PROJECT REPORT

A NOVEL METHOD FOR

HANDWRITTEN DIGIT RECOGNITION

submitted by

PNT2022TMID20912

Sri Vinayaga Barathan - 412419205088

Jaysuriya N - 412419205032

Arunachalam PL - 412419205009

Udhayakumar G - 412419205097

Keshav Kumar P - 412419205042

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 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)

Ayush Kumar Agrawal and Vineet Kumar Awasthi

An artificial neural network has one hidden layer between the input and output layers, whereas a deep neural network has numerous hidden layers with input and output layers. Deep neural networks use several hidden layers to increase model performance and achieve higher accuracy compared to accuracy of machine learning models. Most researchers do their research in the area of pattern recognition. In the field of pattern recognition, there are many patterns that can be used, including handwritten numbers, characters, pictures, faces, sounds, and speech. This study focuses on the classification and recognition of handwritten digits.1000 were utilized as test samples and 1000 were training samples.10000 picture samples make up the USPS dataset, of which 7291 serve as training samples and 2007 serve as testing samples. We've used the proposed deep neural network technique in this paper to classify and identify data from the ARDIS and USPS datasets. The suggested model consists of six layers with softmax and relu activation functions. After model implementation, accuracy for ARDIS samples reached 98.70% testing and 99.76% training, which is greater than accuracy from prior research. Additionally, using the USPS

samples dataset, 98.22% training accuracy and 93.01% testing accuracy were attained. When compared to earlier methodologies, the data show that deep neural networks perform incredibly well.

Recognition of isolated and simply connected handwritten numerals, Pattern Recognition. (1986)

M. Shridhar and A. Badreldin

In this paper the authors describe the results of their investigation into the development of a recognition algorithm for identifying numerals that may be isolated or connected, broken or continuous. Using a structural classification scheme, the recognition algorithm is derived as a tree classifier. In an extensive test experiment, an accuracy of 99% was realized with isolated numerals. When connected numerals were also included a recognition accuracy of 93% was obtained.

Handwritten Character Recognition using Neural Network and TensorFlow (2019)

Megha Agarwal, Shalika, Vinam Tomar, Priyanka Gupta

The offline handwritten character recognition in this study will be carried out using Tensorflow and a convolutional neural network. a process known as using SoftMax Regression, one may assign probabilities to one of the many characters in the handwritten text that offers the range of values from 0 to 1, summed to 1. The objective is to create software that is extremely accurate and that has a minimum level of spatial and temporal complexity. It was determined that strategies for feature extraction like diagonal and direction are significantly better at producing high accuracy. Outcomes in comparison to other conventional vertical and horizontal techniques moreover use the best Neural network tried layers provides the benefit of a higher accurate outcome by having a high noise tolerance. The feed forward model in neural networks is the back-propagation algorithm that was primarily used to classify the characters, recognise them, and receive training continually more. In addition to these, normalizing along with feature extraction, the results were better and more effective. Character recognition is the outcome of accuracy. The paper will describe the best approach to get more than 90% accuracy in the field of Handwritten Character Recognition (HCR).

Handwritten Digit Recognition of MNIST dataset using Deep Learning state-of-the-art Artificial Neural Network (ANN) and Convolutional Neural Network (CNN) (2021)

Drishti Beohar, A. Rasool

Handwritten digit recognition is an intricate assignment that is vital for developing applications, in computer vision digit recognition is one of the major applications. There has been a copious exploration done in the Handwritten Character Recognition utilizing different deep learning models. Deep learning is rapidly increasing in demand due to its resemblance to the human brain. The two major Deep learning algorithms Artificial Neural Network and Convolutional Neural Network which have been compared in this paper considering their feature extraction and classification stages of recognition. The models were trained using categorical cross-entropy loss and ADAM optimizer on the MNIST dataset. Backpropagation along with Gradient Descent is being used to train the networks along with reLU activations in the network which do automatic feature extraction. In neural networks, Convolution Neural Network (ConvNets or Convolutional neural networks) is one of the primary classifiers to do image recognition, image classification tasks in Computer Vision.

2.3 PROBLEM STATEMENT DEFINITION

Handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image.

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 and so on.

CHAPTER 3
IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

Empathy map

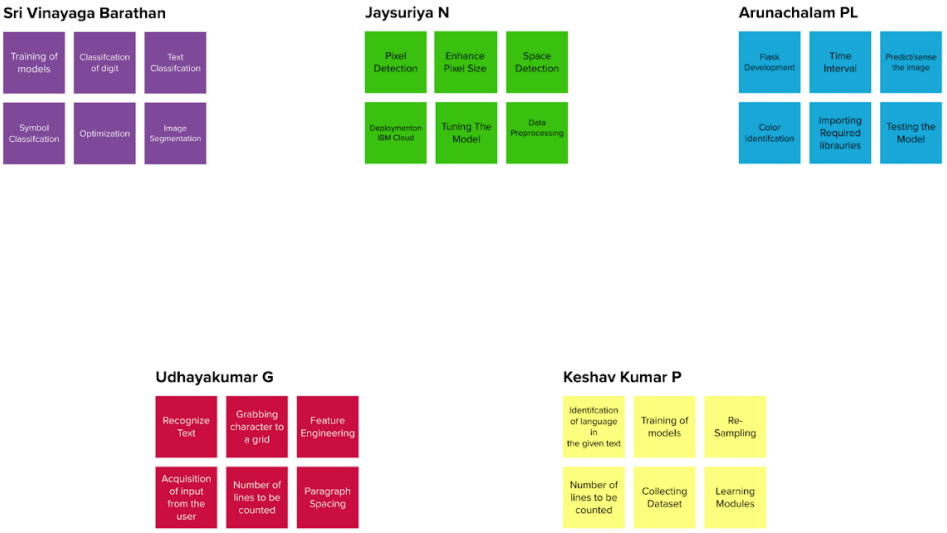
3.2 IDEATION & BRAINSTORMING

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

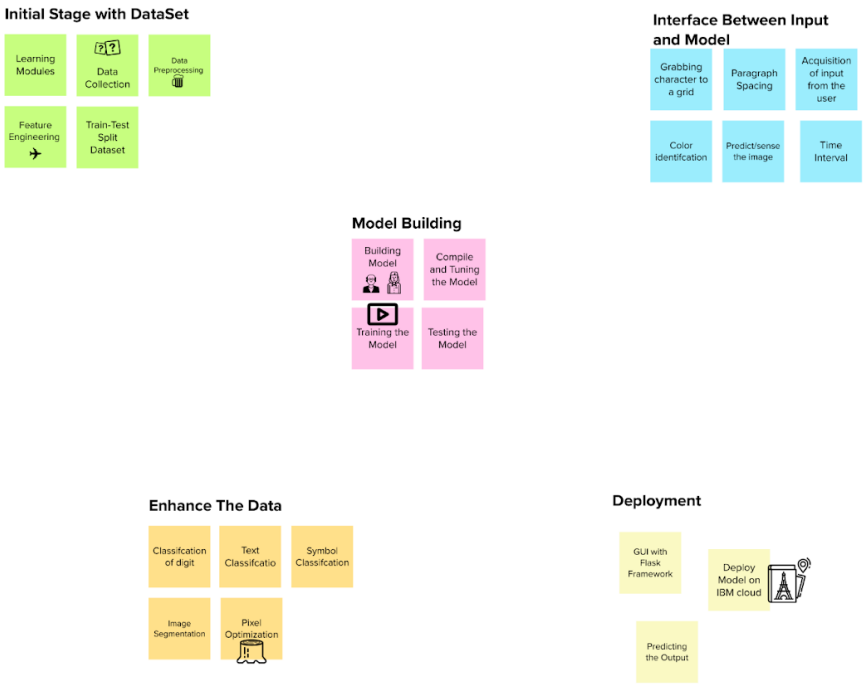


3

Group ideas

Use this space to group similar ideas from the brainstorm. Each group should have a title that describes what the ideas have in common. If a group is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

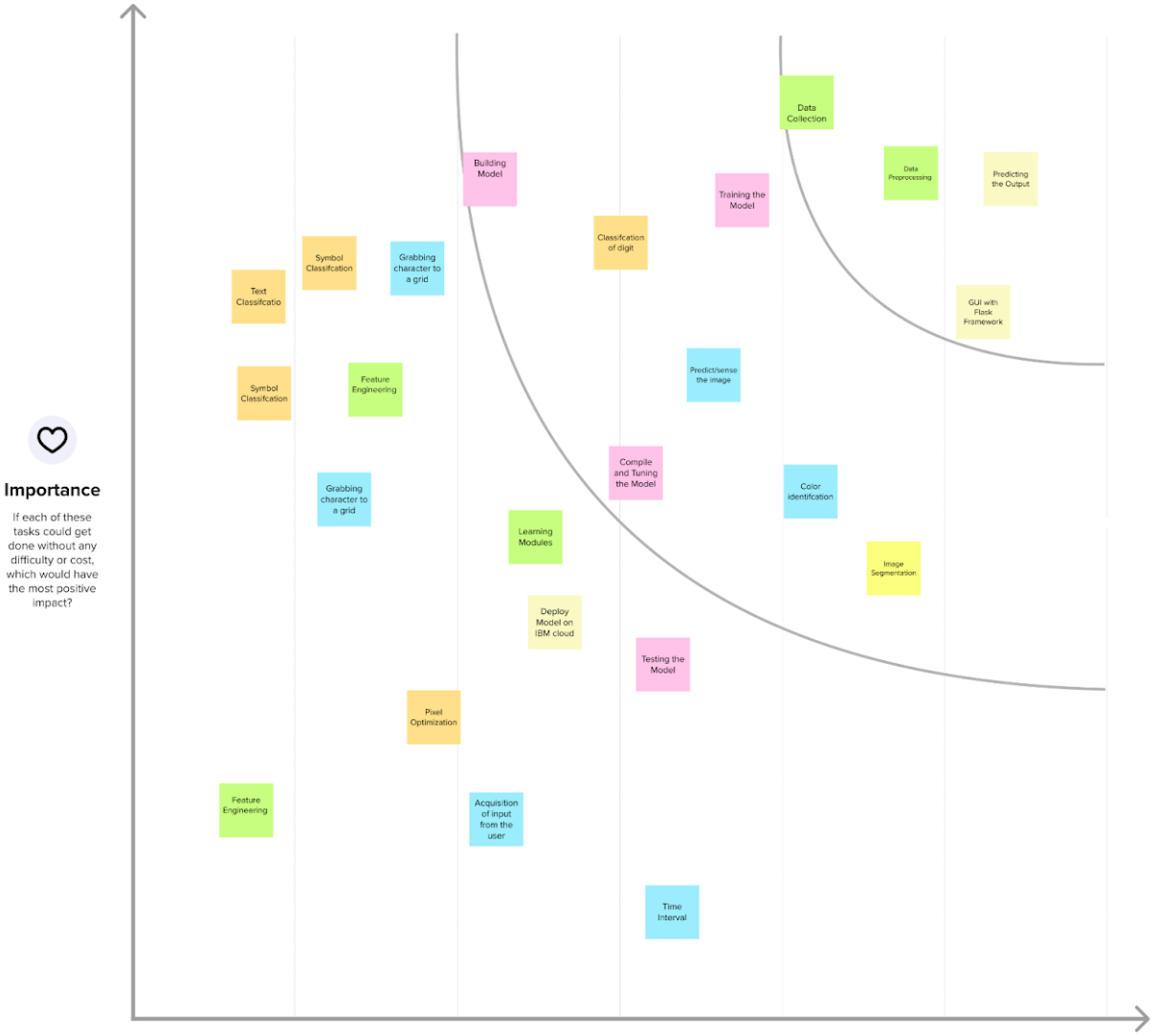


4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes



Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

Feasability

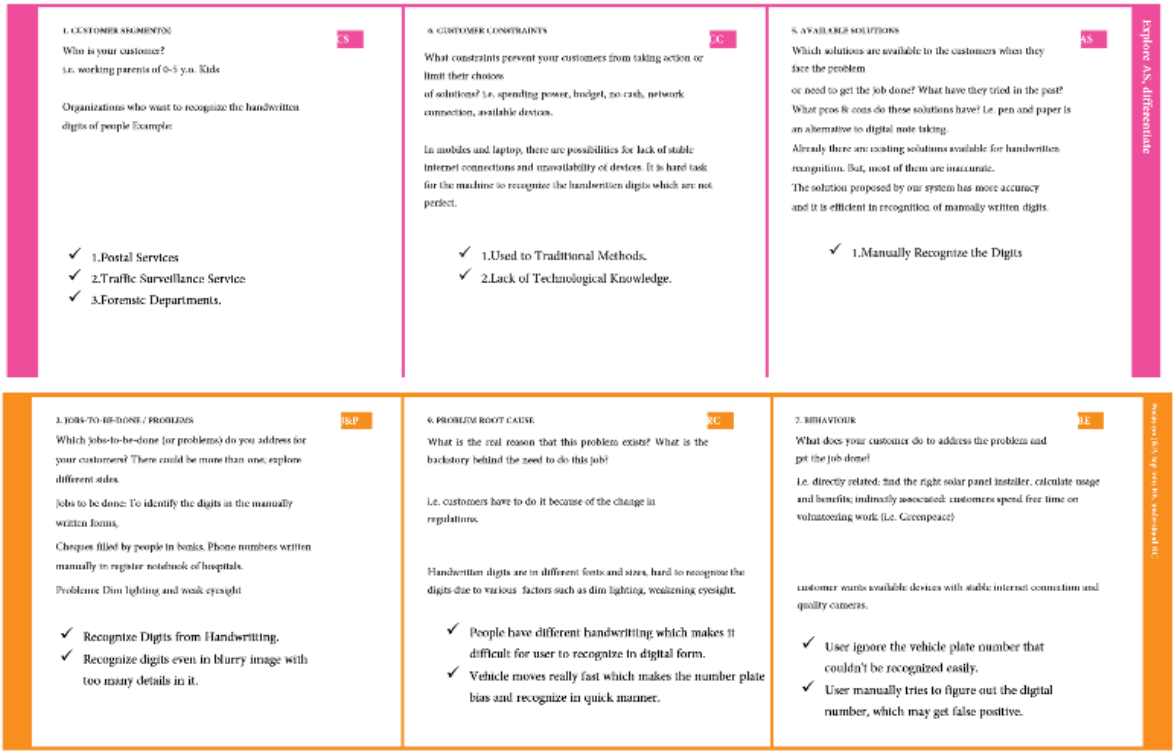
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

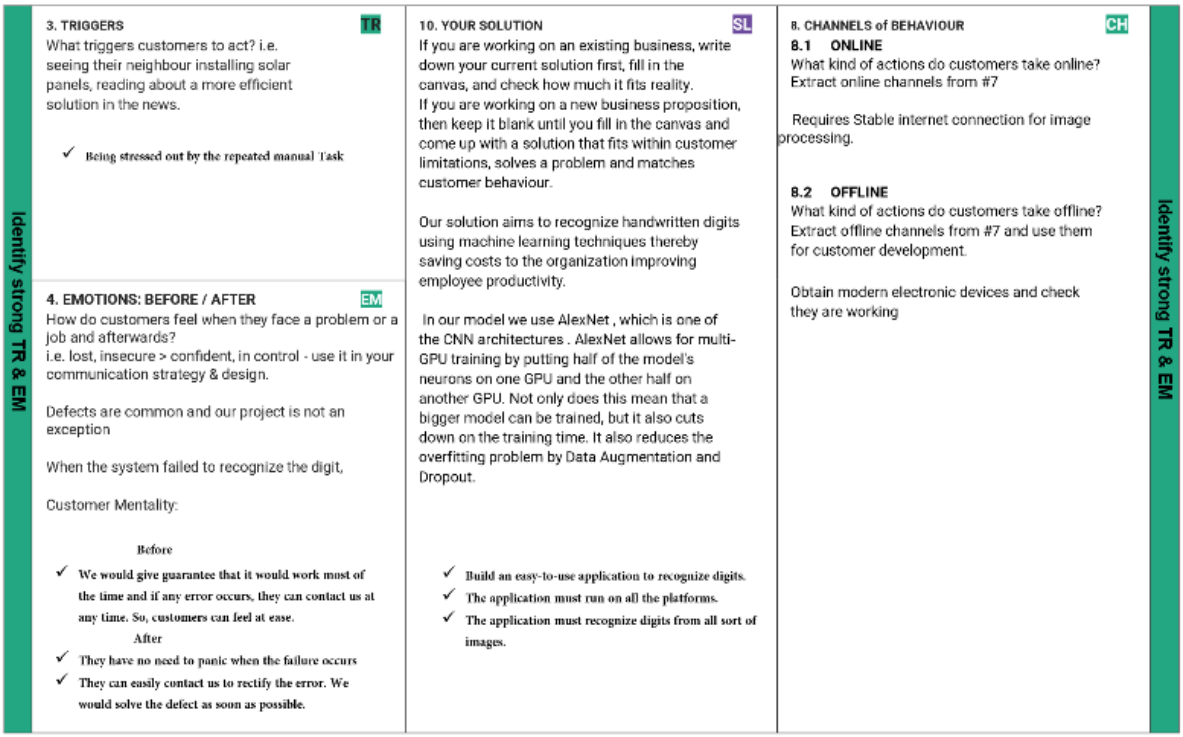
3.3 PROPOSED SOLUTION

S.NO	PARAMETER	DESCRIPTION
1	Problem Statement (Problem to be solved)	Developing a novel handwritten recognition system using ML The handwritten digit recognition system is a way to tackle the problem which uses the image of a digit and recognizes the digit present in the image.
2	Idea / Solution description	Convolutional Neural Networks (CNN) has become one of the most appealing approaches and has been an ultimate factor in a variety of recent success and challenging machine learning applications. In our model we use AlexNet , which is one of the CNN architectures . AlexNet allows for multi-GPU training by putting half of the model's neurons on one GPU and the other half on another GPU. Not only does this mean that a bigger model can be trained, but it also cuts down on the training time. It also reduces the overfitting problem by Data Augmentation and Dropout.
3	Novelty / Uniqueness	Handwritten Digit Recognition is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers,touch defenses, etc. And classify them into 10 predefined classes(0-9).This is the existing method along with this we add some features to make our project unique among them.
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 (Revenue Model)	It is used in the detection of vehicle numbers, banks for reading cheques, post offices for arranging letters, and many other tasks.
6	Scalability of the Solution	The application can easily be scaled to accept multiple inputs and process them parallelly to further increase efficiency.

3.4 PROBLEM SOLUTION FIT

Project Title: A Novel Method For Handwritten Digit Recognition System | Project Design Phase-I - Solution Fit Template | Team ID: PNT2022TMID20912





CHAPTER 4
REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

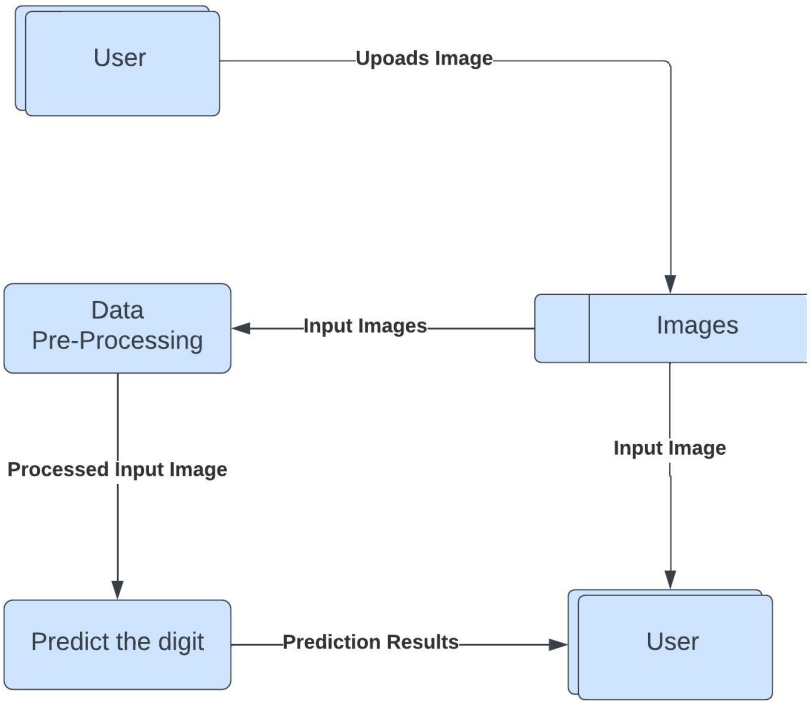
FR No.	Functional Requirement (Epi.)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Upload image	Image upload via files/folders Image upload via drive mage upload via web Image upload via scan/camera
FR-4	Spelling Support	Identification of handwriting of different styles & fonts Spelling Check
FR-5	Translation	Extraction of handwritten digits from the image Conversion of handwritten digits into machine readable form
FR-6	Log out	Includes log out/sign out

4.2 NON FUNCTIONAL REQUIREMENTS

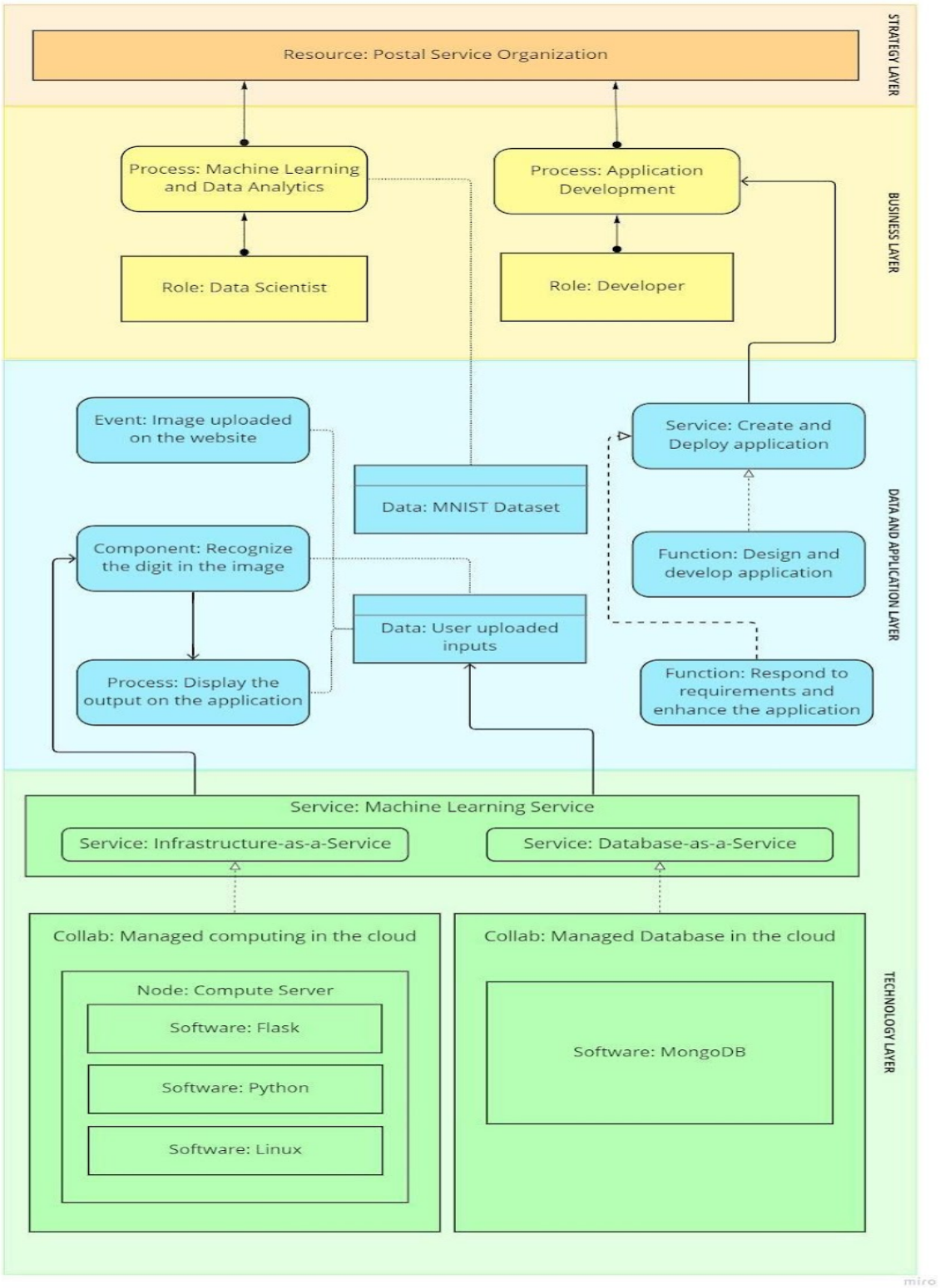
NFR	NON-FUNCTIONAL REQUIREMENTS	DESCRIPTION
NFR-1	Usability	System design must be easily understandable and should be easy to navigate without any problem
NFR-2	Security	Automatic authentication of users with their unique username and password
NFR-3	Reliability	Frequent updation of database and when update fails, it will rollback
NFR-4	Performance	Should reduce the delay in information when hundreds of request are given
NFR-5	Availability	Availability of system functionality and services for use
NFR-6	Scalability	The website traffic limit must be scalable enough to support 2 lakhs users at a time

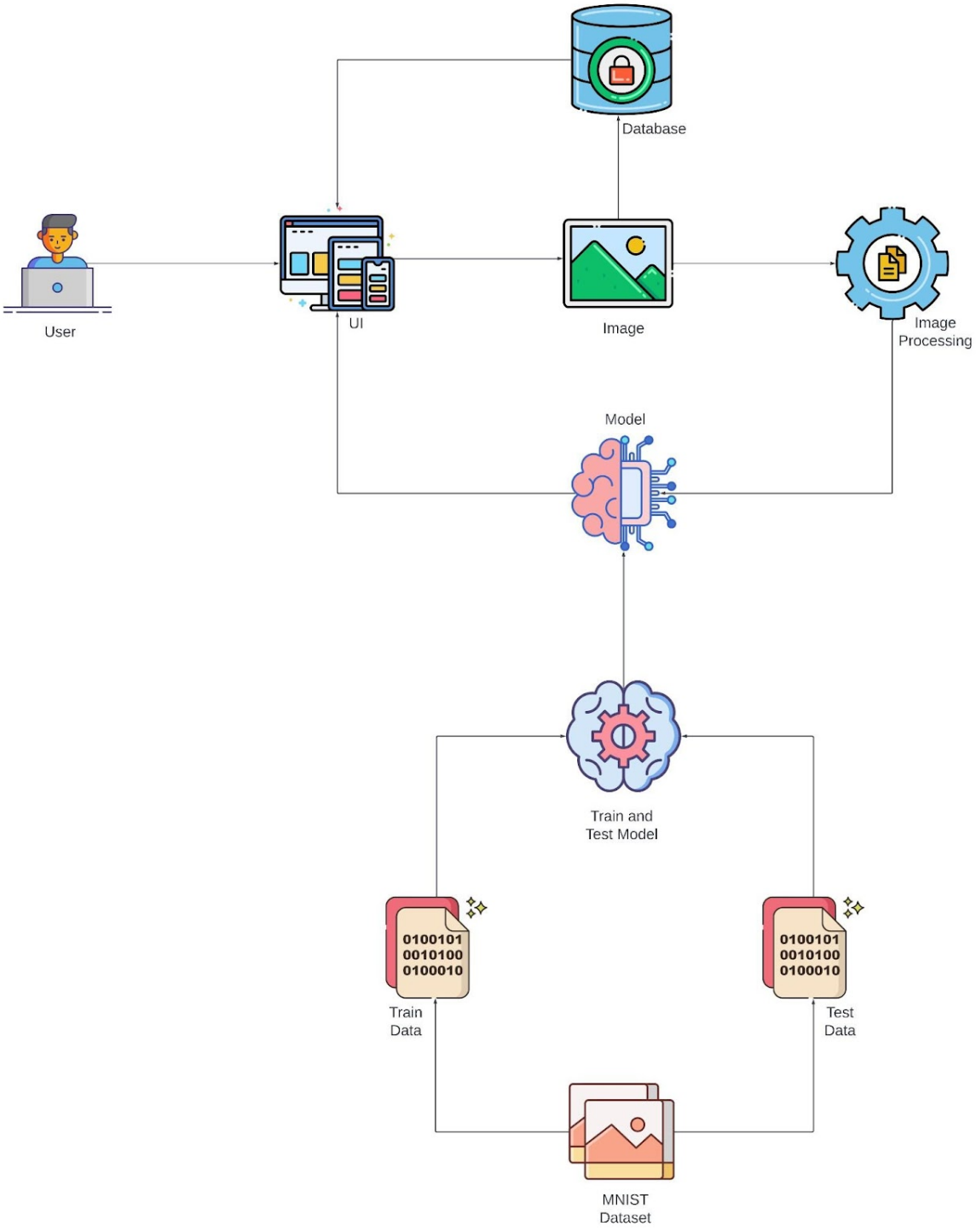
CHAPTER 5
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM



5.2 SOLUTION & 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

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, I can log into the application by entering my email & password	I can log in to the application	Medium	Sprint-3
		USN-5	As a user, On this prediction page, I get to choose the image.	I can choose the image from our local system and predict the output.	High	Sprint-3
		USN-6	As a user, I can view the application's home page where I can read the instructions to use this application.	I can read instructions also and the home page is user-friendly.	Low	Sprint-1
		USN-7	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	High	Sprint-3

				and any virtual storage.		
	Predict	USN-8	As a user, I will train and test the input to get the maximum accuracy of output.	I am able to train and test the application until it gets maximum accuracy.	High	Sprint-4
		USN-9	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)	Accessibility	USN-10	As a user, I can use the web application virtually anywhere	I can use the application on any device with a browser	Medium	Sprint-4

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	Find and Import required Libraries	USN-1	Collecting the required libraries for the development	2	High	Udhayakumar G Arunachalam PL
Sprint-1	Data Pre-Processing	USN-2	Collected Data are pre-processed for Data Processing	2	High	Sri Vinayaga Barathan Jaysuriya N
Sprint-2	Normalizing the Data	USN-3	Data is normalized for clarity and higher quality	2	Medium	Sri Vinayaga Barathan P Keshav Kumar
Sprint-2	Categorizing the data	USN-4	Labels are categorized	2	Medium	Jaysuriya N Arunachalam PL
Sprint-2	Building the model	USN-5	Model is built	2	High	Sri Vinayaga Barathan P Keshav Kumar Udhayakumar G
Sprint-2	Training and testing the model	USN-6	The built model is trained with processed data and tested	2	High	Sri Vinayaga Barathan Jaysuriya N
Sprint-3	Improve the model	USN-7	Model is further refined for better experience	2	Medium	Arunachalam PL P Keshav Kumar Udhayakumar G
Sprint-3	Save the model	USN-8	Model is saved with final fixes and ready to integrate	2	High	Sri Vinayaga Barathan Jaysuriya N
Sprint-3	Build the User Interface	USN-9	The UI is built	2	Medium	Sri Vinayaga Barathan Udhayakumar G
Sprint-4	Integrate and train the model on IBM Cloud	USN-10	The model is integrated with IBM Cloud and trained	2	High	Jaysuriya N P Keshav Kumar
Sprint-4	Test the Application	USN-10	The application is tested	2	High	Sri Vinayaga Barathan Arunachalam PL

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	2	6 Days	18 Oct 2022	23 Oct 2022	2	25 Oct 2022
Sprint-2	2	6 Days	25 Oct 2022	30 Oct 2022	2	30 Oct 2022
Sprint-3	2	6 Days	01 Nov 2022	06 Nov 2022	2	06 Nov 2022
Sprint-4	2	6 Days	07 Nov 2022	12 Nov 2022	2	12 Nov 2022

CHAPTER 7

CODING & SOLUTIONING

Import Required Librauries

In [52]:
```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
from keras.utils import np_utils
import tensorflow as tf
from tensorflow.keras.layers import Conv2D, Dense, Flatten
```

In [2]:
```
print(tf.__version__)
```

```
2.9.2
```

In [3]:
```
mnist_ds = tf.keras.datasets.mnist.load_data()
```

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

In [4]:
```
mnist_ds
```

Building The Model

In [53]:
```
model = tf.keras.models.Sequential([tf.keras.layers.Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation="relu"),
                                    tf.keras.layers.Conv2D(32, (3, 3), activation="relu"),
                                    tf.keras.layers.Flatten(),
                                    tf.keras.layers.Dense(10, activation=tf.nn.softmax)])
```

In [54]:
```
model.compile(loss='categorical_crossentropy', optimizer="Adam", metrics=["accuracy"])
```

In [55]:
```
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
_________________________________________________________________
```

Training The Model

In [56]:
```
model.fit(training_images, training_labels, batch_size=32, epochs=5, validation_data=(test_images,test_labels))
```

```
Epoch 1/5
1875/1875 [==============================] - 16s 4ms/step - loss: 0.1254 - accuracy: 0.9625 - val_loss: 0.0519 - val_accuracy: 0.9831
Epoch 2/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0464 - accuracy: 0.9861 - val_loss: 0.0385 - val_accuracy: 0.9868
Epoch 3/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0296 - accuracy: 0.9907 - val_loss: 0.0409 - val_accuracy: 0.9872
Epoch 4/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0201 - accuracy: 0.9937 - val_loss: 0.0403 - val_accuracy: 0.9878
Epoch 5/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0139 - accuracy: 0.9957 - val_loss: 0.0457 - val_accuracy: 0.9872
```

Out[56]:

Test The Model

In [58]:
```
metrics = model.evaluate(test_images, test_labels, verbose=0)

print("Test Loss -> {}  \nTest Accuracy -> {}".format(metrics[0],metrics[1]))
```

```
Test Loss -> 0.04573516175150871
Test Accuracy -> 0.9872000217437744
```

In [67]:
```
model.predict(test_images[2:8])
```

```
1/1 [==============================] - 0s 15ms/step
```

Out[67]:
```
array([[2.32068427e-08, 9.99983430e-01, 8.10439190e-07, 1.28179977e-07,
        9.64922492e-06, 1.83879649e-06, 1.62838049e-07, 1.56461965e-06,
        2.34936374e-06, 3.22469944e-08],
       [9.99998927e-01, 1.04071238e-13, 7.69856399e-07, 1.84226245e-09,
        3.37900985e-13, 4.71777106e-09, 8.84182239e-09, 2.02508791e-11,
        3.22932721e-07, 9.56373647e-09],
       [8.73478698e-13, 4.26847549e-13, 1.15858136e-10, 3.97662771e-11,
        9.99999881e-01, 2.68545906e-12, 8.56604648e-11, 7.41609482e-11,
        4.87753553e-08, 1.05269102e-07],
       [2.15423035e-09, 9.99581635e-01, 2.15949945e-06, 1.08863398e-08,
        2.10376020e-05, 2.63231090e-08, 3.26978977e-08, 3.81208694e-04,
        1.27898356e-05, 1.17525337e-06],
       [2.41032138e-18, 9.36788008e-11, 3.97475330e-10, 3.54850779e-13,
        9.99299288e-01, 6.94019908e-09, 6.61158953e-14, 4.28452246e-10,
        7.00477336e-04, 2.29253416e-07],
       [1.89875802e-16, 2.21634187e-11, 1.76986703e-09, 2.65193867e-09,
        2.56875592e-06, 2.88839956e-08, 1.60236594e-14, 2.67538752e-11,
        5.46009005e-06, 9.99991894e-01]], dtype=float32)
```

In [74]:
```
history=model.predict(np.array([test_images[7]]))
history
```

```
1/1 [==============================] - 0s 17ms/step
```

Out[74]:
```
array([[1.8987580e-16, 2.2163419e-11, 1.7698670e-09, 2.6519387e-09,
        2.5687532e-06, 2.8883996e-08, 1.6023692e-14, 2.6753875e-11,
        5.4600901e-06, 9.9999189e-01]], dtype=float32)
```

In [75]:
```
np.argmax(history, axis=1)
```

Out[75]:
```
array([9])
```

In [73]:
```
#It predicted as 9
```

Let us see , It is correct or not?

In [78]:
```
tl=test_labels[7]
tl
```

Out[78]:
```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 1.], dtype=float32)
```

In [81]:
```
np.argmax(tl)
```

Out[81]:
```
9
```

It Predicted Correctly!!!

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

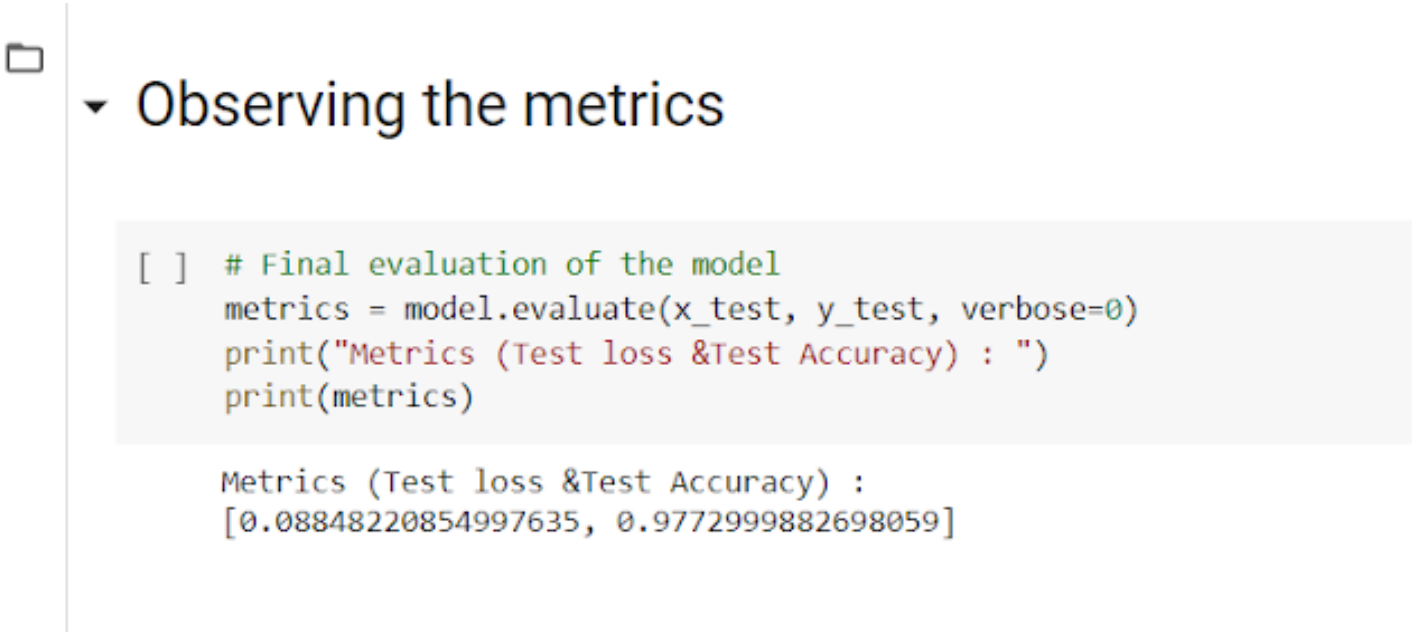
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Total
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won’t Fix	1	0	1	0	2
Total	6	1	4	3	14

8.2.2 TEST CASE ANALYSIS

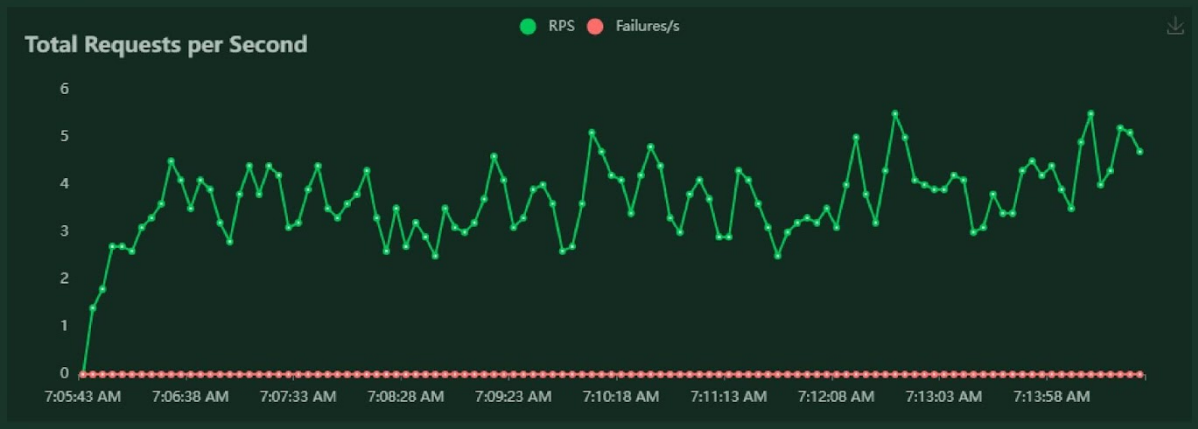
Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	3	7
Security	2	0	1	1
Performance	3	0	1	2
Exception Reporting	2	0	0	2

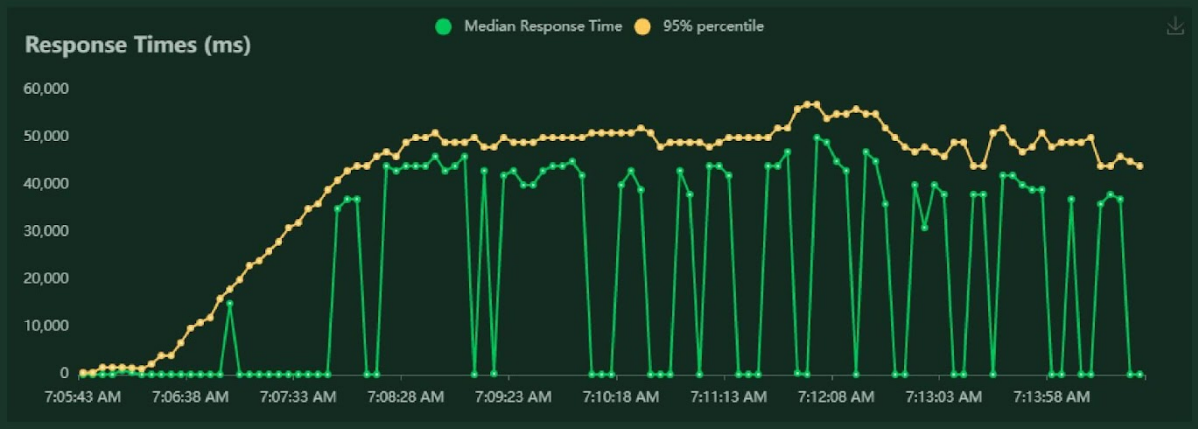
CHAPTER 9
RESULTS

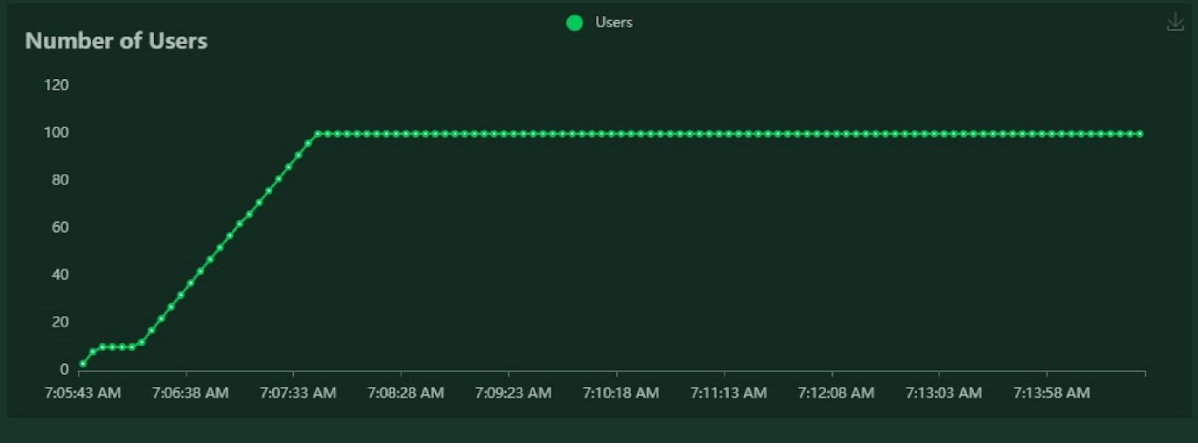
9.1 PERFORMANCE METRICS



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

Building The Model

In [53]:
```
model = tf.keras.models.Sequential([tf.keras.layers.Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation="relu"),
                                    tf.keras.layers.Conv2D(32, (3, 3), activation="relu"),
                                    tf.keras.layers.Flatten(),
                                    tf.keras.layers.Dense(10, activation=tf.nn.softmax)])
```

In [54]:
```
model.compile(loss='categorical_crossentropy', optimizer="Adam", metrics=["accuracy"])
```

In [55]:
```
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
_________________________________________________________________
```

Training The Model

In [56]:
```
model.fit(training_images, training_labels, batch_size=32, epochs=5, validation_data=(test_images,test_labels))
```

```
Epoch 1/5
1875/1875 [==============================] - 16s 4ms/step - loss: 0.1254 - accuracy: 0.9625 - val_loss: 0.0519 - val_accuracy: 0.9831
Epoch 2/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0464 - accuracy: 0.9861 - val_loss: 0.0385 - val_accuracy: 0.9868
Epoch 3/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0296 - accuracy: 0.9907 - val_loss: 0.0409 - val_accuracy: 0.9872
Epoch 4/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0201 - accuracy: 0.9937 - val_loss: 0.0403 - val_accuracy: 0.9878
Epoch 5/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0139 - accuracy: 0.9957 - val_loss: 0.0457 - val_accuracy: 0.9872
```

Out[56]:

Test The Model

In [58]:
```
metrics = model.evaluate(test_images, test_labels, verbose=0)

print("Test Loss -> {}  \nTest Accuracy -> {}".format(metrics[0],metrics[1]))
```

```
Test Loss -> 0.04573516175150871
Test Accuracy -> 0.9872000217437744
```

FLASK APP

```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()
```

RECOGNIZER

```python
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))

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/digitRecog.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">
            <h1 class="heading__main">Handwritten Digit Recognizer</h1>
                <h2 class="heading__sub">Web Application to detect Handwritten digits</h2>

                <div class="form-wrapper">
                    <marquee width="100%" direction="left" height="60px">
                        <b><-----Provide an image for which you want to get the clear identity-------></b>
                        </marquee><br>
                    <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>
    </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;
        background-image: url('https://www.zastavki.com/pictures/1600x900/2015/Backgrounds_Orange_gradient_background_096901_25.jpg');

}

.container {
        width: 100%;
        height: 100%;
        display: flex;
        flex-direction: column;
        justify-content: center;
        align-items: center;
        background-color: rgba(102, 100, 100, 0.79);
}

.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(239, 12, 12, 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;
        width: 700px;
        height: 500px;
        border: 1px solid rgb(255, 98, 0);
}

.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: rgb(255, 255, 255);
        background-color: rgb(34, 14, 213);
        box-shadow: 0 5px 10px rgb(249, 127, 51);
}
```

```
80
81  .form-wrapper .upload #up_btn {
82          display: none;
83
84  }
85
86  .form-wrapper .upload label {
87          font-size: 1rem;
88          font-weight: 600;
89          color: rgb(255, 255, 255);
90          height: 100%;
91          width: 100%;
92          padding: 10px;
93          display: block;
94          background-color: blueviolet;
95          text-align: left;
96
97  }
98
99  .form-wrapper .upload svg {
100         height: 15px;
101         width: auto;
102         padding-right: 8px;
103         margin-bottom: -2px;
104 }
105
106 @media screen and (max-width: 700px) {
107         .upload-container {
108                 height: 20rem;
109                 width: 18rem;
110                 margin-top: 3.5rem;
111                 margin-bottom: -8rem;
112         }
113
114         .heading .heading__main {
115                 margin-top: -6rem;
116                 font-size: 2rem;
117                 padding-bottom: 1rem;
118         }
119 }
```

PREDICT PAGE (HTML)

```
<html>
        <head>
                <style>
                        .bt{
        background-color: gray;
        color: rgb(14, 13, 13);
        border: 1px solid #eee;
        border-radius: 20px;
        box-shadow: 5px 5px 5px #eee;
        text-shadow: none;
        text-align: center;
}
.button {
  border-radius: 4px;
  background-color: #043217;
  border: none;
  color: #FFFFFF;
  text-align: center;
  font-size: 22px;
  padding: 10px;
  width: 200px;
  transition: all 0.5s;
  cursor: pointer;
  margin: 5px;
}

```

```
.button span {
  cursor: pointer;
  display: inline-block;
  position: relative;
  transition: 0.5s;
}

.button span:after {
  content: ' \00AB';
  position: absolute;
  opacity: 0;
  top: 0;
  left: -20px;
  transition: 0.5s;
}

.button:hover span {
  padding-left: 25px;
}

.button:hover span:after {
  opacity: 1;
  left: 0;
}
                </style>
                <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')}}" />

       <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5ID
                <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}}" /><br>


                                </div>
                                <img src="https://media3.giphy.com/media/XwcRf1O9HD0Sk6RaRM/giphy.gif" height="250">
                                <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>

                <form action="/" >
                        <div class="bt">
                                <button type="submit" class="button"><span>Choose another pic !  </span></button>
                </div>
                  </form>


        </body>
</html>
```

PREDICT PAGE (CSS)

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

body {
        color: rgb(255, 255, 255);
        font-family: "Overpass", sans-serif;
        background-image: url('https://wallpaperaccess.com/full/1092567.png');
}

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(124, 189, 245);
        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(129, 175, 231);
}

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

```
54
55  .result-wrapper .result-container .value {
56          font-size: 6rem;
57  }
58
59  .result-wrapper .result-container .accuracy {
60          margin-top: -1rem;
61  }
62
63  .other_predictions {
64          display: flex;
65          justify-content: center;
66          align-items: center;
67          flex-wrap: wrap;
68          column-gap: 1rem;
69          row-gap: 1rem;
70          font-weight: 700;
71          border: 2px dotted black;
72  }
73
74  .other_predictions .value {
75          display: flex;
76          justify-content: center;
77          align-items: center;
78          flex-direction: column;
79          width: 5rem;
80          height: 5rem;
81          box-shadow: 0 0 7px rgb(158, 157, 157);
82          border: 2px dotted black;
83  }
84
85  .other_predictions .value div {
86          margin-top: -1.2rem;
87          border: 2px dotted black;
88
89  }
90
91  @media screen and (max-width: 700px) {
92          h1 {
93                  font-size: 2.3rem;
94          }
95
96          .result-wrapper .input-image-container,
97          .result-wrapper .result-container {
98                  width: 7rem;
99                  height: 7rem;
100         }
101
102         .result-wrapper .result-container .value {
103                 font-size: 4rem;
104         }
105 }
```

JavaScript

```
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';
});
```

GITHUB

https://github.com/IBM-EPBL/IBM-Project-6305-1658826027

PROJECT DEMO

https://youtu.be/-1iLDvnY5PE