PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGITAL RECOGNITION

submitted by

TEAM ID : PNT2022TMID16152

BOOMIKA P	- 927619BIT4010
DHARANI S	- 927619BIT4015
JANANIPRIYA P V	- 927619BIT4040
KARTHIKA K	- 927619BIT4044

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 PROJECT OVERVIEW

The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes. The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image.

Handwritten Digit Recognition is the ability of computer systems to recognize handwritten digits from various sources, such as images, documents, and so on. This project aims to let users take advantage of machine learning to reduce manual tasks in recognizing digits.

1.2 PURPOSE

Digit recognition systems are capable of recognizing the digits from different sources like emails, bank cheque, papers, images, etc. and in different real-world scenarios for online handwriting recognition on computer tablets or system, recognize number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (tax forms) and so on.

CHAPTER 2
LITERATURE SURVEY

2.1 EXISTING PROBLEM

The fundamental problem with handwritten digit recognition is that handwritten digits do not always have the same size, width, orientation, and margins since they vary from person to person. Additionally, there would be issues with identifying the numbers because of similarities between numerals like 1 and 7, 5 and6, 3 and 8, 2 and 5, 2 and 7, etc. Finally, the individuality and variation of each individual's handwriting influence the structure and appearance of the digits.

2.2 REFERENCES

An Efficient And Improved Scheme For Handwritten Digit Recognition Based On Convolutional Neural Network (2019) - ***Ali, Saqib and Shaukat, Zeeshan and Azeem, Muhammad and Sakhawat, Zareen and Mahmood, Tariq and others***

This study uses rectified linear units (ReLU) activation and a convolutional neural network (CNN) that incorporates the Deeplearning4j (DL4J) architecture to recognize handwritten digits. The proposed CNN framework has all the necessary parameters for a high level of MNIST digit classification accuracy. The system's training takes into account the time factor as well. The system is also tested by altering the number of CNN layers for additional accuracy verification. It is importantto note that the CNN architecture consists of two convolutional layers, the first with 32 filters and a 5x5 window size and the second with 64 filters and a 7x7 window size. In comparison to earlier proposed systems, the experimental findings show that the proposed CNN architecture for the MNIST dataset demonstrates great performance in terms of time and accuracy. As a result, handwritten numbers are detected with a recognition rate of 99.89% and high precision (99.21%) in a short amount of time.

Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN) (2020) - ***Ahlawat, Savita and Choudhary, Amit and Nayyar, Anand and Singh, Saurabh and Yoon, Byungun***

This paper's primary goal was to enhance handwritten digit recognition ability. To avoid difficult pre-processing, expensive feature extraction, and a complex ensemble (classifier combination) method of a standard recognition system, they examined different convolutional neural network variations. Their current work makes suggestions on the function of several hyper-parameters through thorough evaluation utilizing an MNIST dataset. They also confirmed that optimizing hyper-parameters is crucial for enhancing CNN architecture performance. With the Adam optimizer for the MNIST database, they were able to surpass many previously published results with a recognition rate of 99.89%. Through the trials, it is made abundantly evident how the performance of handwritten digit recognition is affected by the number of convolutional layers in CNN architecture. According to the paper, evolutionary algorithms can be explored for optimizing convolutional filter kernel sizes, CNN learning parameters, and the quantity of layers and learning rates.

Improved Handwritten Digit Recognition Using Quantum K-Nearest Neighbor Algorithm (2019) - ***Wang, Yixian and Wang, Ruijin and Li, Dongfen and Adu–Gyamfi, Daniel and Tian, Kaibin& anZhu,Yixin***

The KNN classical machine learning technique is used in this research to enable quantum parallel computing and superposition. They used the KNN algorithm with quantum acceleration to enhance handwritten digit recognition. When dealing with more complicated and sizable handwritten digital data sets, their suggested method considerably lowered the computational time complexity of the traditional KNN algorithm. The paper offered a theoretical investigation of how quantum concepts can be applied to machine learning. Finally, they established a fundamental operational concept and procedure for machine learning with quantum acceleration

Handwritten Digit Recognition Using Machine And Deep Learning Algorithms(2021) - *Pashine, Samay and Dixit, Ritik and Kushwah, Rishika*

In this study, they developed three deep and machine learning-based models for handwritten digit recognition using MNIST datasets. To determine which model was the most accurate, they compared them based on their individual properties.Support vector machines are among the simplest classifiers, making them faster than other algorithms and providing the highest training accuracy rate in this situation. However, due to their simplicity, SVMs cannot categorize complicated andambiguous images as accurately as MLP and CNN algorithms can. In their research,they discovered that CNN produced the most precise outcomes for handwritten digit recognition. This led them to the conclusion that CNN is the most effective solution for all types of prediction issues, including those using picture data. Next, by comparing the execution times of the algorithms, they determined that increasing the number of epochs without changing the configuration of the algorithm is pointless due to the limitation of a certain model, and they discovered that beyond a certain number of epochs, the model begins over-fitting the dataset and provides biased predictions.

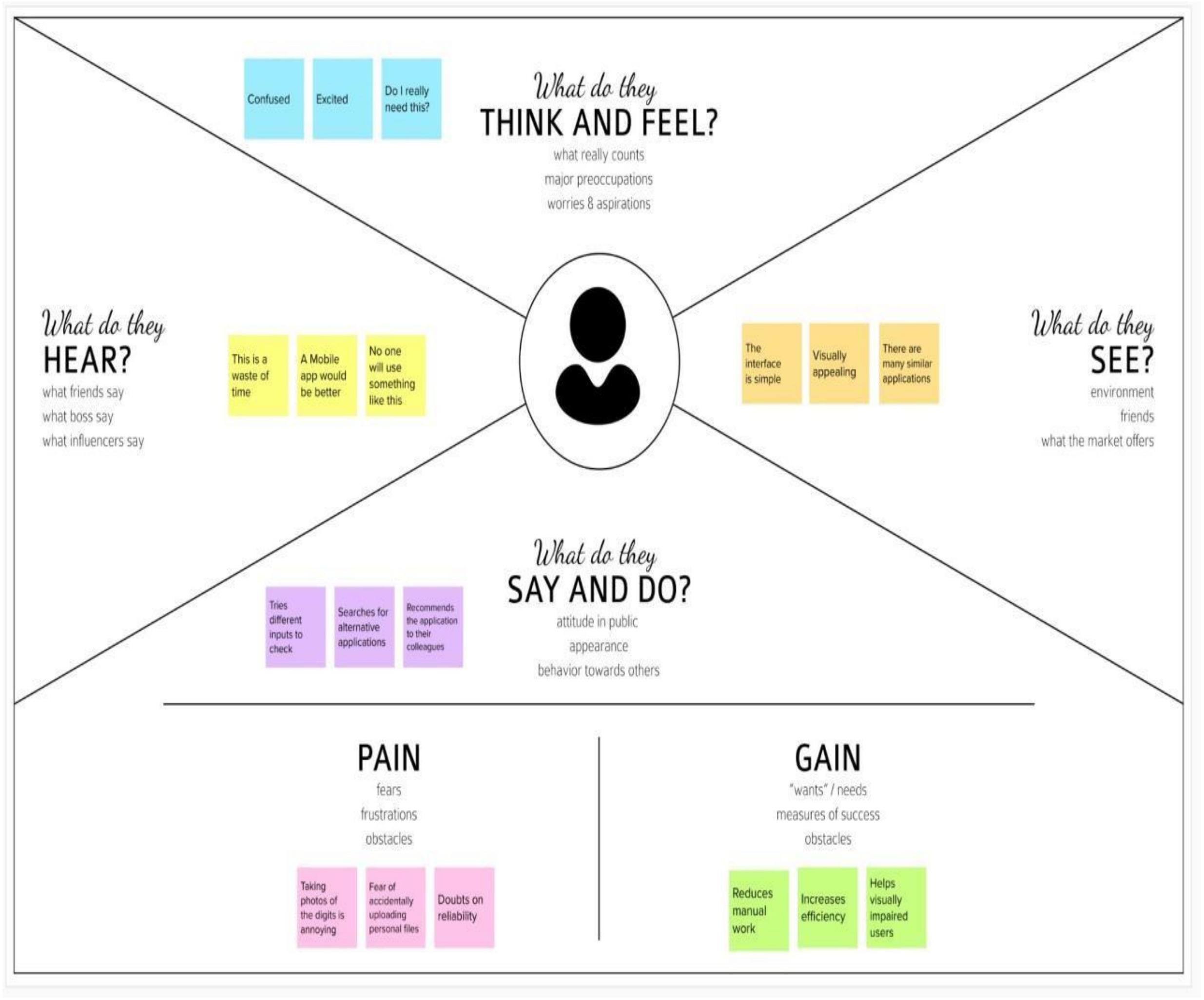
2.3 PROBLEM STATEMENT DEFINITION

The handwritten digits are not always of the same size, width, orientation and justified to margins as they differ from writing of person to person, so the general problem would while classifying the digits due to the similarity between digits such as 1 and 7, 5 and 6, 3 and 8,2 and5,2 and 7,etc. This problem is faced more when many people write a single digit with a variety of different handwritings. Lastly, the uniqueness and varity in the handwritingds of different individuals also influence the formation and appearance of the digits.

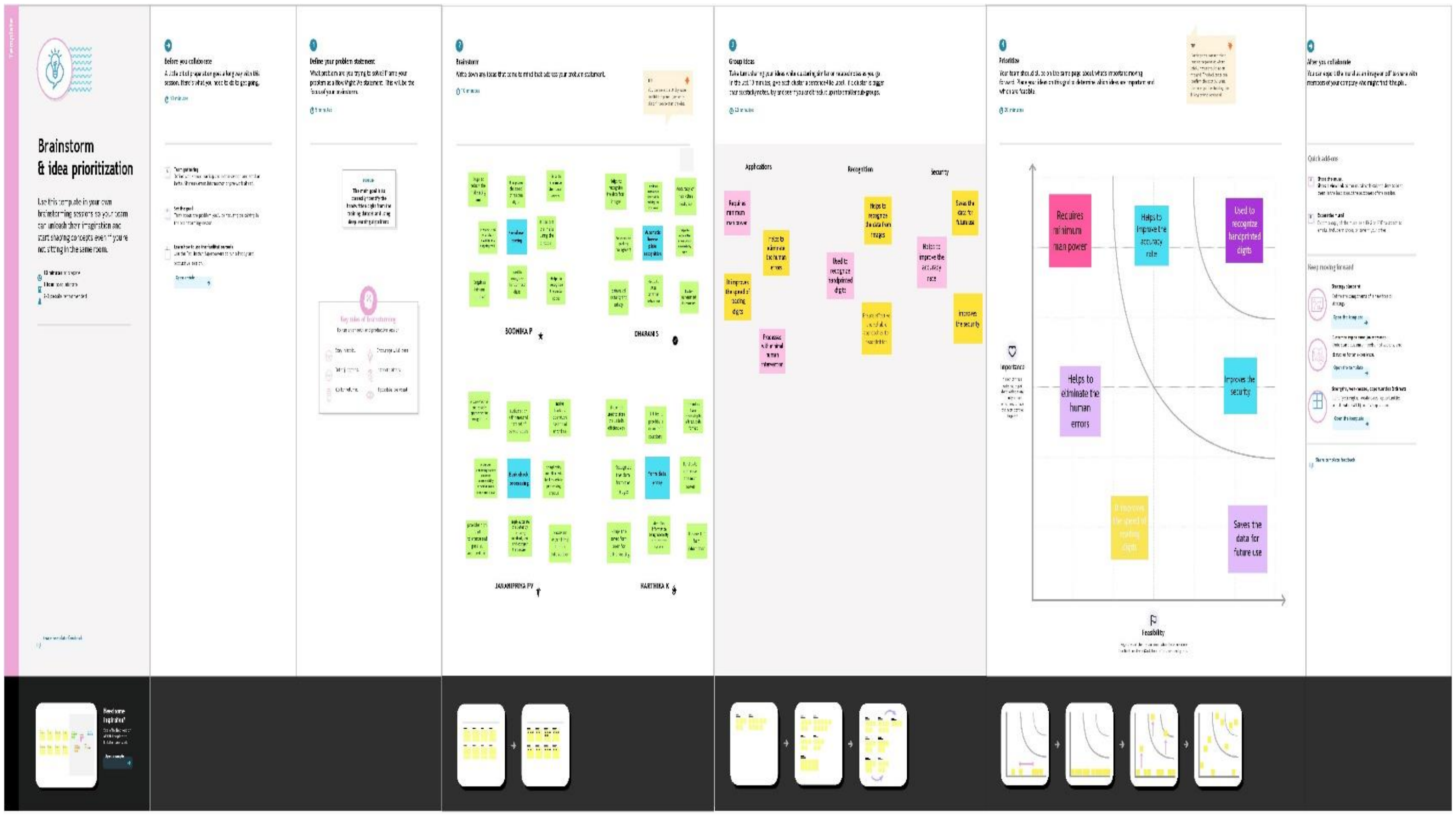
CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



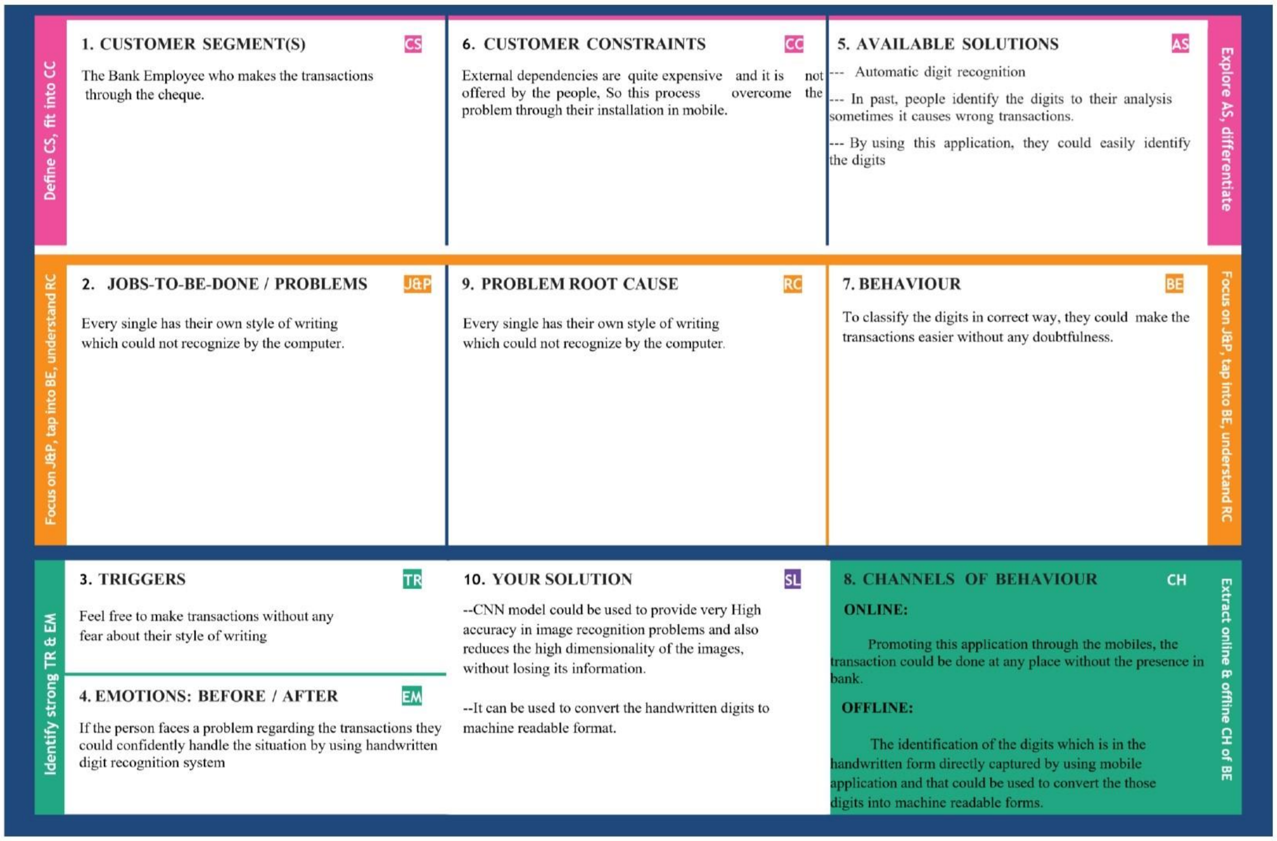
3.2 IDEATION & BRAINSTORMING


Brainstorm
& idea prioritization
Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.
Before you collaborate
Define your problem statement
Brainstorm
Group ideas
Prioritize
After you collaborate
Applications
Recognition
Security
Requires minimum man power
Helps to eliminate the human errors
It improves the speed of reading digits
Processes with minimal human intervention
Helps to recognize the data from images
Used to recognize handprinted digits
Helps to improve the accuracy rate
Saves the data for future use
Improves the security
Importance
Feasibility
DHARANI S
KARTHIKA K

3.3 PROPOSED SOLUTION

S.NO	PARAMETER	DESCRIPTION
1	Problem Statement	To create an application that recognizes handwritten digits
2	Idea / Solution Description	The application takes an imageas the input and accurately detects the digits in it.
3	Novelty / Uniqueness	Instead of recognizing every text, the application accurately recognizes only the digits
4	Social Impact / Customer Satisfaction	This application reduces the manual tasks that need to be performed. This improves productivity in the workplace.
5	Business Model	The application can be integrated with traffic surveillance cameras to recognize vehicle number plates The application can be integrated with Postal systemsto recognize the pin codes effectively
6	Scalability of the Solution	The application can easily be scaled to accept multiple inputs and process them parallelly to further increase efficiency

3.4 PROBLEM SOLUTION FIT



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

The Bank Employee who makes the transactions through the cheque.

6. CUSTOMER CONSTRAINTS CC

External dependencies are quite expensive and it is not offered by the people, So this process overcome the problem through their installation in mobile.

5. AVAILABLE SOLUTIONS AS

--- Automatic digit recognition

--- In past, people identify the digits to their analysis sometimes it causes wrong transactions.

--- By using this application, they could easily identify the digits

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Every single has their own style of writing which could not recognize by the computer.

9. PROBLEM ROOT CAUSE RC

Every single has their own style of writing which could not recognize by the computer.

7. BEHAVIOUR BE

To classify the digits in correct way, they could make the transactions easier without any doubtfulness.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

Feel free to make transactions without any fear about their style of writing

4. EMOTIONS: BEFORE / AFTER EM

If the person faces a problem regarding the transactions they could confidently handle the situation by using handwritten digit recognition system

10. YOUR SOLUTION SL

--CNN model could be used to provide very High accuracy in image recognition problems and also reduces the high dimensionality of the images, without losing its information.

--It can be used to convert the handwritten digits to machine readable format.

8. CHANNELS OF BEHAVIOUR CH

ONLINE:

Promoting this application through the mobiles, the transaction could be done at any place without the presence in bank.

OFFLINE:

The identification of the digits which is in the handwritten form directly captured by using mobile application and that could be used to convert the those digits into machine readable forms.

Extract online & offline CH of BE

CHAPTER 4
REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FR.NO	FUNCTIONAL REQUIREMENTS	SUB REQUIREMENTS
FR-1	Model Creation	Get access the MNIST dataset
		Analyze the dataset
		Define a CNN model
		Train and Test the Model
FR-2	Application Development	Create a website to let the user recognize handwritten digits.
		Create a home page to upload images
		Create a result page to display the results
		Host the website to let the users use it from anywhere
FR-3	Input Image Upload	Let users upload images of various formats.
		Let users upload images of various size
		Prevent users from uploading unsupported image formats
		Pre-Process the image to use it on the model

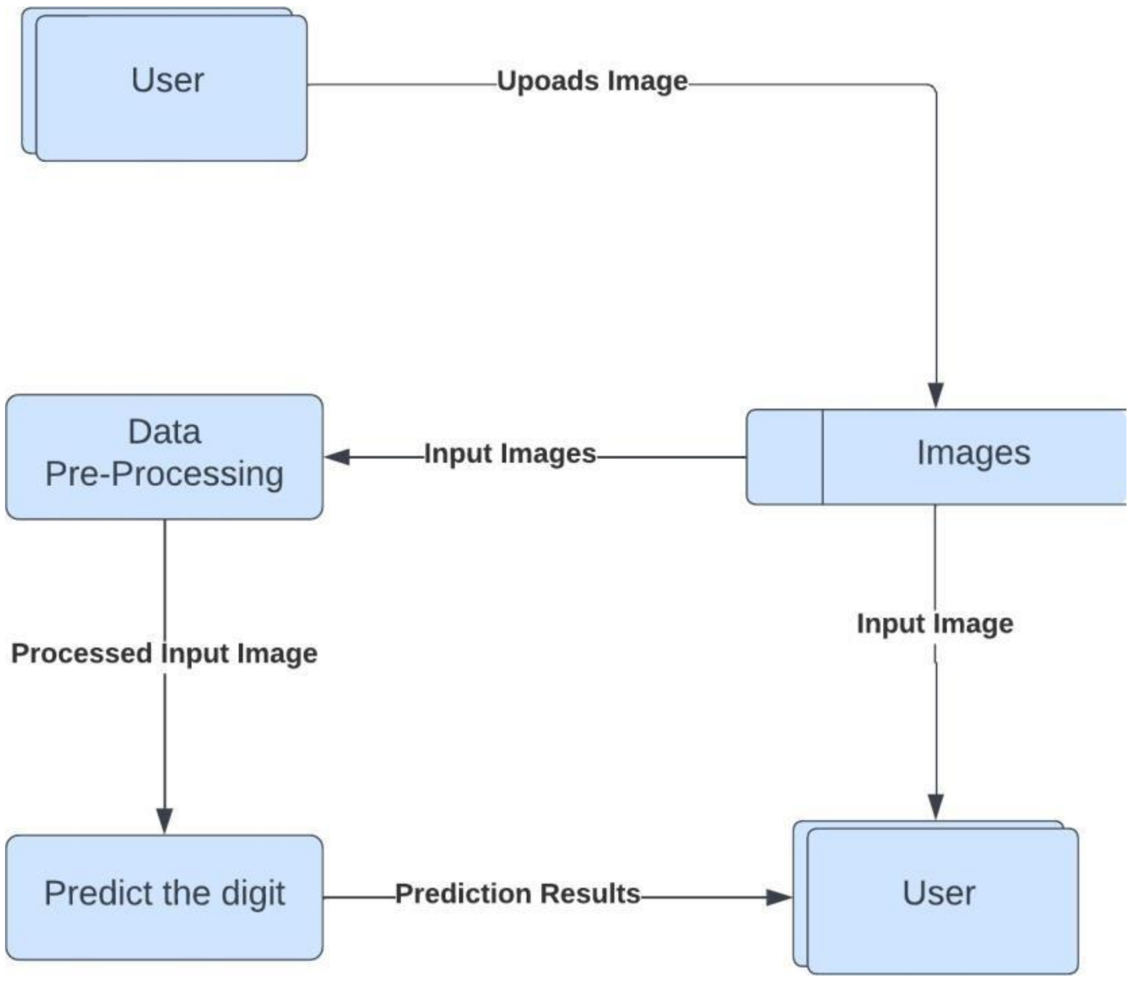
		Create a database to store all the input images
FR-4	Display Results	Display the result from the model
		Display input image
		Display accuracy the result
		Display other possible predictions with their respective accuracy

4.2 NON FUNCTIONAL REQUIREMENTS

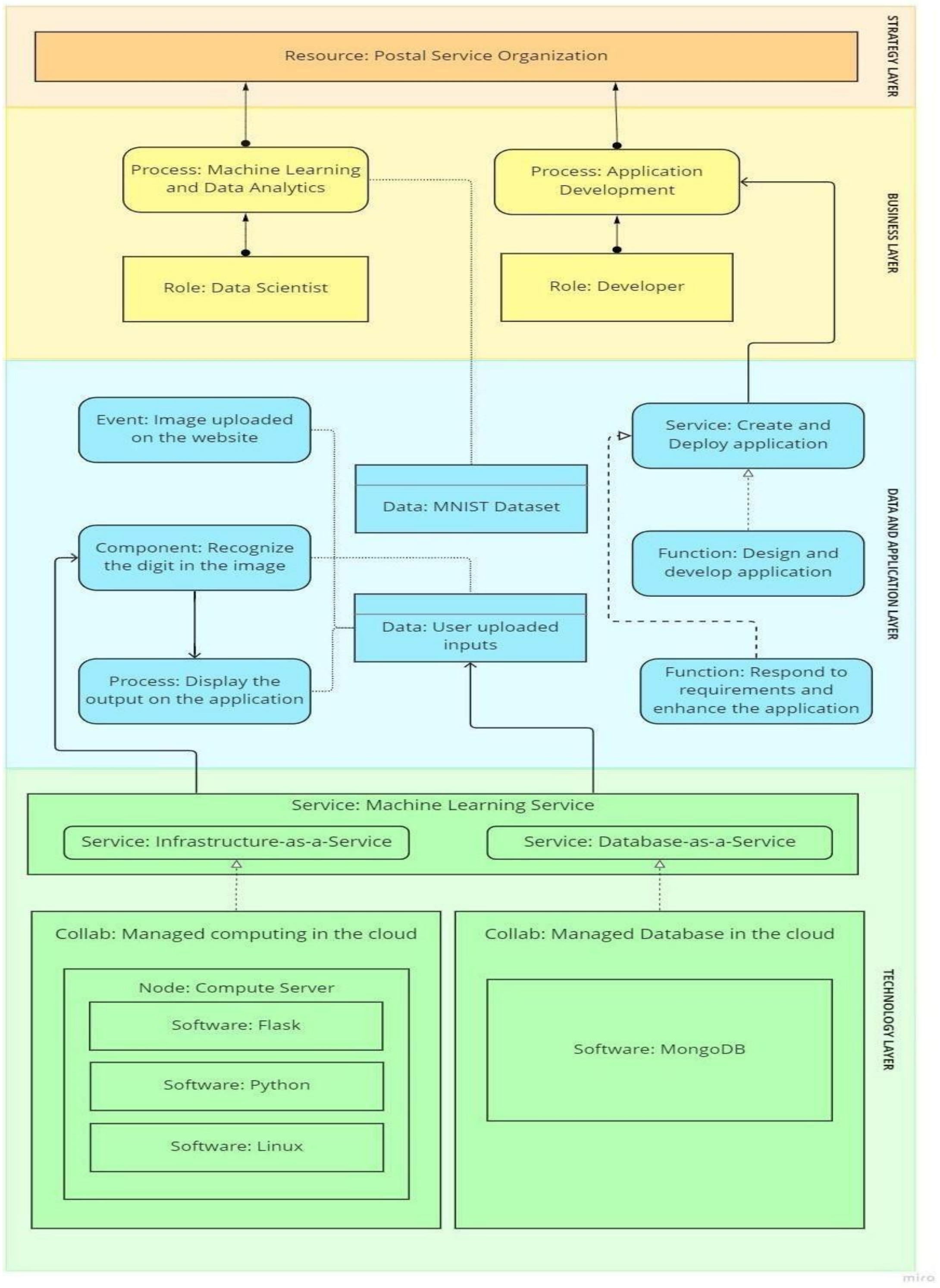
NFR	**NON-FUNCTIONAL REQUIREMENTS**	**DESCRIPTION**
NFR-1	Usability	The application must be usable in all devices
NFR-2	Security	The application must protect user uploaded image
NFR-3	Reliability	The application must give an accurate result as much as possible
NFR-4	Performance	The application must be fast and quick to load up
NFR-5	Availability	The application must be available to use all the time
NFR-6	Scalability	The application must scale along with the user base

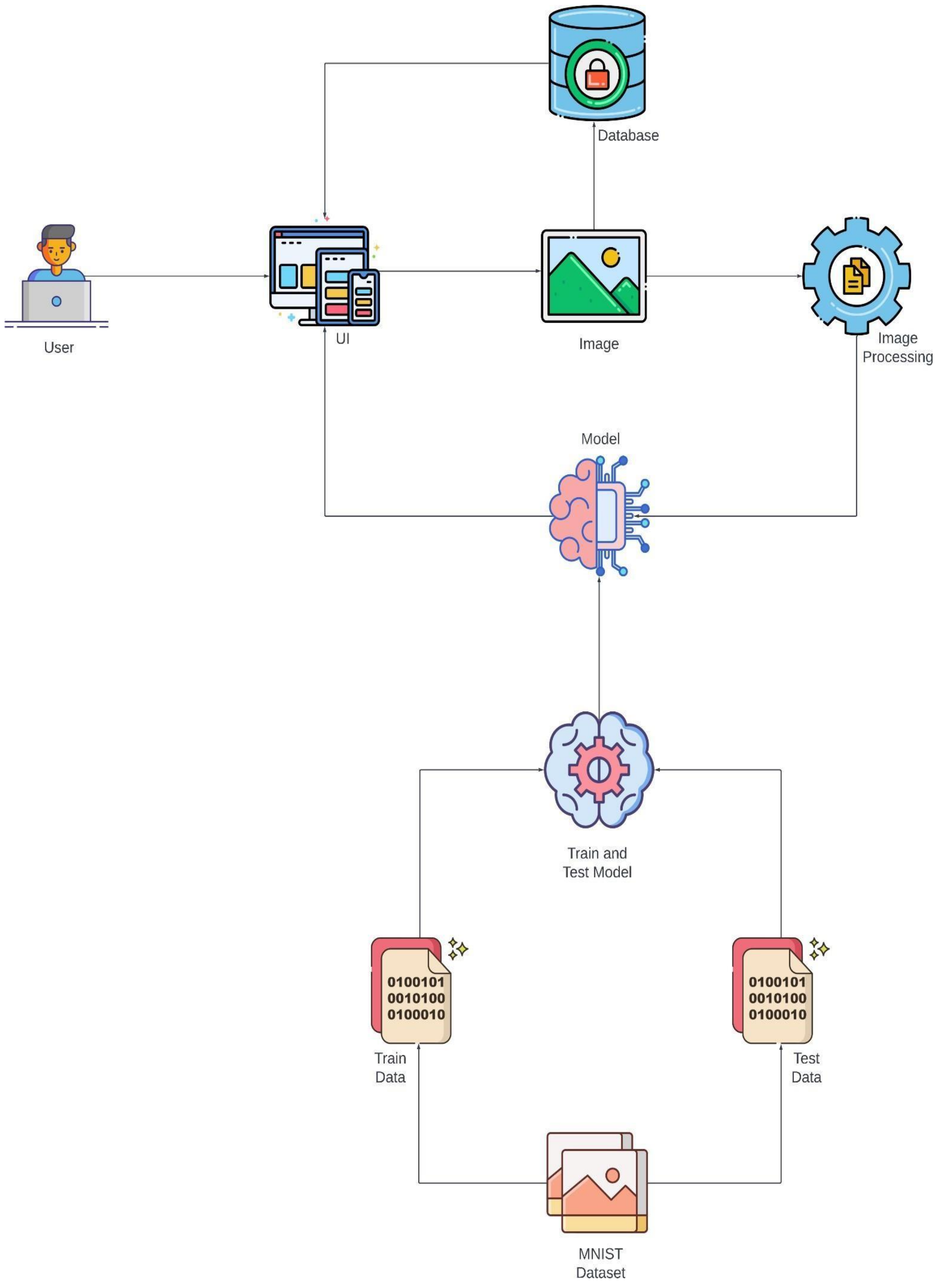
CHAPTER 5
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM



5.2 SOLUTION & TECHNICAL ARCHITECTURE


STRATEGY LAYER
Resource: Postal Service Organization
BUSINESS LAYER
Process: Machine Learning and Data Analytics
Process: Application Development
Role: Data Scientist
Role: Developer
DATA AND APPLICATION LAYER
Event: Image uploaded on the website
Service: Create and Deploy application
Data: MNIST Dataset
Component: Recognize the digit in the image
Function: Design and develop application
Data: User uploaded inputs
Process: Display the output on the application
Function: Respond to requirements and enhance the application
TECHNOLOGY LAYER
Service: Machine Learning Service
Service: Infrastructure-as-a-Service
Service: Database-as-a-Service
Collab: Managed computing in the cloud
Collab: Managed Database in the cloud
Node: Compute Server
Software: Flask
Software: Python
Software: Linux
Software: MongoDB
miro


Database
User
UI
Image
Image Processing
Model
Train and Test Model
0100101 0010100 0100010
Train Data
0100101 0010100 0100010
Test Data
MNIST Dataset

5.3 USER STORIES

User Type	Functional Requirements	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Customer	Accessing the Application	USN-1	As a user, I should be able to access the application from anywhere and use on any devices	User can access the application using the browser on any device	High	Sprint-4
	Uploading Image	USN-2	As a user, I should be able to upload images to predict the digits	User can upload images	High	Sprint-3
	Viewing the Results	USN-3	As a user, I should be able to view the results	The result of the prediction is displayed	High	Sprint-3
	Viewing Other Prediction	USN-4	As a user, I should be able to see other close predictions	The accuracy of other values must be displayed	Medium	Sprint-4
	Usage Instruction	USN-5	As a user, I should have a usage instruction to know how to use the application	The usage instruction is displayed on the home page	Medium	Sprint-4

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

SPRINT	USER STORY / TASK	STORY POINTS	PRIORITY	TEAM MEMBERS
Sprint - I	Get the dataset	3	High	Boomika P
	Explore the data	2	Medium	Boomika P Dharani S
	Data Pre-Processing	3	High	Jananipriya P V Karthika K
	Prepare training and testing data	3	High	Jananipriya P V Karthika K
Sprint - II	Create the model	3	High	Boomika P
	Train the model	3	High	Karthika K
	Test the model	3	High	Dharani S
Sprint - III	Improve the model	2	Medium	Jananipriya P V Karthika K
	Save the model	3	High	Jananipriya P V
	Build the Home Page	3	High	Boomika P Dharani S
	Setup a database to store input images	2	Medium	Jananipriya P V
Sprint - IV	Build the results page	3	High	Boomika P Dharani S

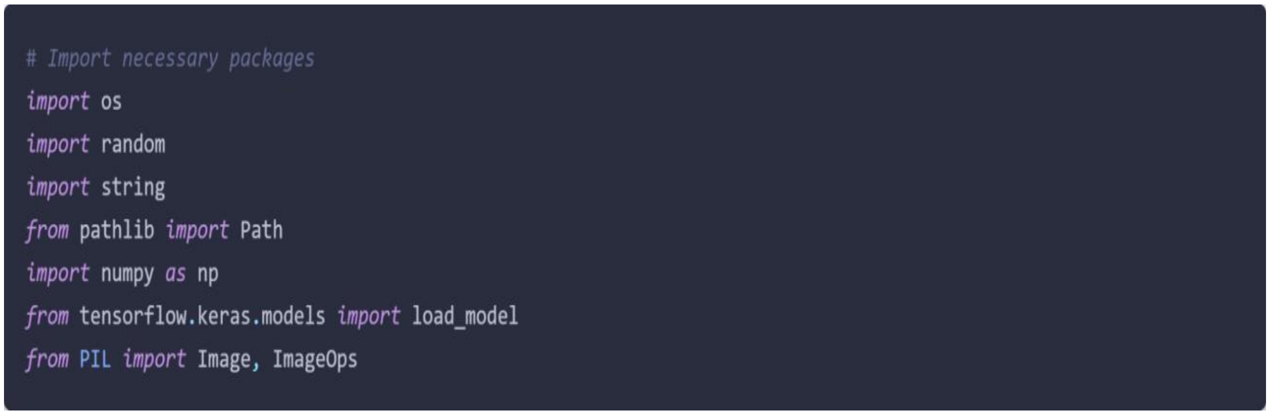
	Integrate the model with the application	3	High	Dharani S Karthika K
	Test the application	3	High	Dharani S Jananipriya P V

6.2 SPRINT DELIVERY SCHEDULE

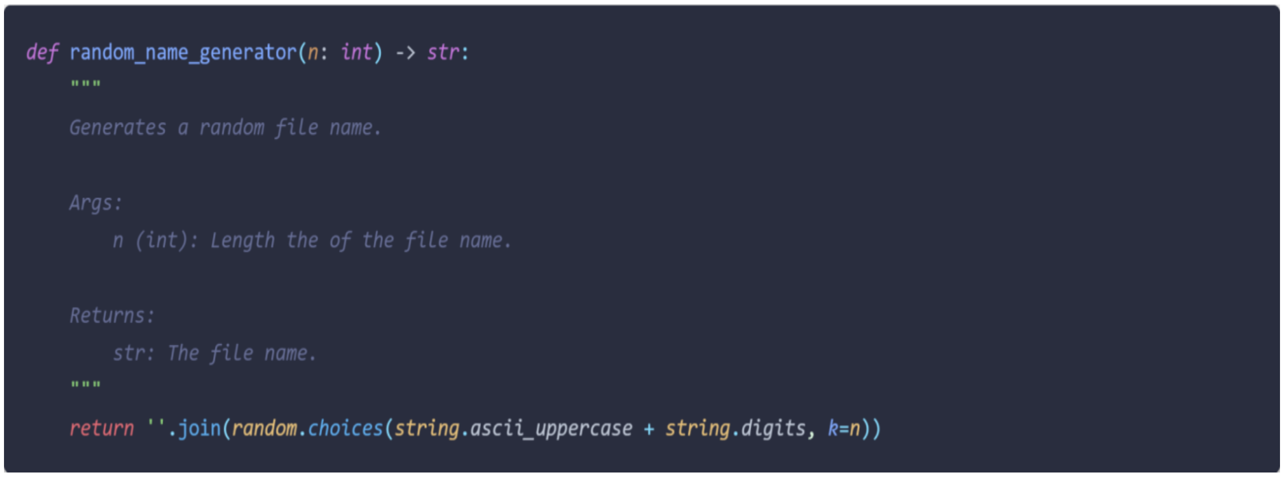
SPRINT	TOTAL STORY POINTS	DURATION	SPRINT START DATE	SPRINT END DATE (PLANNED)	STORY POINTS COMPLETED (AS ON PLANNED DATE)	SPRINT RELEASE DATE (ACTUAL)
Sprint - I	11	6 Days	24 Oct 2022	29 Oct 2022	11	29 Oct 2022
Sprint - II	9	6 Days	31 Oct 2022	05 Nov 2022	9	05 Nov 2022
Sprint - III	10	6 Days	07 Oct 2022	12 Nov 2022	10	12 Nov 2022
Sprint - IV	9	6 Days	14 Nov2022	19 Nov 2022	9	19 Nov 2022

CHAPTER 7

CODING & SOLUTIONING



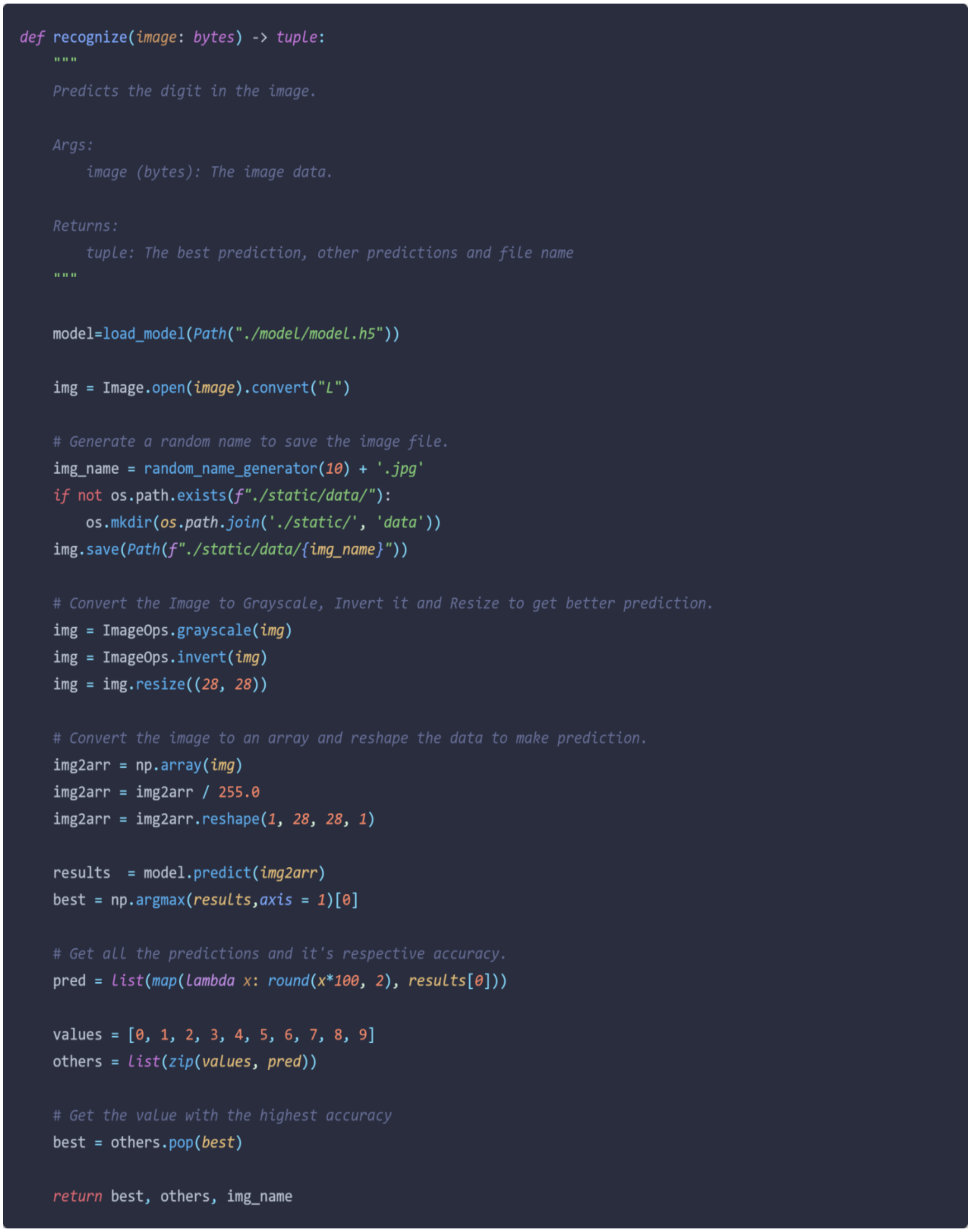
```python
# Import necessary packages
import os
import random
import string
from pathlib import Path
import numpy as np
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps
```



```python
def random_name_generator(n: int) -> str:
    """
    Generates a random file name.

    Args:
        n (int): Length the of the file name.

    Returns:
        str: The file name.
    """
    return ''.join(random.choices(string.ascii_uppercase + string.digits, k=n))
```



```python
def recognize(image: bytes) -> tuple:
    """
    Predicts the digit in the image.

    Args:
        image (bytes): The image data.

    Returns:
        tuple: The best prediction, other predictions and file name
    """

    model=load_model(Path("./model/model.h5"))

    img = Image.open(image).convert("L")

    # Generate a random name to save the image file.
    img_name = random_name_generator(10) + '.jpg'
    if not os.path.exists(f"./static/data/"):
        os.mkdir(os.path.join('./static/', 'data'))
    img.save(Path(f"./static/data/{img_name}"))

    # Convert the Image to Grayscale, Invert it and Resize to get better prediction.
    img = ImageOps.grayscale(img)
    img = ImageOps.invert(img)
    img = img.resize((28, 28))

    # Convert the image to an array and reshape the data to make prediction.
    img2arr = np.array(img)
    img2arr = img2arr / 255.0
    img2arr = img2arr.reshape(1, 28, 28, 1)

    results  = model.predict(img2arr)
    best = np.argmax(results,axis = 1)[0]

    # Get all the predictions and it's respective accuracy.
    pred = list(map(lambda x: round(x*100, 2), results[0]))

    values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
    others = list(zip(values, pred))

    # Get the value with the highest accuracy
    best = others.pop(best)

    return best, others, img_name
```

CHAPTER 8
TESTING

8.1 TEST CASES

Test caseID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
HP_TC_001	UI	Home Page	Verify UI elements in the Home Page	The Home page must be displayed properly	Working as expected	PASS
HP_TC_002	UI	Home Page	Check if the UIelements are displayed properly in different screen sizes	The Home page must be displayed properly in all sizes	The UI is not displayed properly in screen size 2560 x 1801 and 768 x 630	FAIL
HP_TC_003	Functional	Home Page	Check if user can upload their file	The input image should be uploaded to the application successfully	Working as expected	PASS
HP_TC_004	Functional	Home Page	Check if user cannot upload unsupported files	The application should not allow user to select a non image file	User is able toupload any file	FAIL
HP_TC_005	Functional	Home Page	Check if the page redirects to the result page once the input is given	The page should redirect to the results page	Working as expected	PASS

BE_TC_001	Functional	Backend	Check if all theroutes are working properly	All the routes should properly work	Working as expected	PASS
M_TC_001	Functional	Model	Check if the model can handle various image sizes	The model should rescale the image and predict the results	Working as expected	PASS
M_TC_002	Functional	Model	Check if the model predicts the digit	The model should predict the number	Working as expected	PASS
M_TC_003	Functional	Model	Check if the model can handle complex input image	The model should predict the number in the complex image	The model fails to identify the digit since the model is not built to handle such data	FAIL
RP_TC_001	UI	Result Page	Verify UI elements in the Result Page	The Result page must be displayed properly	Working as expected	PASS
RP_TC_002	UI	Result Page	Check if the input image is displayed properly	The input image should be displayed properly	The size of the input image exceeds the display container	FAIL
RP_TC_003	UI	Result Page	Check if the result is displayed properly	The result should be displayed properly	Working as expected	PASS
RP_TC_004	UI	Result Page	Check if the other predictions are displayed properly	The other predictions should be displayed properly	Working as expected	PASS

8.2 USER ACCEPTANCE TESTING

8.2.1 DEFECT ANALYSIS

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Total
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won't Fix	1	0	1	0	2
Total	6	1	4	3	14

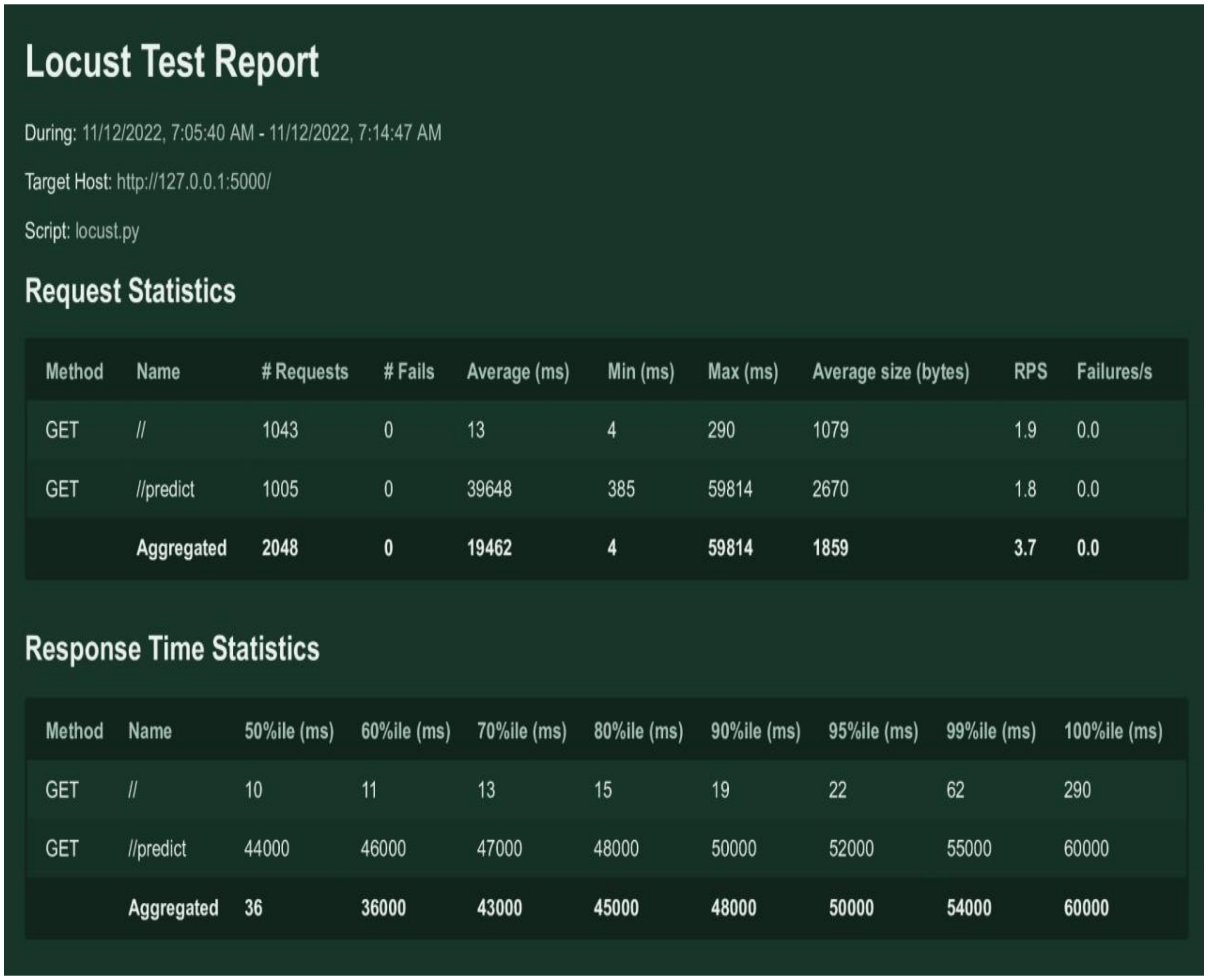
8.2.2 TEST CASE ANALYSIS

Section	Total Cases	Not Tested	Fail	Pass
Client Application	1	0	3	7
Security	2	0	1	1
Performance	3	0	1	2
Exception Reporting	2	0	0	2

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS



Locust Test Report

During: 11/12/2022, 7:05:40 AM - 11/12/2022, 7:14:47 AM

Target Host: http://127.0.0.1:5000/

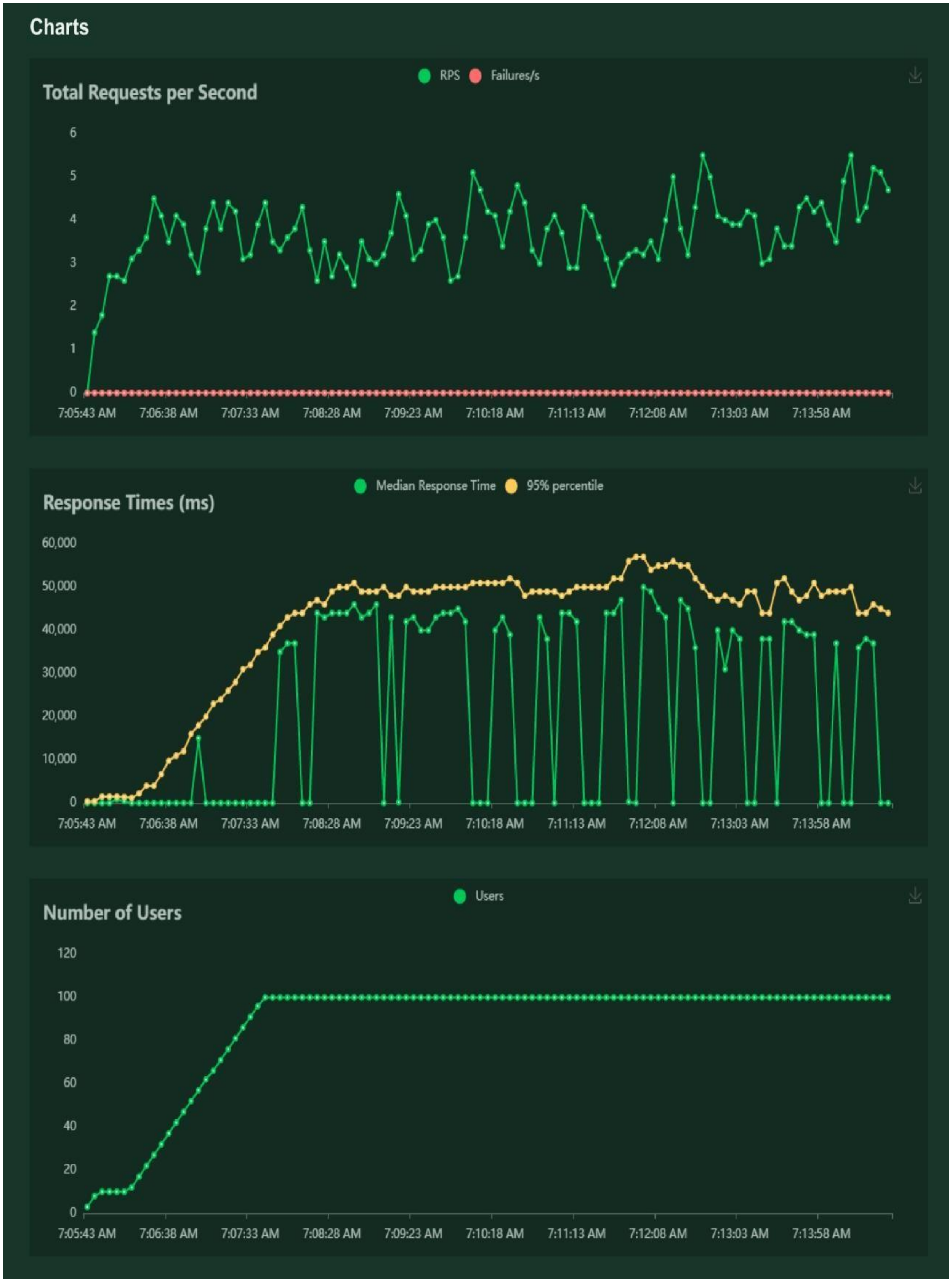
Script: locust.py

Request Statistics

Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	//	1043	0	13	4	290	1079	1.9	0.0
GET	//predict	1005	0	39648	385	59814	2670	1.8	0.0
	Aggregated	**2048**	**0**	**19462**	**4**	**59814**	**1859**	**3.7**	**0.0**

Response Time Statistics

Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	//	10	11	13	15	19	22	62	290
GET	//predict	44000	46000	47000	48000	50000	52000	55000	60000
	Aggregated	**36**	**36000**	**43000**	**45000**	**48000**	**50000**	**54000**	**60000**



Charts
Total Requests per Second
● RPS ● Failures/s
6
5
4
3
2
1
0
7:05:43 AM
7:06:38 AM
7:07:33 AM
7:08:28 AM
7:09:23 AM
7:10:18 AM
7:11:13 AM
7:12:08 AM
7:13:03 AM
7:13:58 AM
Response Times (ms)
● Median Response Time ● 95% percentile
60,000
50,000
40,000
30,000
20,000
10,000
0
7:05:43 AM
7:06:38 AM
7:07:33 AM
7:08:28 AM
7:09:23 AM
7:10:18 AM
7:11:13 AM
7:12:08 AM
7:13:03 AM
7:13:58 AM
Number of Users
● Users
120
100
80
60
40
20
0
7:05:43 AM
7:06:38 AM
7:07:33 AM
7:08:28 AM
7:09:23 AM
7:10:18 AM
7:11:13 AM
7:12:08 AM
7:13:03 AM
7:13:58 AM

CHAPTER 10

ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Reduces manual work
- More accurate than average human
- Capable of handling a lot of data
- Can be used anywhere from any device

DISADVANTAGES

- Cannot handle complex data
- All the data must be in digital format
- Requires a high performance server for faster predictions
- Prone to occasional errors

CHAPTER 11
CONCLUSION

This project demonstrated a web application that uses machine learning to recognize handwritten numbers. Flask, HTML, CSS, JavaScript, and a few other technologies were used to create this project. The model predicts the handwritten digit using a CNN network. During testing, the model achieved a 99.61% recognition rate. The proposed project is scalable and can easily handle a huge number of users. Since it is a web application, it is compatible with any device that can run a browser. This project is extremely useful in real-world scenarios such as recognizing number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (tax forms) and so on. There is so much room for improvement, which can be implemented in subsequent versions.

CHAPTER 12

FUTURE SCOPE

This project is far from complete and there is a lot of room for improvement. Some of the improvements that can be made to this project are as follows:

- Add support to detect from digits multiple images and save the results
- Add support to detect multiple digits
- Improve model to detect digits from complex images
- Add support to different languages to help users from all over the world

This project has endless potential and can always be enhanced to become better. Implementing this concept in the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency.

APPENDIX

SOURCE CODE

MODEL CREATION

```
{
"cells": [
{
"cell_type": "code",
"execution_count": 1,
"id": "a9535ae5",
"metadata": {},
"outputs": [],
"source": [
"from tensorflow.keras.models import load_model"
]
},
{
"cell_type": "code",
"execution_count": 2,
"id": "55b16a6c",
"metadata": {},
"outputs": [],
"source": [
"model=load_model(r'C:/Users/Dharani Saravanan/Pictures/models/mnistCNN.h5')"
]
},
{
"cell_type": "code",
"execution_count": 3,
"id": "839c90d9",
"metadata": {},
"outputs": [],
"source": [
"from PIL import Image"
]
},
{
"cell_type": "code",
"execution_count": 4,
"id": "f065d1c7",
"metadata": {},
"outputs": [],
"source": [
"import numpy as np"
]
},
{
"cell_type": "code",
"execution_count": 21,
"id": "4bc71bf5",
"metadata": {},
}
],
"source": [
"for index in range(4):\n",
"    img=Image.open('C:\\Users\\Dharani Saravanan\\data' +str(index)+ 'IMB IMG.png').convert(\"L\")\n",
"    img=img.resize((28,28))\n",
"    im2arr=np.array(img)\n",
"    im2arr=im2arr.reshape(1,28,28,1)\n",
"    y_pred=model.predict(im2arr)\n",
"    print(y_pred)"
]
},
{
"cell_type": "code",
"execution_count": null,
"id": "14d8550a",
"metadata": {},
"outputs": [],
"source": []
},
{
"cell_type": "code",
"execution_count": null,
"id": "78e4809f",
"metadata": {},
"outputs": [],
"source": []
}
],
"metadata": {
"kernelspec": {
"display_name": "Python 3 (ipykernel)",
"language": "python",
"name": "python3"
},
"language_info": {
"codemirror_mode": {
"name": "ipython",
"version": 3
},
"file_extension": ".py",
"mimetype": "text/x-python",
"name": "python",
"nbconvert_exporter": "python",
"pygments_lexer": "ipython3",
"version": "3.9.13"
}
},
"nbformat": 4,
"nbformat_minor": 5
}
```

FLASK APP

```
from flask import Flask , render_template , request
from werkzeug.utils import secure_filename
from PIL import Image
from matplotlib import pyplot
import numpy as np
import tensorflow as tf
#from load_model import model
import os
app = Flask(_name_)
app.debug = True
```

```
@app.route("/")
def upload():
 return render_template('index.html')
@app.route('/', methods = ['GET', 'POST'])
def upload_file():
# Create a directory in a known location to save files to.
 uploads_dir = os.path.join(r'D:\desktop2020\ibm\MNIST-Web-App-master', 'static')
 model = tf.keras.models.load_model("mnist_ann_model.h5")
 if request.method == 'POST':
 f = request.files['file']
 f.save(os.path.join(uploads_dir , secure_filename(f.filename)))
 img = Image.open(f).convert('L')
 img = img.resize((28,28) , Image.ANTIALIAS)
 data = ((np.asarray(img))/255.0)
 pred= model.predict(data.reshape(1,28,28,1))
 return render_template('prediction.html' ,out = str(pred[0].argmax()) , im = f.filename)
@app.route('/index')
 def index():
 return render_template('index.html')
 if _name_ == '_main_':
 app.run(debug = True)
```

HOME PAGE (HTML)

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Digit Recognition WebApp</title>
<link rel="stylesheet" type="text/css" href="{{ url_for('static',filename='css/style.css') }}">
</head>
<body>
 <section>
  <h1 class="welcome">IBM PROJECT
<div id="team_id">TEAM ID : PNT2022TMID16152</div>
</h1>
</section>
```

```
<section id="title">
<p style="font-size:35px;color:white;">HANDWRITTEN RECOGNITION SYSTEM</p>
<br><br>
<p style="color:white;">
The website is designed to predict the handwritten digit.
</p>
<p style="color:white;">
Handwriting recognition is one of the compelling research works going on because every individual in this world
has their own style of writing. It is the capability of the computer to identify and understand
handwritten digits or characters automatically. Because of the progress in the field of science and technology,
everything is being digitalized to reduce human effort.</p>
<br>
<p style="color:white;"> Hence, there comes a need for handwritten digit recognition in many real-time applications.
MNIST data set is widely used for this recognition process and it has 70000 handwritten digits.
We use Artificial neural networks to train these images and build a deep learning model.
Web application is created where the user can upload an image of a handwritten digit.
This image is analyzed by the model and the detected result is returned on to UI</p>
</section>
<form action="/" method="POST" enctype="multipart/form-data">
<div style="text-align:center">
<p style="color:white;">SELECT IMAGE<p><input type="file" name="file">
<button type="submit"> Submit </button>
</div>
</form>
</body>
</html>
```

HOME PAGE (CSS)

```
body{
background-image: url("img.png");
width: 90%;
height: 200%;
margin: auto;
text-align: center;
padding: 50px;
}
form{
margin-left: 205px;
```

```
margin-top: 200px;
padding: 20px;
width: 610px;
border-radius: 10px;
}
input{
color: rgb(93, 92, 92);
background-color: rgb(253, 252, 252);
font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif
}
#confidence{
font-family: 'Josefin Sans', sans-serif;
margin-top: 7.5%;
}
#content{
margin: 0 auto;
padding: 2% 15%;
padding-bottom: 0;
}
.welcome{
text-align: center;
position: relative;
color: honeydew;
background-color:#2b0e66;
padding-top: 1%;
padding-bottom: 1%;
font-weight: bold;
font-family: 'Prompt', sans-serif;
}
#team_id{
text-align: right;
font-size: 20px;
padding-right: 3%;
}
#result{
font-size: 5rem;
}
#title{
padding: 1.5% 15%;
margin: 0 auto;
text-align: center;
}
button{
color: white;
```

```
background-color: rgb(54, 47, 47);
text-align: center;
}
p{
font-family: 'Source Code Pro', monospace,sans-serif;
margin-top: 1%
```

PREDICTION PAGE (HTML)

```
<!DOCTYPE html>
<html lang=""en>
<head>
<meta charset="UTF-8">
<title>Digit Recognition WebApp</title>
<link rel="stylesheet" type="text/css" href="{{ url_for('static',filename='css/style2.css') }}">
</head>
<body>
<h4>WELCOME TO RECOGNITION</h4>
<a href="/index">GO TO HOME PAGE</a>
<img class = 'predict_img' src = "static/{{im}}" alt ='No Image found'  width="280" height="250">
<h2 class ='output'> DIGIT RECOGNITION IS: <font color = "Yellow"><b>
<u>'{{out}}'</u></b></font></h2>
</body>
</html>
```

PREDICTION PAGE (CSS)

```
body{
background-image: url("img3.png");
height: 400px;
width:550px;
}
form{
margin-left: 205px !important;
margin-top: 20px !important;
}
input{
color: rgb(93, 92, 92);
background-color: rgb(211, 16, 16);
font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif
}
button{
```

```
color: white;
background-color: rgb(54, 47, 47);
}
.predict_img{
border: 10px solid rgb(255, 66, 68);
border-radius: 10px;
opacity: 0.1;
margin-top: -1px;
margin-left: 215px;
}
.output{
color: rgb(255, 254, 254);
margin-top: 1px;
margin-left: 170px;
margin-bottom: 1px;
}
.wrapper{
width: 100%;
max-width: 1180px;
padding: 0 10px;
margin: 0 auto;
}
.wrapper:after{
display: block;
content: "";
clear: both;
}
h1.logo{
background-image: url(images/logo1.png);
background-repeat: no-repeat;
width: 100px;
text-indent: -10000px;
float: left;
}
header{
background-color:none;
top:0;
}
header nav{
float: right;
}
header nav h2{
text-indent: -10000px;
height: 0;
```

```
margin: 0;
}
header nav li{
float: left;
list-style-type: none;
margin: 10px 20px;
}
header nav li a{
text-decoration: none;
color: #201C1C;
font-size: 18px;
}
header nav li a:hover{
color: #FDFCFC;
transition: 0.2s ease-in;
}
#headtext{
font-family: Helvetica;
margin-top: 30px;
text-align: center;
font-size: 25px;
top: 300px;
display: block;
}
btn1{
border-radius: 10px;
width: 9%;
background-color: #18C13A;
padding: 5px;
font-size: 20px;
color: white;
margin-left: 47%
}
#btn1:hover{
background-color: #0C9B28;
  }
```

GITHUB

https://github.com/IBM-EPBL/IBM-Project-17412-1659669660

PROJECT DEMO

https://drive.google.com/drive/folders/1TIJqNdCZRlluCxQdL39zcN4mr1o7YkhL