PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION

submitted by

PNT2022TMID02158

Jeevamutharasi S	-	2116190701079
Jeevitha C	-	2116190701081
Kaushik A	-	2116190701090
Krithic Kumar UB	-	2116190701103

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 . PROJECT OVERVIEW

Handwritten digit recognition is the ability of a computer system to recognize the handwritten inputs like digits, characters etc. from a wide variety of sources like emails, papers, images, letters etc. This has been a topic of research for decades. Some of the research areas include signature verification, bank check processing, postal address interpretation from envelopes etc. Here comes the use of Deep Learning. In the past decade, deep learning has become the hot tool for Image Processing, object detection, handwritten digit and character recognition etc. A lot of machine learning tools have been developed like scikit-learn, scipy-image etc. and pybrains, Keras, Theano, Tensorflow by Google, TFLearn etc. for Deep Learning. These tools make the applications robust and therefore more accurate.

The Artificial Neural Networks can almost mimic the human brain and are a key ingredient in image processing field. For example, Convolutional Neural Networks with Back Propagation for Image Processing, Deep Mind by Google for creating Art by learning from existing artist styles etc.. Handwriting Recognition has an active community of academics studying it.

Classification of images and patterns has been one of the major implementation of Machine Learning and Artificial Intelligence. People are continuously trying to make computers intelligent so that they can do almost all the work done by humans Handwriting recognition system is the most basic and an important step towards this huge and interesting area of Computer Vision.

1.2. PURPOSE

Digit recognition systems are able to identifynumbers from a varietyof sources, including emails, bank checks, papers, images, etc. They can also be used in a variety of real- world situations, such as online handwriting recognition on computer tabletsor systems, identifying vehicle licence plates, processing bank cheque amounts, and reading numbers from forms that have been filled out by hand (such as tax forms).

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 becauseof 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

Hermans et al. have addressed the MNIST handwritten digit classification problem. In this context, 10 iterations are used for each image in the MNIST dataset; in other words, each input digit is repeated for 10 masking periods. In their experiments, the authors focused on both an MNIST handwritten digit classification dataset, and a TIMIT phoneme classification dataset. In both MNIST and TIMT datasets, the authors found that optimizing the input encoding can make great improvements over random masks.

Mohapatra et al. proposed a new method for classifying MNIST handwritten digit images. In their new method, the authors used the discrete cosine space-frequency transform to extract image features and artificial neural network classifiers to solve the classification problem. In order to reduce the computational cost, the authors proposed to normalize all the images of the MNIST handwritten digit dataset and exclude undesirable boundary pixels.

Kussul and Baidyk proposed a new neural classifier limited receptive area (LIRA) for MNIST handwritten digit images classification. In the LIRA classifier, the sensor layer is followed with the associative layer, and the trainable connections are used to connect the associative layer with the output layer. Experiments with MNIST handwritten digit images show that the LIRA classifier has achieved a classification accuracy of 99.41%.

In order to classify MNIST handwritten digit images, Ahlawata and Choudharyb proposed to build a hybrid classification model by integrating convolutional neural networks and support vector machines (SVM). In this context, the authors used convolutional neural networks to extract the features of the image, while SVM was used as a binary classifier. Based on experimental results the authors have achieved a classification accuracy of 99.28%.

Chazal et al. proposed to use identical network topologies to compare between two weight optimization methods using MNIST handwritten digit classification database. In the first weight optimization methods, the authors use the extreme learning machine algorithm. While backpropagation algorithm is used in the second weight optimization methods. Based on their experimental results, the authors found that the weight optimization method that uses the extreme learning machine is much faster than the one that uses the backpropagation algorithm. Ma and Zhang adopted deep analysis with multi-feature extraction to build a handwritten digit classification method. In order to exclude negative information and maintain relevant features, the images of various sizes were normalized, and projection features were extracted from pre-processed images. Distribution features and projection features are also used to classify MNIST handwritten digit datasets.

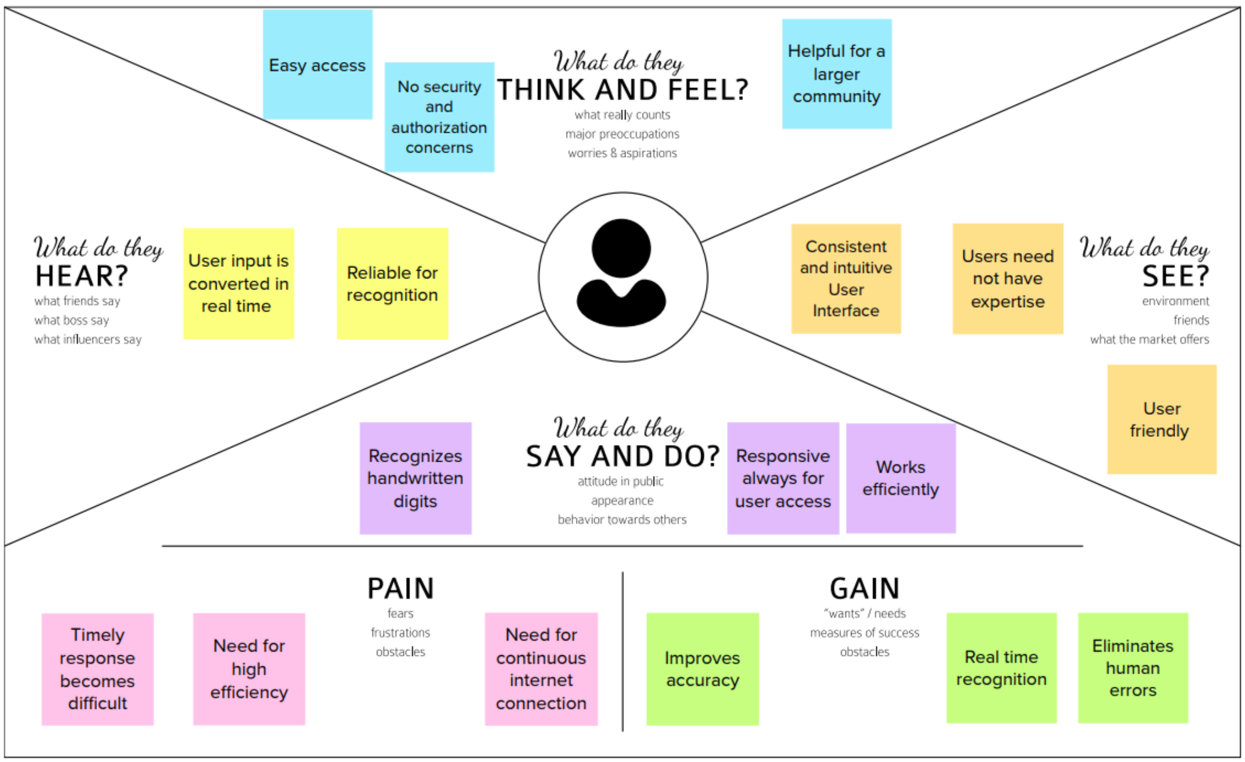
2.3. PROBLEM STATEMENT DEFINITION

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.

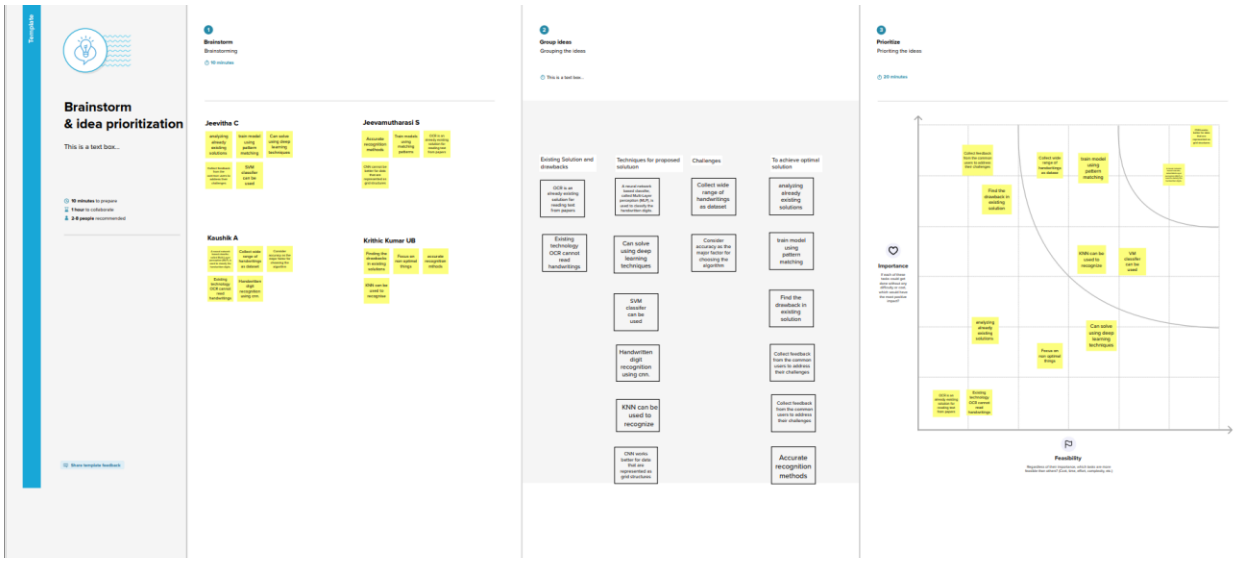
CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1. EMPATHY MAP CANVAS



3.2. IDEATION & BRAINSTORMING



3.3. PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Requirement of a novel method to recognize handwritten digits using Conventional NeuralNetworks (CNN) using the MNIST dataset with higher accuracy and reliability.
2.	Idea / Solution description	The solution is to first normalize the image andpass the normalized image to the ResNet (a variation of CNN) and extract digit specific features. We then passon these features to a fully connected layer to get the classprediction.

3.	Novelty / Uniqueness	Based on an examination of the thickness and form of the numerical picture, it can accurately and efficiently identify the digits.
4.	Social Impact/ Customer Satisfaction	It can be used in traffic signals to find thevehicle number registration in case of violation of trafficrules.
5.	Business Model(Revenue Model)	The accuracy and faster rate at which the digits are recognized helps in reducing human labourcost .
6.	Scalability of the Solution	Ability to recognize continuous numbers in banking and other sectorswhere the accuracy of numbers in highly in demand. Ability to recognize realtime characters andwords recognition.

3.4. PROBLEM SOLUTIONFIT

Problem-Solution fit canvas 2.0

Purpose / Vision: A Novel Method for Handwritten Digit Recognition System

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) — CS

A peíson who needs to íead postal addíesses, bank check amounts, and forms. Also if a person doesn't have proper eye sight he or she cannot read the signatures properly and they cannot be sure about the authenticity of the thing or document which has been signed.

6. CUSTOMER CONSTRAINTS — CC

It is a haíd task foí the machine because handwíitten digits aíe not peífect and can be made with many diffeíent flavoís. Ihe handwíitten digit íecognition is the solution to this píoblem which uses the image of a digit and íecognizes the digit píesent in the image.

5. AVAILABLE SOLUTIONS — AS

The capability of a computer to fell the novel handwritten integers from different sources like images, papers, touch defences

Explore AS, differentiate

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

2. JOBS-TO-BE-DONE / PROBLEMS — J&P

Offline handwriting recognition systems are less accurate than online systems because only spatial information is available for offline systems, while both spatial and temporal information is available for online systems.

9. PROBLEM ROOT CAUSE — RC

It is easy for the human to perform a task accurately by practicing it repeatedly and memorizing it for the next time

7. BEHAVIOUR — BE

Behavioral characteristics through text processing and handwriting recognition, with the objective of in corporating the obtained results with futuristic artificial intelligence systems that can employ text processing and handwriting recognition as individualistic signatory features

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

Identify strong TR & EM

3. TRIGGERS — TR

The live recognition rate highly depends on the digit skew, as automatic de-skewing was not implemented, but manually performed.

4. EMOTIONS: BEFORE / AFTER — EM

Handwriting and signature biometrics have a long history in the literature,especially in terms of identity recognition and/or verification; nevertheless, it reveals more information therefore provides more opportunities for personal characteristics estimation, particularly, emotional state

10. YOUR SOLUTION — SL

We integrated the handwritten recognition model into the full text recognition system by augmenting the script identification model with an additional classification between printed text and handwritten text.

8. CHANNELS of BEHAVIOUR — CH

8.1 ONLINE

Online handwriting recognition involves the automatic conversion of text as it is written on a special digitizer or PDA, where a sensor picks up the pen-tip movements as well as pen-up/pen-down switching.

8.2 OFFLINE

K-NN combined with preprocessing methods is capable of achieving great performance apart from Neural Network when used as a classification algorithm in offline handwritten digit recognition.

Extract online & offline CH of BE

CHAPTER 4
REQUIREMENT ANALYSIS

4.1. FUNCTIONAL REQUIREMENT

FR No:	Functional Requirement and description
FR-1	**ImageData**: Handwritten digitrecognition is the ability of a computer to recognize the human handwritten digits from different sources like images,papers, touch screens, etc, and classify them into 10 predefined classes (0-9). This has been a topicof boundless-research in the fieldof deep learning. In the realmof 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 spaceis allotted to a website on a server. Shared, dedicated, VPS,and reseller hosting are the fourbasic varieties.
FR-3	**Digit_Classifier_Model:** To train a convolutional network to predict the digit from an image, use the MNISTdatabase of handwritten digits. get the training and validation data first.
FR-4	**MNIST dataset:** The MNIST dataset is an acronym that stands for the Modified National Institute of Standards and Technology 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.
FR-5	databases, software, virtual storage, and networking, among others. In layman's terms, Cloud Computing is defined as a virtual platform that **allows you to store and access your data over the internet without any limitations**.

4.2. NON FUNCTIONAL REQUIREMENTS

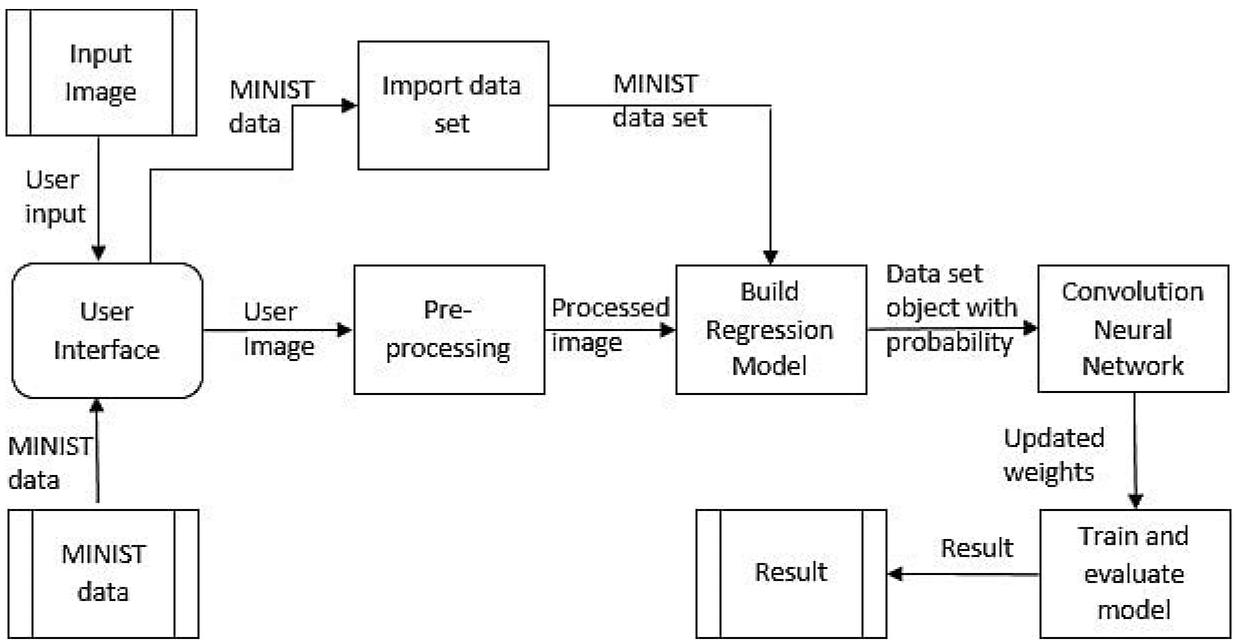
NFR No.	Non-Functional Requirement
NFR-1	**Usability:** Handwritten character recognition is one of the practically important issuesin pattern recognition applications. The applications of digit recognition include postal mail sorting, bank check processing, form data entry, etc. One of the very significant 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	**Reliability:** 1. The system not only produces a classification of the digit but also a rich description of the instantiation parameters which can yieldinformation such as the writing style. 2. The generative models can perform recognition driven segmentation. 3. The methodinvolves a relative.
NFR-3	**Performance:** The neural networkuses the examples to automatically inferrules for recognizing handwritten digits. Furthermore, by increasing the number of training examples, the network can learn more about handwriting, and so improveits accuracy. There are a numberof ways and algorithms to recognize handwritten digits, including Deep Learning/CNN, SVM, Gaussian NaiveBayes, KNN, Decision Trees,Random Forests, etc.

NFR-4	**Accuracy:** Optical Character Recognition (OCR) technology provides higher than 99% accuracy with typed characters in high qualityimages. However, the diversity in human writing types, spacing differences, and irregularities of handwriting causes less accurate character recognition.

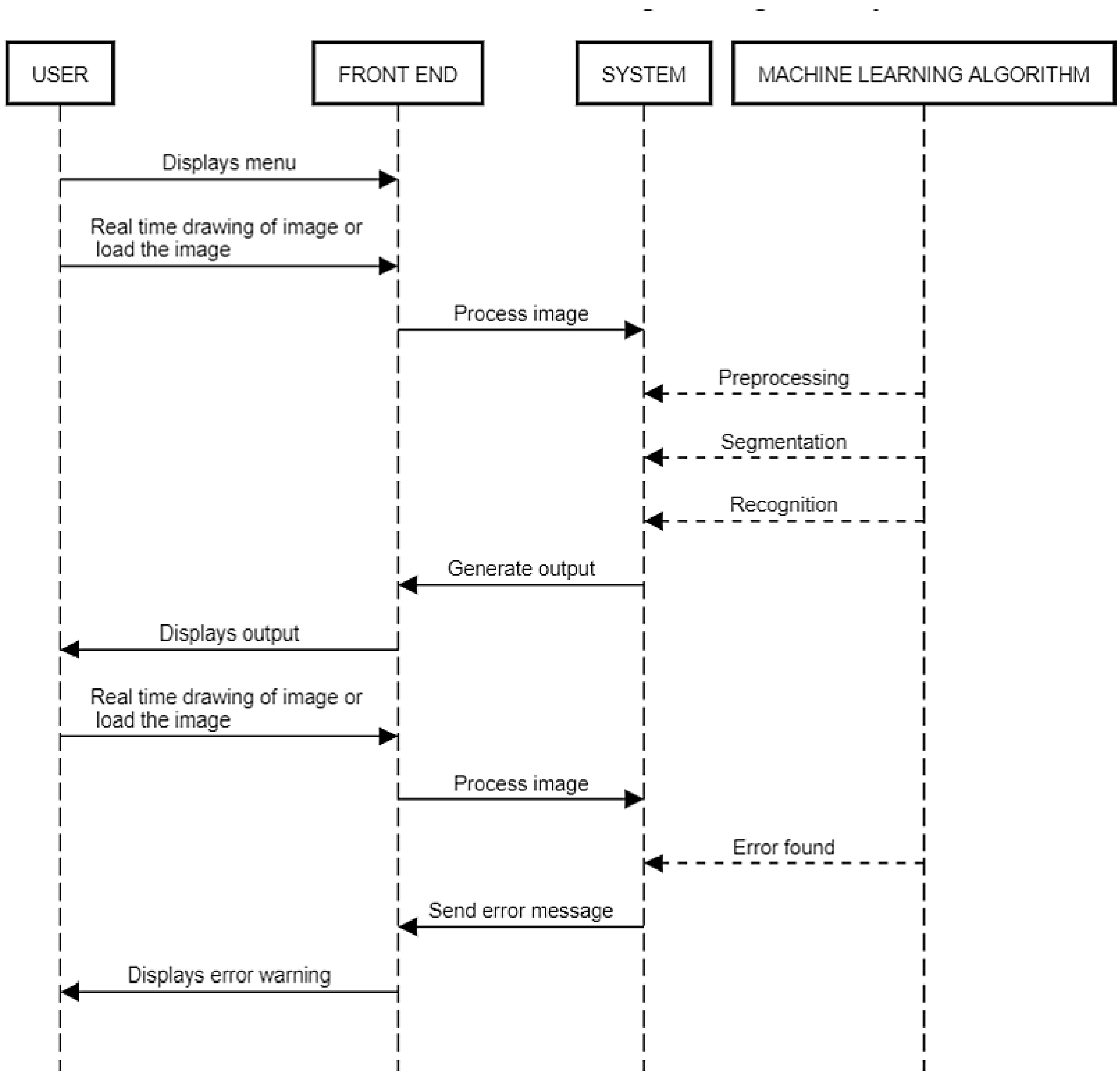
CHAPTER 5

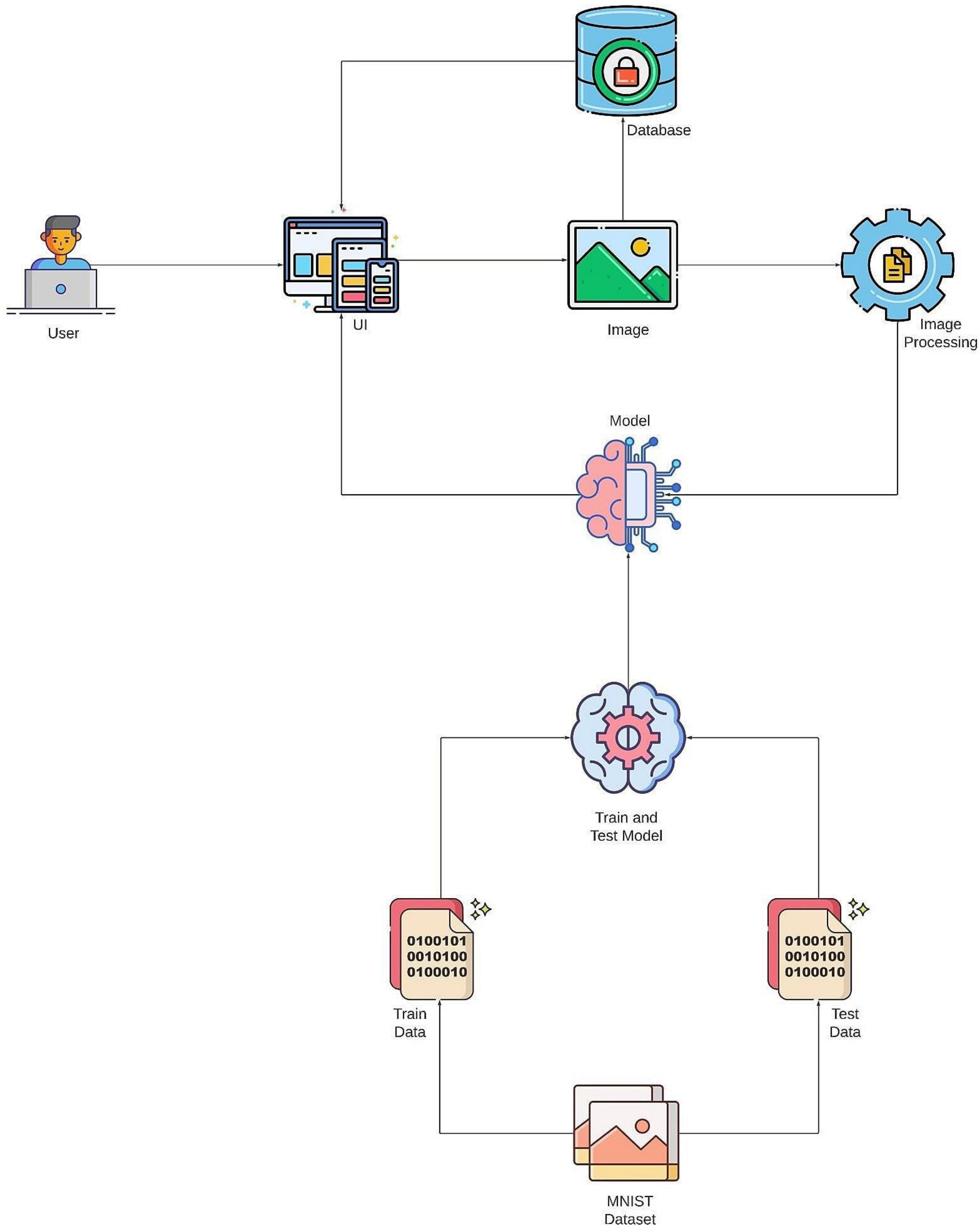
PROJECT DESIGN

5.1. DATA FLOW DIAGRAM



5.2. **SOLUTION AND TECHNICAL ARCHITECTURE**




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**

UserType	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Any common individual		USN-1	Receiving the digital form of the handwritten digits with a very high accuracy	Either write it on the webpage or scan the image of the written digit	High	Sprint-1
Bank officials	Separate registration	USN-2	Helps in understanding the amount and account number entered in demand draft and cheques in banks	Useful in characterizin g thedigits in banks	High	Sprint-2
Customer (Web user)	Home	USN-3	As a user, I can view the guideto use the webapp	I can view the awareness of this application and its limitations.	Low	Sprint-1
		USN-4	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-1
Any comm on person	Login	USN-5	As a user, I can log into the application by entering email& password		Low	Sprint-1

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 withdifferent handwritings.	10	Low	Jeevamutharasi S Kaushik A
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	Jeevitha C Krithic Kumar UB
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	Jeevamutharasi S Jeevitha C
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High	Kaushik A Krithic Kumar UB

Sprint-2	Compiling themodel	USN-5	With both the training data defined and model defined, it's time to configure the learning process.	2	Medium	Jeevitha C Kaushik A
Sprint-2	Train &test the model	USN-6	As a user, letus train our model with our image dataset.	6	Medium	Jeevamutharasi S Krithic Kumar UB
Sprint-2	Save the model	USN-7	As a user, the model is saved & integrated with an androidapplication or web application in order to predict something.	2	Low	Jeevitha C Krithic Kumar UB
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	Jeevamutharasi S Kaushik A
Sprint-3		USN-9	As a user, I can know the details of the fundamental usage of the application.	5	Low	Jeevamutharasi S Krithic Kumar UB

Sprint-3		USN-10	As a user, I can see the predicted / recognized digits in the application.	5	Medium	Jeevitha C Kaushik A
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	Jeevamutharasi S Jeevitha C
Sprint-4	Cloud Deployment	USN-12	As a user, I can access the webapplication and makethe use of the product from anywhere.	10	High	Kaushik A Krithic Kumar UB

6.2. SPRINT DELIVERYSCHEDULE

SPRINT	TOTAL STORY POINTS	DURATION	SPRINT START DATE	SPRINT END DATE (PLANNED)	STORY POINTS COMPLETED (AS ON PLANNED DATE)	SPRINT RELEASEDATE (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 Nov 2022	19 Nov 2022	9	19 Nov 2022

CHAPTER 7

CODING & SOLUTIONING

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from PIL import Image, ImageOps
from keras.utils import np_utils
from tensorflow.keras.datasets import mnist
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import Conv2D, Dense, Flatten
```

+ Code + Markdown

```
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 0s 0us/step
```

```
print(X_train.shape)
print(X_test.shape)
```

```
(60000, 28, 28)
(10000, 28, 28)
```

```
plt.imshow(X_train[0])
```

```python
def random_name_generator(n: int) -> str:
    return "".join(random.choices(string.ascii_uppercase + string.digits, k=n))


def recognize(image: bytes) -> tuple:
    model = load_model(Path("./model/model.h5"))

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

    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}"))

    img = ImageOps.grayscale(img)
    img = ImageOps.invert(img)
    img = img.resize((28, 28))

    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]

    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))

    best = others.pop(best)
```

CHAPTER 8

TESTING

8.1. TEST CASES

Testcase ID	Feature Type	Component	Test Scenario	Expected Result	ActualResult	Status
HP_TC_001	UI	HomePage	Verify UI elements in the Home Page	The Home page mustbe displayed properly	Working as expected	PASS
HP_TC_002	UI	HomePage	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 inscreen size 2560 x 1801 and 768 x 630	FAIL
HP_TC_003	Functional	HomePage	Check if user can upload theirfile	The inputimage should be uploaded to application successfully	Working as expected	PASS
HP_TC_004	Functional	HomePage	Check if user cannot upload unsupported files	The application should not allow userto select a non imagefile	User is able to upload any file	FAIL
HP_TC_005	Functional	HomePage	Check if page redirects result pageonce input is given	The page should redirect to the results page	Working as expected	PASS
BE_TC_001	Functional	Backend	Checkif 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 imagesizes	The modelshould rescale theimage and predict the results	Working as expected	PASS
M_TC_002	Functional	Model	Check if the model predicts the digit	The model shouldpredict the number	Working as expected	PASS
M_TC_003	Functional	Model	Checkif the modelcan handlecomplex input image	The modelshould 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 theinput image exceeds the display container	FAIL
RP_TC_003	UI	Result Page	Check if the resultis displayed properly	The result shouldbe displayed properly	Working as expected	PASS
RP_TC_004	UI	Result Page	Checkif 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
Fixe d	4	1	0	1	6
NotReproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won’tFix	1	0	1	0	2
Total	6	1	4	3	14

8.2.2 TEST CASE ANALYSIS

Section	Tota l Cases	Not Teste d	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

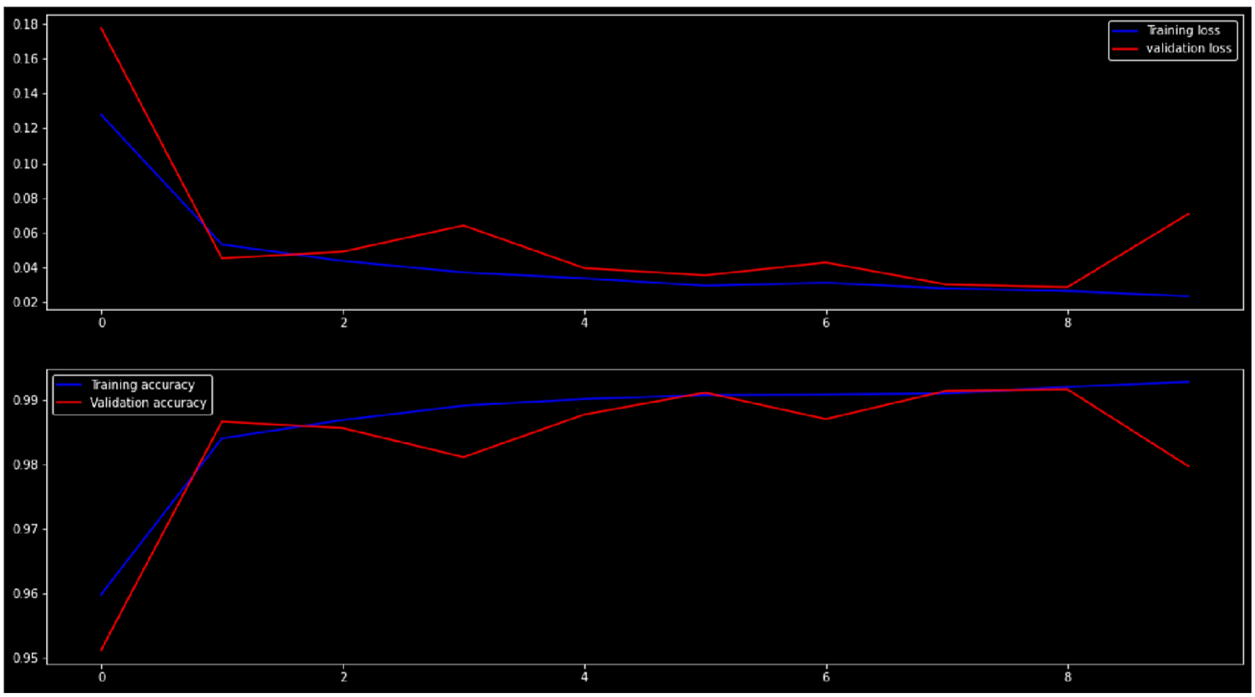
9.1 PERFORMANCE METRICS

9.1.1 MODEL SUMMARY

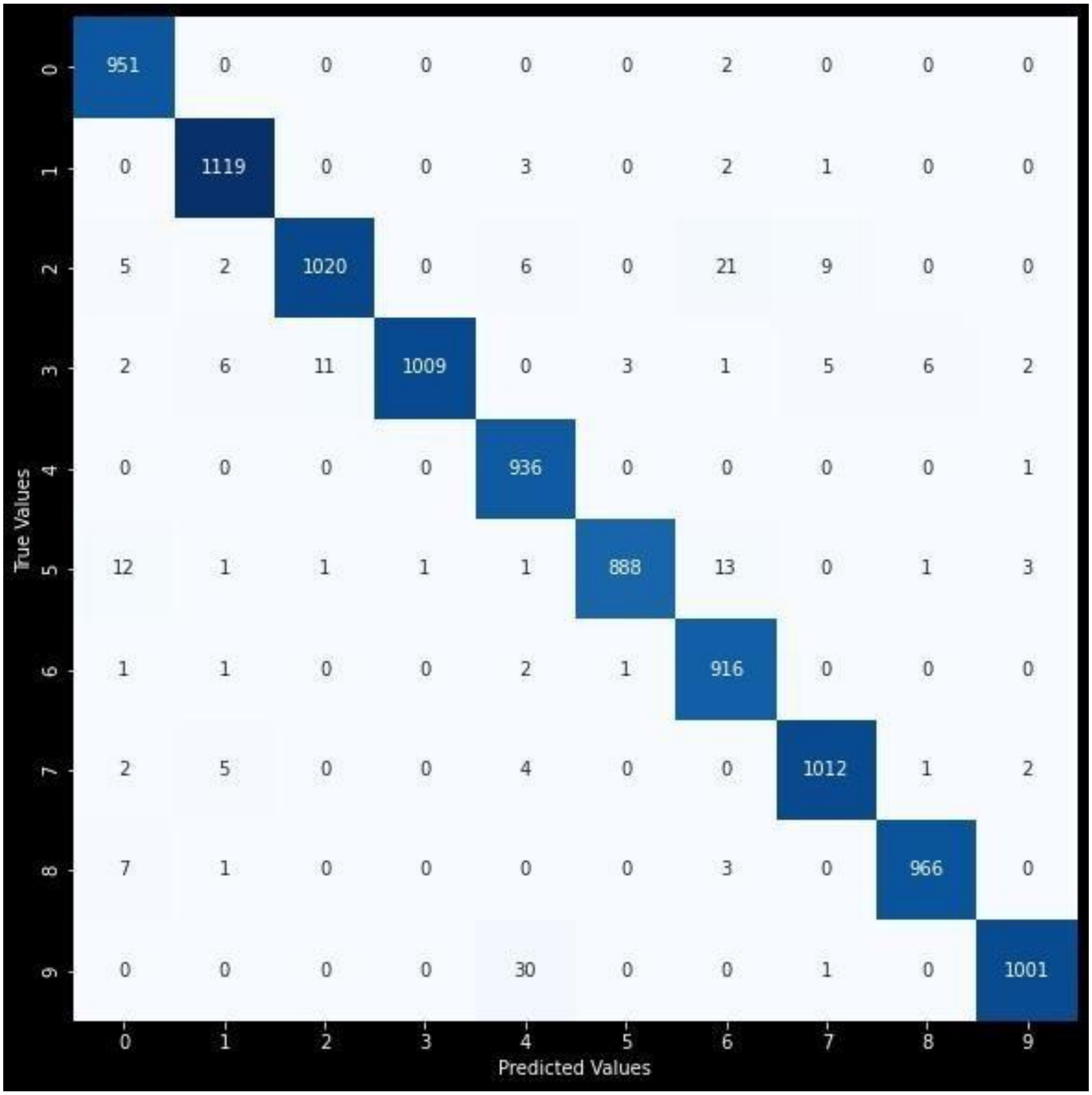
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 64)        640

 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 %



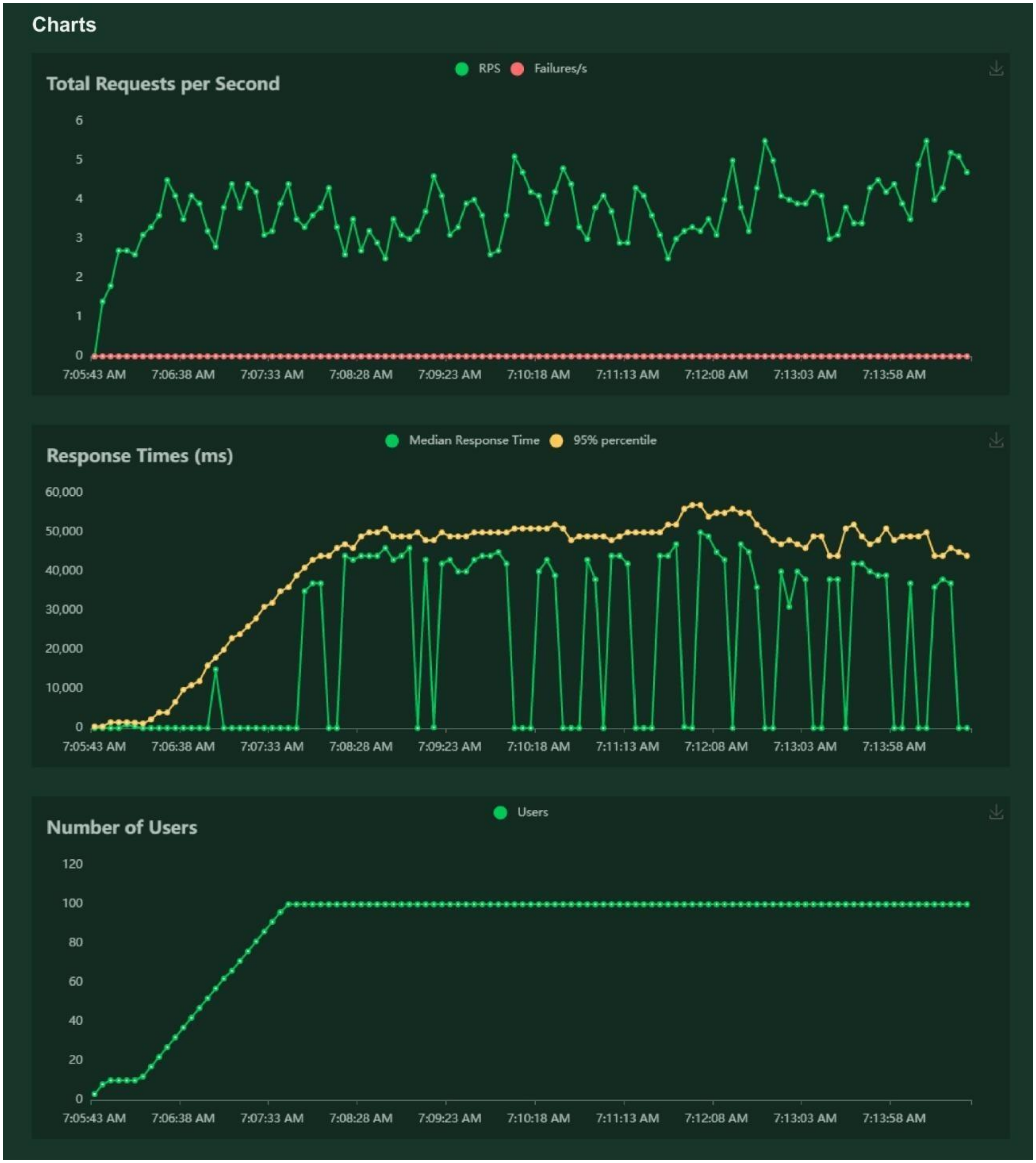
9.1.3 CONFUSION MATRIX



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**

CHAPTER 10

ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Can be used anywhere from any device
- Manual work can be reduced
- Lot of data can be added
- Tends to be more accurate than humans

DISADVANTAGES

- Complex data cant be handled
- Digital format is expected
- Requires a fast server
- Occasional errors might occur

CHAPTER 11

CONCLUSION

This project shows and suggests a web app that uses machine learning to identify handwritten numbers. The tech stack used here for the project are Flask, HTML, CSS, JavaScript. CNN network model is used to predict the handwritten digits. The model achieved a 99.61% recognition rate during the testing. This project is scalable and can easily handle a huge number of users.

This system is compatible with any device that can run a browser since it is a web application. 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.
In subsequent versions much more improvement can be made.

CHAPTER 12

FUTURE SCOPE

This project has a lot of room for improvement and improvements can be made in the next versions. Some of the improvements that can be made to this project are:

- Multiple digits detection support can be added.
- Model to detect digits from complex images can be improved.
- Add support to detect from digits multiple images and save the results
- Multilingual support can be added to help users from all over the world

Implementing this concept in the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency. This system has endless advancement in the next versions and can always be improved to be better than this.

APPENDIX

SOURCE CODE

PYTHON AND FLASK CODE FILES

```python
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()
```

```python
import os
import random
import string
import numpy as np

from pathlib import Path
from PIL import Image, ImageOps
from tensorflow.keras.models import load_model


def random_name_generator(n: int) -> str:
    return "".join(random.choices(string.ascii_uppercase + string.digits, k=n))


def recognize(image: bytes) -> tuple:
    model = load_model(Path("./model/model.h5"))

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

    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}"))

    img = ImageOps.grayscale(img)
    img = ImageOps.invert(img)
    img = img.resize((28, 28))

    img2arr = np.array(img)
    img2arr = img2arr / 255.0
    # img2arr = img2arr.reshape(1, 28, 28, 1)
    img2arr = img2arr.reshape(1, 784)

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

    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))

    best = others.pop(best)

    return best, others, img_name
```

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from PIL import Image, ImageOps
from keras.utils import np_utils
from tensorflow.keras.datasets import mnist
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import Conv2D, Dense, Flatten
```

Load the data

```python
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 0s 0us/step
```

Data Analysis

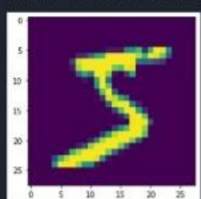
```python
print(x_train.shape)
print(x_test.shape)
```

```
(60000, 28, 28)
(10000, 28, 28)
```

```python
plt.imshow(x_train[0])
```

```
<matplotlib.image.AxesImage at 0x7f287692dd10>
```

```
<matplotlib.image.AxesImage at 0x7f287692dd10>
```



Data Pre-Processing

```python
x_train = x_train.reshape(60000, 28, 28, 1).astype('float32')
x_test = x_test.reshape(10000, 28, 28, 1).astype('float32')
```

```python
n_classes = 10
y_train = np_utils.to_categorical(y_train, n_classes)
y_test = np_utils.to_categorical(y_test, n_classes)
```

```python
y_train[0]
```

```
array([0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)
```

Create model

```python
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(n_classes, activation="softmax"))
```

Test the saved model

```python
model=load_model("model.h5")
```

```python
img = Image.open("/content/sample_data/sample.png").convert("L")
img = np.array(img).reshape(1, 28, 28, 1)

res  = model.predict(img)
res = np.argmax(res, axis = 1)
res = pd.Series(res, name="Label")
print(res)
```

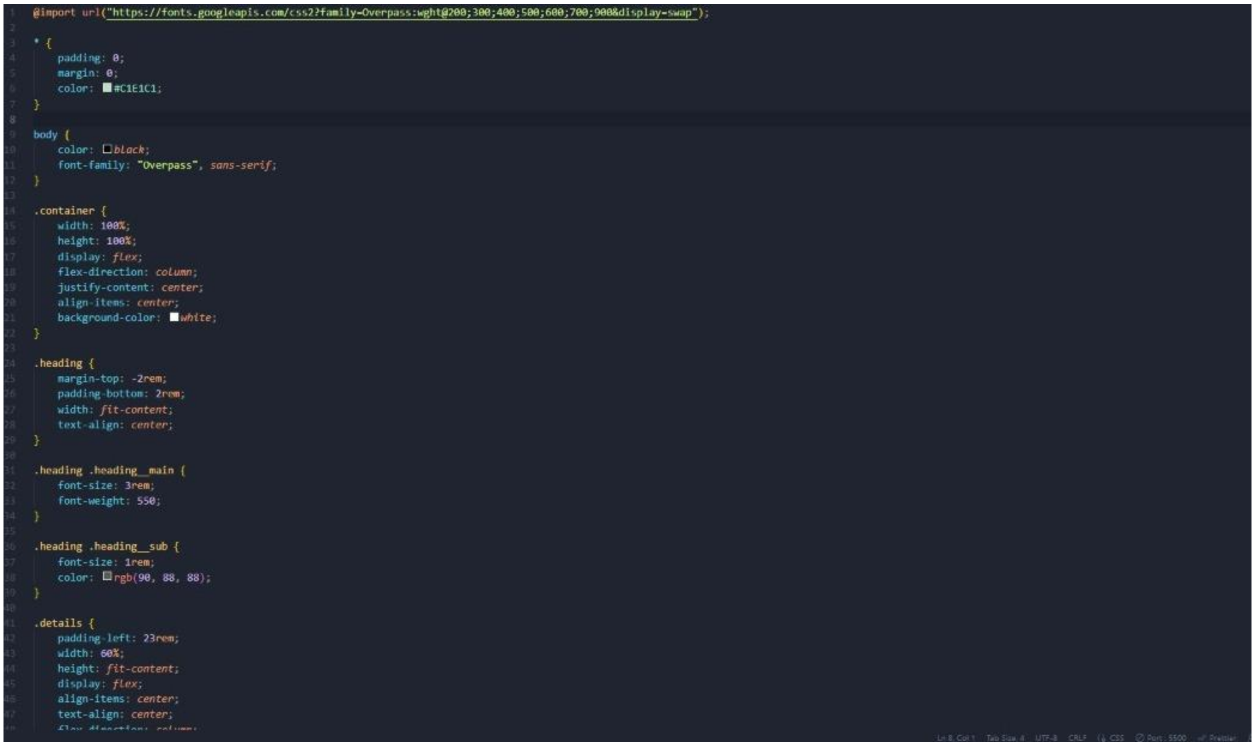
```
1/1 [==============================] - 0s 19ms/step
0    8
Name: Label, dtype: int64
```

HTML FILES

```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>
                <br>
                <br>
                <div class="details">
                    <h4>Team ID: PNT2022TMID02158</h4>
                    <h4>Student Names: Jeevitha C, Jeevamutharasi S, Kaushik A,
                        Krithic Kumar U B</h4>
                    <h4>
                        Project Description: Handwritten digit recognition has recently been of very
                        interest among the researchers because of the evolution of various Machine
                        Learning, Deep Learning and Computer Vision algorithms. This is highly useful in
                        finding the numbers that are written using hands. The illegible handwritten
                        digits are converted to their respective digits.
                    </h4>
                </div>
                <br>
                <br>
            </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>
```

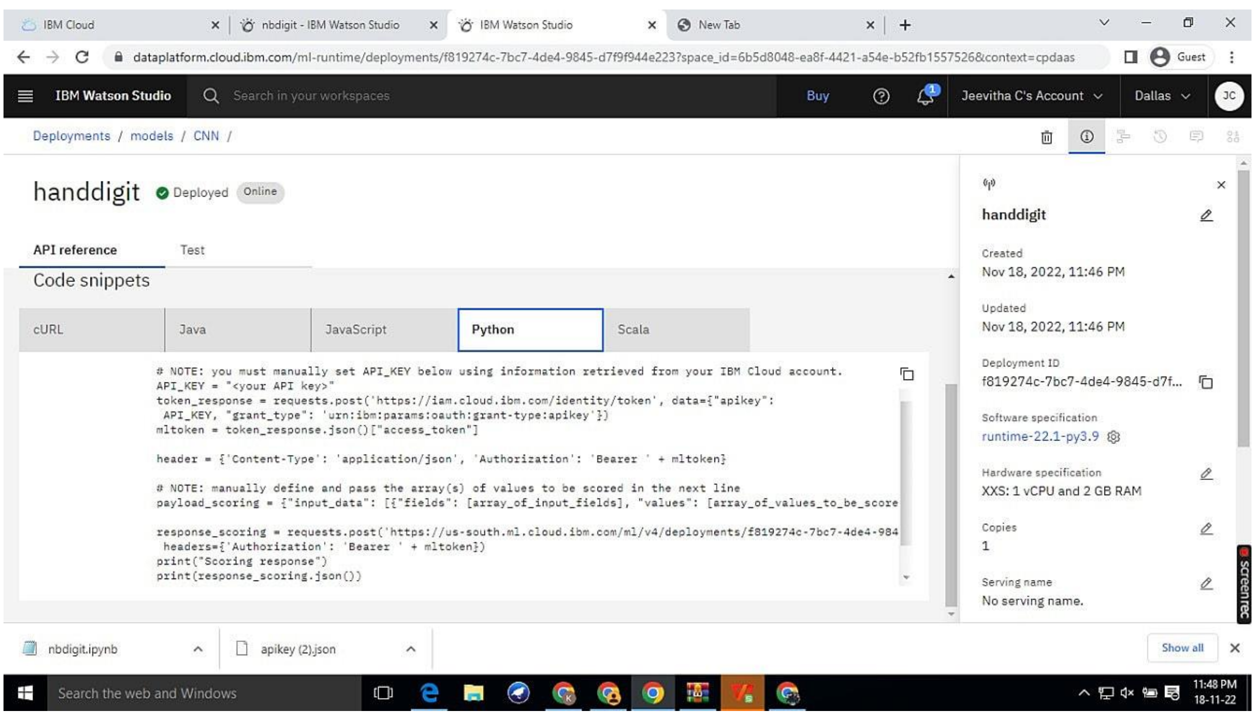
CSS FILES

```css
.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;
    }
}
```



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

* {
    padding: 0;
    margin: 0;
    color: #C1E1C1;
}

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

.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);
}

.details {
    padding-left: 23rem;
    width: 60%;
    height: fit-content;
    display: flex;
    align-items: center;
    text-align: center;
```

TEST THE MODEL ON IBM CLOUD



GITHUB PROFILE

https://github.com/IBM-EPBL/IBM-Project-7452-1658857275

PROJECT DEMO VIDEO LINK

https://drive.google.com/file/d/1_COaKs-uLwqjiPDuPa35znMKL1pmti-8/view?usp=drivesdk