PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION

submitted by

PNT2022TMID28539

Sri Sai Krishnaa R C - 312819104075

Mukesh Kanna G - 312819104052

Mohan M - 312819104049

Sarath Kumar D - 312819104067

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 PROJECT OVERVIEW

Machine learning and deep learning play an important role in computer technology and artificial intelligence. With the use of deep learning and machine learning, human effort can be reduced in recognizing, learning, predictions and in many more areas.

Handwritten Digit Recognition is the ability of computer systems to recognise 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 and 6, 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

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.

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 important to 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 Quantum K-Nearest Neighbor Algorithm (2019)

Wang, Yuxiang and Wang, Ruijin and Li, Dongfen and Adu-Gyamfi, Daniel and Tian, Kaibin and Zhu, 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 and ambiguous 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

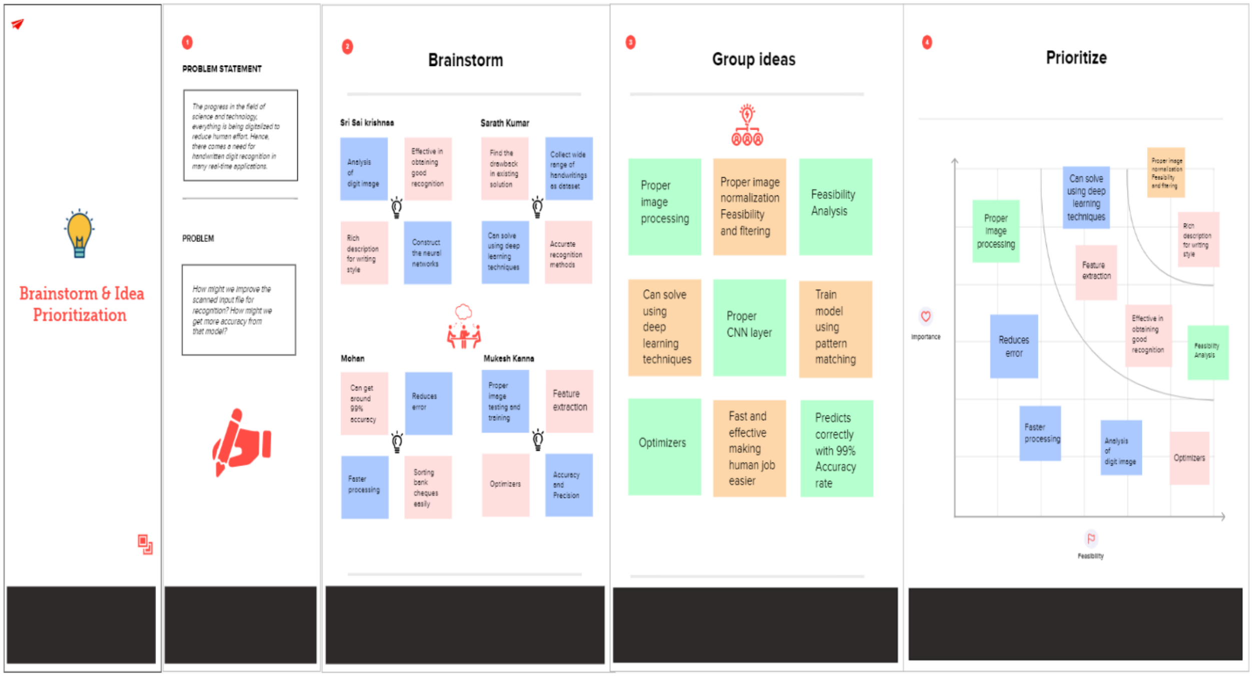
For years, the traffic department has been combating traffic law violators. These offenders endanger not only their own lives, but also the lives of other individuals. Punishing these offenders is critical to ensuring that others do not become like them. Identification of these offenders is next to impossible because it is impossible for the average individual to write down the license plate of a reckless driver. Therefore, the goal of this project is to help the traffic department identify these offenders and reduce traffic violations as a result.

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

3.2 IDEATION & BRAINSTORMING



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 image as 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 tra c surveillance cameras to recognize vehicle number plates The application can be integrated with Postal systems to recognize the pin codes e ectively
6	Scalability of the Solution	The application can easily be scaled to accept multiple inputs and process them parallelly to further increase e ciency

3.4 PROBLEM SOLUTION FIT

Project Design Phase-I - Solution Fit

Project Title: A Novel Method for Handwritten Digit Recognition System

TeamID: PNT2022TMID28539

Define CS, fit into CC

1. CUSTOMERSEGMENT(S) CS

- Medical data Transcriptions
- Banking
- Digital Government
- Schools and Colleges

2. CUSTOMERCONSTRAINTS CC

- Speed and Accuracy of the system
- Lack of reliable internet connections, unavailability of gadgets like mobile phones and computers, inaccessibility of appropriate cameras.
- Size of the Vocabulary

3. AVAILABLESOLUTIONS AS

- Free OCR API
- Using this system, they can resolve this type of problems
- Human centric data feed

Explore AS, differentiate

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

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

- Each and every handwriting has its own characteristics and uniqueness.
- Its difficult to understand the different people's handwriting digit.

Adaptive learning module with ML to learn from its own instances and gets updated

- To design a system that recognizes a wide range of handwriting script

5. PROBLEMROOTCAUSE RC

- The handwriting is differed from person to person
- Hand-written digits are in varying fonts and sizes; thus, they are becoming increasingly difficult to ascertain due to various factors such as weakening eyesight, time constraints, etc.
- Not everyone can understand everyone's handwriting

6. BEHAVIOUR BE

- Designing the best software that more quickly and accurately identifies the handwritten digits
- Provision for real-time handwritten update in case if the application used by fixed and same user

Customer wants reliable internet connections and high-quality cameras.

- Know the market trends and adapts accordingly

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

CHAPTER 4
REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FR No.	Sub Requirement (Story / Sub-Task)
FR-1	Image Data: Handwritten digit recognition refers to a computer's capacity to identify human handwritten digits from a variety of sources, such as photographs, documents, touch screens, etc., and categorise them into ten established classifications (0-9). In the realm of deep learning, this has been the subject of countless studies.
FR-2	Website: Web hosting makes the code, graphics, and other items that make up a website accessible online. A server hosts every website you've ever visited. The type of hosting determines how much space is allotted to a website on a server. Shared, dedicated, VPS, and reseller hosting are the four basic varieties.
FR-3	Digit Classifier Model: To train a convolutional network to predict the digit from an image, use the MNIST database of handwritten digits. get the training and validation data first.
FR-4	Cloud: The cloud offers a range of IT services, including virtual storage, networking, servers, databases, and applications. In plain English, cloud computing is described as a virtual platform that enables unlimited storage and access to your data over the internet.
FR-5	Modified National Institute of Standards and Technology dataset: The abbreviation MNIST stands for the MNIST dataset. It is a collection of 60,000 tiny square grayscale photographs, each measuring 28 by 28, comprising handwritten single digits between 0 and 9.

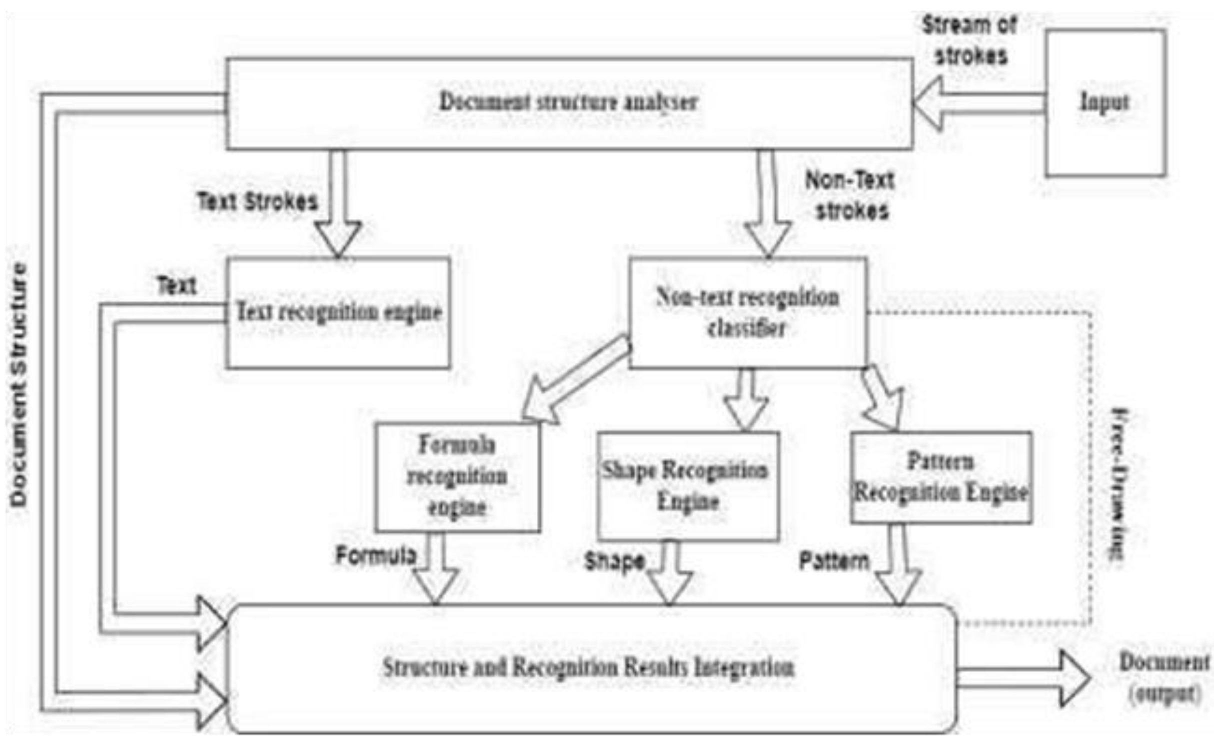
4.2 NON FUNCTIONAL REQUIREMENTS

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	Usability	One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Applications for digit recognition include filling out forms, processing bank checks, and sorting mail.
NFR-2	Security	1) The system generates a thorough description of the instantiation parameters, which might reveal information like the writing style, in addition to a categorization of the digit. 2) The generative uses a relatively. The generative models are capable of segmentation driven by recognition. 3) The procedure uses a relatively.
NFR-3	**Reliability**	The samples are used by the neural network to automatically deduce rules for reading handwritten digits. Furthermore, the network may learn more about handwriting and hence enhance its accuracy by increasing the quantity of training instances. Numerous techniques and algorithms, such as Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests, etc., can be used to recognize handwritten numbers.

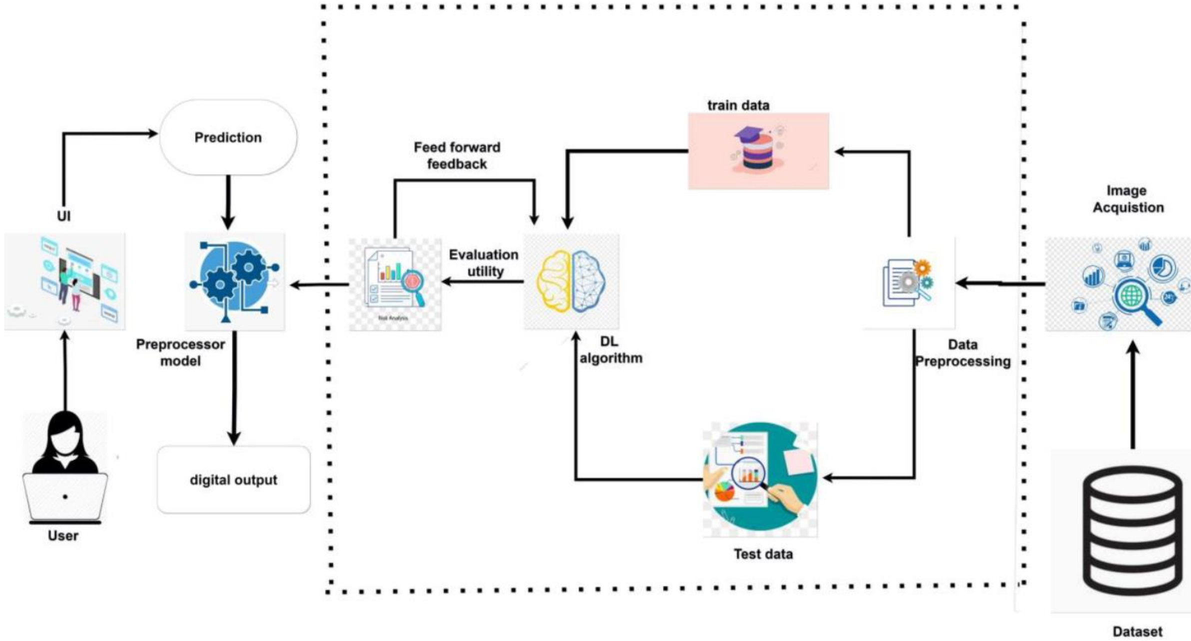
NFR-4	**Accuracy**	With typed text in high-quality photos, optical character recognition (OCR) technology offers accuracy rates of greater than 99%. However, variances in spacing, abnormalities in handwriting, and the variety of human writing styles result in less precise character identification.
NFR-5	**Availability**	Access to information restricted to each other
NFR-6	**Scalability**	The task of handwritten digit recognition, using a classifier, has great importance and use such as online handwriting recognition on computer tablets, recognize zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand (for example - tax forms) and so on.

CHAPTER 5
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM



5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Home	USN-1	As a user, I can view the guide and awareness to use this application.	I can view the awareness to use this application and its limitations.	Low	Sprint-1

		USN-2	As a user, I'm allowed to view the guided video to use the interface of this application.	I can gain knowledge to use this application by a practical method.	Low	Sprint-1

		USN-3	As a user, I can read the instructions to use this application.	I can read instructions also to use it in a user-friendly method.	Low	Sprint-2
	Recognize	USN-4	As a user, In this prediction page I get to choose the image.	I can choose the image from our local system and predict the output.	High	Sprint-2
	Predict	USN-6	As a user, I'm Allowed to upload and choose the image to be uploaded	I can upload and choose the image from the system storage and also in any virtual storage.	Medium	Sprint-3
		USN-7	As a user, I will train and test the input to get the maximum accuracy of output.	I can able to train and test the application until it gets maximum	High	Sprint-4

				accuracy of the result.		
		USN-8	As a user, I can access the MNIST data set	I can access the MNIST data set to produce the accurate result.	Medium	Sprint-3
Customer (Web user)	Home	USN-9	As a user, I can view the guide to use the web app.	I can view the awareness of this application and its limitations.	Low	Sprint-1

	Recognize	USN-10	As a user, I can use the web application virtually anywhere.	I can use the application portably anywhere.	High	Sprint-1
		USN-11	As it is an open source, can use it cost freely.	I can use it without any payment to be paid for it to access.	Medium	Sprint-2
		USN-12	As it is a web application, it is installation free	I can use it without the installation of the application or any software.	Medium	Sprint-4

	Predict	USN-13	As a user, I'm Allowed to upload and choose the image to be uploaded	I can upload and choose the image from the system storage and also in any virtual storage.	Medium	Sprint-3

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	As a user, I can collect the dataset from various resources with different handwritings.	10	Low	Mukesh Kanna, Sarath Kumar
Sprint-1	Data Preprocessing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test.	10	Medium	Sri Sai Krishnaa, Mukesh kanna
Sprint-2	Model Building	USN-3	As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit.	5	High	Mohan, Sarath Kumar
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High	Mohan, Sarath Kumar, Sri Sai Krishnaa

Sprint	Functional Requiremen t (Epic)	User Story Number	User Story / Task	Story Point s	Priorit y	Team Members
Sprint-2	Compiling the model	USN-5	With both the training data defined and model defined, it's time to configure the learning process.	2	Mediu m	Mukesh Kanna, Mohan
Sprint-2	Train & test the model	USN-6	As a user, let us train our model with our image dataset.	6	Mediu m	Sarath Kumar, Mukesh Kanna
Sprint-2	Save the model	USN-7	As a user, the model is saved & integrated with an android application or web application in order to predict something.	2	Low	Sri Sai Krishnaa,
Sprint-3	Building UI Application	USN-8	As a user, I will upload the handwritten digit image to the application by clicking a upload button.	5	High	Sri Sai Krishnaa, Sarath Kumar
Sprint-3		USN-9	As a user, I can know the details of the fundamental usage of the application.	5	Low	Mohan
Sprint-3		USN-10	As a user, I can see the predicted / recognized digits in the application.	5	Mediu m	Mukesh kanna, Sri Sai krishnaa
Sprint-4	Train the model on IBM	USN-11	As a user, I train the model on IBM and integrate flask/Django with scoring end point.	10	High	Sarath Kumar, Mohan
Sprint-4	Cloud Deployment	USN-12	As a user, I can access the web application and make the use of the product from anywhere.	10	High	Mukesh Kanna, Sri Sai Krishnaa,

						Sarath Kumar, Mohan

6.2 SPRINT DELIVERY SCHEDULE

Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

CHAPTER 7

CODING & SOLUTIONING

```python
from flask import Flask, render_template, request# Flask-It is our framework which we are going to use to run/serve our application.
#request-for accessing file which was uploaded by the user on our application.

from PIL import Image #used for manipulating image uploaded by the user.
import numpy as np #used for numerical analysis
from tensorflow.keras.models import load_model#to load our model trained with MNIST data
import tensorflow as tf#to run our model.
```

```python
@app.route('/') #default route
def upload_file():
   return render_template('main.html') #rendering html page
@app.route('/about') #Main page route
def upload_file1():
   return render_template('main.html') #rendering html page
@app.route('/upload') #main page route
def upload_file2():
   return render_template('index6.html')
```

```
@app.route('/predict', methods = ['POST']) #route for our prediction
def upload_image_file():
    if request.method == 'POST':
        img = Image.open(request.files['file'].stream).convert("L")# convert image to monochrome
        img = img.resize((28,28))# resizing of input image
        im2arr = np.array(img)#converting to image
        im2arr = im2arr.reshape(1,28,28,1) #reshaping according to our requirement
        y_pred = model.predict_classes(im2arr) #predicting the results
        print(y_pred) #printing our result in prompt
        #return 'Predicted Number: ' + str(y_pred) #returning our output
```

CHAPTER 8
TESTING

8.1 TEST CASES

Test case ID	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 UI elements 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 to upload 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 the routes 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	1	3
Duplicate	1	0	0	0	1
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	1	1	2
Skipped	0	0	1	1	2

Won’t Fix	1	0	1	0	2
Total	7	1	6	4	18

8.2.2 TEST CASE ANALYSIS

Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	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

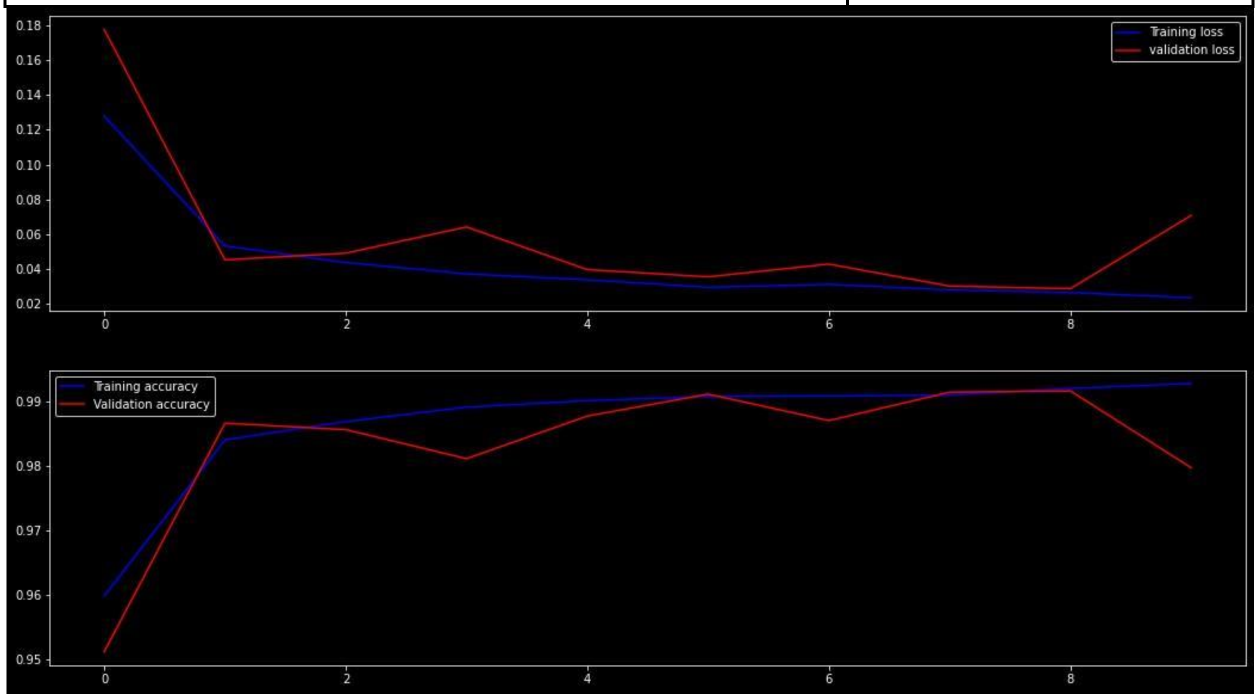
9.1.1 MODEL SUMMARY

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 64)        640

 conv2d_1 (Conv2D)           (None, 24, 24, 32)        18464

 flatten (Flatten)           (None, 18432)             0

 dense (Dense)               (None, 10)                184330

=================================================================
Total params: 203,434
Trainable params: 203,434
Non-trainable params: 0
_________________________________________________________________
```

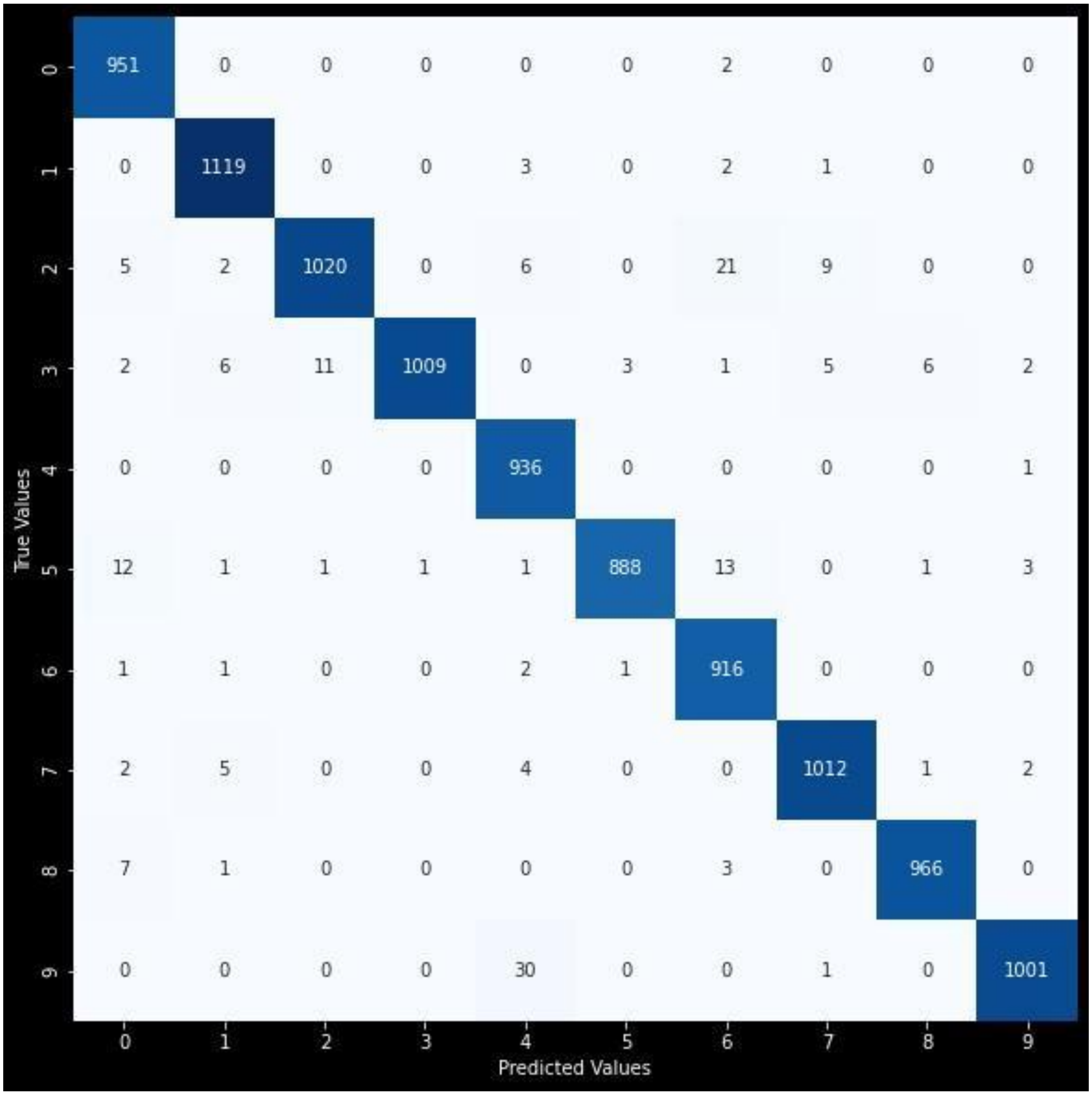
9.1.2 ACCURACY

CONTENT	VALUE
Training Accuracy	99.14%
Training Loss	2.70%
Validation Accuracy	97.76%

Validation Loss	10.36%



9.1.3 CONFUSION MATRIX



9.1.4 CLASSIFICATION REPORT

	precision	recall	f1-score	support
0	1.00	0.97	0.98	980
1	0.99	0.99	0.99	1135
2	0.96	0.99	0.97	1032
3	0.97	1.00	0.98	1010
4	1.00	0.95	0.98	982
5	0.96	1.00	0.98	892
6	0.99	0.96	0.97	958
7	0.99	0.98	0.99	1028
8	0.99	0.99	0.99	974
9	0.97	0.99	0.98	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.98	0.98	0.98	10000

9.1.5 APPLICATION TEST REPORT

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/

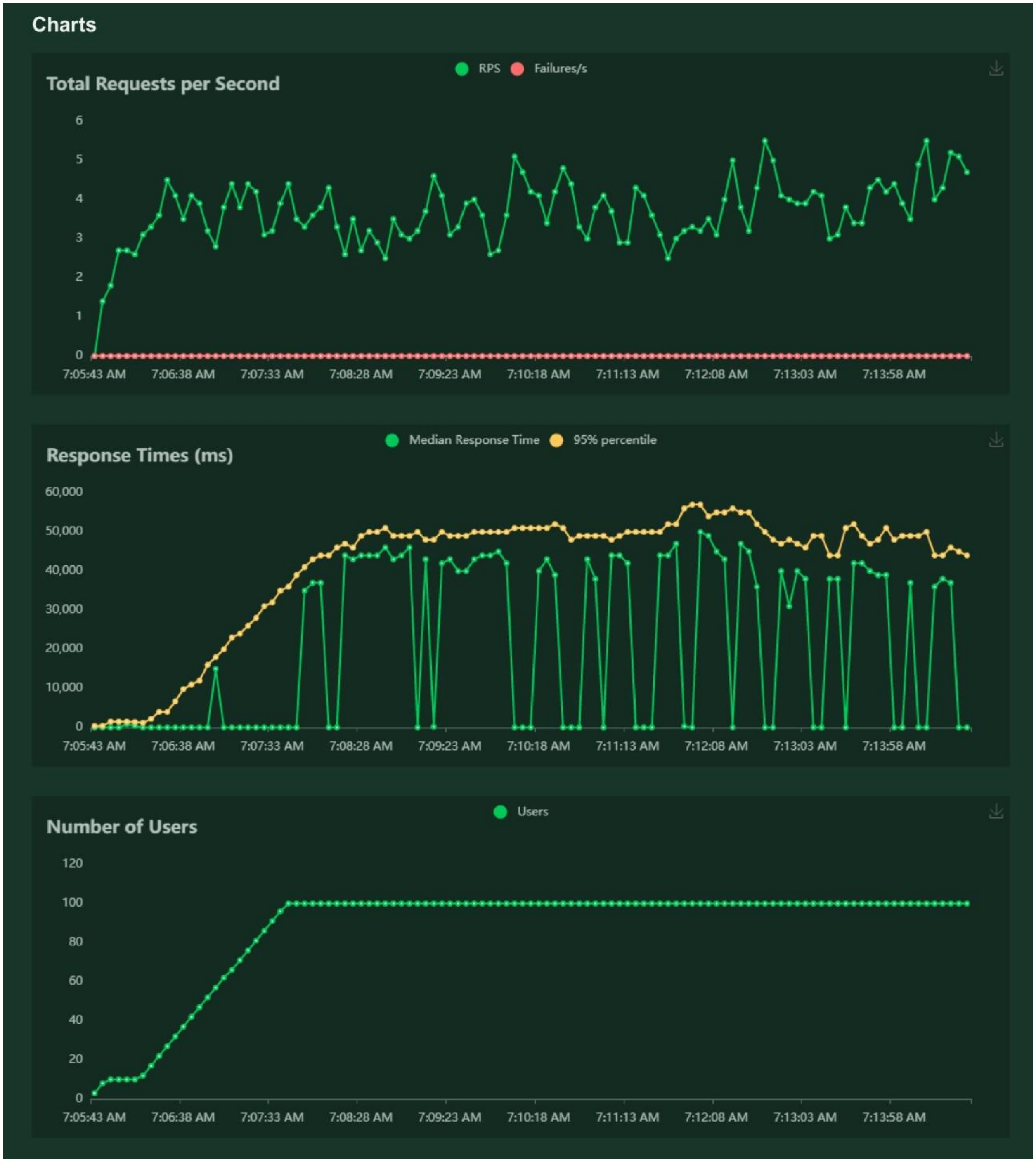
Script: locust.py

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 recognise 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

```
# Load the necessary packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from keras.utils import np_utils
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, Dense, Flatten
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps

# Load the data
(X_train, y_train), (X_test, y_test) = mnist.load_data()

# Data pre-processing
X_train = X_train.reshape(60000, 28, 28, 1).astype('float32')
X_test = X_test.reshape(10000, 28, 28, 1).astype('float32')

number_of_classes = 10
Y_train = np_utils.to_categorical(y_train, number_of_classes)
Y_test = np_utils.to_categorical(y_test, number_of_classes)
```

```
# Create the model
model = Sequential()
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation="relu"))
model.add(Conv2D(32, (3, 3), activation="relu"))
model.add(Flatten())
model.add(Dense(number_of_classes, activation="softmax"))

model.compile(loss='categorical_crossentropy', optimizer="Adam", metrics=["accuracy"])

# Train the model
model.fit(X_train, Y_train, batch_size=32, epochs=5, validation_data=(X_test,Y_test))

# Evaluate the model
metrics = model.evaluate(X_test, Y_test, verbose=0)
print("Metrics (Test Loss & Test Accuracy): ")
print(metrics)

# Save the model
model.save("model.h5")
```

```
# Test the saved model
model=load_model("model.h5")

img = Image.open("sample.png").convert("L")
img = img.resize((28, 28))
img2arr = np.array(img)
img2arr = img2arr.reshape(1, 28, 28, 1)
results  = model.predict(img2arr)
results = np.argmax(results,axis = 1)
results = pd.Series(results,name="Label")
print(results)
```

FLASK APP

```
from flask import Flask,render_template,request
from recognizer import recognize

app=Flask(__name__)

@app.route('/')
def main():
    return render_template("home.html")


@app.route('/predict',methods=['POST'])
def predict():
    if request.method=='POST':
      image = request.files.get('photo', '')
      best, others, img_name = recognize(image)
      return render_template("predict.html", best=best, others=others, img_name=img_name)


if __name__=="__main__":
    app.run()
```

RECOGNIZER

```
# 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
```

HOME PAGE (HTML)

```html
<html>
    <head>
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <title>Handwritten Digit Recognition</title>
        <link rel="icon" type="image/svg" sizes="32x32" href="{{url_for('static',filename='images/icon.svg')}}"
 />
        <link rel="stylesheet" href="{{url_for('static',filename='css/main.css')}}" />
        <script src="https://unpkg.com/feather-icons"></script>
        <script defer src="{{url_for('static',filename='js/script.js')}}"></script>
    </head>
    <body>
        <div class="container">
            <div class="heading">
                <h1 class="heading__main">Handwritten Digit Recognizer</h1>
                <h2 class="heading__sub">Easily analyze and detect handwritten digits</h2>
            </div>
            <div class="upload-container">
                <div class="form-wrapper">
                    <form class="upload" action="/predict" method="post" enctype="multipart/form-data">
                        <label id="label" for="upload-image"><i data-feather="file-plus"></i>Select File</label>
                        <input type="file" name="photo" id="upload-image" hidden />
                        <button type="submit" id="up_btn"></button>
                    </form>
                    <img id="loading" src="{{url_for('static',filename='images/loading.gif')}}">
                </div>
            </div>
        </div>
    </body>
</html>
```

HOME PAGE (CSS)

```css
@import url("https://fonts.googleapis.com/css2?family=Overpass:wght@200;300;400;500;600;700;900&display=swap");

* {
    padding: 0;
    margin: 0;
}

body {
    color: black;
    font-family: "Overpass", sans-serif;
}
```

```css
.container {
    width: 100%;
    height: 100%;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    background-color: white;
}

.heading {
    margin-top: -2rem;
    padding-bottom: 2rem;
    width: fit-content;
    text-align: center;
}

.heading .heading__main {
    font-size: 3rem;
    font-weight: 550;
}

.heading .heading__sub {
    font-size: 1rem;
    color: rgb(90, 88, 88);
}

.upload-container {
    box-shadow: 0 0 20px rgb(172, 170, 170);
    width: 40rem;
    height: 25rem;
    padding: 1.5rem;
}

.form-wrapper {
    background-color: rgba(190, 190, 190, 0.5);
    width: 100%;
    height: 100%;
    display: flex;
    border: 1px dashed black;
    justify-content: center;
    align-items: center;
}

.form-wrapper #loading {
    display: none;
    position: absolute;
}
```

```css
.form-wrapper .upload {
    display: flex;
    justify-content: center;
    align-items: center;
    width: 8rem;
    height: -webkit-fit-content;
    height: -moz-fit-content;
    height: fit-content;
    border-radius: 6px;
    color: white;
    background-color: rgb(114, 96, 182);
    box-shadow: 0 5px 10px rgb(146, 135, 247);
}

.form-wrapper .upload #up_btn {
    display: none;
}

.form-wrapper .upload label {
    font-size: 1rem;
    font-weight: 600;
    color: white;
    height: 100%;
    width: 100%;
    padding: 10px;
    display: block;
}

.form-wrapper .upload svg {
    height: 15px;
    width: auto;
    padding-right: 8px;
    margin-bottom: -2px;
}

@media screen and (max-width: 700px) {
    .upload-container {
        height: 20rem;
        width: 18rem;
        margin-top: 3.5rem;
        margin-bottom: -8rem;
    }

    .heading .heading__main {
        margin-top: -6rem;
        font-size: 2rem;
        padding-bottom: 1rem;
    }
}
```

HOME PAGE (JS)

```
feather.replace(); // Load feather icons

form = document.querySelector('.upload')
loading = document.querySelector("#loading")
select = document.querySelector("#upload-image");

select.addEventListener("change", (e) => {
    e.preventDefault();

    form.submit()
    form.style.visibility = "hidden";
    loading.style.display = 'flex';
});
```

PREDICT PAGE (HTML)

```
<html>
    <head>
        <title>Prediction | Handwritten Digit Recognition</title>
        <link rel="stylesheet" href="{{url_for('static',filename='css/predict.css')}}" />
        <link rel="icon" type="image/svg" sizes="32x32" href="{{url_for('static',filename='images/icon.svg')}}"
 />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    </head>
    <body>
        <div class="container">
            <h1>Prediction</h1>
            <div class="result-wrapper">
                <div class="input-image-container">
                    <img src="{{url_for('static',filename='data/')}}{{img_name}}" />
                </div>
                <div class="result-container">
                    <div class="value">{{best.0}}</div>
                    <div class="accuracy">{{best.1}}%</div>
                </div>
            </div>
            <h1>Other Predictions</h1>
            <div class="other_predictions">
                {% for x in others %}
                <div class="value">
                    <h2>{{x.0}}</h2>
                    <div class="accuracy">{{x.1}}%</div>
                </div>
                {% endfor %}
            </div>
        </div>
    </body>
</html>
```

```css
@import url("https://fonts.googleapis.com/css2?family=Overpass:wght@200;300;400;500;600;700;900&display=swap");

body {
    color: black;
    font-family: "Overpass", sans-serif;
}

h1 {
    padding-top: 2rem;
}

.container {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
}

.result-wrapper {
    width: -webkit-fit-content;
    width: -moz-fit-content;
    width: fit-content;
    height: -webkit-fit-content;
    height: -moz-fit-content;
    height: fit-content;
    box-shadow: 0 0 10px rgb(126, 125, 125);
    padding: 1.5rem;
    display: flex;
    justify-content: center;
    align-items: center;
    -moz-column-gap: 1rem;
    column-gap: 1rem;
}

.result-wrapper .input-image-container,
.result-wrapper .result-container {
    width: 15rem;
    height: 15rem;
    border: 1px dashed black;
    justify-content: center;
    display: flex;
    align-items: center;
    flex-direction: column;
    background-color: rgb(209, 206, 206);
}
```

```css
.result-wrapper .input-image-container img {
    width: 60%;
    height: 60%;
    background-color: aqua;
    background-size: contain;
}

.result-wrapper .result-container .value {
    font-size: 6rem;
}

.result-wrapper .result-container .accuracy {
    margin-top: -1rem;
}

.other_predictions {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-wrap: wrap;
    column-gap: 1rem;
    row-gap: 1rem;
    font-weight: 700;
}

.other_predictions .value {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
    width: 5rem;
    height: 5rem;
    box-shadow: 0 0 7px rgb(158, 157, 157);
}

.other_predictions .value div {
    margin-top: -1.2rem;
}

@media screen and (max-width: 700px) {
    h1 {
        font-size: 2.3rem;
    }

    .result-wrapper .input-image-container,
    .result-wrapper .result-container {
        width: 7rem;
        height: 7rem;
    }

    .result-wrapper .result-container .value {
        font-size: 4rem;
    }
}
```

GITHUB

https://github.com/IBM-EPBL/IBM-Project-35347-1660283908

PROJECT DEMO

https://drive.google.com/drive/folders/1Fe93uSc5A8eqF_ccM-pC6gYEvHS4WQSU?usp=sharing