Designing a Framework for Enhancing Images using DNN, GAN and MIRNet

by

SASHANK SHEKHAR SHUKLA 200014325013

SALMA FAIZ 209000000048

SHIKHA SHRIVASTAVA 200014325005

Submitted in Partial Fulfillment of the Requirements for the award of the degree

of

MASTER OF COMPUTER APPLICATION

Under the Guidance

of

Er. Anshu Singh

Faculty of Engineering and Technology , University of Lucknow

Lucknow, (U.P.) India.

Certificate

We hereby certify that the work, which is being presented in the report, entitled **Designing a Framework for Enhancing Images using DNN, GAN and MIRNet**, in fulfillment of the requirement for the award of the degree of **Master of Computer Application** and submitted to the institution is an authentic record of my/our own work carried out during the period *January-2022* to *May-2022* under the supervision of Er. Anshu Singh. We also cited the reference about the text(s)/figure(s)/table(s) from where they have been taken.

Er. Anshu Singh

In−charge CSE Departement

Date

The final copy of this project has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Candidate's Declaration

We hereby certify that we have properly checked and verified all the items as prescribed in the check-list and ensure that my project is in the proper format as specified in the guideline for major project.
We declare that the work containing in this report is my own work. We understand that plagiarism is defined as any one or combination of the following:

(1) To steal and pass off (the ideas or words of another) as one's own

(2) To use (another's production) without crediting the source

(3) To commit literary theft

(4) To present as new and original idea or product derived from an existing source.

We understand that plagiarism involves an intentional act by the plagiarist of using someone else's work/ideas completely/partially and claiming authorship/originality of the work/ideas. Verbatim copy as well as close resemblance to some else's work constitute plagiarism.
We have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programmes, experiments, results, websites, that are not my original contribution. We have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.
We affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, We shall be fully responsible and answerable. My faculty supervisor(s) will not be responsible for the same.

Signature:
Name: Sashank Shekhar Shukla
Roll. No: 200014325013
Date: $2^{nd} June$ 2022

Signature:
Name: Shikha Srivastava
Roll. No: 200014325005
Date: $2^{nd} June$ 2022

Signature:
Name: Salma Faiz
Roll. No: 209000000048
Date: $2^{nd} June$ 2022

Abstract

As the technology around us is evolving rapidly day by day, it is bringing major changes to us and our surroundings, and one of the fields is "Image Technology". Few decades ago, people were able to capture their moments, then few years latter they advanced to futuristic features like using images for security purposes, navigation, and entertainment. Now, with the collaboration of artificial intelligence, it has been so advanced that it can paint a person as how has he been looked like in past centuries to how will he look after few decades. Here we have knowledge available about Image Processing and Artificial Intelligence. So, we want to utilize this knowledge to assist the forensics operations. We are about to design a software which will assist the forensics operation like objects detection, a face matching system, upscaling, denoising, smoothing, coloring, and sharpening images to study a image in details, so that a forensic expert can infer the knowledge easily with speed and accuracy. This system will be very helpful to forensic experts. Along with it, this software have a new Input Output method. It supports I/O from remote devices existing in LAN.

KEYWORDS : Object detection, Face matching, Upscaling, Denoising, Smoothing, Coloring, Artificial Intelligence and Image Processing. Input/Output over LAN

Acknowledgments

We would like to express our deepest appreciation to our keen and intuitive mentor **Er. Anshu Singh** for guiding us through out the entire project work and helping with new concepts and technologies.

We are extremely thankful to our brilliant and resourceful faculty **Er. Namrata Singh** for making the completion of this enormous project possible. We would also like to extend our deepest gratitude to our young and dynamic project coordinator **Er. Pankaj Roy** for helping us with related works. We are deeply indebted to our intelligent faculty **Er. Rohit Yadav** who exhorted us about LAN, Routing and Shearing using Internet Protocol. We would like to express our deepest appreciation to our keen and intuitive faculty **Er. Prachi Verma** for her crucial insight into concepts of Machine Learning, which helped us a lot. We gratefully acknowledge the assistance of our humble and intelligent faculty **Er. Sandeep Kumar** whose lessons on transformation helped us a lot through out the development. Last but not the least, we are extremely grateful to vary friendly and humble crew member of University's Aid Staff, Mr. Rishabh, he waited for us so that we can code few more.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

1.1 Context

This System uses the various concepts of Image Processing and Machine Learning to give users a much accurate, satisfying and deciding results. The context of the system discussed here is the digital image forensics, as it takes images as inputs, and provides various tools to manipulate the image data in order to gather some meaningful information or to make the data more meaningful.

1.2 Motivation

Rapidly emerging technology is changing our surrounding and lifestyle at very fast rate. As few years ago, technology was supposed to perform tasks leading to the future, but now a days, it's capable to perform task of the future, as well as the past. For example, earlier, software's were able to clear the face in a passport size photo, then they evolved to portrait a person few years ahead and back. Now they are advanced in such a way that they can portrait a person as he would has looked few centuries ago, and his kids may look like in the future. With the capability of machines to learn automatically with the help of Machine Learning Technology, much complex tasks can be performed with ease. As the advancement of the technology has been found misused at certain places, it is necessary to get control over it, which is nearly impossible, but validation of such thing is in our hands. So, it is necessary to have a system that can validate anything that has been passed through any technology. It motivated us to design a system that can validate a major participating thing present widely around us, we decided to design a system which can assist human being with

the validation process of an image, and other things to assist the forensic operation.

1.3 Objectives

The name of this project is self-explaining the objective of the project, in brief, this project is supposed to assist a forensic operation with image data. This project will provide various tools like AI powered upscaling [1], difference based object cutting, Noise reduction [9], AI powered recoloring, Glare removal, AI powered face matching [10], various filters, pixel level control to replace a specific color or a range of colors. With such ground breaking tools, this project will have very good scope in government's investigation organizations as well as private sectors. This project is not limited to just one sector, it can be used by various organizations in matching faces, engineers, doctors and students can use it to cut objects from a diagram, people can re-colorize and enhance the quality of their old images. For fun, people can easily modify the color of an object. All of these with ease.

1.4 Scope of the Project

Till now, a lot of Image Processing Software for desktop are available out there in the world, but those software do not give results automatically, those software require high level knowledge for quality output, and work manually. This major disadvantage of available systems open wide scope for our project.

1.4.1 In Forensics

This project is capable of matching faces [10], up scaling low quality images [2], finding missing objects [6],removing Low Light [2], recolouring an image [4] and removing noise from an image [9]. Such features are a great help to Forensic Department where these features can be used for analysis of a night time noisy low resolution image.

1.4.2 In Industries

Industries can use Re-colouring feature to automatically change the colour channel of an image.

1.4.3 In Entertainment

This project is capable of converting a black and white movie into a coloured one, also can increase the quality of old 360p movie into a good quality movie with resolution up to 4K.

1.4.4 In Securing a place

This project contains a face matching module which can be separately implemented with any other python or C++ project very easily.

1.4.5 In Education Sectors

Several students face problem while reading from low quality of images, specially when shared over WhatsApp, which automatically reduces the quality of the image while sharing. The high resolution module is very useful in this scenario.

1.5 Project Modules

1.5.1 Database Administration Module

This module manages the database, in this project, database contains mainly the images and a contact number associated with the input image. This is one of the important modules of our proposed system as it is designed not just to handle input images, but also provide security to database from SQL Injection, Intruders and any other miscellaneous activity against database.

1.5.2 Denoising Module

With the explosion in the number of digital images taken every day, the demand for more accurate and visually pleasing images is increasing. However, the images captured by modern cameras are inevitably degraded by noise, which leads to deteriorated visual image quality. Therefore, we developed a denoising module.

1.5.3 Face Matching Module

It is developed from scratch which takes two images as input, and uses a pre-trained model based on Convolution Siamese Neural Network [10], which compares two images and returns 1 or 0 as output, 1 if faces match, otherwise 0.

1.5.4 File Management Module

In the beginning, it was expected that images from the form data will be accepted by the programming modules, but when implemented, it failed to load, so to overcome this, we developed this module which basically receives the form image data, stores it into file system, to prevent duplicate data, it automatically gives a unique name to the image file, then saves it into media directory, and file path along with name is saved into imagebase table.

1.5.5 High Resolution Module

Upscaling playes very important role when it comes to analyse a low quality image. This module uses No-GAN architecture to upscale a low resolution image. This module increases the quality of an image upto **70 times.** [11]

1.5.6 Image Sharpening Module

Sometimes we need to increase the details of an image so that we can extract the feature of edge from that image more accurately, this model is about it. It takes an image as input and uses matrix multiplication method to sharpen the image, the image matrix is multiplied with a

kernel specially designed for sharpening, which values are decided on the basis of through study and research work on the behaviour of different types of kernel [12].

1.5.7 Image Database Module

This module stores and manages data for users of the website. this module provide URL's for the saved image, instead of saving the file into it. and further when images are retrived, is loads all the image urls, the visits each url, saves the data and sends it for display.

1.5.8 Kernel Module

In image processing, a kernel, convolution matrix, or mask is a small matrix used for blurring, sharpening, embossing, edge detection, and more. This is accomplished by doing a convolution between the kernel and an image, like Gaussian blur, edge-detection etc.

1.5.9 Local Area Network Management Module

Remote input and output is the new feature of this application, this module makes it possible. This module provides ability to give input from any device that that is connected to the network. It sends response of the request to devices connected in the network.

1.5.10 Low Light Removal Module

This module takes a low light image as input and uses MiRNET to balance the brightness and exposure of the input image [2].

1.5.11 Recolouring Module

It is one of the most sophisticated modules present in our project. This module takes a gray-scale image as input and uses a pretrained model to give output, which is a 3 channel matrix data which further can be increased to 4 channels [4].

1.5.12 Remote Input Module

This module activates the server with 0.0.0.0 IP address, which allows all the devices int the network to communicate with host device using host's IP address. So, devices connected into a network can send data to the host computer using it's IP address.

1.5.13 Remote Output Module

This module forwards the results to the client device, as request is made using remote input module, request is saved, then when results are ready, it is send back to the client, but this module is designed in such a way that if a user want output on a different device, he can get it, if he wants output on multiple devices, this feature is also available.

1.5.14 Secret Text Finder Module

This module takes an image as input and performs a difference based operation on the image matrix data in order to fetch out hidden structures present in the image.

1.5.15 Server Management Module

This module manages the server part of the application, It basically handle the requests to prevent various types of attack from any intruder, like XSS, Man in the Middle and clickjacking.

1.5.16 Single Session Hosting Module

It is necessary module that enables the remote input and output from the user and also helps in sending results to multiple devices while maintaining the security and a single session.

1.5.17 System Input Module

This module communicate with database of the application as well as the file system of the host device on which it runs.This module reads input from file, renames it, saves it into media directory, then makes URL and seves the URL into database.

1.5.18 System Output Module

This module allows user to save the result on the host machine.

1.5.19 Training Module

This module is for face matching feature, it takes labeled directory and reads data as labeled data, then performs shuffling and trains the freshly created Convolutional Siamese Neural Network Architecture. [10]

1.5.20 Web Tutorial Module

To make it easy to understand what this project is about, how it is working, and how to operate it. All helpful information is available in this module.

Chapter 2

Methodology

2.1 Proposed hypothesis

i By coloured image makes more sense than black and white one.

ii A neural network designed to recognise objects can also be modified to recognise faces.

iii Adding convolution layer to a neural network may improve it's accuracy by several times.

iv A learning model can perform better than a general function.

v A linear variation can be highlighted by performing double difference.

vi Matrices gives astonishing results when multiplied with each other.

2.2 Algorithms

2.2.1 Algorithm to search image containing a face in a directory

[TO CREATE AND TRAIN NEURAL NETWORK]

STEP 1 : Start

STEP 2 : Input image

STEP 3 : Reshape image to 100 pixel by 100 pixel

STEP 4 : Add 2D convolutional layer(feature=64, kernel=(10,10), activation=RelU)

STEP 5 : Add 2D maxpooling layer(feature=64, poolSize=(2,2), padding=same)

STEP 6 : Add 2D convolutional layer(feature=128, kernel=(7,7), activation=RelU)

STEP 7 : Add 2D maxpooling layer(feature=64, poolSize=(2,2), padding=same)

STEP 8 : Add 2D convolutional layer(feature=128, kernel=(4,4), activation=RelU)

STEP 9 : Add 2D maxpooling layer(feature=64, poolSize=(2,2), padding=same)

STEP 10: Add 2D convolutional layer(feature=256, kernel=(4,4), activation=RelU)

STEP 11: Flatten the resultant matrix of STEP 10

STEP 12: Dense the flattened output to 4096 units using sigmoid activation and save

STEP 13: Pass the input image and saved result to Model function in keras

STEP 14: Add L1 distance layer to the model

STEP 15: Train the model with labelled images and output

STEP 16: Save the model

STEP 17: Stop

[**TO USE THE SAVED MODEL**]

STEP 1: Load the model

STEP 2: Input the test image

STEP 3: For each image in directory

STEP 4: Input validation image and test image to model

STEP 5: If model returns 1, goto the step 7

STEP 6: If no more validation image, goto the step 8

STEP 7: Display the matched image and image name

STEP 8: Stop

2.2.2 Algorithm to upscale a low resolution image into a high resolution image using pretrained GAN

[**ESRGAN : ENHANCED SUPER RESOLUTION GENERARIVE ADVERSARIAL NETWORK**]

STEP 1: Load the pretrained model

STEP 2: Input image

STEP 3: image = image * 1.0/255

STEP 4: data = Unsqueeze image

STEP 5: Pass data to model

STEP 6: Output High Resolution Image

STEP 7: Exit

[ALGORITHM TO UNSQUEEZE the image data]

STEP 1: data equals empty array

STEP 2: For each channel in image

STEP 3: For each row in channel

STEP 4: For each column in row

STEP 5: append value of image[channel][row][column] to data

STEP 6: Return data

STEP 7: Exit

2.2.3 Algorithm to correct a low light image using pretrained ESRGAN

[ESRGAN : ENHANCED SUPER RESOLUTION GENERARIVE ADVERSARIAL NETWORK]

STEP 1: Load the pretrained model

STEP 2: Input image

STEP 3: $image = image * 1.0/255$

STEP 4: data = Unsqueeze image

STEP 5: Create inputFrames of size row=256 and column = 256 from data

STEP 6: for each inputFrame in inputFrames

STEP 7: pass inputFrame to model

STEP 8: replace inputFrame with the output of step 7

STEP 9: merge frames to create the output picture

STEP 7: Exit

2.2.4 Algorithm to recolour a black and white image using pre-trained NO-GAN Model

[NO-GAN : GAN That requires no-training]

STEP 1: Load the pretrained model

STEP 2: Input black and white image

STEP 3: Input the render factor

STEP 4: pass the input image and render factor to the model

STEP 5: Pass data to model

STEP 6: Output 3 channel colour image

STEP 7: Exit

2.2.5 Algorithm to Denoise/Sharp an image using Non-Local Mean

[NL-MEAN : NON LOCAL MEAN]

STEP 1: Input the image, patch size and search window

STEP 2: For each pixel in image

STEP 3: Cp = Summation of squared Euclidean distance of patch pixels

STEP 4: $pixel = 1/Cp * (SummationOfPixel * squaredEuclideanDistanceOfPatchPixels$

STEP 5: Reconstruct image from manipulates pixel values as output

STEP 6: Exit

2.3 Analytical validation

2.3.1 Different Methods to detect faces

METHOD	TEST
Humans	95.5
Hierarchial Bayesian Program Learning	95.2
Affine Model	81.8
Hierarchical Deep	65.2
Deep Boltzmann Machine	62.0
Simple Stroke	35.2
1-Nearest Neighbour	21.7
Siamese Neural Net	58.3
Convolutional Siamese Net	**92.0**

Table 2.1: Comparison Data

So, for our face detection system, we used Convolutional Siamese Net because it has 92.0 accuracy score, which is good and more feasible to modify for faces. [10]

2.3.2 Denoising Examples from SIDD

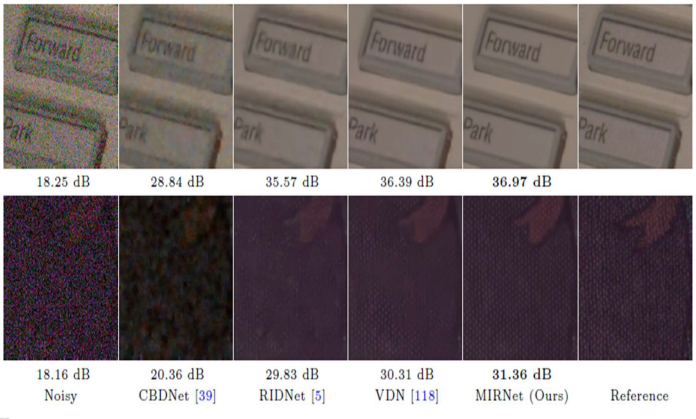


Figure 2.1: Denoising

2.3.3 High-Resolution for upscaling an image

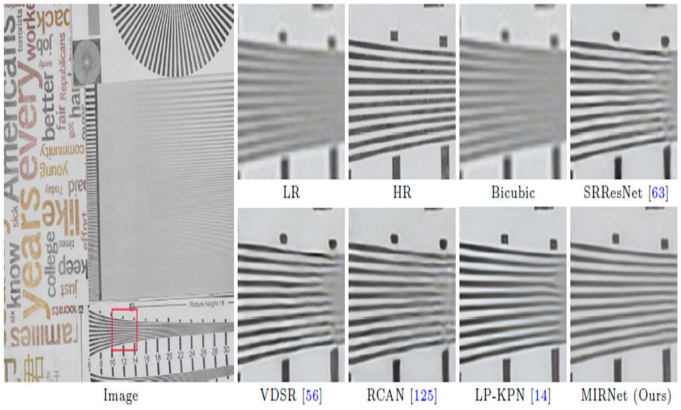


Figure 2.2: Upscaling

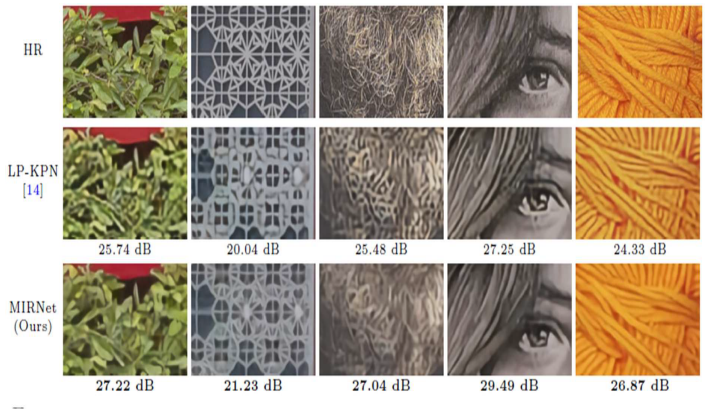


Figure 2.3: Results

Chapter 3

Software and Hardware Requirement

3.1 Hardware Specification Details

This software is very sophisticated in terms of development, even though it is developed using modular approach, it is impossible execute some of it's modules on most of the existing systems. The Hardware requirements for this software are discussed below

3.1.1 Developer End

(1) RAM :

(a) Capacity : 60 Gigabyte

(b) Frequency : 3200 Mega Hertz

(c) Type : DDR4

(d) Column Address Strobe Latency : Class 17

(e) Memory : Buffered

(2) CPU :

(a) Frequency : 4.7 Gigahertz

(b) Core : 14

(c) Hyper-threading : Enable, if core¡10

(d) L1 Cache : 12 Megabyte

(e) L cache : 24 Megabyte

(f) Memory Support : Yes, Double Data Rate

(g) DirectX : 11.0

(h) OpenGL : 4.3

(i) Instruction Set : 64-Bit

(3) GPU :

(a) Brand : Nvidia [only]

(b) CUDA cores : 5888

(c) Clock : 1.78 Gigahertz

(d) Memory : 12 Gigabyte

(e) Memory Type : GDDR6

(f) Tensor cores : 3^{rd} Gen

(g) Architecture : Ampere

(h) Encoder : 7^{th} Gen

(i) Decoder : 5^{th} Gen

(j) CUDA Capability : 8.6

(4) HD Monitor

(5) Peripherals

3.1.2 User End/Deployment End

(1) RAM :

(a) Capacity : 8 Gigabyte

(b) Frequency : 2400 Mega Hertz

(c) Type : DDR4

(d) Column Address Strobe Latency : Class 15

(e) Memory : Buffered

(2) CPU :

(a) Frequency : 2.2 Gigahertz

(b) Core : 2

(c) Hyper-threading : Enable, if core¡2

(d) L1 Cache : 512 Kilobyte

(e) L cache : 1 Megabyte

(f) Memory Support : Yes, Double Data Rate

(g) DirectX : 7.0

(h) OpenGL : 3.2

(i) Instruction Set : 64-Bit

(3) GPU :

(a) Brand : Nvidia [only]

(b) CUDA cores : 896

(c) Clock : 1.65 Gigahertz

(d) Memory : 4 Gigabyte

(e) Memory Type : GDDR5

(f) Tensor cores : 2^{nd} Gen

(g) Architecture : Ampere

(h) Encoder : 5^{th} Gen

(i) Decoder : 3^{rd} Gen

(j) CUDA Capability : N/A

(4) HD Monitor

(5) Peripherals

3.2 Software Specification Details

This software is built using open-source softwares only, zero paid packages or application programming interfaces are used in this project.

3.2.1 Developer End

(1) OS : Windows 10 Pro 21h2

(2) base language compiler/interpreter : Python 3.7

(3) IDE : Spyder 5.0

(4) Editor : Visual studio code 1.67

(5) Environment : conda

(6) Server : Django 4.0.3

(7) CUDA 11.3

(8) TensorboardX 2.9.0

(9) tensorflow 2.9.1

(10) opencv-python 4.5.5

(11) Jupyter Lab 3.0

(12) Browser

3.2.2 User End/Deployment End

(1) OS : Windows 10 Pro 21h2

(2) base language compiler/interpreter : Python 3.7

(3) Environment : conda

(4) Server : Django 4.0.3

(5) CUDA 11.3

(6) TensorboardX 2.9.0

(7) tensorflow 2.9.1

(8) opencv-python 4.5.5

(9) Browser

Chapter 4

System Design

4.1 Introduction

This System integrates several Artificial Intelligence modules with Image Processing concepts of Image Processing and Machine Learning to give users a much accurate, satisfying and deciding results. It uses GAN, No-GAN, MirNET and Convolutional Siamese Network which are trained on highly configured systems, and the end results are very satisfying.

4.2 Architecture Diagram

This is very heavy application, so it have very complex system architecture, but we have tried to make is simple as much as possible.

This system takes input from user, uses two methods for input, first on PC, and second by Remote device. Then the input is send to the server, where user's selection decides the procedure of the execution. There are several modules in that environment, some are Pre-trained Artificial Intelligence modules and some are traditional coded modules. These modules run on the system, and are only managed by server, this new concept gave us freedom to send outputs to a user which didn't gave input, but wants the output. And, further, output is served on both, offline, as well as on remote device.

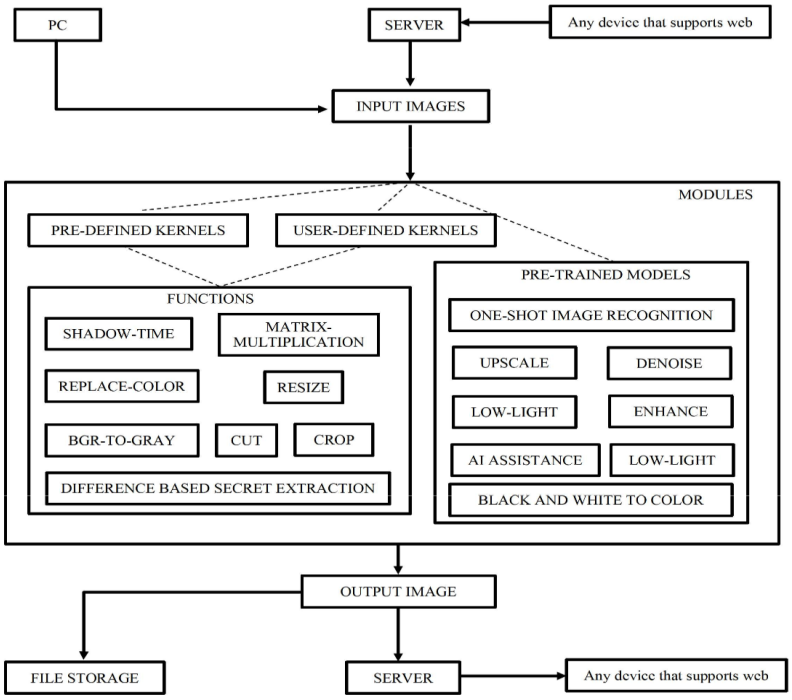


Figure 4.1: Project Architecture

4.3 Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system modeling its process aspects.
Often it is a preliminary step used to create an overview of the system that can later be elaborated. DFDs can also be used for the visualization of data processing (structured design) and show what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes or information about whether processes will operate in sequence or in parallel.
Processes or functions are shown as circles or sometimes as rounded rectangles. A process shows a part of the system that transforms inputs into outputs. The label should be a word, phrase, or short sentence that says what the process does—for example, "Find information on DFDs."

Flows are shown as arrowed lines (either straight or curved). The label should say what kind of information or item moves along the flow—for example, "Web link."

Stores (places where data are stored) are shown as two parallel lines or an open-ended rectangle. Not all systems have stores.

The label is generally the plural of the name of the items carried by the flow into and out of the store—for example, "Web links." (This implies that the same items go in and out, so the flows into and out of the store will have the same labels.)

Terminators, external entities or people with which the system communicates, are shown as rectangles. The label is the name of the terminating entity (for example, "Web application book"), person, or group of people.

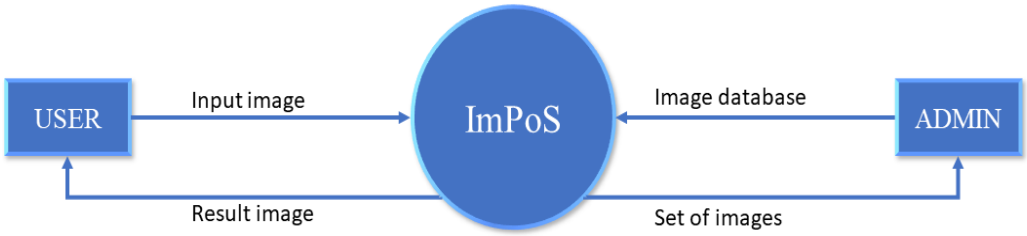


Figure 4.2: Level - 0 DFD

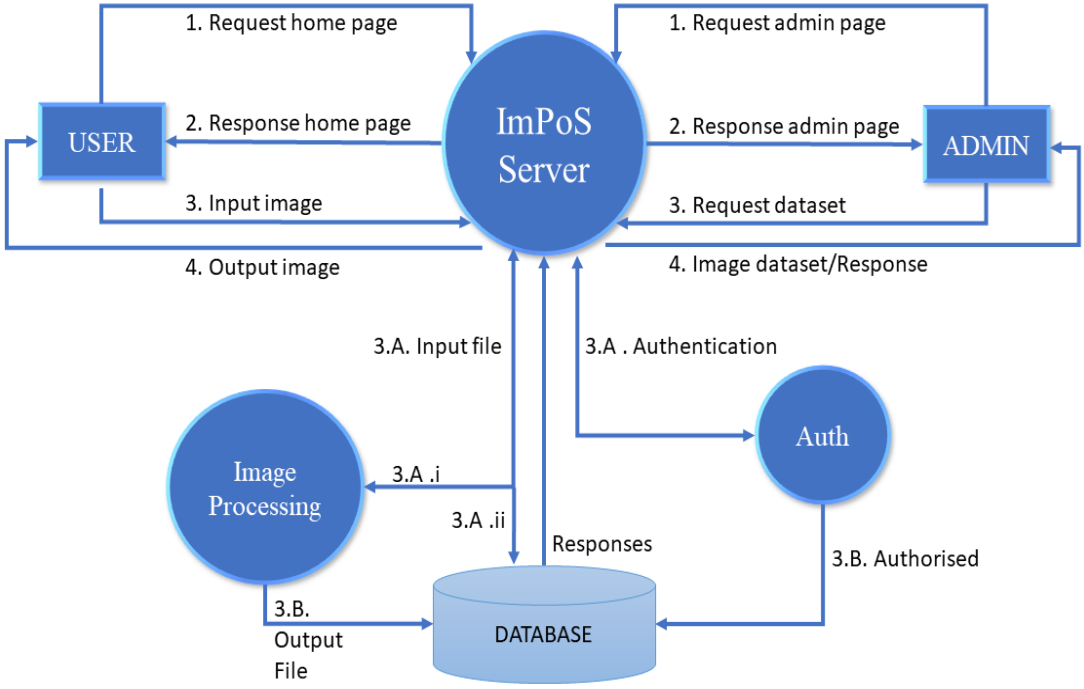


Figure 4.3: Level - 1 DFD

4.4 ER diagram

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how "entities" such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the inter-connectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs. Detailed procedure id shown already.

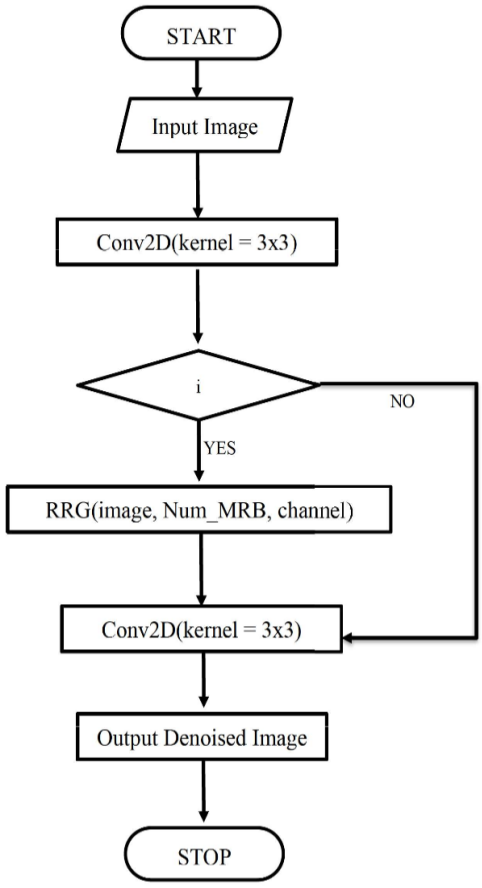


Figure 4.4: MiRNet

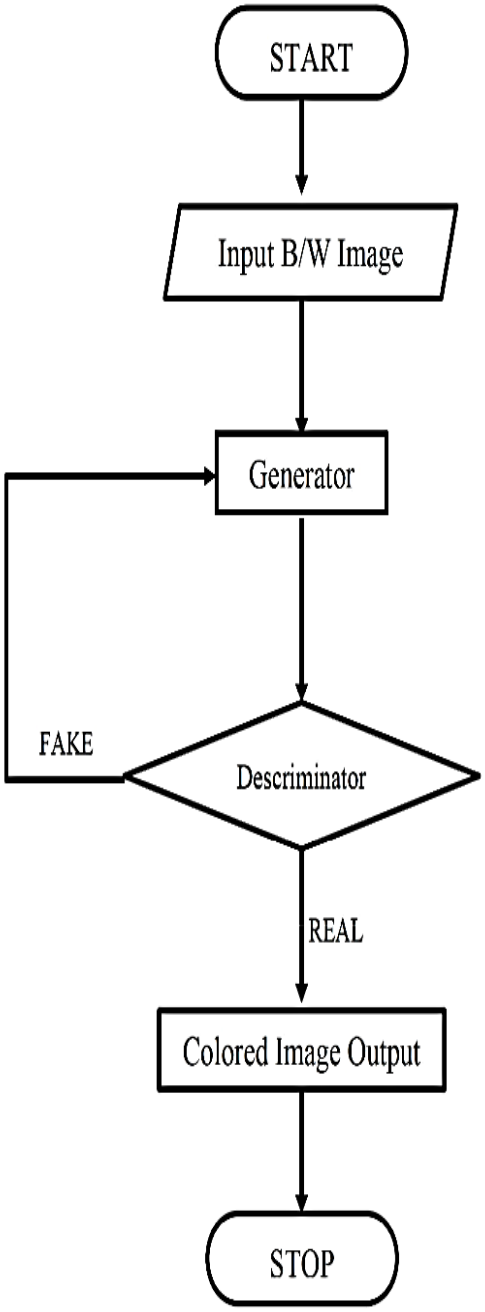


Figure 4.5: No-GAN

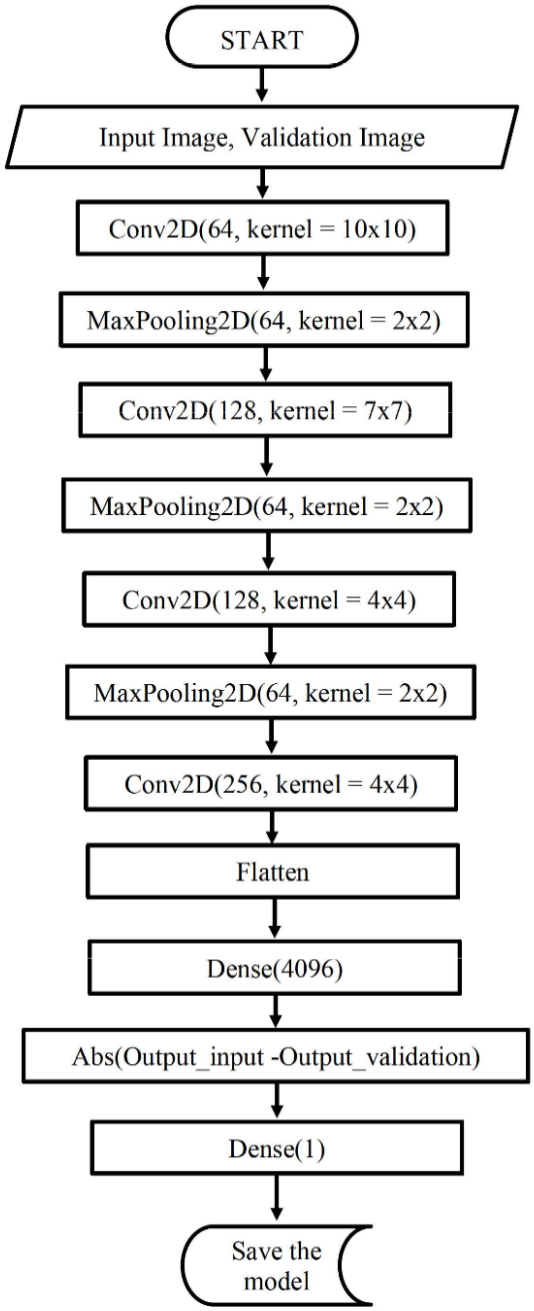


Figure 4.6: Convolutions Siamese Training

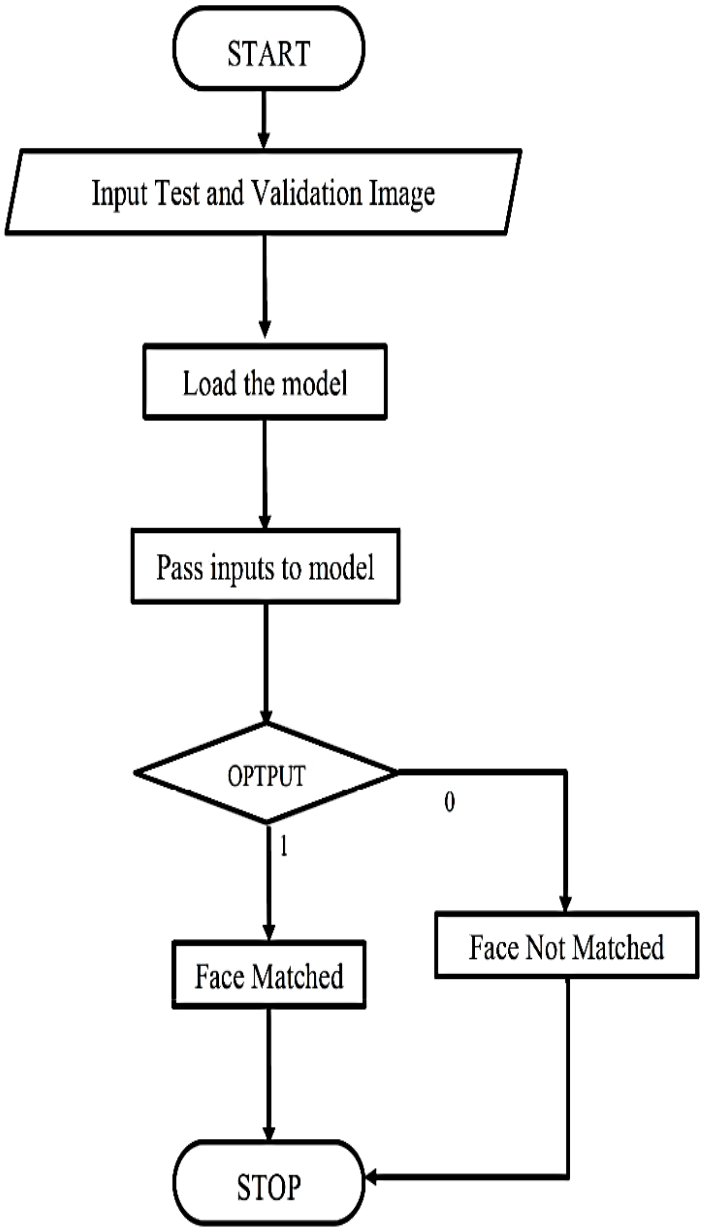


Figure 4.7: Convolutions Siamese Testing

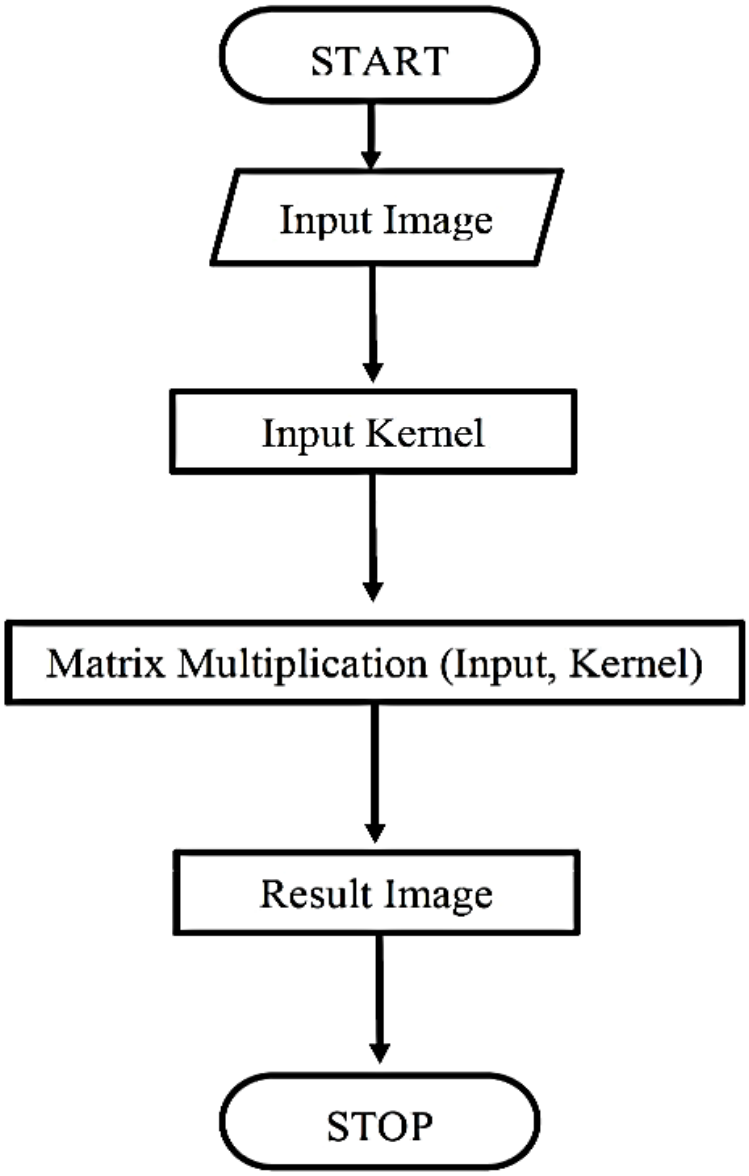


Figure 4.8: Manual Operations

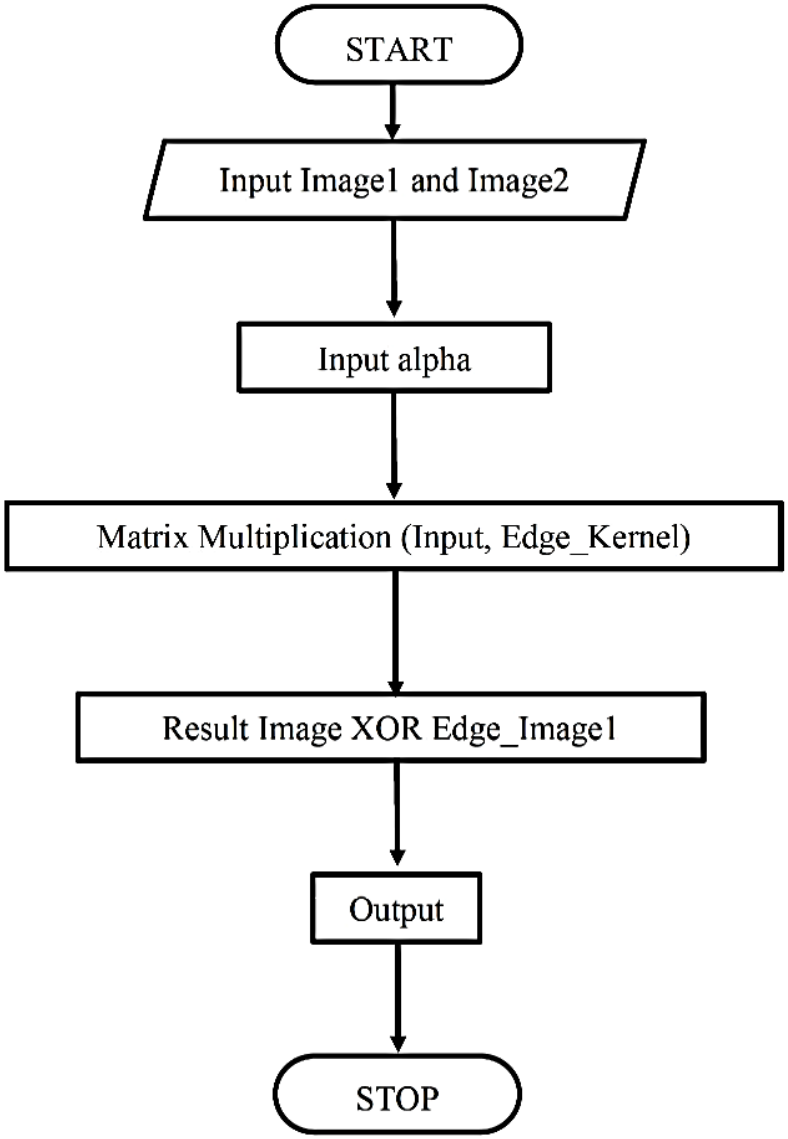


Figure 4.9: Missing/Misplaced objects

4.5 Database Schema

CODE

```
from enum import unique
from django.db import models
class image base(models.Model):
   contact = models.IntegerField(primary key=True, null=False)
   image = models.FileField(null=True)
   def str(self):
      return __str__(self.contact)
```

CONTACT	IMAGE
7398688462	Ang.jpg
6386550905	toon.jpeg
7880348563	file.png

Table 4.1: Database

Chapter 5

Implementation

5.1 Introduction

As the technology around us is evolving rapidly day by day, it is bringing major changes to us and our surroundings, and one of the fields is "Image Technology". Few decades ago, people were able to capture their moments, then few years latter they advanced to futuristic features like using images for security purposes, navigation, and entertainment. Now, with the collaboration of artificial intelligence, it has been so advanced that it can paint a person as how has he been looked like in past centuries to how will he look after few decades. Here we have knowledge available about Image Processing and Artificial Intelligence. So, we want to utilize this knowledge to assist the forensics operations. We are about to design a software which will assist the forensics operation like objects detection, a face matching system, upscaling, denoising, smoothing, coloring, and sharpening images to study a image in details, so that a forensic expert can infer the knowledge easily with speed and accuracy. This system will be very helpful to forensic experts.

5.2 Description of Developed System

This System uses the various concepts of Image Processing and Machine Learning to give users a much accurate, satisfying and deciding results. The context of the system discussed here is the digital image forensics, as it takes images as inputs, and provides various tools to manipulate

the image data in order to gather some meaningful information or to make the data more meaningful.

Rapidly emerging technology is changing our surrounding and lifestyle at very fast rate. As few years ago, technology was supposed to perform tasks leading to the future, but now a days, it's capable to perform task of the future, as well as the past. For example, earlier, software's were able to clear the face in a passport size photo, then they evolved to portrait a person few years ahead and back. Now they are advanced in such a way that they can portrait a person as he would has looked few centuries ago, and his kids may look like in the future. With the capability of machines to learn automatically with the help of Machine Learning Technology, much complex tasks can be performed with ease. As the advancement of the technology has been found misused at certain places, it is necessary to get control over it, which is nearly impossible, but validation of such thing is in our hands. So, it is necessary to have a system that can validate anything that has been passed through any technology. It motivated us to design a system that can validate a major participating thing present widely around us, we decided to design a system which can assist human being with the validation process of an image, and other things to assist the forensic operation.

5.3 Technical Details of Implemented System

When someone develop a project with full efforts and outcome fulfill expectations, this moment gives very beautiful feeling of success. As our aim was to implement our study in the field of Machine Learning and Computer Graphics Animation, we planned the environment for our project. At first it seemed impossible to gather this much powerful ecosystem for the development of our project, we thoroughly searched for alternate options available to fulfill out vision of developing a system which gives extraordinary results automatically without any human interference.

So, after thorough investigation, we got our best option, Google Colab to perform sophisticated operations like training a neural network, creating a trained model. Then we choose to perform predictions and results on offline device. This is how the training and development part is done.

Now to use these modules, we had two options, first one was Desktop GUI and Second one was Web Application. We selected web application for our project because of it's advantages over Desktop applications. **Advantages of Web Application over Desktop that made us to select GUI using Web**

(1) Web Applications are more platform independent then Desktop Applications

(2) Designing GUI elements is very easy in Web

(3) It is easy to scale a Web Application

(4) Web Applications can be highly interactive and beautiful

(5) Web Applications provide more stability

5.4 Screenshots of Developed System

Figure 5.1: Home Page

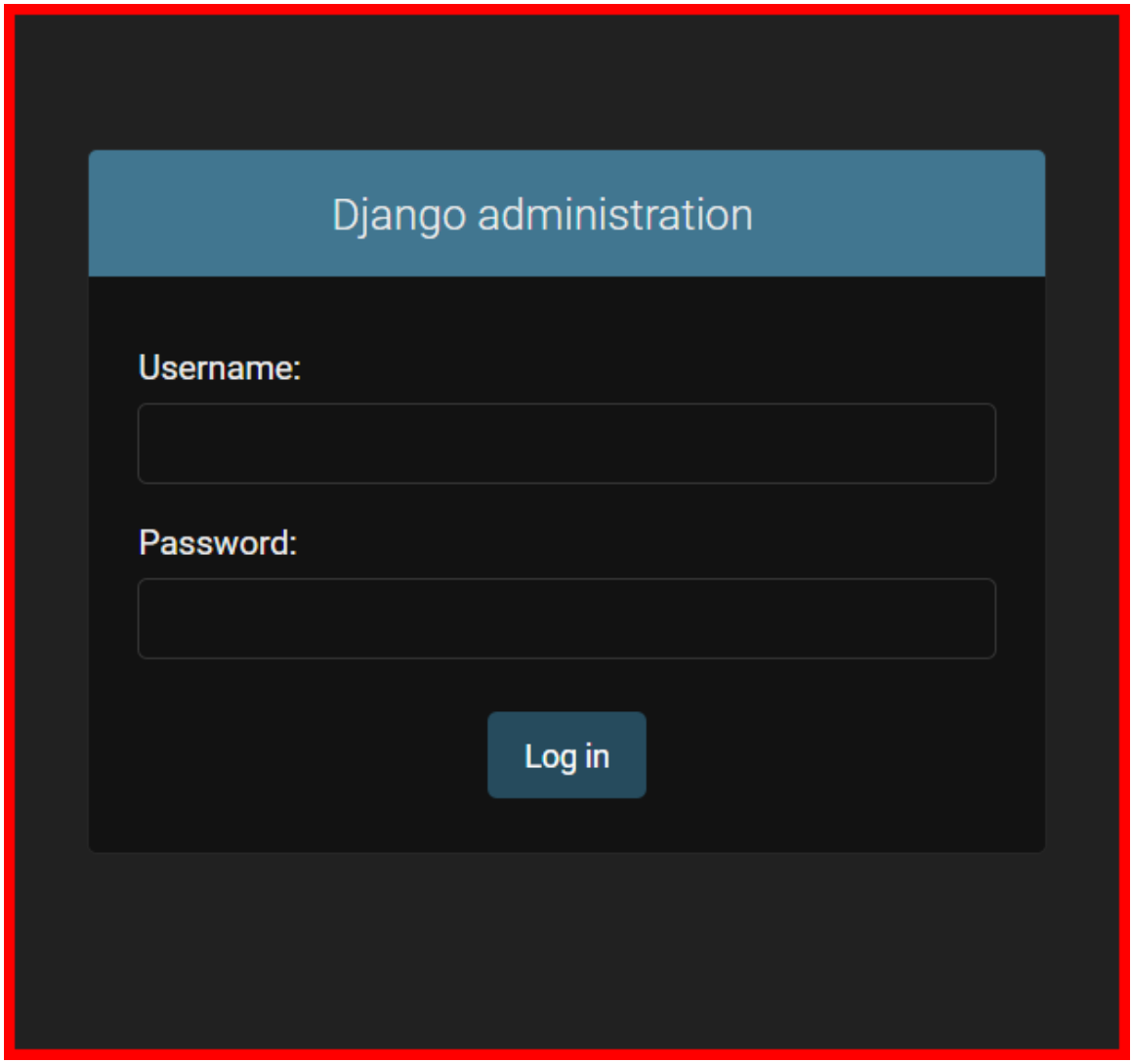


Figure 5.2: Authentication page

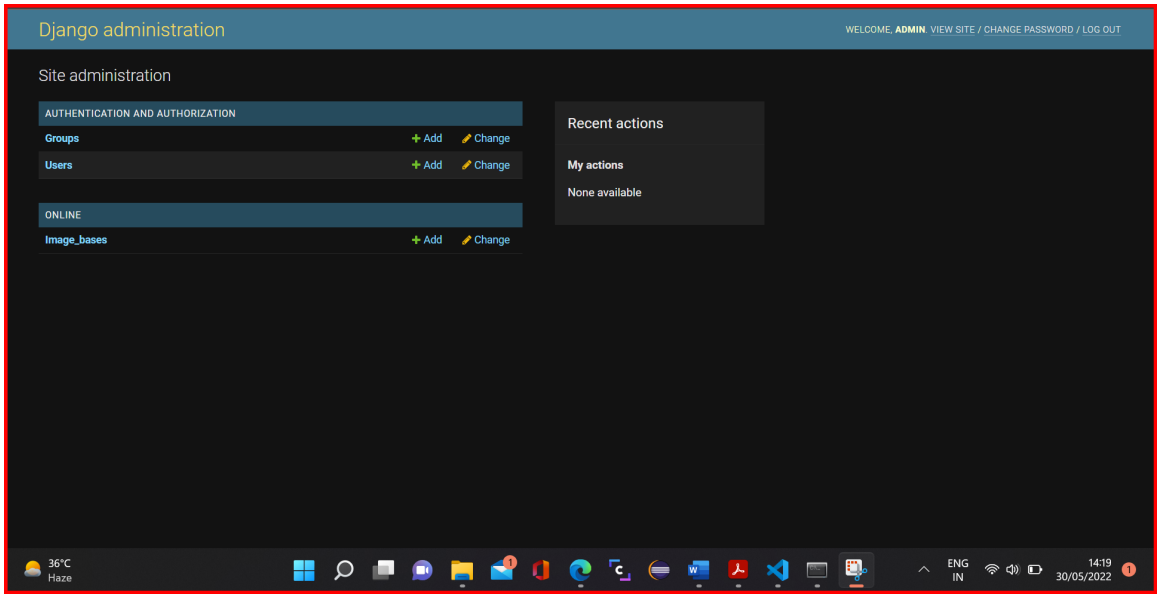


Figure 5.3: Admin page

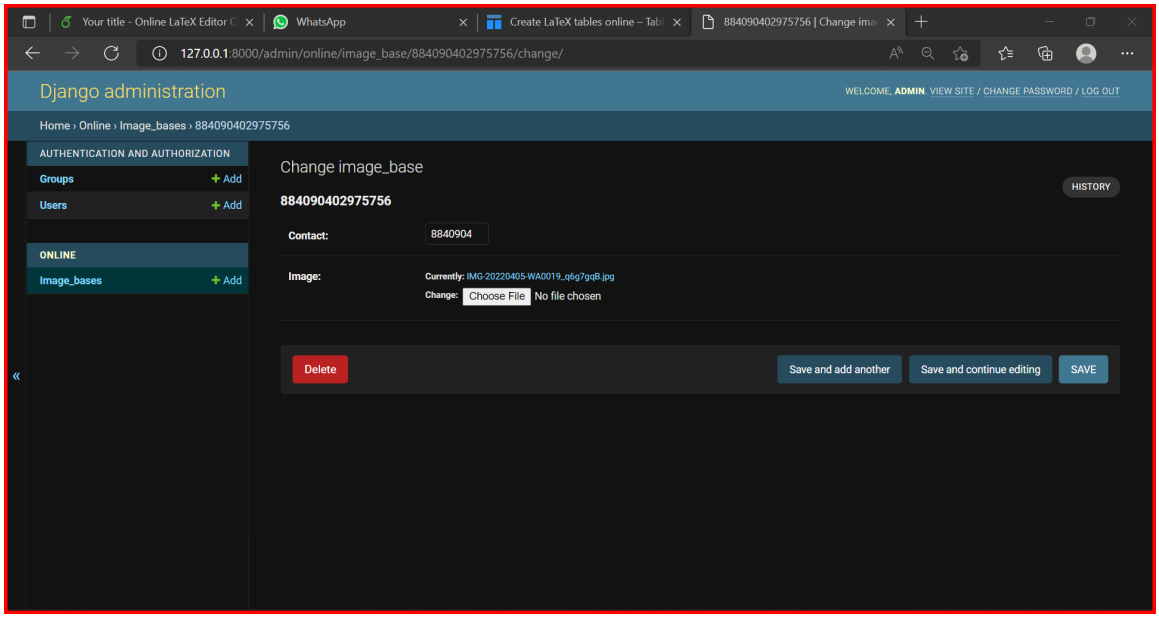


Figure 5.4: Image Base

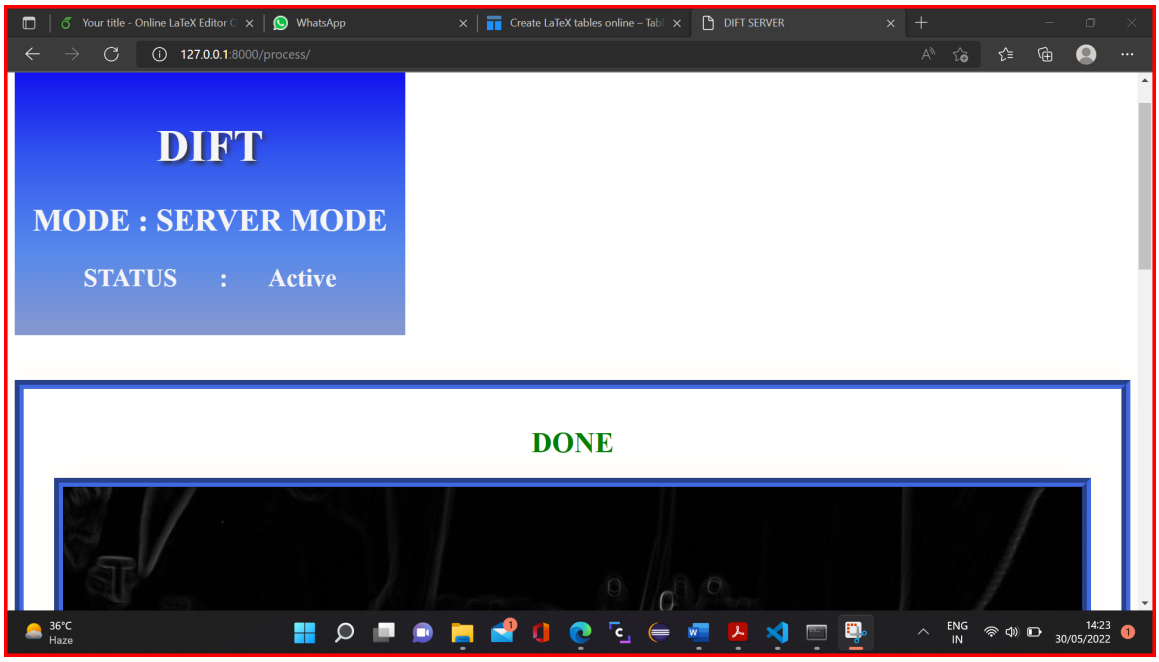


Figure 5.5: Result [TOP]

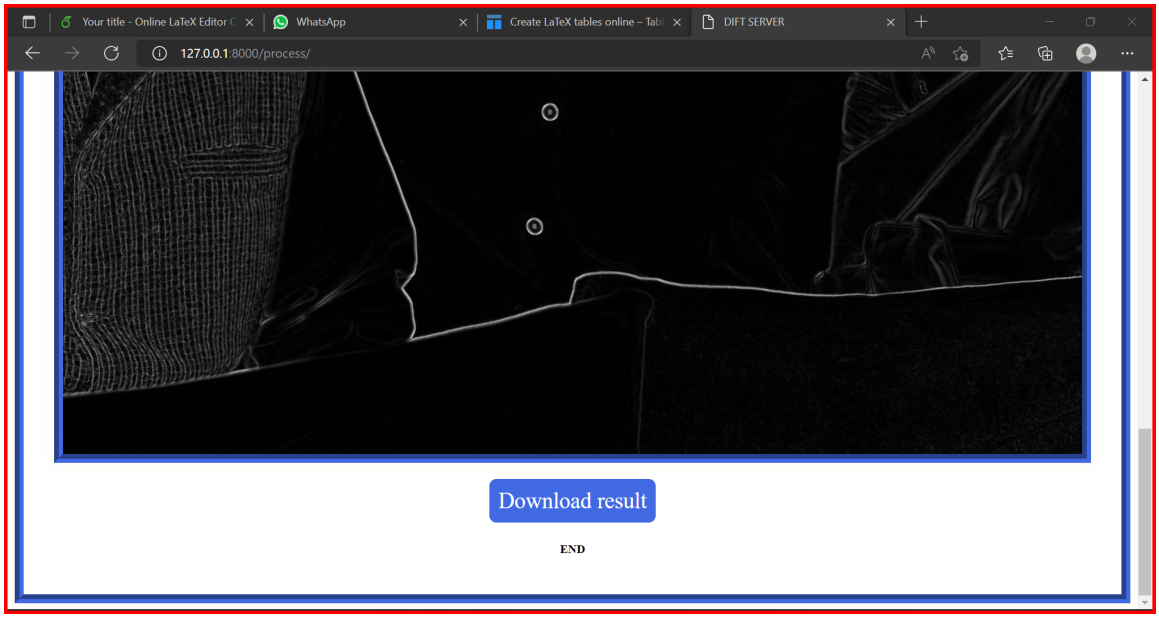


Figure 5.6: Result [BOTTOM]

Figure 5.7: Rocolouring Result

Figure 5.8: High Resolution Result

5.5 Screenshots of Developed System code

5.5.1 Code for Server Management

```
from http.client import HTTPResponse
from django.shortcuts import render
from django.http.response import HttpResponseRedirect
import sys
import cv2 as cv
from online.models import image_base
import os
def index(request):
    return render(request, 'index.html')
# Create your views here.
def process(request):
    if request.method == 'POST':
        ph_no = request.POST['phone']
        img_data = request.FILES['photo']
        try:
            image_base.objects.create(image=img_data, contact=ph_no)
            data = image_base.objects.get(contact=ph_no)
            sobel(data)
            return render(request, 'done.html',{"data":data})
        except:
            return render(request, 'failed.html')
```

5.5.2 Code for upscaling

```
import os.path as osp
import glob
import cv2
import numpy as np
import torch
import RRDBNet_arch as arch
```

```
7
8  model_path = 'models/RRDB_ESRGAN_x4.pth'  # models/RRDB_ESRGAN_x4.pth OR models/
       RRDB_PSNR_x4.pth
9  #device = torch.device('cuda')  # if you want to run on CPU, change 'cuda' -> cpu
10 device = torch.device('cpu')
11
12 test_img_folder = 'LR/*'
13
14 model = arch.RRDBNet(3, 3, 64, 23, gc=32)
15 model.load_state_dict(torch.load(model_path), strict=True)
16 model.eval()
17 model = model.to(device)
18
19 print('Model path {:s}. \nTesting...'.format(model_path))
20
21 idx = 0
22 for path in glob.glob(test_img_folder):
23     idx += 1
24     base = osp.splitext(osp.basename(path))[0]
25     print(idx, base)
26     # read images
27     img = cv2.imread(path, cv2.IMREAD_COLOR)
28     img = img * 1.0 / 255
29     img = torch.from_numpy(np.transpose(img[:, :, [2, 1, 0]], (2, 0, 1))).float()
30     img_LR = img.unsqueeze(0)
31     img_LR = img_LR.to(device)
32
33     with torch.no_grad():
34         output = model(img_LR).data.squeeze().float().cpu().clamp_(0, 1).numpy()
35     output = np.transpose(output[[2, 1, 0], :, :], (1, 2, 0))
36     output = (output * 255.0).round()
37     cv2.imwrite('results/{:s}_rlt.png'.format(base), output)
```

5.5.3 Code for removing Low light

```
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 31 16:32:22 2022

@author: K K's NOTEBOOK
"""

from tensorflow.keras.models import load_model
from PIL import Image
import numpy as np
import keras


low_light = load_model('low_light.h5')

llim = Image.open('im.jpg').convert('RGB')

llim = llim.resize((256,256),Image.NEAREST)

image = keras.preprocessing.image.img_to_array(llim)

image = image.astype('float32') / 255.0

image = np.expand_dims(image, axis = 0)

output = low_light.predict(image)

output_image = output[0] * 255.0

output_image = output_image.clip(0,255)

```

```
output_image = output_image.reshape((np.shape(output_image)[0],np.shape(
    output_image)[1],3))

output_image = np.uint32(output_image)

res = Image.fromarray(output_image.astype('uint8'),'RGB')

res.show()
```

5.5.4 Code for denoising image

```
# -*- coding: utf-8 -*-
"""
Created on Sat Apr  9 01:52:41 2022

@author: K K's NOTEBOOK
"""

# importing libraries
import numpy as np
import cv2
from matplotlib import pyplot as plt

p1 = 1
p2 = 1
p3 = 5
p4 = 13

# Reading image from folder where it is stored
img_1 = cv2.imread('inputs/i/2.png')
img_2 = cv2.imread('inputs/i/3.png')
img_3 = cv2.imread('inputs/i/4.png')
for i in range(0,5):
```

```
    dst= cv2.fastNlMeansDenoisingColored(img_1, None, p1, p2, p3, p4)
    cv2.imwrite("outputs/low/ {} {} {} {} .png".format(p1,p2,p3,p4), dst)
    p1+=3
    dst= cv2.fastNlMeansDenoisingColored(img_1, None, p1, p2, p3, p4)
    cv2.imwrite("outputs/low/ {} {} {} {} .png".format(p1,p2,p3,p4), dst)
    p2+=3
    dst= cv2.fastNlMeansDenoisingColored(img_1, None, p1, p2, p3, p4)
    cv2.imwrite("outputs/low/ {} {} {} {} .png".format(p1,p2,p3,p4), dst)
    p3+=2
    dst= cv2.fastNlMeansDenoisingColored(img_1, None, p1, p2, p3, p4)
    cv2.imwrite("outputs/low/ {} {} {} {} .png".format(p1,p2,p3,p4), dst)
    p4+=4
    dst= cv2.fastNlMeansDenoisingColored(img_1, None, p1, p2, p3, p4)
    cv2.imwrite("outputs/low/ {} {} {} {} .png".format(p1,p2,p3,p4), dst)
# denoising of image saving it into dst image'''
dst_1= cv2.fastNlMeansDenoisingColored(img_1, None, 10, 10, 7, 15)
dst_2 = cv2.fastNlMeansDenoisingColored(img_2, None, 10, 10, 7,15)
dst_3 = cv2.fastNlMeansDenoisingColored(img_3, None, 10, 10, 7,15)

# Plotting of source and destination image
cv2.imwrite("outputs/2.png", dst_1)
cv2.imwrite("outputs/3.png", dst_2)
cv2.imwrite("outputs/4.png", dst_3)
```

5.5.5 Code of Siamese Architecture

```
inp = Input(shape=(100,100,3), name='input_image')

c1 = Conv2D(64, (10,10), activation='relu')(inp)
m1 = MaxPooling2D(64, (2,2), padding='same')(c1)

c2 = Conv2D(128, (7,7), activation='relu')(m1)
m2 = MaxPooling2D(64, (2,2), padding='same')(c2)
```

```
8
9  c3 = Conv2D(128, (4,4), activation = 'relu')(m2)
10 m3 = MaxPooling2D(64, (2,2), padding='same')(c3)
11
12 c4 = Conv2D(256, (4,4), activation='relu')(m3)
13 f1 = Flatten()(c4)
14 d1 = Dense(4096, activation='sigmoid')(f1)
15 model = Model(inputs=[inp], outputs=[d1], name='embedding')
```

Chapter 6

Testing and Validation

6.1 Introduction

Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.

6.2 Testing

6.2.1 Acceptance testing:

Verifying whether the whole system works as intended.And it passed the acceptance testing. We have developed this project on windows,and found that in order to port the software to Mac OS or Linux, it need few modifications, everything else is accepted.

6.2.2 Integration testing:

Ensuring that software components or functions operate together, and initially we faced few serious issues which were later removed.

6.2.3 Unit testing:

Validating that each software unit performs as expected. A unit is the smallest testable component of an application. It took very very long time, and also so many errors, which we removed by studying the functions in details.

6.2.4 Performance testing:

Testing how the software performs under different workloads. Load testing, for example, is used to evaluate performance under real-life load conditions. And we found that the resources specification decides the performance of our Software.

6.2.5 Regression testing:

Checking whether new features break or degrade functionality. Sanity testing can be used to verify menus, functions and commands at the surface level, when there is no time for a full regression test. At first, new features broke the pipeline, but after few changes in code, it passed regression testing.

6.2.6 Stress testing:

Testing how much strain the system can take before it fails. Considered to be a type of non-functional testing. And we found that normal laptops and desktops are not able to able run this software.

6.2.7 Usability testing:

Validating how well a customer can use a system or web application to complete a task. It was short and quick, because we put our best effort in designing the User Interface, we kept it simple that is why people who tested, found it clean and simple.

MODULES	TEST CASES	TOTAL PASSED
DATABASE	24	21
DENOISING	41	41
FACE MATCHING	7	6
HIGH RESOLUTION	9	5
IMAGE SHARPENING	27	27
IMAGE DATABASE	33	31
KERNEL MODULE	4	4
REMOTE I/O	81	72
SERVER	103	102

Table 6.1: Testing details

6.3 Result validation

OBSERVATION TABLE : FACE RECOGNITION

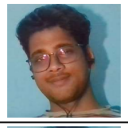
IMAGE 1	IMAGE 2	OUTPUT
		1
		0
		1
		1
		0
		0

Figure 6.1: Facematching

[10]

OBSERVATION TABLE : UPSCALING

INPUT IMAGES	OUTPUT IMAGES
	
	

Figure 6.2: High Resolution 1

Figure 6.3: High Resolution 2

[1]

OBSERVATION TABLE : LOW LIGHT IMAGE CLEARING USING MIRNet

INPUT IMAGES	OUTPUT IMAGES
	
	
	

Figure 6.4: Low light removal

[3]

OBSERVATION TABLE : RE-COLORING BLACK AND WHITE IMAGE

INPUT IMAGE 	**RENDER FACTOR = 1** 
RENDER FACTOR = 10 	**RENDER FACTOR = 20** 

Figure 6.5: Recolouring 1

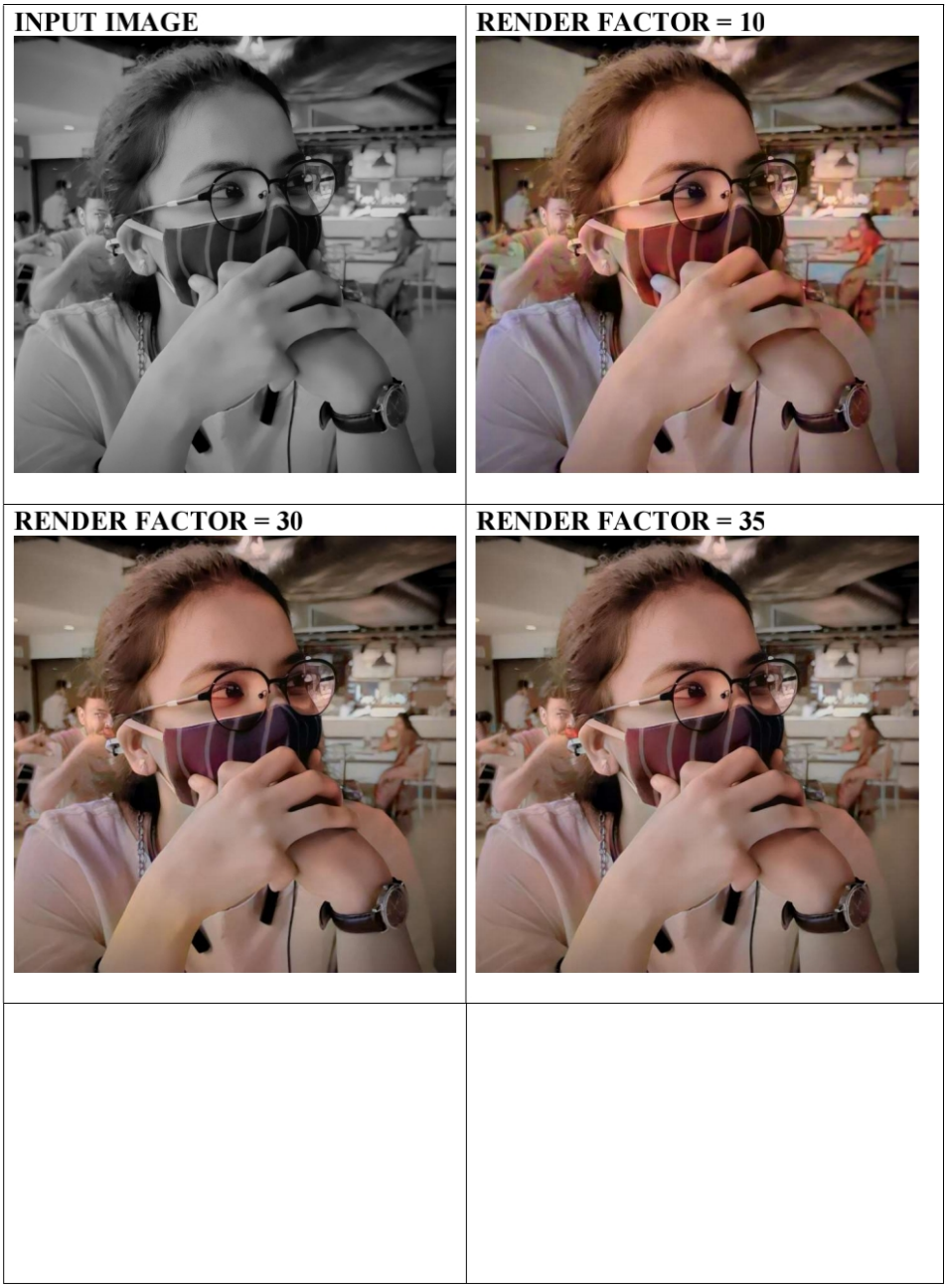


Figure 6.6: Recolouring 2

[4]

Figure 6.7: Recolouring 3

OBSERVATION TABLE : DENOISING IMAGE USING FAST NON LOCAL MEAN METHOD

INPUT IMAGE

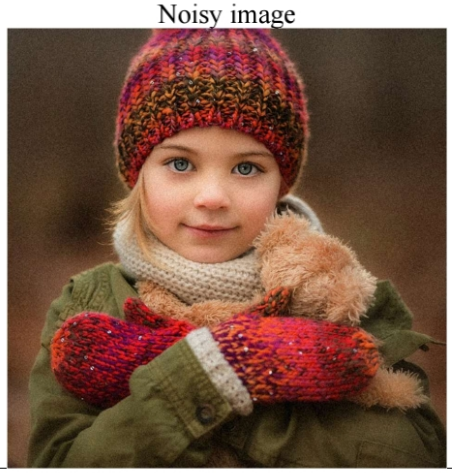
Noisy image 	1-1-5-13 
4-4-5-13 	7-7-9-21 

Figure 6.8: Denoising 1

EFFECT OF COLOUR FACTOR @ 10-10-7-XX [XX=COLOR FACTOR]

At colour luminance factor = 5, image is still noisy, at 15, it's good but very few details are lost, and at < 25, there is major detail loss in the image.

Figure 6.9: Denoising 2

Noisy

Filtered 10-10-9-21

NOTE : In the above name format **XX-XX-XX-XX,** The first number represents the patch size to compute weights, second for window size to compute weighted average for given pixel, third for luminance strength and fourth for colour luminance.

OBSERVATION : The **Fast Non Local Mean** method to denoise image doesn't show effect up to 4-4-5-13, gives better result in between 10-7-19-21 to 13-13-11-13. And after 13-13-11-13 it starts loosing the details of the image at high rate.

Figure 6.10: Denoising 3

[5]

EFFECT OF STRENGTH FACTOR FOR LUMEN COMPONENT

@10-10-XX-15 [XX = STRENGTH FACTOR]

OBSERVATION : From above table, it is clear that details are well preserved at strength factor = 3, but contains noise too, and at strength factor after 7 and higher, details are compromised but noise is highly removed.

Figure 6.11: Denoising 4

So, best parameters for denoising an image using fast non local mean methods are :

Good Patch size to compute weight = 10

Good Window size to compute weighted average for given pixel = 10

Good Strength factor = 7

And good Luminance colour factor = 15

Now, testing some noisy images with parameters = **10-10-7-15**

INPUT IMAGES	CORRESPONDING OUTPUT
	

Figure 6.12: Denoising 5

[9]

6.4 Conclusion

Our developed system is giving extraordinary results, we worked very very hard on this project, and we achieved more than our vision. It's very beautiful experience.

Chapter 7

Discussions and conclusion

7.1 Contributions

We put a lot of effort in developing this project, it was very heavy project, has many advance modules which are not possible to develop on simple or even good machines available around us. So we took some helps from open source and some from experts to design and develop this software. We coded modules online and used Google's resources to compile sophisticated blocks like training and modeling. Later on, we saved trained models so that we can use it on offline machines.
We coded few modules like secret text detection, Remote I/O, Face Matching Artificial Intelligence from scratch and the whole server with so many new concepts and features.

7.2 Limitations

This is a heavy software so it have some limitations.

(1) Requires high processing power

(2) Requires more disk space

7.3 Future scope

This idea have very bright future, in future, it can be coded to perform operations automatically, analyse images automatically, for example, when an image is given to it, it will be able to find automatically that what needs to be done so that this image becomes perfect as expected.

Bibliography

[1] Generative Adversarial Nets, Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozairy, Aaron Courville, Yoshua Bengioz D´epartement d'informatique et de recherche op´erationnelle Universit´e de Montr´eal Montr´eal, QC H3C 3J7, arXiv:1406.2661v1 [stat.ML] 10 Jun 2014.

[2] Learning Enriched Features for Real Image Restoration and Enhancement, Syed Waqas Zamir1, Aditya Arora1, Salman Khan1;2, Munawar Hayat1;2, Fahad Shahbaz Khan1;2, Ming-Hsuan Yang3;4, and Ling Shao1;2 1 Inception Institute of Arti

cial Intelligence, UAE 2 Mohamed bin Zayed University of Arti

cial Intelligence, UAE 3 University of California, Merced, USA 4 Google Research

[3] Densely connected normalizing flows, Matej Grci´c, Ivan Grubiši´c and Siniša Šegvi´c Faculty of Electrical Engineering and Computing University of Zagreb matej.grcic@fer.hr ivan.grubisic@fer.hr sinisa.segvic@fer.hr

[4] BIAS IN AUTOMATED IMAGE COLORIZATION: METRICS AND ERROR TYPES Frank Stapel Floris Weers Doina Bucur University of Twente, arXiv:2202.08143v1 [cs.CV] 16 Feb 2022

[5] PSEUDO NUMERICAL METHODS FOR DIFFUSION MODELS ON MANIFOLDS Luping Liu, Yi Ren, Zhijie Lin , Zhou Zhao Zhejiang University fluping.liu,rayeren,linzhijie,zhaozhoug@zju.edu.cn, arXiv:2202.09778v1 [cs.CV] 20 Feb 2022

[6] Sobel Edge Detection Based on Weighted Nuclear Norm Minimization Image Denoising Run Tian , Guiling Sun *, Xiaochao Liu and Bowen Zheng, College of Electronic Information and Optical Engineering, Nankai University, Tianjin 300350, China; tianr9616@163.com (R.T.); liuxiaochao3@163.com (X.L.); zhengbwen@mail.nankai.edu.cn (B.Z.) Feb 2022

[7] Tian, R.; Sun, G.; Liu, X.; Zheng, B. Sobel Edge Detection Based onWeighted Nuclear Norm Minimization Image Denoising. Electronics 2021

[8] Fast adaptive and selective mean filter for the removal of high density salt and paper noise. Samsad Beagum Sheik Fareed, Sheeja Shaik Khader, IET Journals, ISSN 1751-9659 year 2018

[9] Non Local Means Denoising, Antoni Buades[1], Bartomeu Coll[2], Jean-Michel Morel3, 1CNRS-Paris Descartes, France, [2] Universitat Illes Balears, Spain, [3] CMLA, ENS Cachan, France, IPOL 2011-9-13

[10] Siamese Neural Networks for One-shot Image Recognition, Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, Department of Computer Science, University of Toronto. Toronto, Ontario, Canada. 2014

[11] Recorrupted-to-Recorrupted: Unsupervised Deep Learning for Image Denoising Tongyao Pang1, Huan Zheng1, Yuhui Quan2, and Hui Ji1 1Department of Mathematics, National University of Singapore, 119076, Singapore 2School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China 2021

[12] Sharpness measure: towards automatic image enhancement, Doron Shaked, Ingeborg Tastal, HP Inc., Conference Paper · October 2005 DOI: 10.1109/ICIP.2005.1529906 · Source: IEEE Xplore

[13] Image Edge Detection: A New Approach Based on Fuzzy Entropy and Fuzzy Divergence Mario Versaci1 ,Francesco Carlo Morabito1 Received: 14 January 2020 / Revised: 3 November 2020 / Accepted: 30 November 2020 / Published online: 6 February 2021 Taiwan Fuzzy Systems Association 2021