identifying gaps or contradictions in the literature, and providing a foundation for the researcher's own study. Ultimately, a literature survey helps situate the researcher's work within the broader context of existing scholarship and guides the development of research questions, hypotheses, and methodologies.

**PHASES OF AI IMAGE ENHANCER:**

## 1. Problem Definition:- Clearly define the objective of the project, which is to develop a system capable of AI Image enhancement.

## 2. Requirement Analysis:- Gather requirements from potential users, stakeholders, and domain experts to understand the desired functionality and usability of the system.

## 3. Design:- Design the architecture of the system, including the components for image processing, feature extraction, classification, and user interface.

## 4. Data Preprocessing:- Clean and preprocess the dataset by resizing images, normalizing pixel values, and removing noise.

## 5. Model Integration:-Integrate a machine learning or deep learning model for image enhancement.

## 6. Evaluation:- Evaluate the trained model's performance using the validation and test sets.

## 7. Integration:- Integrate the trained model into a user-friendly interface, allowing users to input hand sign images and receive predictions of the corresponding alphabet letters.

## 8. Testing:- Conduct thorough testing of the system to identify and fix any bugs or issues.

## 9. Deployment:- Deploy the system in a production environment, such as a web application, mobile app, or standalone software.

## 10. Maintenance and Updates:- Provide ongoing maintenance and support for the deployed system, including bug fixes, performance enhancements, and updates to accommodate changes in technology or user requirements.

## ClOUDINARY’S ALGORITHM:

## Introduction:

## Cloudinary’s advanced media management platform leverages cutting-edge machine learning algorithms to empower intelligent image and video processing. These algorithms form the backbone of Cloudinary's capabilities, enabling automatic enhancements and transformations that streamline media workflows. The platform employs a variety of machine learning techniques, each tailored to optimize specific types of media tasks. Supervised learning algorithms enhance media quality and detect features from labeled data, while unsupervised learning algorithms discover patterns and segment images without pre-labeled data. Cloudinary also integrates deep learning algorithms, inspired by neural networks, to handle complex transformations and intricate media enhancements. These powerful algorithms enable Cloudinary to provide dynamic, scalable, and efficient media management solutions for diverse user needs.

## Working of Cloudinary’s machine learning algorithms:

## 1. Data Collection: The first step involves gathering relevant data that represents the problem domain. This data can include various features or attributes that influence the outcome.

## 2. Data Preprocessing: Before feeding the data into the algorithm, it undergoes preprocessing steps such as cleaning (handling missing values, removing outliers), normalization (scaling features to a standard range), and feature engineering (selecting, transforming, or creating new features).

## 3. Validation: After training, the model's performance is evaluated on a separate portion of the data called the validation set. This helps assess how well the model generalizes to unseen data and whether it's overfitting or underfitting.

## 4. Testing: Once the model is trained and validated, it's evaluated on a final portion of the data called the test set to estimate its performance in real-world scenarios.

## 5. Monitoring and Maintenance: The model's performance is continuously monitored, and updates or retraining may be necessary to adapt to changing data distributions or business requirements.

## Advantages:

## 1. Accuracy: Machine learning algorithms can often achieve higher accuracy than traditional statistical or rule-based approaches. By learning patterns and relationships from data, they can make predictions or classifications with greater precision, especially in complex and high-dimensional datasets.

## 2. Adaptability: Machine learning algorithms are capable of learning from new data and adapting their behavior accordingly. They can continuously improve their performance over time as they encounter more examples and feedback, making them suitable for dynamic and evolving environments.

## Disadvantages:

## 1. Data Dependency: Machine learning algorithms heavily rely on the quality and quantity of the data they are trained on. If the training data is biased, incomplete, or unrepresentative of the real-world scenarios, the model's performance may suffer, leading to inaccurate predictions or biased outcomes.

## 2. Interpretability: Some machine learning algorithms, particularly complex deep learning models, lack interpretability, making it difficult to understand how they arrive at their predictions or decisions. This lack of transparency can be problematic in critical applications such as healthcare or finance, where decision-making must be explainable and accountable.

## Cloudinary Console:

## 

## Cloudinary Console

## 

## IMPORTED LIBRARIES:

## React:

## React is a popular JavaScript library for building user interfaces, particularly for web applications. It's maintained by Facebook and a community of developers. React is known for its efficiency, component reusability, and declarative nature, making it a popular choice for building interactive and dynamic user interfaces..

## Next.js:

## Next.js is a popular open-source React framework that simplifies the development of server-side rendered (SSR) React applications. It's maintained by Vercel and a large community of developers.

## Cloudinary:

Cloudinary is a cloud-based image and video management platform. It provides a range of features for manipulating and optimizing media assets, including Image Transformation, Responsive Images, Media Optimization, Digital Asset Management. Next.js and Cloudinary are often used together in web development projects, leveraging Next.js for building efficient React applications and Cloudinary for managing and optimizing media assets.

**SOFTWARE REQUIREMENT SPECIFICATION**

**INTRODUCTION:**

A Software Requirements Specification (SRS) is a foundational document in software development, serving as a detailed blueprint for the entire project. Comprising essential information about the software's functionality and behaviour, it acts as a bridge between the client's needs and the development team's implementation strategies. Typically, an SRS includes various sections detailing the functional and non-functional requirements, user interfaces, system constraints, and other critical aspects of the software.

The primary purpose of an SRS is to define what the software should accomplish and how it should perform under different scenarios. By clearly outlining the scope and objectives of the project, it helps stakeholders align their expectations and provides a reference point for evaluating the final product. Additionally, the SRS serves as a communication tool, facilitating effective collaboration between clients, developers, testers, and other project stakeholders.

In essence, the SRS acts as a contract between the client and the development team, ensuring that everyone involved has a shared understanding of the project's goals and requirements. It helps minimize misunderstandings and ambiguities, reducing the risk of scope creep and ensuring that the final product meets the client's expectations. Throughout the software development lifecycle, the SRS serves as a guiding document, informing design decisions, development priorities, and testing strategies to ultimately deliver a successful software solution.

**OVERALL DESCRIPTION:**

This section will give an overview of the whole system. The system will be explained in its context to show how the system interact with other system and introduce the basic functionality of it. It will also describe what type of user will use the system and what functionality is available for each type.

The Software Requirements Specification (SRS) is a comprehensive document that outlines the detailed description of the requirements for a software system. It serves as a contract between the stakeholders involved in the software development process, providing a clear understanding of what the software should accomplish. The SRS document includes an introduction that provides an overview of the software system, its purpose, and the intended audience. It also covers the system overview, functional requirements, non-functional requirements.

**COMPLETE STRUCTURE OF THE PROJECT:**

**Requirements:**

|  |  |
| --- | --- |
| **SOFTWARE REQUIREMENT** | |
| **Front end** | JavaScript |
| **Operating System** | Windows 10 and above |
| **Development Tool** | Visual Studio Code |
| **HARDWARE REQUIREMENTS** | |
| **Processor** | Intel i5 |
| **RAM** | 4 GB |

Table 3.1

**FUNCTIONAL REQUIREMENT:**

1. **Image Upload:**

Users should be able to upload images from their devices or external sources to the application.

1. **Image Enhancement Options:**

Provide various enhancement options such as restoration, generative fill, color correction, and artistic effects.

1. **Preview and Comparison:**

Allow users to preview enhanced images and compare them with the original for visual assessment.

1. **Customization Settings:**

Offer customization settings for enhancement parameters such as brightness, contrast, saturation, and filters.

1. **Save and Download:**

Enable users to save enhanced images and download them in different formats and resolutions.

1. **Responsive Design:**

Ensure a responsive design that adapts to different screen sizes and devices for a consistent user experience.

**SYSTEM DESIGN DESCRIPTION**

**INTRODUCTION:**

System design:

Systems Design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. It involves translating user requirements into a detailed blueprint that guides the implementation phase. The goal is to create a well-organized and efficient structure that meets the intended purpose while considering factors like scalability, maintainability, and performance.

The introduction to system design sets the stage for the subsequent sections of the design document. It provides an overview of the project, its objectives, and the intended audience. The introduction typically includes a brief description of the problem or opportunity that the system aims to address, along with an explanation of the importance and relevance of the system in the given context.

Design:

A software design is a description of the structure of the software to be implemented the data, which is part of the system, the interfaces between the system components and sometimes the algorithms used. Designers do not arrive at a finished design immediately but develop the design iteratively through several different versions.

Objectives of system design:

* Practicality: We need a system that should be targeting the set of audiences(users) corresponding to which they are designing.
* Accuracy: Above system design should be designed in such a way it fulfils nearly all requirements around which it is designed be it functional o non-functional requirements.
* Completeness: System design should meet all user requirements
* Efficient: The system design should be such that it should not overuse surpassing the cost of resources nor under use as it will by now, we know will result in low thorough put (output) and less response time(latency).

**FUNCTIONAL DECOMPOSITION:**

Functional decomposition is a system design and analysis method that involves breaking down a complex system into smaller, more manageable components. It uses top-down approach where you start with the overall system or process and then divide that into its sub-components or sub-processes.

**SYSTEM ARCHITECTURE:**

Software Architecture defines fundamental organization of a system and more simply defines a structured solution. It defines how components of a software system are assembled, their relationship and communication between them. It serves as a blueprint for software application and development basis for developer team.

System software architecture:

Input image

API request

Operation Detection

Image Processing

Output

Cloudinary model

Fig:4.3.1 System software architecture

System technical architecture:

Output

(CSV&HTML)

Cloudinary model

Error

ABSENTEES

Input Image

API Request and Image processing

Fig:4.3.2 System technical architecture

**SYSTEM BLOCK DIAGRAM:**

Operation

Detection

API

Request

Image input

Output

Image

API Response

Image Transformation

Fig 4.4 System block diagram

The proposed system follows a vision-driven methodology to perform AI-based image enhancement and transformation on digital images. The enhancement process consists of three phases: data collection, data pre-processing, and feature extraction and transformation. After the data is collected and pre-processed, the feature extraction process is initiated. During this process, image features and landmarks are extracted as key points from the input images. These essential data points are then used to perform various enhancements, such as image restoration and generative fill. The enhanced images are further processed using Cloudinary's API and custom algorithms to achieve high-quality transformations. Finally, the enhanced images are displayed to the user in real time through a seamless and intuitive web interface..

**4.5 DESCRIPTION OF THE PROGRAM:**

**4.5.1 Context flow diagram:**

A context flow diagram, also known as a system context diagram or simply a context diagram, is a visual representation that provides an overview of a system and its interactions with external entities. It's often used in systems engineering, software engineering, and business analysis to illustrate the scope and boundaries of a system.

In a context flow diagram:

* System Boundary: The system being analyzed is represented by a single central box. This box contains the name or identifier of the system.
* External Entities: These are entities outside the system boundary that interact with the system in some way. They could be users, other systems, hardware devices, or even data sources like databases.
* Data Flows: Arrows are used to show the flow of data or information between the system and external entities. These arrows represent the inputs and outputs exchanged between the system and its external entities.
* Data Labels: Each arrow typically includes a label describing the type of data or information being exchanged.

**Input image**

User

AI Image Enhancer

**Cloudinary**

**SDK Output**

Fig 4.5.1 Context flow diagram

**4.6 DATA FLOW DIAGRAM:**

A data flow diagram (DFD) is a graphical representation of the flow of data through a system or process. It illustrates how data is input into a system, processed, and then output in a structured and visual manner. In a DFD, processes, data stores, external entities, and data flows are represented by various symbols connected by arrows to show the flow direction.

By using DFD, we can easily understand the overall functionality of a system because diagram represents the incoming data flow, outgoing data flow, and stored data in a graphical format.

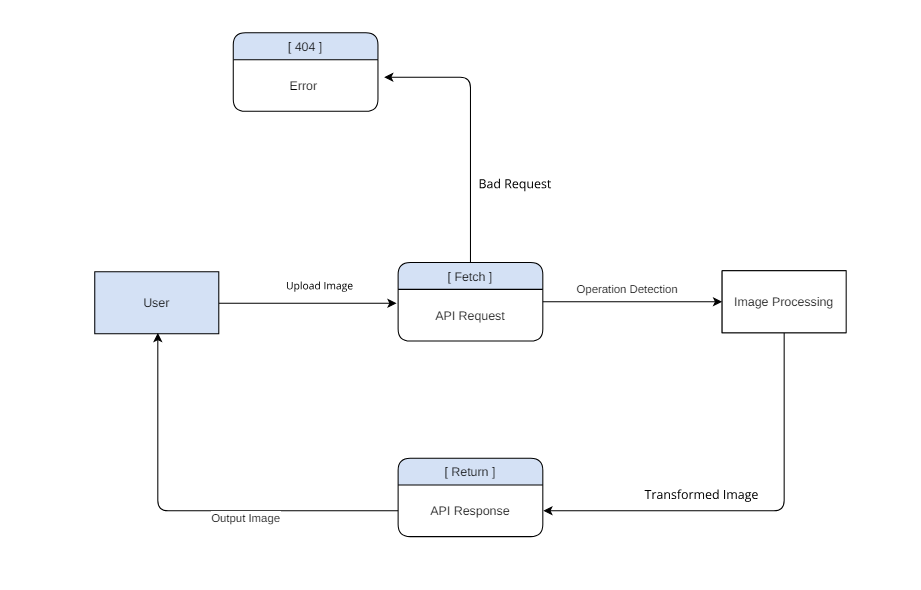


Fig 4.6 Data flow diagram

**MAIN PROGRAM CODE**

"use client"

import { zodResolver } from "@hookform/resolvers/zod"

import { useForm } from "react-hook-form"

import { z } from "zod"

import { Button } from "@/components/ui/button"

import {

Select,

SelectContent,

SelectItem,

SelectTrigger,

SelectValue,

} from "@/components/ui/select"

import { Form } from "@/components/ui/form"

import { Input } from "@/components/ui/input"

import { CustomField } from "./CustomField"

import { aspectRatioOptions,defaultValues, transformationTypes } from "@/constants"

import { AspectRatioKey, debounce, deepMergeObjects } from "@/lib/utils"

import { useEffect, useState} from "react"

import MediaUploader from "./MediaUploader"

import TransformedImage from "./TransformedImage"

export const formSchema = z.object({

title: z.string(),

aspectRatio: z.string().optional(),

color: z.string().optional(),

prompt: z.string().optional(),

publicId: z.string(),

from:z.string().optional(),

})

export default function TransformationForm({ type, config = null }: TransformationFormProps) {

const [image, setImage] = useState(null);

const transformationType = transformationTypes[type];

const [newTransformation, setNewTransformation] = useState<Transformations | null>(null);

const [isTransforming, setIsTransforming] = useState(false);

const [transformationConfig, setTransformationConfig] = useState(config)

//~Defining form using Form Schema:

const form = useForm<z.infer<typeof formSchema>>({

resolver: zodResolver(formSchema),

defaultValues: defaultValues,

})

//~SubmitHandler :

async function onSubmit(values: z.infer<typeof formSchema>) {

return;

}

//~OnChange field for Select :

const onSelectFieldHandler =(value:string,onChange:(value:string)=>void)=>{

const imageSize = aspectRatioOptions[value as AspectRatioKey];

setImage((prevState: any) => ({

...prevState,

aspectRatio: imageSize.aspectRatio,

width: imageSize.width,

height: imageSize.height,

}))

setNewTransformation(transformationType.config);

return onChange(value)

}

//~Onchange field for Input field :

const onInputChangeHandler = (fieldName: string, value: string, type: string, onChangeField: (value: string) => void) => {

debounce(() => {

setNewTransformation((prevState: any) => ({

...prevState,

[type]: {

...prevState?.[type],

[fieldName === 'prompt' ? 'prompt' :fieldName === 'from'? 'from' :'to' ]: value

}

}))

}, 1000)();

return onChangeField(value)

}

//~Transformation button handler :

const onTransformHandler = async ()=>{

setIsTransforming(true)

setTransformationConfig(

deepMergeObjects(newTransformation, transformationConfig)

)

setNewTransformation(null)

}

useEffect(() => {

if(image && (type === 'restore')) {

setNewTransformation(transformationType.config)

}

}, [image, transformationType.config, type])

return (

<Form {...form}>

<form onSubmit={form.handleSubmit(onSubmit)} className="space-y-4">

{type === 'fill' && (

<CustomField

control={form.control}

name="aspectRatio"

formLabel="Aspect Ratio"

className="w-full"

render={({ field }) => (

<Select

onValueChange={(value) => onSelectFieldHandler(value, field.onChange)}

value={field.value}

>

<SelectTrigger className="select-field">

<SelectValue placeholder="Select size" />

</SelectTrigger>

<SelectContent>

{Object.keys(aspectRatioOptions).map((key) => (

<SelectItem key={key} value={key} className="select-item">

{aspectRatioOptions[key as AspectRatioKey].label}

</SelectItem>

))}

</SelectContent>

</Select>

)}

/>

)}

{/\* //=MediaUploaderField: \*/}

<div className="media-uploader-field">

<CustomField

control={form.control}

name="publicId"

className="flex size-full flex-col"

render={({ field }) => (

<MediaUploader

onValueChange={field.onChange}

setImage={setImage}

publicId={field.value}

image={image}

type={type}

/>

)}

/>

<TransformedImage

image={image}

type={type}

title={form.getValues().title}

isTransforming={isTransforming}

setIsTransforming={setIsTransforming}

transformationConfig={transformationConfig}

hasDownload={true}

/>

</div>

<div className="flex flex-col gap-4">

<Button

type="button"

className="submit-button capitalize"

disabled={isTransforming || newTransformation === null}

onClick={onTransformHandler}

>

{isTransforming ? 'Transforming...' : 'Apply Transformation'}

</Button>

</div>

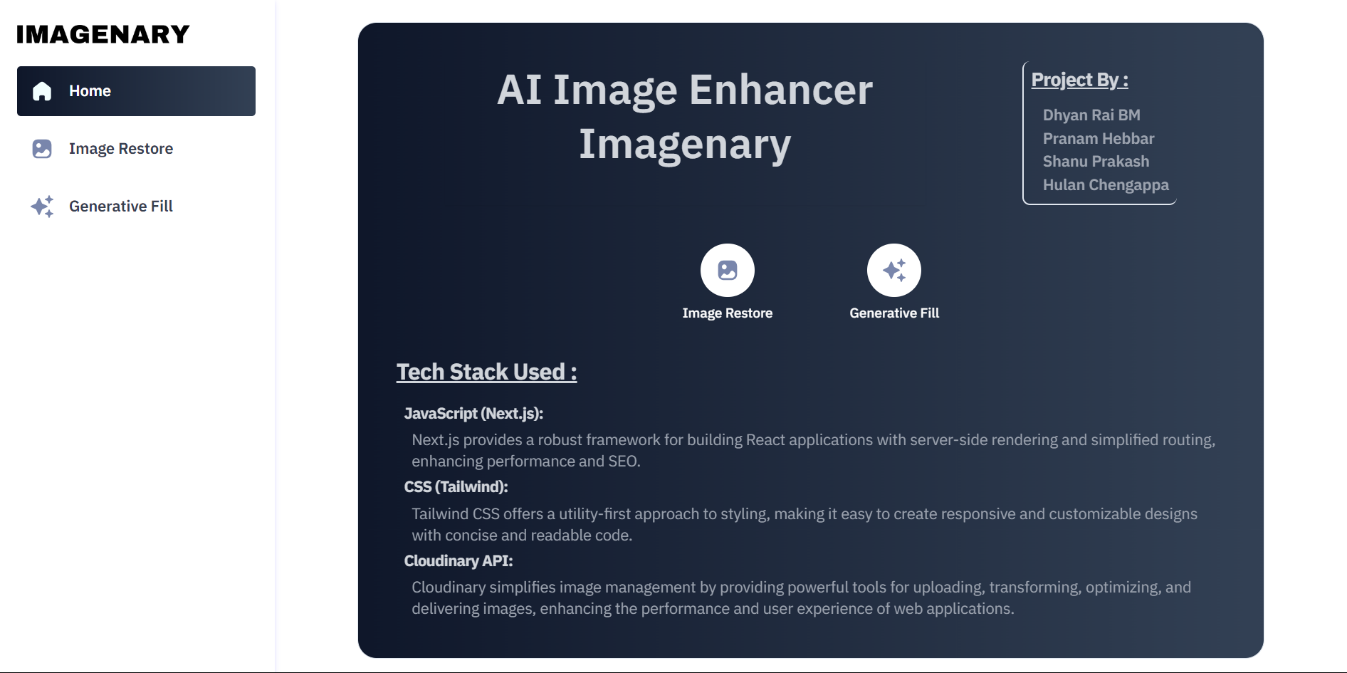
</form>

</Form>

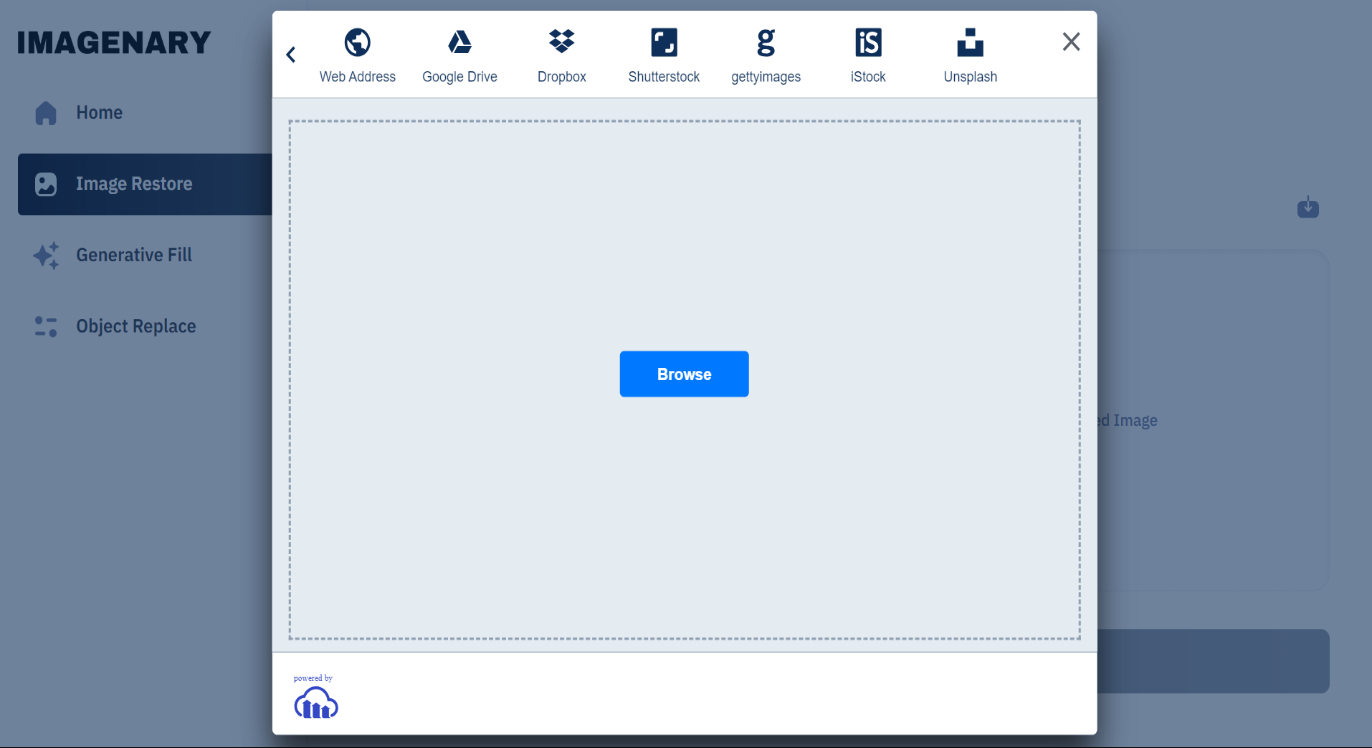
)

}

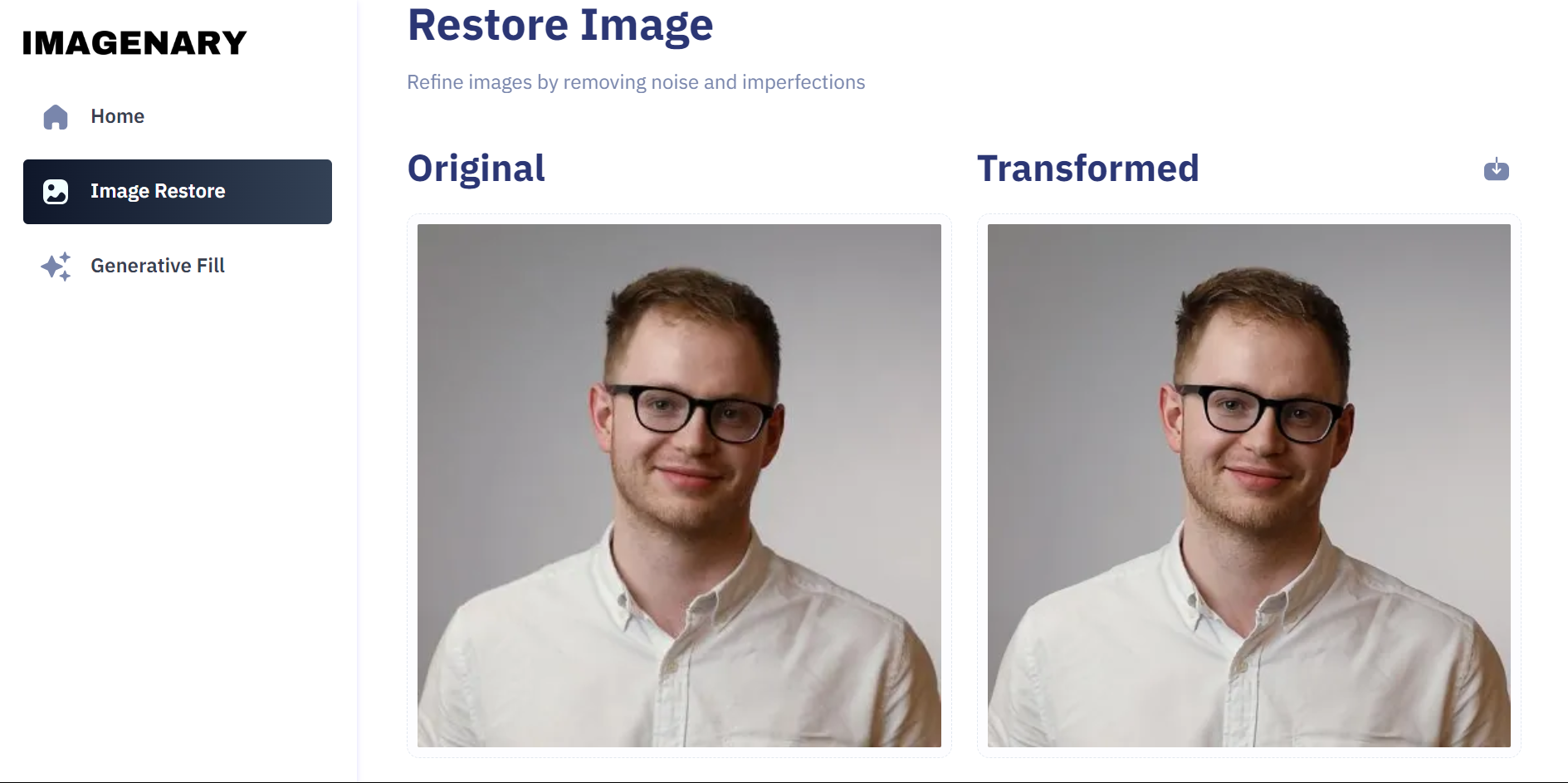
**USER INTERFACE**



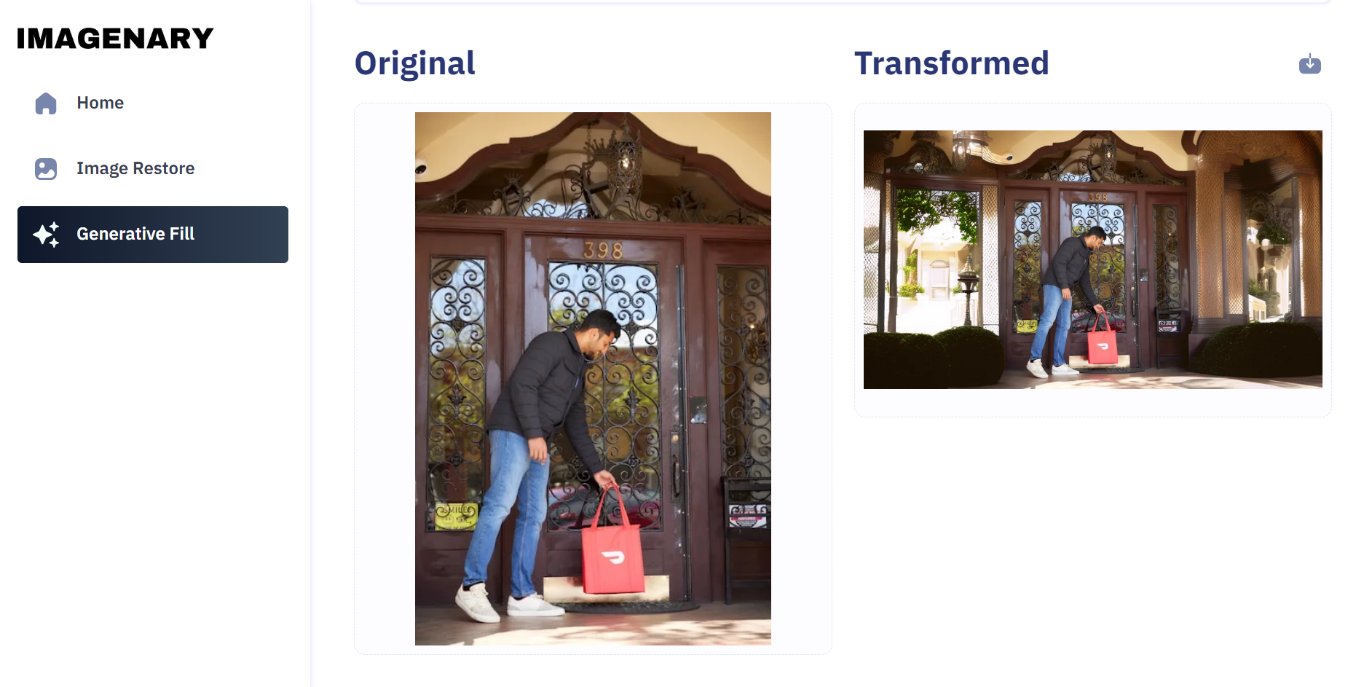
* 1. Home page User Interface



6.2 Cloudinary Image Upload Panel



6 .3 Image Restore - Before and After



6.4 Generative Fill - Before and After

**TESTING**

**SYSTEM TESTING:**

In the system the behaviour of whole system product is tested as defined by the scope of the development project. It may include tests based on risks or requirement specification, business process, use cases or other high-level description of the system behaviour. Interaction with the operating system, resources. System testing is most often the final test to verify that the system to be delivered meets the specification and its purpose.

**TESTING IN REAL-TIME:**

Using live feed, real-time image enhancement and transformation were demonstrated with promising results. In a real-time testing scenario for Imagenary, users upload their digital images via the web interface, and the app performs immediate enhancements, such as image restoration and generative fill.

The process is streamlined to ensure swift and seamless user interaction. Once an image is uploaded, the app leverages the Cloudinary API and custom algorithms to process and enhance the image in real time. The enhanced image is then displayedback to the user almost instantaneously, showcasing the app's capability to deliverhigh-quality image transformations with minimal delay. This real-time testing confirms Imagenary's effectiveness and efficiency in providing powerful image enhancement tools accessible to everyone.

**FUTURE SCOPE AND ENHANCEMENT**

The proposed Imagenary system is an advanced AI-driven platform designed to enhance and transform digital images. While it currently excels in features such as image restoration and generative fill, its capabilities can be further extended to include a broader range of image enhancement techniques. Instead of focusing solely on basic enhancements, future iterations could incorporate more complex transformations like facial feature refinement and artistic style transfer, increasing the app's versatility.

One potential development is to provide more contextual enhancements, allowing users to select specific areas of an image for targeted adjustments. This would increase the precision and relevance of the enhancements. Additionally, the system could be expanded to support batch processing, enabling users to apply enhancements to multiple images simultaneously, thus improving workflow efficiency.

A future goal could be to develop a complete product that caters to a wider audience, including professional photographers, digital artists, and casual users. By putting the entire system online, any user could utilize their device's camera or upload images to quickly and effectively enhance them. This would bridge the gap between professional-grade image enhancement and user-friendly accessibility.

Ultimately, by continuously refining the algorithms and expanding the range of supported transformations, Imagenary aims to become a comprehensive solution for all digital image enhancement needs. Integrating the system with mobile applications and providing language selection options would further enhance its usability and appeal, making high-quality image transformation tools accessible to a global audience.

**CONCLUSION**

In conclusion, our Imagenary project marks a significant advancement in the realm of AI-driven image enhancement, offering a powerful platform for users to restore and transform their digital images. Through the integration of cutting-edge AI technologies and the Cloudinary API, we have developed a system that not only performs complex image enhancements with precision but also makes these tools accessible to a broad audience. Our results highlight the potential of Imagenary to enhance visual quality and unlock creative possibilities across various industries.

While the project has delivered impressive capabilities, there are still opportunities for growth and improvement. Future work could focus on expanding the range of supported image transformations, enhancing the system's performance under diverse conditions, and integrating more comprehensive training data. Additionally, exploring mobile integration and user-specific customization options could further increase the platform's usability and appeal.

Overall, the Imagenary project demonstrates the transformative power of AI and cloud-based technologies in the field of digital image enhancement. By continuing to innovate and refine our approach, we aim to provide even more robust and versatile tools that empower users to achieve their creative visions and improve the quality of their digital imagery.

**REFERENCES.**

1. React Documentation. Available at: https://reactjs.org/docs/getting-started.html (accessed: 07.03.2024).
2. Tailwind CSS Documentation. Available at: <https://tailwindcss.com/docs> (accessed: 07.03.2024).
3. Next.js Documentation. Available at: https://nextjs.org/docs (accessed: 01.06.2024).
4. "Learning React: A Hands-On Guide to Building Web Applications Using React and Redux". Publisher: Addison-Wesley Professional, 2017. Available at: https://www.oreilly.com/library/view/learning-react-a/9780134843579/
5. ShadCn UI .Available at: (https://ui.shadcn.com/)

-