PROJECT

A NOVEL METHOD FOR HANDWRITTEN

DIGIT credit

Submitted By

Team ID : PNT2022TMID09888

Team Size : 4
Team Leader : Nishalini G
Team member : Archana S
Team member : Preethy Mariya Mathew
Team member : Savitha V

TABLE OF CONTENTS

CHAPTER 1

unveiling

a. PROJECT OVERVIEW

Machine learning deep learning play an important role in computer applied science and artificial intelligence. With the use of deep learning machine learning can be reduced in recognize, predictions and in many more areas.

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

b. PURPOSE

Digit recognition systems are capable of recognizing the digits fromdifferent sources like emails bank quench, 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 quench amounts, numeric entries in forms filled up by hand (tax forms)and so on.

CHAPTER 2

LITERATURE SURVEY

a.EXISTING PROBLEM

The fundamental problem with handwritten digit recognition is thathandwritten 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 handwritinginfluence the structure and appearance of the digits.

REFERENCES

Improved Handwritten Digit Recognition Using Convolution Neural Networks(CNN) (2020)

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 convolution 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%. it is made through the trials.abundantly evident how the performance of handwritten digit recognition is affected by the number of convolution layers in CNN architecture.

According to the paper, evolutionary algorithms can be explored for optimizing convolution filter kernel sizes, CNN learning parameters, and the quantity of layers and learning rates.

PROJECT

An Efficient And Improved Scheme For Handwritten Digit Recognition Based On Convolution Neural Network (2019)

This study uses rectified linear units (RELU) activation and a convolution neural network (CNN) that incorporates the Deep learning 4J (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 convolution layers, the first with32 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 lathe 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 Neighbour Algorithm (2019)

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 paperoffered 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 DeepLearning Algorithms(2021)

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 basedon their 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 picturedata. 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 limit of a certainmodel and they

discovered that beyond a certain number of epochs ,the model begins over-fitting the dataset and provides based predictions.

a.PROBLEM STATEMENT DEFINITION

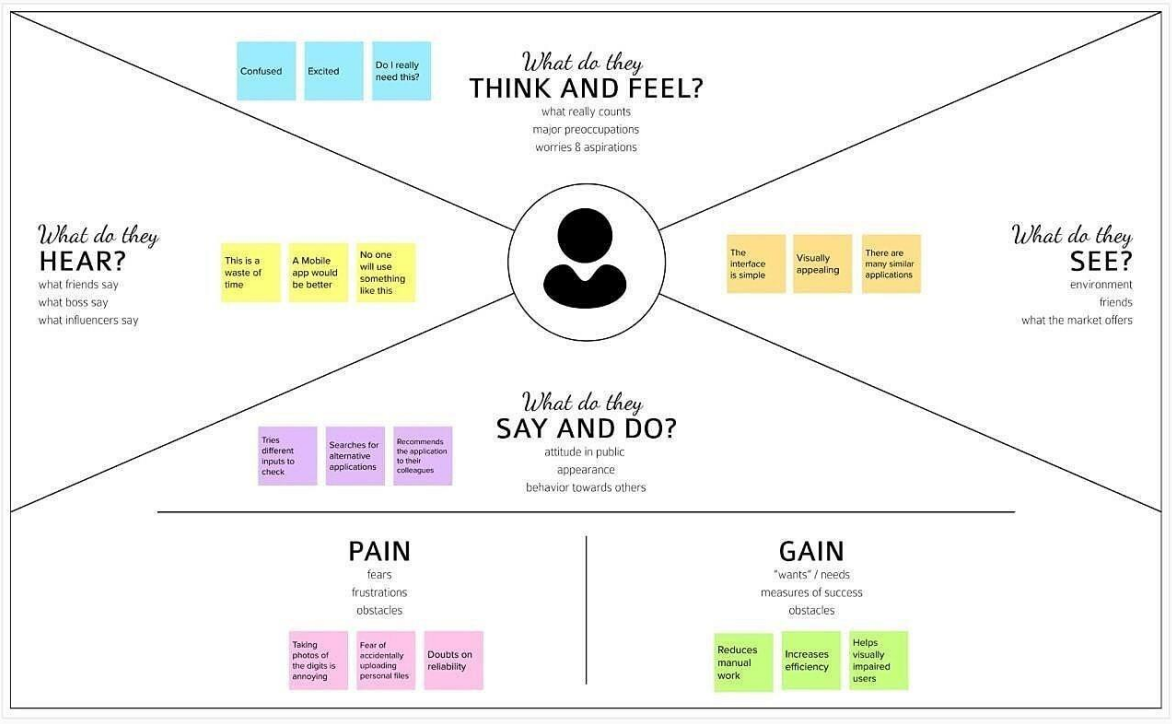
For years, the traffic department has been combating traffic law violators.These offenders endanger not only their own lives, but also the lives of other individuals. Punishing these offenders is critical to ensuring that others do not become like them. Identification of these offenders is next to impossible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

PROJECT

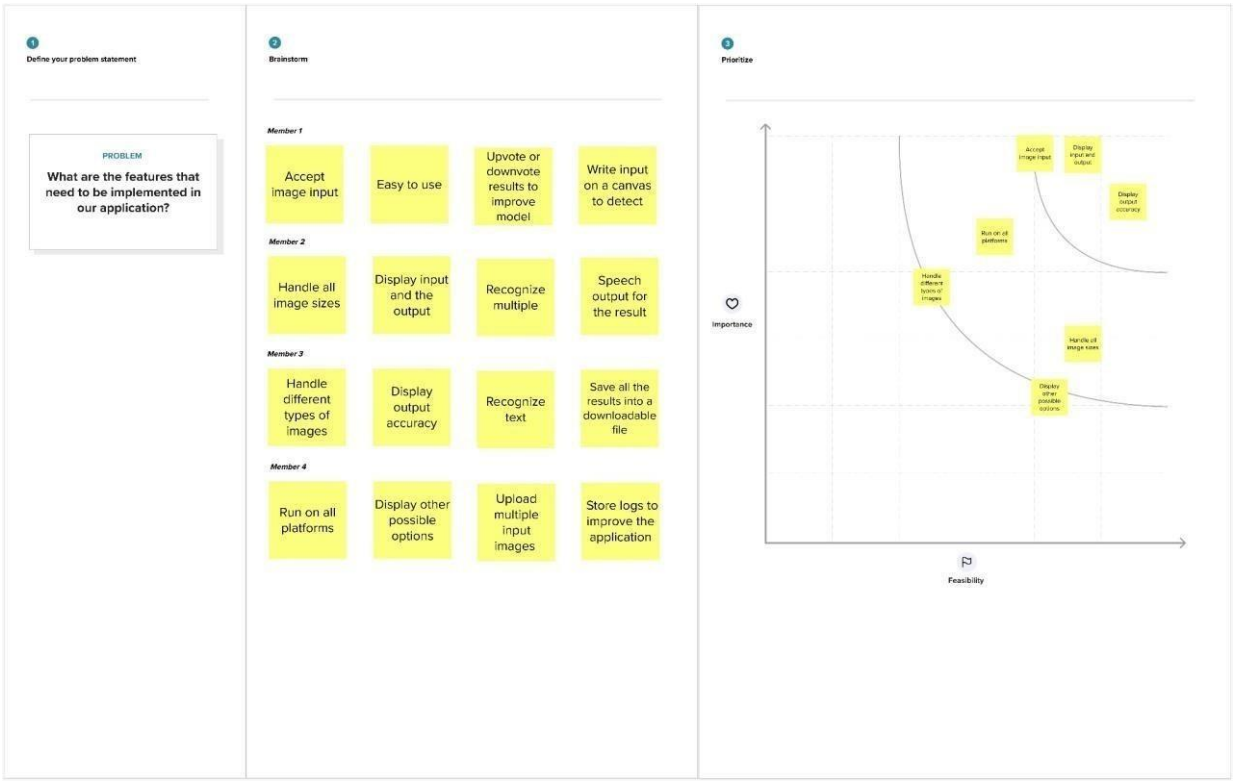
CHAPTER 3

IDEATION AND PROPOSEDSOLUTION

3.1 EMPATHY MAP CANVAS



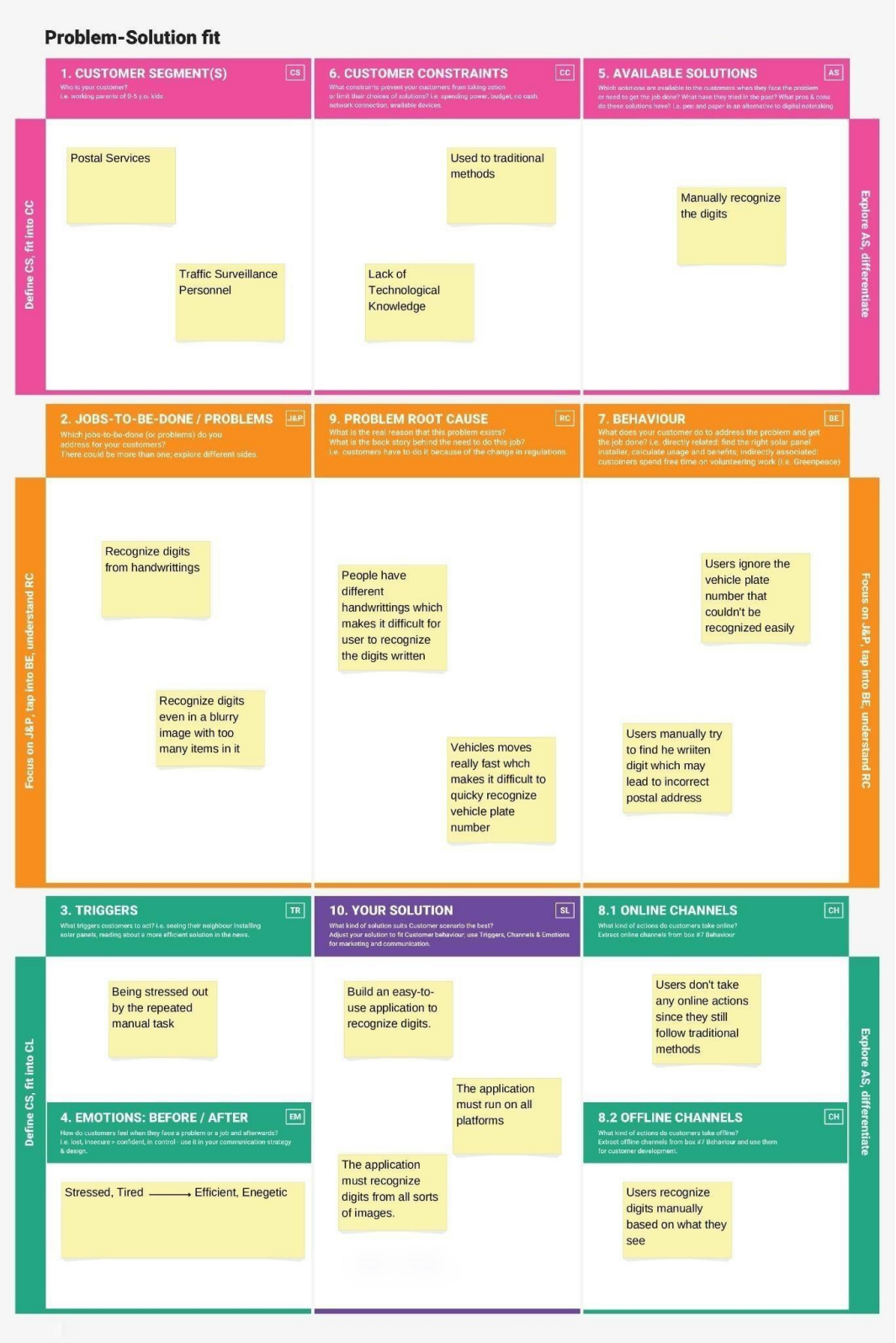
3.2 IDEATION & BRAINSTORMING



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 / CustomerSatisfaction	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 systems to recognize the pin codes effectively
6	Scalability of the Solution	The application can easily be scaled to accept multiple inputs and process them parallel to increase efficiency further

3.4 PROBLEM SOLUTION FIT



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 recognizehandwritten digits.
		Create a homepage 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 the irrespective accuracy

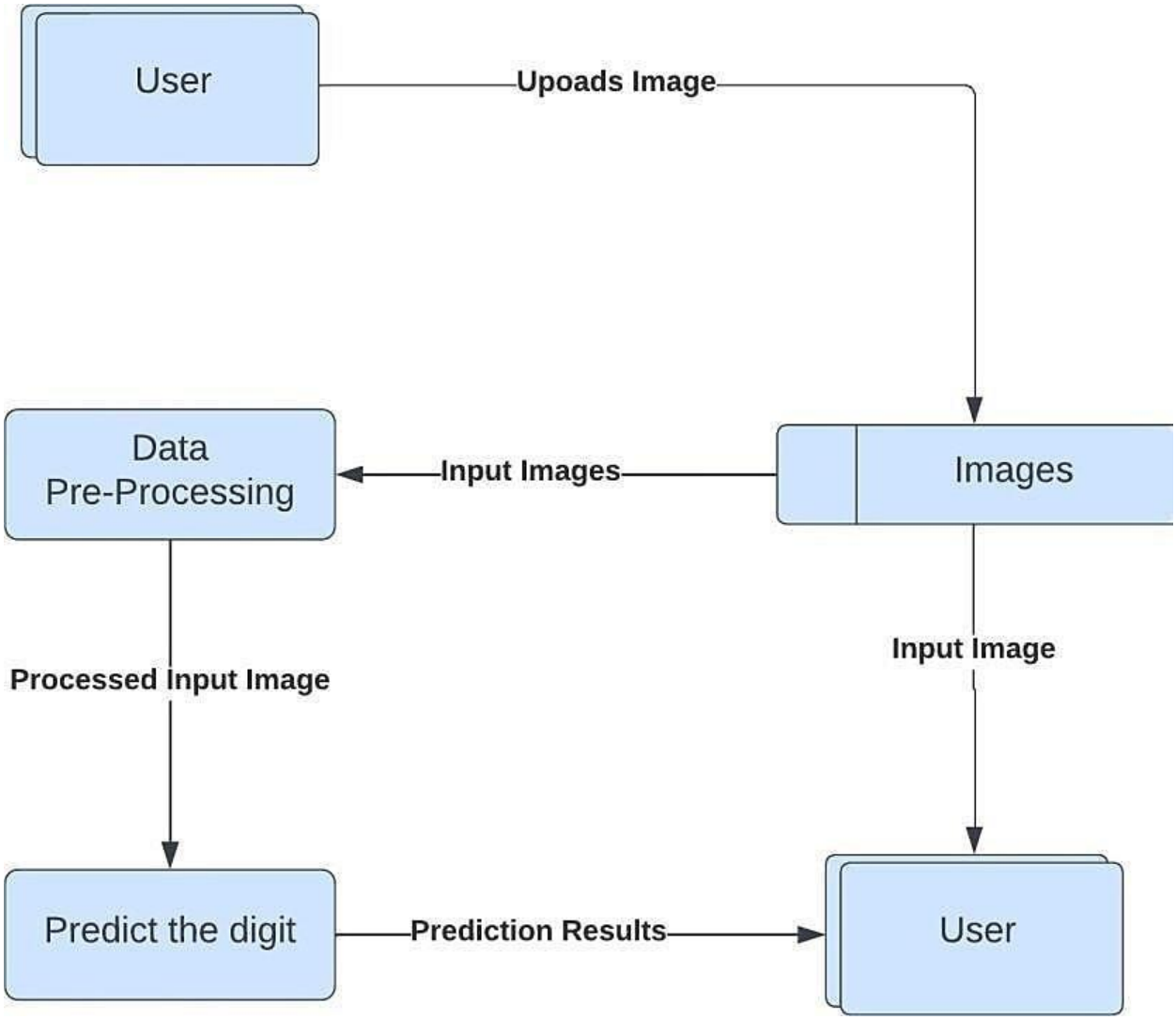
4.2 NON FUNCTIONAL REQUIREMENTS

NFR	NON-FUNCTIONAL REQUIREMENTS	DESCRIPTION
NFR-1	Usability	The application must be usable in all devices
NFR-2	protection	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	handiness	The application must be available to use all the time
NFR-6	Scalability	The application must scale along with the user base

CHAPTER 5

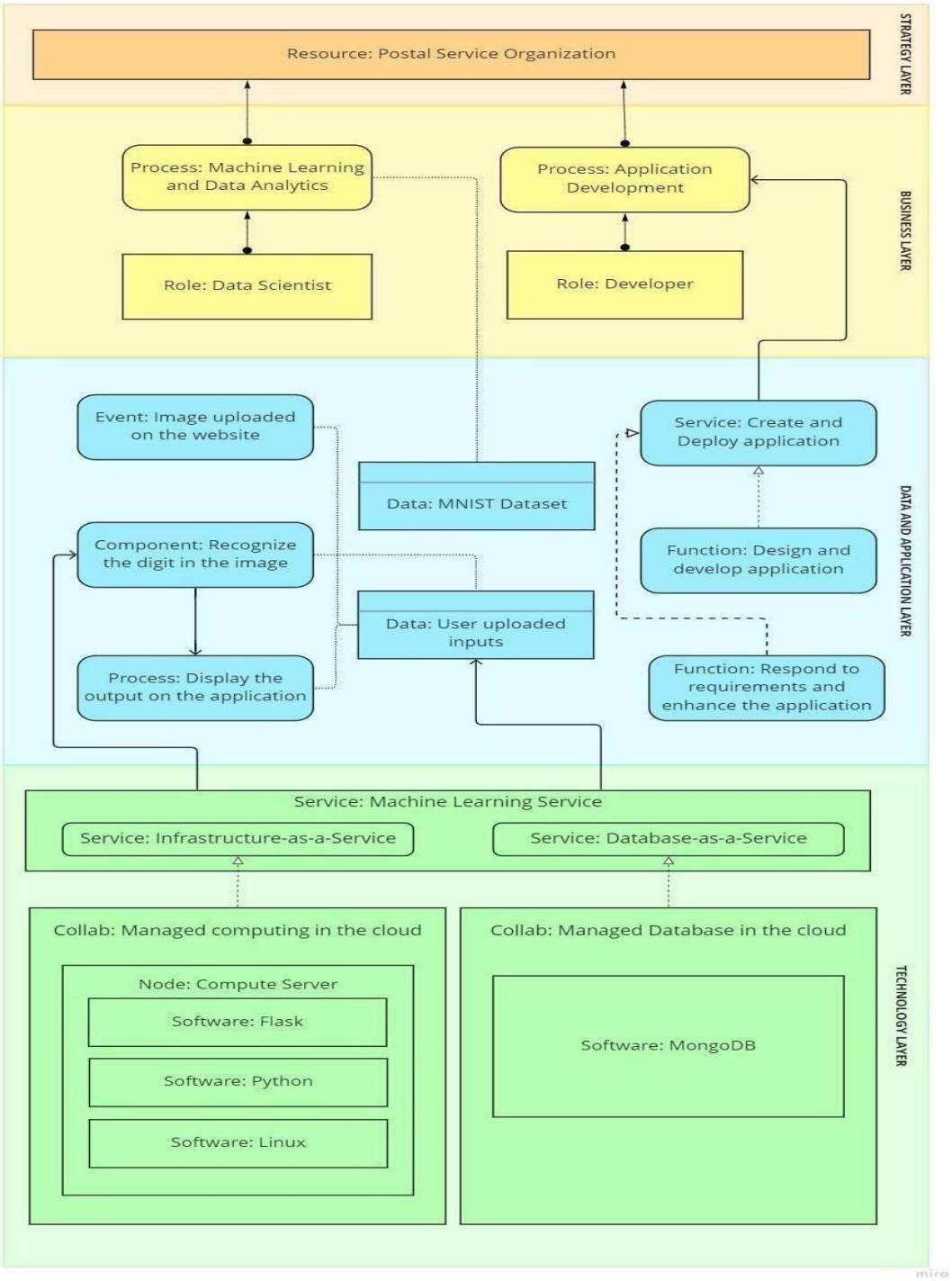
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

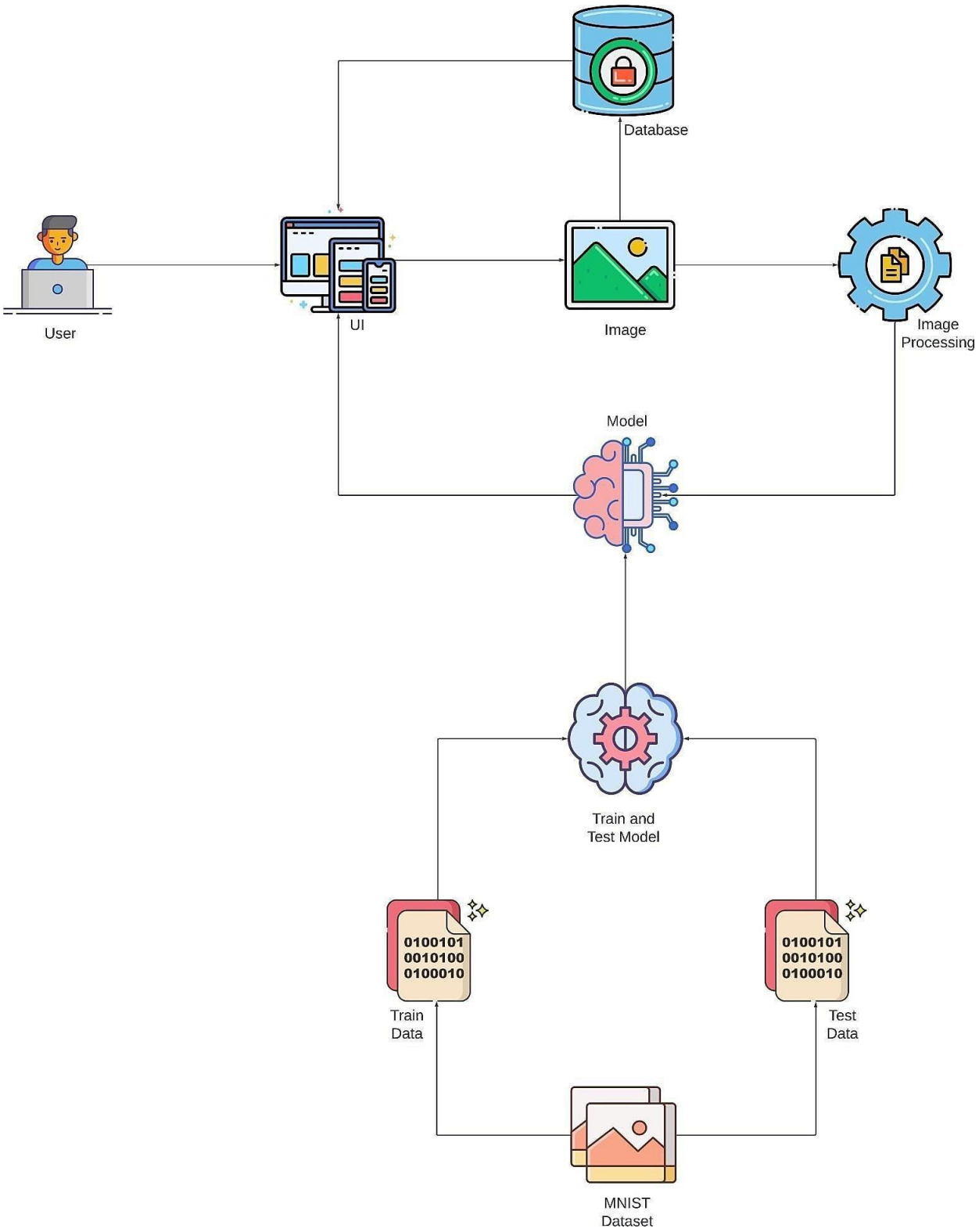


5.2 SOLUTION & TECHNICAL ARCHITECTURE

PROJECT


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
Node: Compute Server
Software: Flask
Software: Python
Software: Linux
Collab: Managed Database in the cloud
Software: MongoDB
miro

PROJECT


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 toaccess 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 uploadimages to predict the digits	User can upload images	High	Sprint-3
	Viewing the Results	USN-3	As a user, I should be able toview 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 predictios	The accuracy of other values must be displayed	Medium	Sprint-4
	Usage Instruction	USN-5	As a user, I should have a usage instruction toknow how to use the applicatio n	The usage instruction is displayed on the home page	Medium	Sprint-4

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

7 Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	5	Low	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-1	Sign up page	USN-2	As a user, I will receive confirmation email once I have registered for the application	4	High	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password	6	High	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-1	Data collection	USN-4	As a user, I need to collect the data with different hand writing to train the model	8	Medium	Nishalini, Archana, Preethy Mariya Mathew, Savitha

PROJECT

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data preprocessing	USN-5	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test	6	Medium	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-2	Model Building	USN-6	As a user, I will get an application with DL model which provides high accuracy of recognized handwritten digit.	5	High	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-2	Add the CNN layers	USN-7	Add input convolutional layer, max-pooling layer, flatten, hidden and output layers to the model	4	Medium	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-2	Compile the model	USN-8	As a user, compile the model for trained dataset.	12	Low	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-2	Train and test the model	USN-9	As a user, train and test the model for the dataset collected and data are validated	8	High	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-2	Save the model	USN-10	As a user, the compiled data are saved and integrated with web application	6	Medium	Nishalini, Archana, Preethy Mariya Mathew, Savitha

PROJECT

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Building user interface application	USN-11	As a user, upload the input image that contains handwritten digits	10	High	Nishalini, Archana, Preethy Mariya Mathew, Savitha
		USN-12	As a user, I can provide the fundamental details about the usage of application to the customer	8	Low	Nishalini, Archana, Preethy mariya Mathew, Savitha
		USN-13	As a user, I can see the predicted or recognized digits in the application	6	Medium	Nishalini, Archana, Preethy Mariya Mathew, Savitha
Sprint-4		USN-14	As a user, train the model in IBM cloud and integrate the result	20	High	Nishalini, Archana, Preethy Mariya Mathew, Savitha

PROJECT

7.1 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	29Oct 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
# 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))
```

PROJECT

```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 case ID	Feature Type	Component	Test Scenario	Expected Result	ActualResult	Status
HP_TC_001	UI	Home Page	Verify UI elements inthe 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 usercan upload their file	The input imageshould be uploaded to the application successfully	Working as expected	PASS
HP_TC_004	Functional	Home Page	Check if user cannot upload unsupportedfiles	The application should not allow user to select anon 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 theinput is given	The page should redirect to the results page	Working as expected	PASS

PROJECT

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 re scale the image and predict the results	Working as expected	PASS
M_TC_002	Functional	Model	Check if themodel predicts thedigit	The model should predict the number	Working as expected	PASS
M_TC_003	Functional	Model	Check if the model can handle complex inputimage	The model should predict the number in the complex image	The model failsto identify thedigit since themodel is not built to handlesuch data	FAIL
RP_TC_001	UI	Result Page	Verify UI elements inthe Result Page	The Result page must be displayed properly	Working as expected	PASS
RP_TC_002	UI	Result Page	Check if the input image isdisplayed 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 theresult is displayed properly	The result should be displayed properly	Working as expected	PASS

RP_TC_004	UI	Result Page	Check if theother predictions are displayedproperly	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	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

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

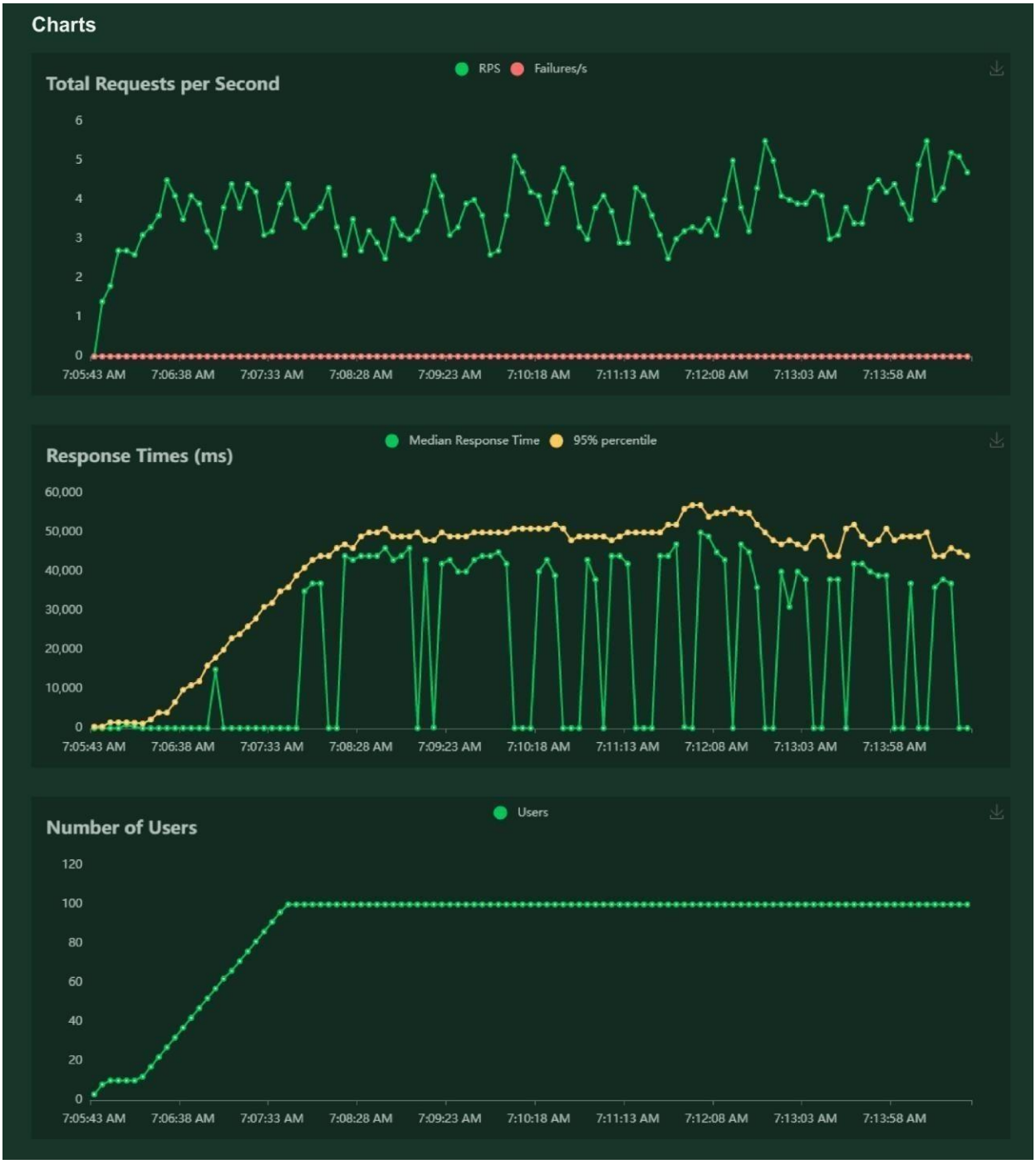
Request Statistics

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**

PROJECT


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 useful in real-world scenarios such as recognizing number plates of vehicles, processing bank quench amounts, numeric entries in forms filled up byhand (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 substantial amount 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

```
# 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")
```

PROJECT

```
# 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()
```

RECOGNIZOR

```
# 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
```

PROJECT

```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
```

PROJECT

HOME PAGE (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)

PROJECT

```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;
}
```

PROJECT

```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;
}
```

PROJECT

```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;
    }
}
```

PROJECT

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>
```

PROJECT

```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);
}
```

PROJECT

```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;
    }
}
```

PROJECT

GITHUB

https://github.com/IBM-EPBL/IBM-Project-14776-1659589755.git