PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

Submitted by

PNT2022TMID35510

Thiruvazhi Dhinesh Kumar S	**2019105594**
Gowtham N	**2019105532**
Dinesh M	**2019105526**
Ruban Chakaravarthi V	**2019105045**

TABLE OF CONTENTS

1. INTRODUCTION

1.1 PROJECT OVERVIEW

1.2 PURPOSE

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

2.2 REFERENCES

2.3 PROBLEM STATEMENT DEFINITION

3. IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP

3.2 IDEATION & BRAINSTORMING

3.3 PROPOSED SOLUTION

3.4 PROBLEM SOLUTION FIT

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

4.2 NON FUNCTIONAL REQUIREMENTS

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

5.2 SOLUTION & TECHNICAL ARCHITECTURE

5.3 USER STORIES

6. PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

6.2 SPRINT DELIVERY SCHEDULE

6.3 REPORT FROM JIRA

7. CODING & SOLUTIONING

7.1 FEATURE 1

7.2 FEATURE 2

8. TESTING

8.1 TEST CASES

8.2 USER ACCEPTANCE TESTING

9. RESULTS

9.1 PERFORMANCE METRICS

10 ADVANTAGES & DISADVANTAGES

11 CONCLUSION

12 FUTURE SCOPE

13 APPENDIX

SOURCE CODE

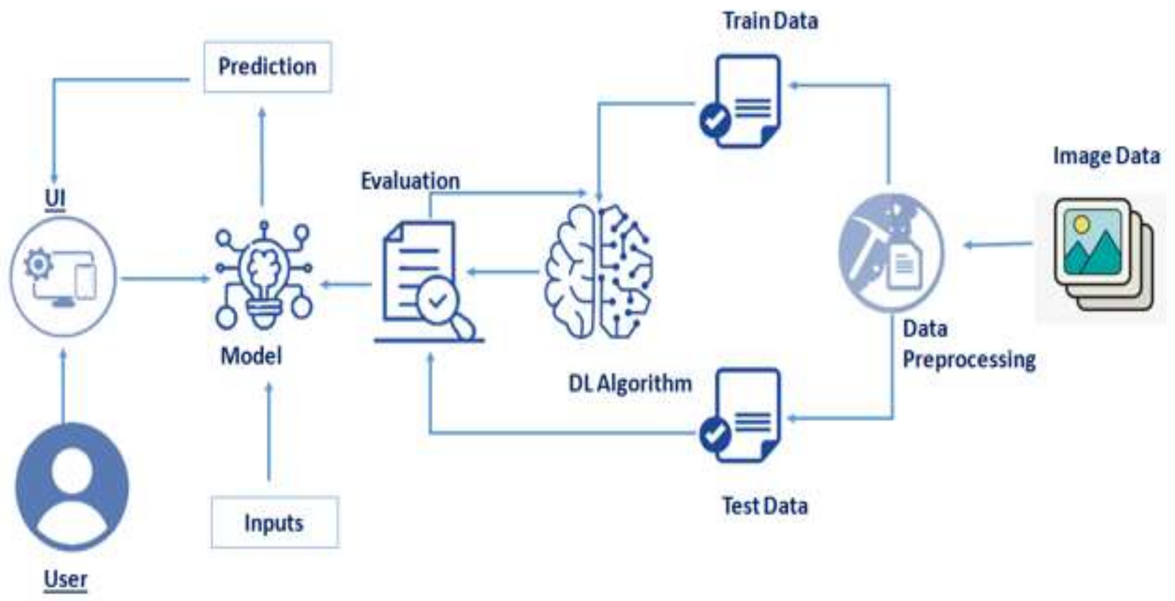
GITHUB & PROJECT DEMO LINK

INTRODUCTION

1.1 PROJECT OVERVIEW

Handwriting recognition is one of the compelling research works going on because every individual in this world has their own style of writing. It is the capability of the computer to identify and understand handwritten digits or characters automatically. Because of the progress in the field of science and technology, everything is being digitalized to reduce human effort. Hence, there comes a need for handwritten digit recognition in many real-time applications. MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. We use Artificial neural networks to train these images and build a deep learning model. Web application is created where the user can upload an image of a handwritten digit. this image is analyzed by the model and the detected result is returned on to UI

Technical Architecture:



1.2 PURPOSE

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 (tax forms) and so on.

LITERATURE SURVEY

2.1 EXISTING PROBLEM:

The issue is that there's a wide range of handwriting – good and bad. This makes it tricky for programmers to provide enough examples of how every character might look. Sometimes, characters look very similar, making it hard for a computer to recognise accurately. 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:

R. Bajaj, L. Dey, S. Chaudhari et al, employed three different kinds of features, namely, the density features, moment features and descriptive component features for classification of Devanagari Numerals. They proposed multi classifier connectionist architecture for increasing the recognition reliability and they obtained 89.6% accuracy for handwritten Devanagari numerals.

Salvador España-Boquera et al, in this paper hybrid Hidden Markov Model (HMM) model is proposed for recognizing unconstrained offline handwritten texts. In this, the structural part of the optical model has been modelled with Markov chains, and a Multilayer Perceptron is used to estimate the emission probabilities. In this paper, different techniques are applied to remove slope and slant from handwritten text and to normalize the size of text images with supervised learning methods. The key features of this recognition system were to develop a system having high accuracy in pre-processing and recognition, which are both based on ANNs.

Yoshimasa Kimura presented a work on how to select features for Character Recognition Using Genetic Algorithms. The author proposes a novel method of feature selection for character recognition using genetic algorithms (GA). The proposed method selects only the genes for which the recognition rate of training samples exceeds the predetermined threshold as a candidate of the parent gene and adopts a reduction ratio in the number of features used for recognition as the fitness value.

Renata F. P. Neves has proposed SVM based offline handwritten digit recognition. Authors claim that SVM outperforms the Multilayer perceptron classifier. Experiment is carried out on NIST SD19 standard dataset. Advantage of MLP is that it is able to segment non-linearly separable classes. However, MLP can easily fall into a region of local minimum, where the training will stop assuming it has achieved an optimal point in the error surface. Another hindrance is defining the best network architecture to solve the problem, considering the number of layers and the amount of perceptron in each hidden layer. Because of these disadvantages, a digit recognizer using the MLP structure may not produce the desired low error rate.

M. Hanmandlu, O.V. Ramana Murthy has presented in their study the recognition of handwritten Hindi and English numerals by representing them in the form of exponential membership functions which serve as a fuzzy model. The recognition is carried out by modifying the exponential membership functions fitted to the fuzzy sets. These fuzzy sets are derived from features consisting of normalized distances obtained using the Box approach. The membership function is modified by two structural parameters that are estimated by optimizing the entropy subject to the attainment of membership function to unity. The overall recognition rate is found to be 95% for Hindi numerals and 98.4% for English numerals.

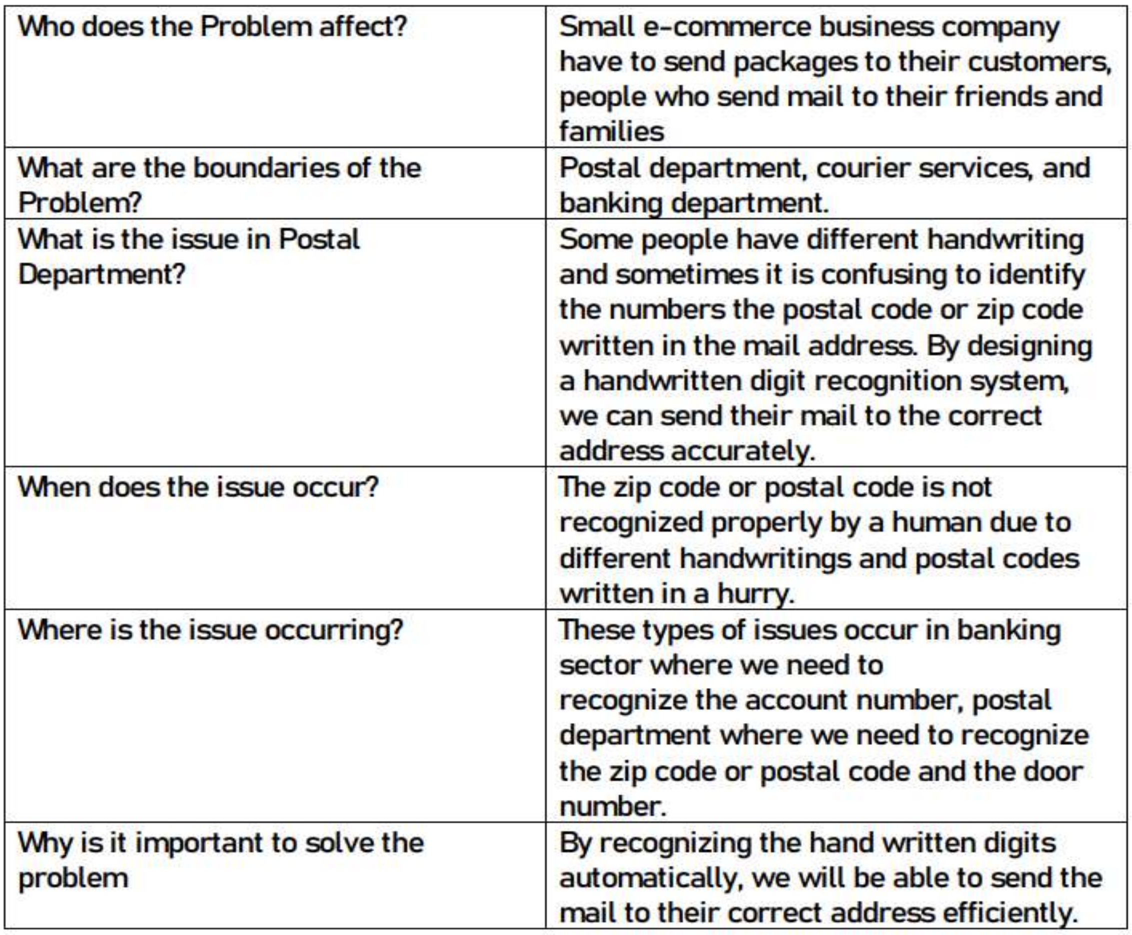
Ragha & Sasikumar describes a system for Kannada characters. In this paper, the moment features are extracted from the Gabor wavelets of pre-processed images of 49 characters. The comparison of moments features of 4 directional images with original images are tested on Multi-Layer Perceptron with Back Propagation Neural

Network. The average performance of the system with these two features together is 92%.

Aparna et al, proposed a method to construct a handwritten Tamil character by executing a sequence of strokes. A structure or shape-based representation of a stroke was used in which a stroke was represented as a string of shape features. Using this string representation, an unknown stroke was identified by comparing it with a database of strokes using a flexible string-matching procedure.

2.3 PROBLEM STATEMENT DEFINITION:

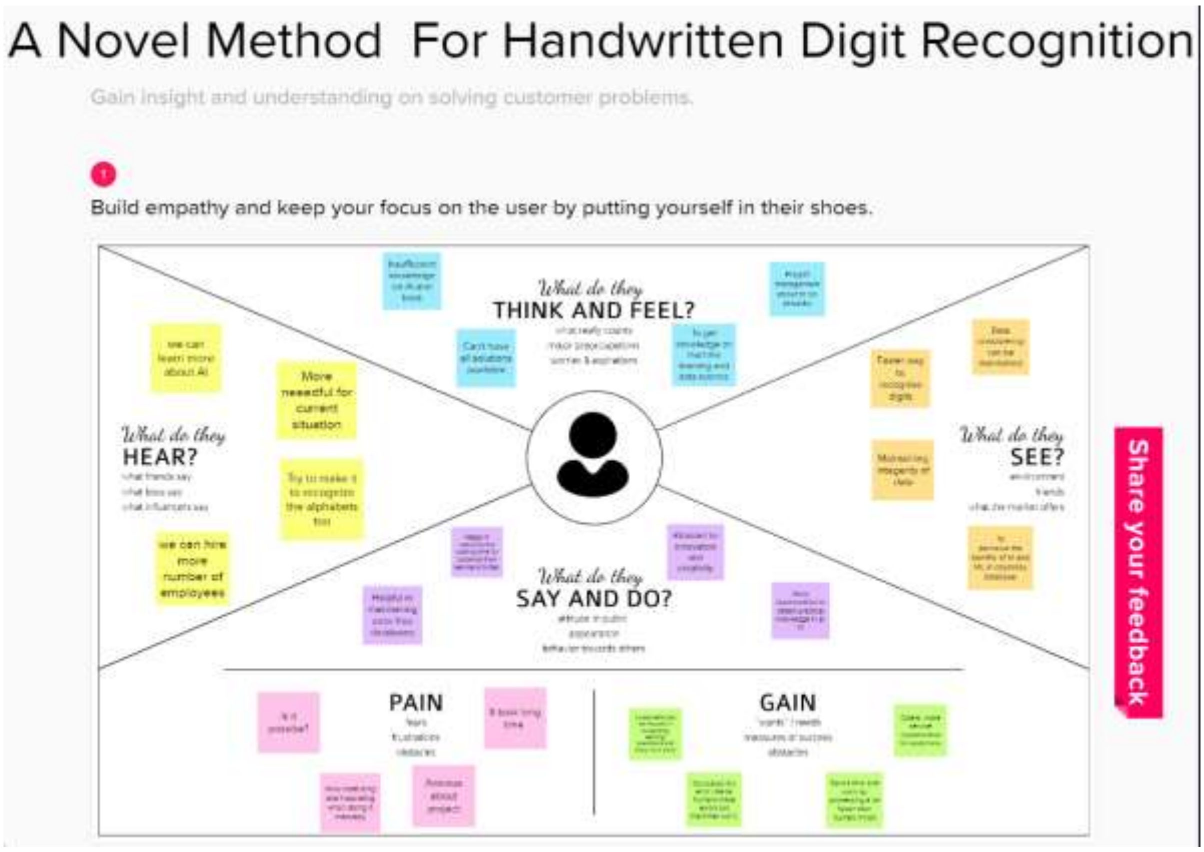
Arun is an employee at the postal office who's job is to segregate mail with different handwritten pin code or postal address quickly and efficiently so that every mail is sent to their correct address. So, a novel method for Handwritten Digit Recognition System needs to be developed.



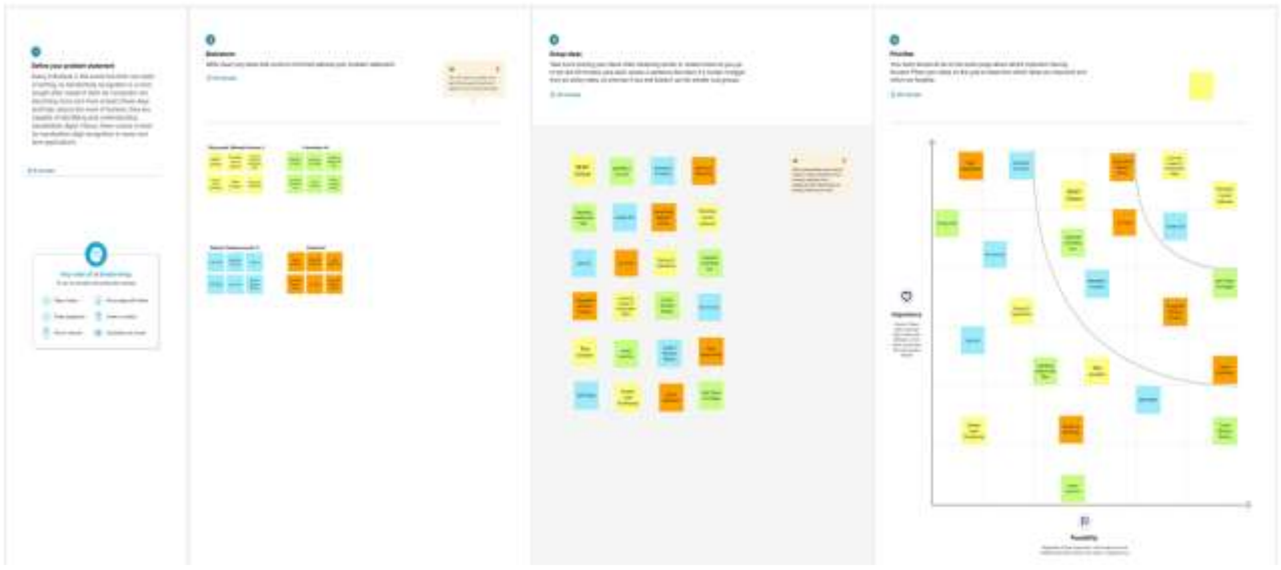
Who does the Problem affect?	Small e-commerce business company have to send packages to their customers, people who send mail to their friends and families
What are the boundaries of the Problem?	Postal department, courier services, and banking department.
What is the issue in Postal Department?	Some people have different handwriting and sometimes it is confusing to identify the numbers the postal code or zip code written in the mail address. By designing a handwritten digit recognition system, we can send their mail to the correct address accurately.
When does the issue occur?	The zip code or postal code is not recognized properly by a human due to different handwritings and postal codes written in a hurry.
Where is the issue occurring?	These types of issues occur in banking sector where we need to recognize the account number, postal department where we need to recognize the zip code or postal code and the door number.
Why is it important to solve the problem	By recognizing the hand written digits automatically, we will be able to send the mail to their correct address efficiently.

<u>**IDEATION & PROPOSED SOLUTION**</u>

3.1 EMPATHY MAP CANVAS:



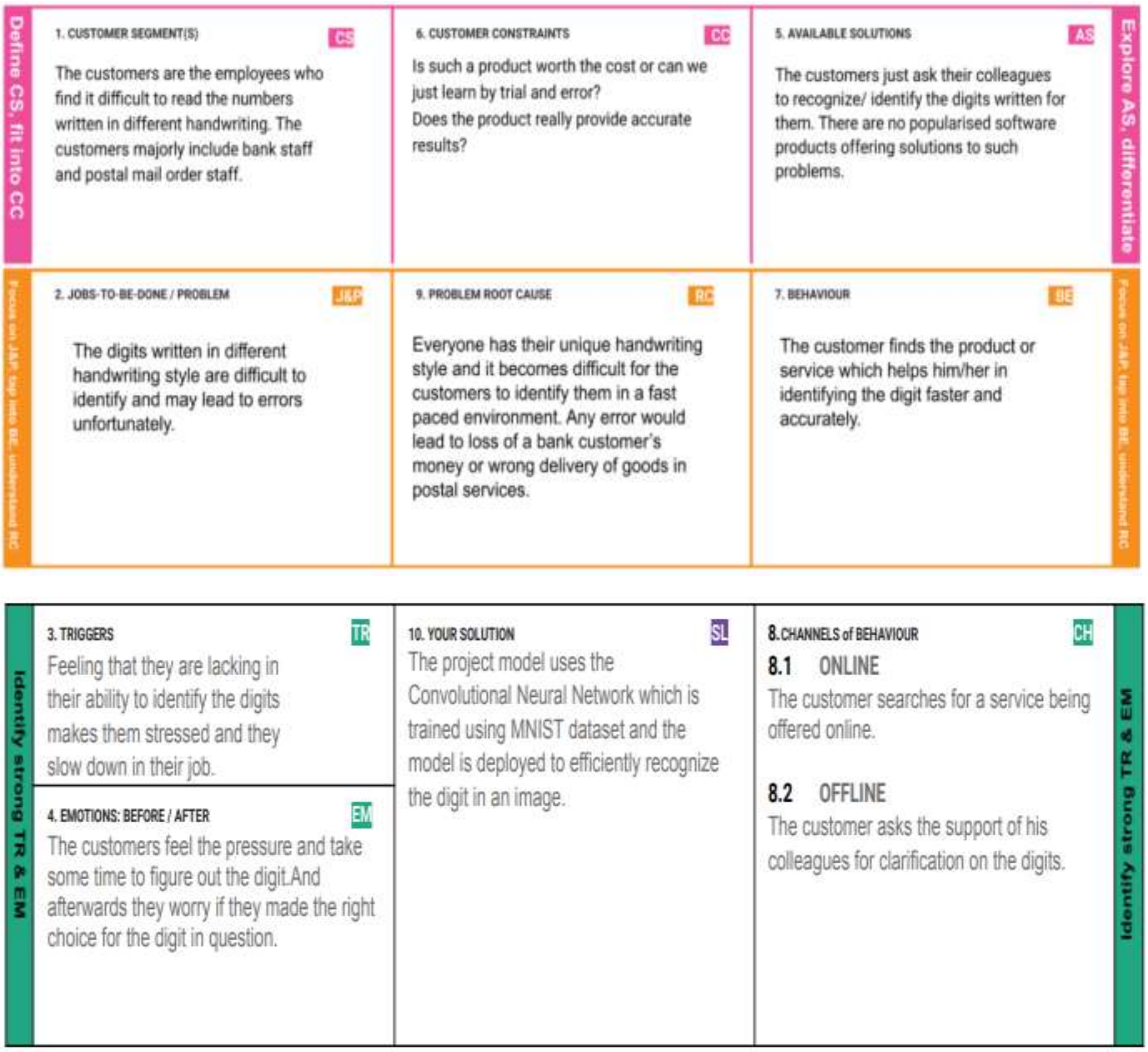
3.2 IDEATION & BRAINSTORMING:



3.3 PROPOSED SOLUTION:

S.NO	PARAMETER	DESCRIPTION
1.	Problem Statement (Problem to be solved)	Handwritten digits do not always have same size, thickness and shape. So, the handwritten digits cannot be recognized correctly.
2.	Idea/Solution description	Handwritten digit recognition system is a way to overcome this problem in which the image of a digit is given as an input to recognize what digit is present in the image.
3.	Novelty/Uniqueness	This handwritten recognition system is meant for only digits, but other existed system is meant to recognize alphabets, expressions etc..
4.	Feasibility of Solution	With a model trained using diverse images , digits can be identified with great accuracy.
5.	Social Impact/Customer Satisfaction	• Postal department and courier services can easily find the digits written. • Old people who have eye sight issues,like far-sightedness can use this system to recognize the handwritten digits correctly
6.	Business Model	This system can be converted into business model by providing services to Banking sector and Postal sector.
7.	Scalability of the Solution	If this model is able to classify different handwritten digits, then it can be used to classify different characters and different types of digits like Arabic numbers.

3.4 PROBLEM SOLUTION FIT:



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

The customers are the employees who find it difficult to read the numbers written in different handwriting. The customers majorly include bank staff and postal mail order staff.

6. CUSTOMER CONSTRAINTS CC

Is such a product worth the cost or can we just learn by trial and error?
Does the product really provide accurate results?

5. AVAILABLE SOLUTIONS AS

The customers just ask their colleagues to recognize/ identify the digits written for them. There are no popularised software products offering solutions to such problems.

Explore AS, differentiate

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

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

The digits written in different handwriting style are difficult to identify and may lead to errors unfortunately.

9. PROBLEM ROOT CAUSE RC

Everyone has their unique handwriting style and it becomes difficult for the customers to identify them in a fast paced environment. Any error would lead to loss of a bank customer's money or wrong delivery of goods in postal services.

7. BEHAVIOUR BE

The customer finds the product or service which helps him/her in identifying the digit faster and accurately.

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

Identify strong TR & EM

3. TRIGGERS TR

Feeling that they are lacking in their ability to identify the digits makes them stressed and they slow down in their job.

4. EMOTIONS: BEFORE / AFTER EM

The customers feel the pressure and take some time to figure out the digit.And afterwards they worry if they made the right choice for the digit in question.

10. YOUR SOLUTION SL

The project model uses the Convolutional Neural Network which is trained using MNIST dataset and the model is deployed to efficiently recognize the digit in an image.

8.CHANNELS of BEHAVIOUR CH

8.1 ONLINE

The customer searches for a service being offered online.

8.2 OFFLINE

The customer asks the support of his colleagues for clarification on the digits.

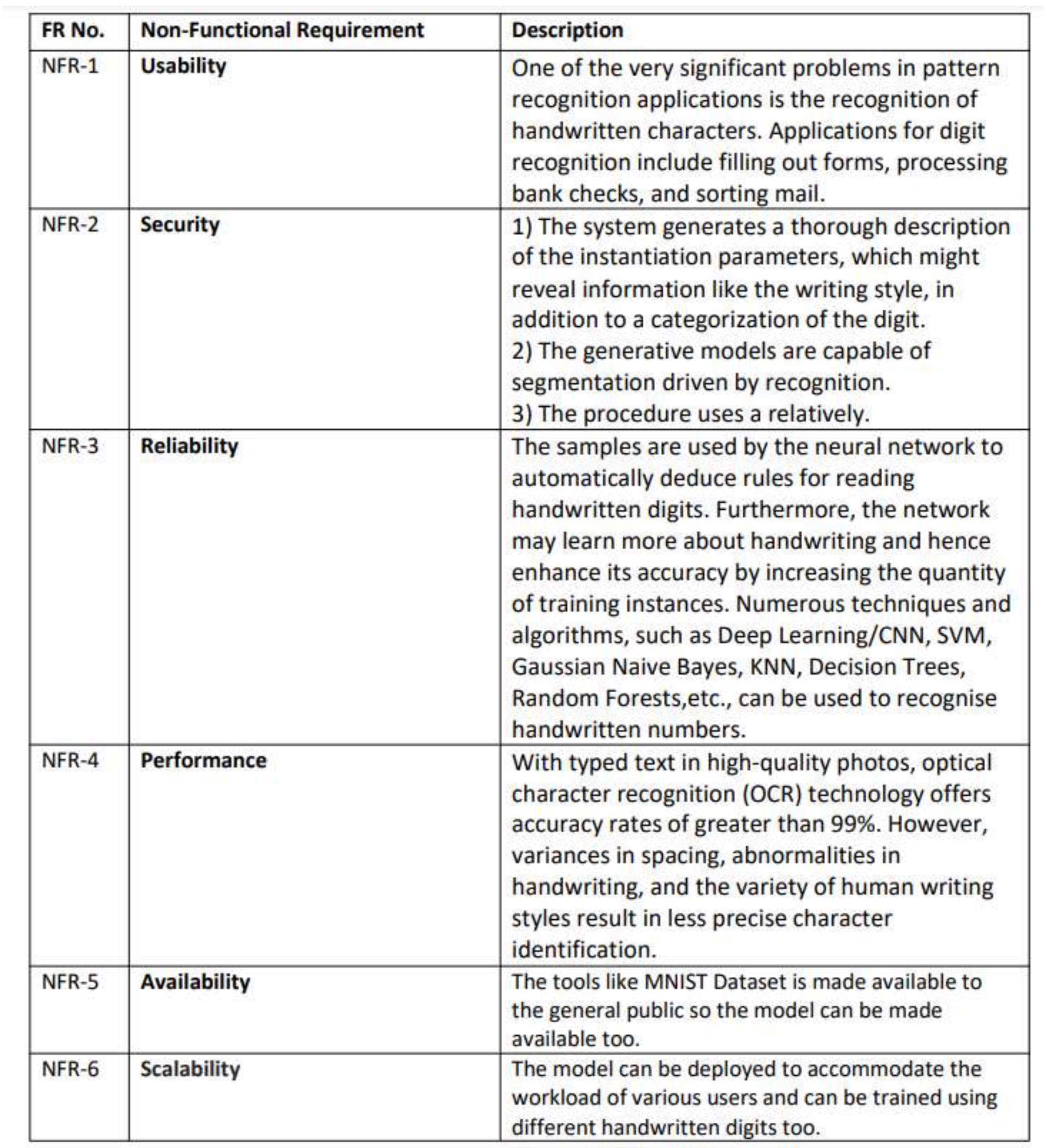
Identify strong TR & EM

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS:

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

4.2 NON-FUNCTIONAL REQUIREMENTS:

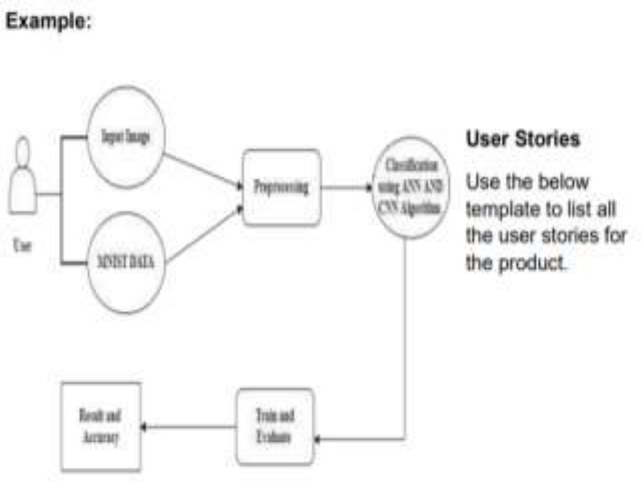


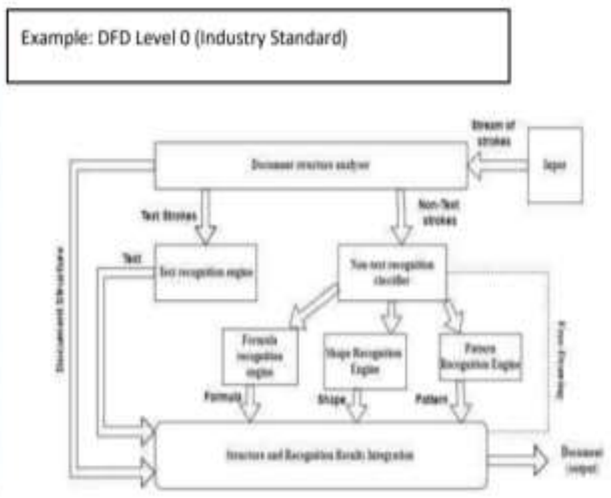
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Applications for digit recognition include filling out forms, processing bank checks, and sorting mail.
NFR-2	**Security**	1) The system generates a thorough description of the instantiation parameters, which might reveal information like the writing style, in addition to a categorization of the digit. 2) The generative models are capable of segmentation driven by recognition. 3) The procedure uses a relatively.
NFR-3	**Reliability**	The samples are used by the neural network to automatically deduce rules for reading handwritten digits. Furthermore, the network may learn more about handwriting and hence enhance its accuracy by increasing the quantity of training instances. Numerous techniques and algorithms, such as Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests,etc., can be used to recognise handwritten numbers.
NFR-4	**Performance**	With typed text in high-quality photos, optical character recognition (OCR) technology offers accuracy rates of greater than 99%. However, variances in spacing, abnormalities in handwriting, and the variety of human writing styles result in less precise character identification.
NFR-5	**Availability**	The tools like MNIST Dataset is made available to the general public so the model can be made available too.
NFR-6	**Scalability**	The model can be deployed to accommodate the workload of various users and can be trained using different handwritten digits too.

PROJECT DESIGN

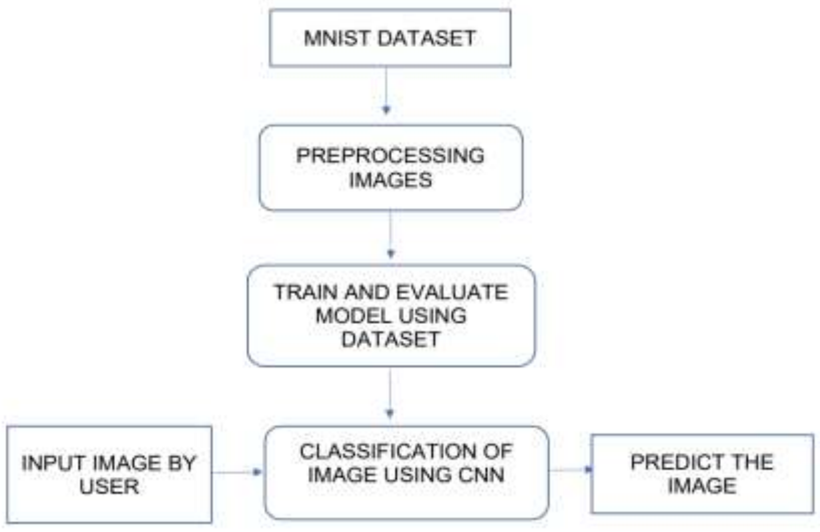
5.1 DATA FLOW DIAGRAMS:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.


Example:
User
Input Image
MNIST DATA
Preprocessing
Classification using ANN AND CNN Algorithms
Train and Evaluate
Result and Accuracy
User Stories
Use the below template to list all the user stories for the product.


Example: DFD Level 0 (Industry Standard)
Stream of strokes
Input
Document structure analyzer
Text Strokes
Non-Text strokes
Text
Text recognition engine
Non-text recognition classifier
Formula recognition engine
Shape Recognition Engine
Pattern Recognition Engine
Formula
Shape
Pattern
Document Structure
Free-Form Ink
Structure and Recognition Results Integration
Document (output)

5.2 SOLUTION & TECHNICAL ARCHITECTURE:


MNIST DATASET
PREPROCESSING IMAGES
TRAIN AND EVALUATE MODEL USING DATASET
INPUT IMAGE BY USER
CLASSIFICATION OF IMAGE USING CNN
PREDICT THE IMAGE

DATASET:

The MNIST data collection, which contains 70000 handwritten digits, is frequently utilized for this recognition method. In order to train these photos and create a deep learning model, we use artificial neural networks. A web application is developed that allows users to upload pictures of handwritten numbers. The model examines this picture. The 60,000 training and 10,000 testing labeled handwritten digit images in the MNIST Handwritten Digit Recognition Dataset.

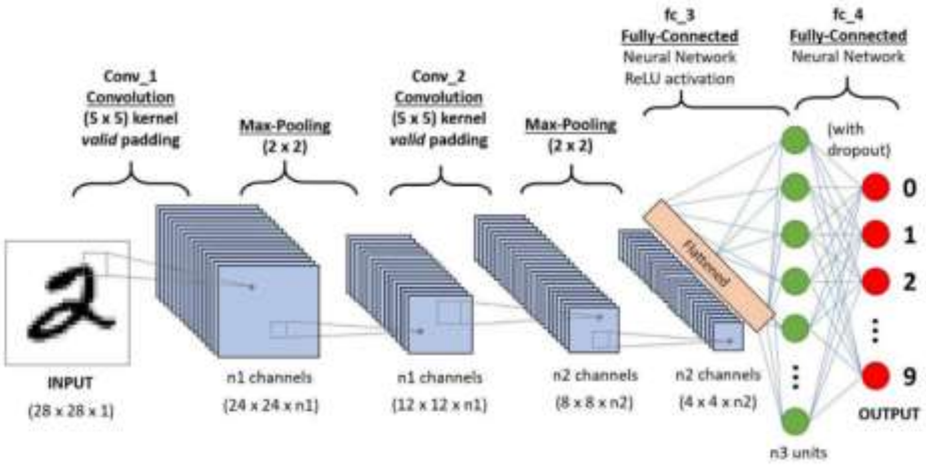
APPROACH:

This project will be approached utilizing a convolutional neural network. (CNN)

● **The convolutional and max-pool layer**: The feature is extracted from the input image and got max pooled to reduce the dimensionality of the image without any changes in the extracted feature.

● **The dense layer**: The flattened output from the max-pool layer is fed to a feed-forward neural network and backpropagation applied to every iteration of training.

● **The output layer**: The nodes in this stratum are referred to as output units. It gives us access to the neural network's final prediction, which may be used to make final predictions.



A neural network is a model of the brain's operations. It is made up of numerous layers with a variety of activations; these activations mimic the neurons in our brain. An attempt is made by a neural network to learn a set of parameters from a set of data that might aid in understanding the underlying relationships. Since neural networks are capable of adapting to changing input, the network can produce the best outcome without having to change the output criterion.

METHODOLOGY:

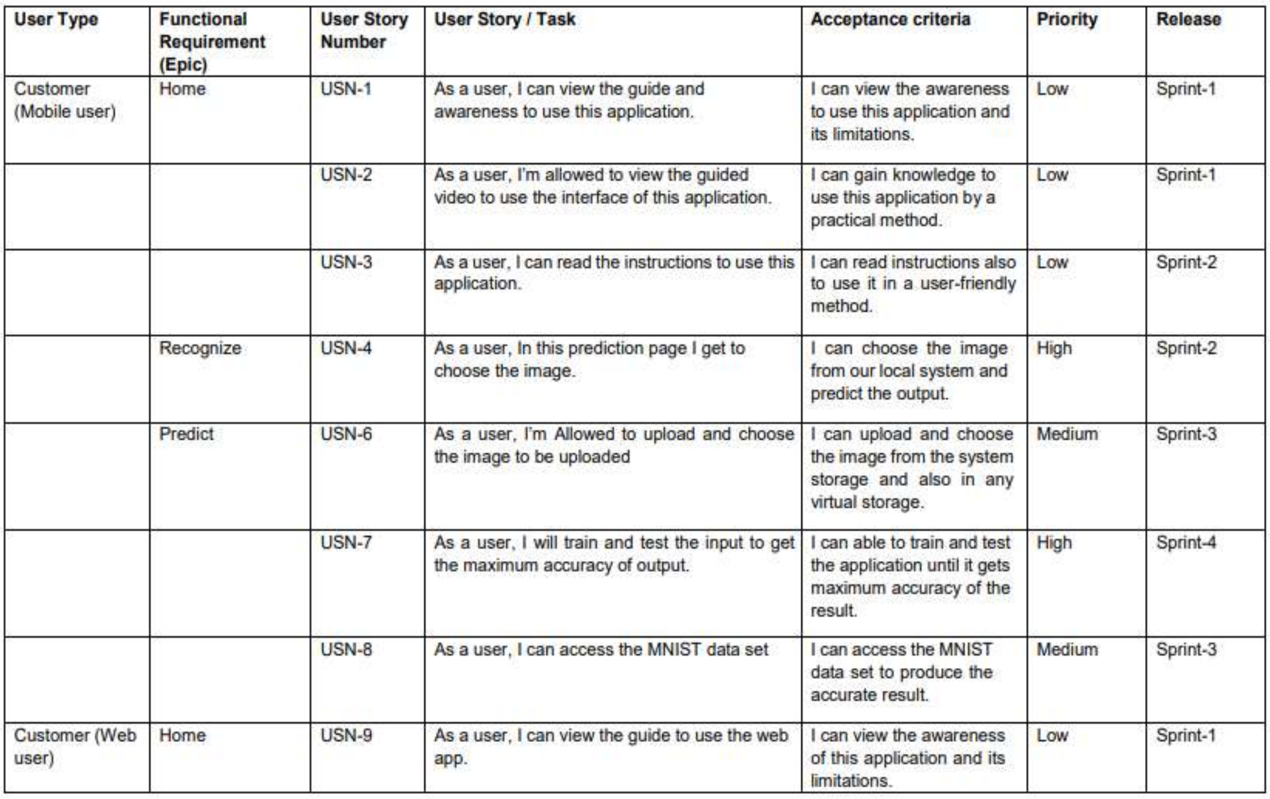
A neural network with one hidden layer and 100 activation units has been put into practice (excluding bias units). The features (X) and labels (Y) were retrieved after the data was loaded from a mat file. To prevent overflow during computation, features are then scaled into a range of [0,1] by dividing by 255. 10,000 testing cases and 60,000 training examples make up the data. With the training data, feedforward is used to calculate the hypothesis, and backpropagation is then used to lower the error between the layers. To combat overfitting, the regularization parameter lambda is set to 0.1. To identify the model that fits the situation the optimizer runs for 70 times. WORKING: After receiving an input, neural networks change it using a number of hidden layers. Each group of neurons in a hidden layer is completely linked to every other neuron in the layer above it. One layer of neurons have perfect independence from one anotherr. The "output layer" is the final layer to be fully connected. CONVOLUTION

LAYER : The foundational component of a CNN is the convolutional layer. The parameters of the layer are a set of learnable filters (or kernels) that cover the entire depth of the input volume but have a narrow receptive field. Each filter is convolved across the width and height of the input volume during the forward pass, computing the dot product between each filter entry and the input to create a two-dimensional activation map of the filter. As a result, the network picks up filters that turn on when it detects a certain kind of feature at a particular spatial location in the input.

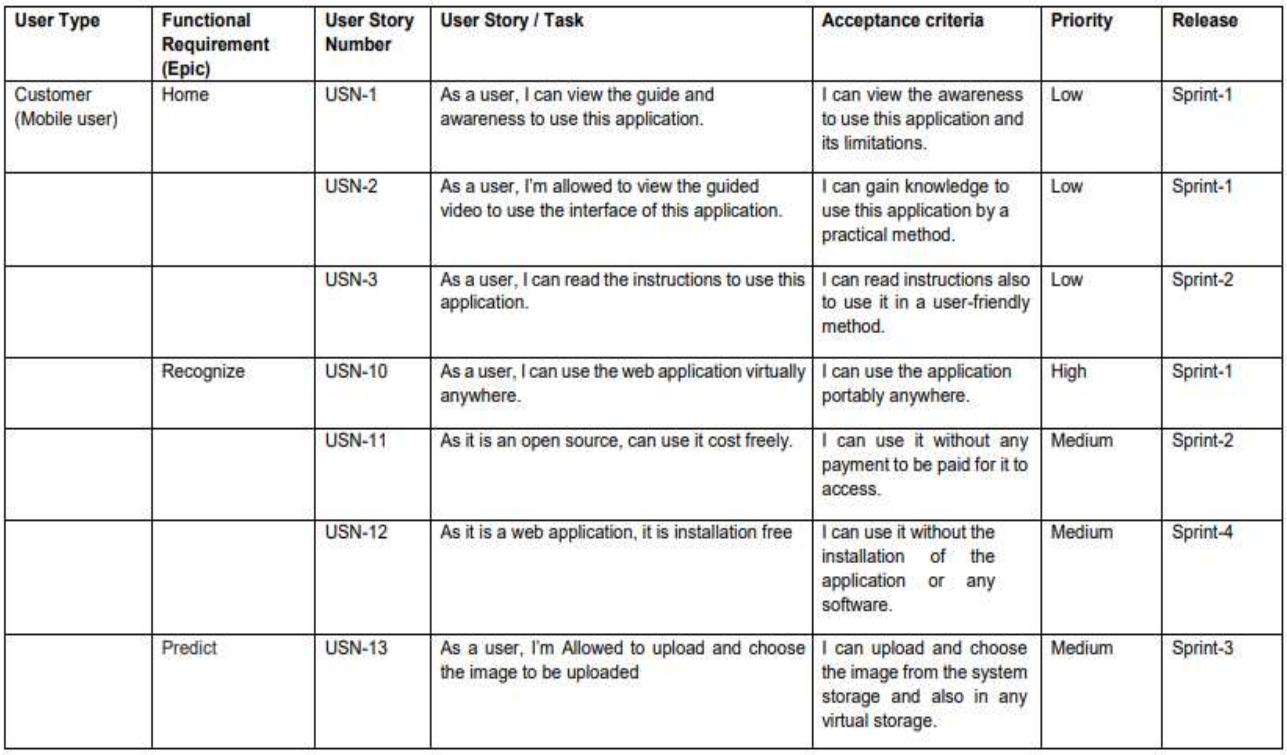
FEATURE EXTRACTION : All neurons in a feature share the same weights .In this way all neurons detect the same feature at different positions in the input image. Reduce the number of free parameters.

TENSORFLOW : An open-source machine learning library for both research and production is called TensorFlow. TensorFlow provides developers of all skill levels with APIs for desktop, mobile, web, and cloud applications. We can achieve text output and sound output by scanning the number digit and converting it to PNG format using the python3 command in the terminal.

5.3 USER STORIES:



User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Home	USN-1	As a user, I can view the guide and awareness to use this application.	I can view the awareness to use this application and its limitations.	Low	Sprint-1
		USN-2	As a user, I'm allowed to view the guided video to use the interface of this application.	I can gain knowledge to use this application by a practical method.	Low	Sprint-1
		USN-3	As a user, I can read the instructions to use this application.	I can read instructions also to use it in a user-friendly method.	Low	Sprint-2
	Recognize	USN-4	As a user, In this prediction page I get to choose the image.	I can choose the image from our local system and predict the output.	High	Sprint-2
	Predict	USN-6	As a user, I'm Allowed to upload and choose the image to be uploaded	I can upload and choose the image from the system storage and also in any virtual storage.	Medium	Sprint-3
		USN-7	As a user, I will train and test the input to get the maximum accuracy of output.	I can able to train and test the application until it gets maximum accuracy of the result.	High	Sprint-4
		USN-8	As a user, I can access the MNIST data set	I can access the MNIST data set to produce the accurate result.	Medium	Sprint-3
Customer (Web user)	Home	USN-9	As a user, I can view the guide to use the web app.	I can view the awareness of this application and its limitations.	Low	Sprint-1



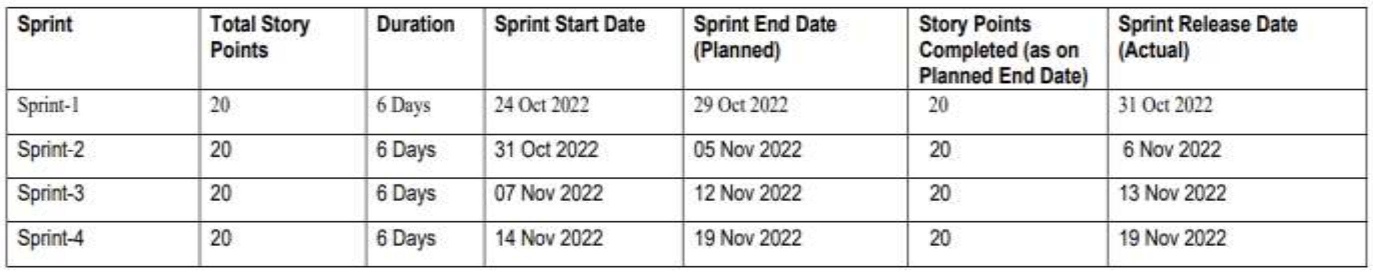
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Home	USN-1	As a user, I can view the guide and awareness to use this application.	I can view the awareness to use this application and its limitations.	Low	Sprint-1
		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-10	As a user, I can use the web application virtually anywhere.	I can use the application portably anywhere.	High	Sprint-1
		USN-11	As it is an open source, can use it cost freely.	I can use it without any payment to be paid for it to access.	Medium	Sprint-2
		USN-12	As it is a web application, it is installation free	I can use it without the installation of the application or any software.	Medium	Sprint-4
	Predict	USN-13	As a user, I'm Allowed to upload and choose the image to be uploaded	I can upload and choose the image from the system storage and also in any virtual storage.	Medium	Sprint-3

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & 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 need to collect the data with different handwriting to train the model	6	High	Thiruvazhi Dhinesh Kumar S, Dinesh M, Gowtham N
Sprint-1	Importing libraries	USN-2	As a user, I have to implement necessary libraries in python packages.	4	Low	Thiruvazhi Dhinesh Kumar S, Dinesh M, Gowtham N
Sprint-1	Data preprocessing	USN-3	As a user, I can load the dataset, handle the missing values, scale and split the data.	10	Medium	Thiruvazhi Dhinesh Kumar S, Dinesh M, Gowtham N
Sprint-2	Model building	USN-4	As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit.	5	High	Ruban Chakaravarthi V, Thiruvazhi Dhinesh Kumar S, Gowtham N
Sprint-2	Add the CNN layers	USN-5	Add input convolutional layer, max-pooling layer, flatten, hidden and output layers to the model.	5	High	Ruban Chakaravarthi V, Thiruvazhi Dhinesh Kumar S, Gowtham N
Sprint- 2	Compile the model	USN-6	As a user, compile the model for trained dataset.	2	Medium	Ruban Chakaravarthi V, Thiruvazhi Dhinesh Kumar S, Gowtham N
Sprint-2	Train and test the model	USN-7	As a user, train and test the model for the dataset collected and data are validated.	4	High	Ruban Chakaravarthi V, Thiruvazhi Dhinesh Kumar S, Gowtham N
Sprint-2	Save the model	USN-8	As a user, the compiled data are saved and integrated with an android application or web application.	2	Low	Ruban Chakaravarthi V, Thiruvazhi Dhinesh Kumar S, Gowtham N
Sprint-3	Building UI application	USN-9	As a user upload the input image that contains handwritten digits.	10	Medium	Dinesh M, Ruban Chakaravarthi V, Thiruvazhi Dhinesh Kumar S
Sprint-3		USN-10	As a user, I can provide the fundamental details about the usage of application to customer.	5	Low	Dinesh M, Ruban Chakaravarthi V, Thiruvazhi Dhinesh Kumar S
Sprint-3		USN-11	As a user, I can see the predicted or recognized digits in the application.	5	Medium	Dinesh M, Ruban Chakaravarthi V, Thiruvazhi Dhinesh Kumar S
Sprint-4	Train the model on IBM	USN-12	As a user train the model in IBM cloud and integrate the results.	10	High	Gowtham N, Dinesh M, Thiruvazhi Dhinesh Kumar S
Sprint-4	Cloud Deployment	USN-13	As a user, I can access the web application and make the use of the product from anywhere.	10	High	Gowtham N, Dinesh M, Thiruvazhi Dhinesh Kumar S

Project Tracker, Velocity & Burndown Chart:



Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	31 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	6 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	13 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

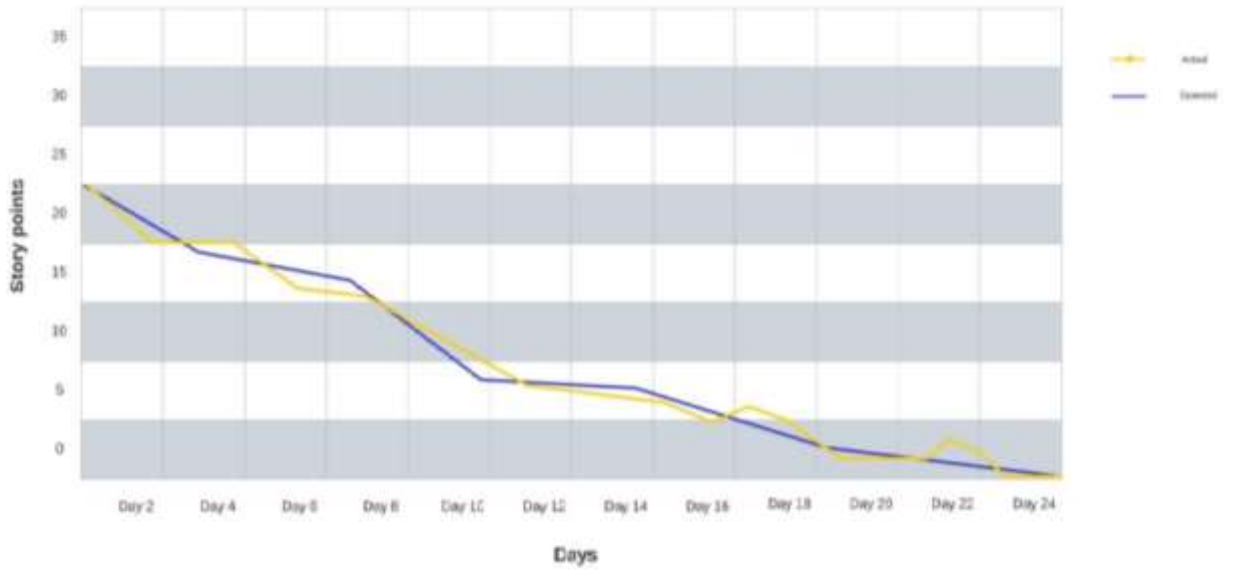
Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day).

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



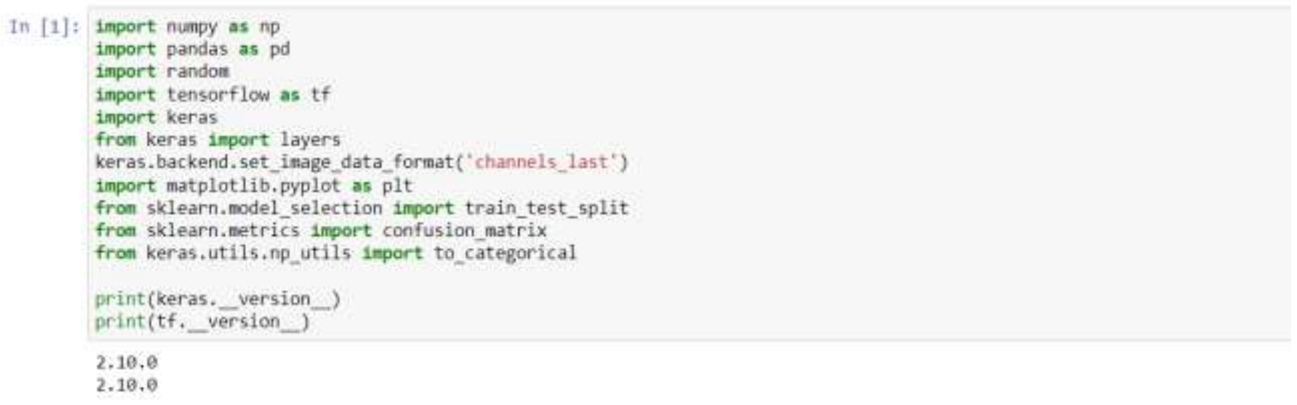
6.2 SPRINT DELIVERY SCHEDULE:

S.No.	Milestones	Activities	Timeline
1	Literature Survey	Literature survey on the handwritten digit recognition system and information gathering.	10 September 2022
2.	Empathy Map	Prepared empathy map canvas to capture the user gains and pains.	10 September 2022
3.	Brainstorming and Ideation	Ideas are listed and top 3 ideas are prioritized based on the feasibility and importance.	17 September 2022
4.	Proposed Solution Document	Proposed solution document is prepared which includes Novelty, feasibility of idea, social impact, scalability of solution, etc.	24 September 2022
5.	Problem Solution Fit	Includes customer segments and customer constraints, the problem root cause and jobs to be done.	1 October 2022
6.	Solution Architecture	From data collection to digit recognition by the web application are represented in architectural diagram.	1 October 2022
7.	Customer Journey Map	Prepare Customer Journey maps to understand user interactions and experiences with the application	8 October 2022
8.	Functional Requirement Document	Functional requirements and non functional requirements alike scalability and accuracy are described.	15 October 2022
9.	Data Flow Diagrams	Data flow diagram and user stories are prepared and four sprint phases are described.	15 October 2022
10.	Technology Architecture	Technical flow graphs are created and the functions of technical stacks are defined.	15 October 2022

CODING AND SOLUTIONING

7.1 FEATURE 1:

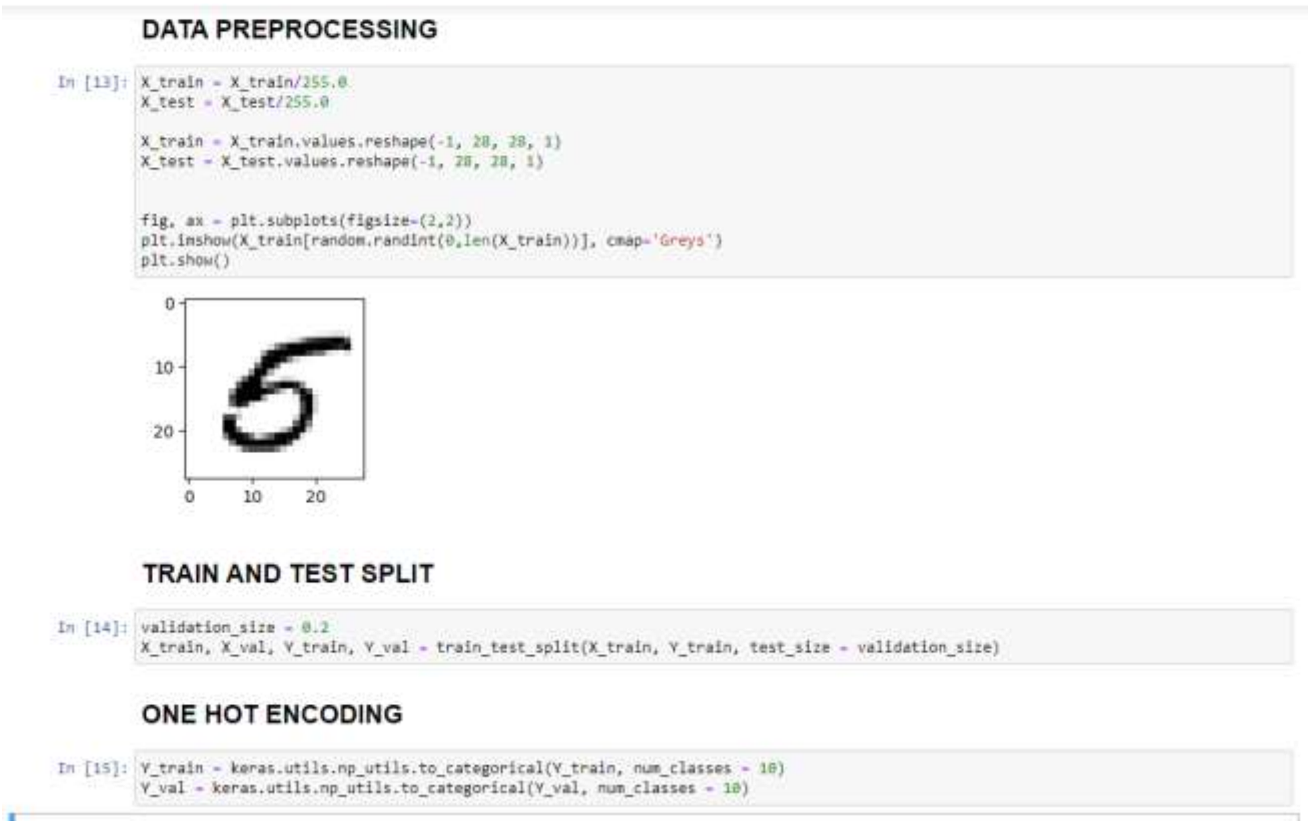
Importing Required Libraries:



```
In [1]: import numpy as np
        import pandas as pd
        import random
        import tensorflow as tf
        import keras
        from keras import layers
        keras.backend.set_image_data_format('channels_last')
        import matplotlib.pyplot as plt
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import confusion_matrix
        from keras.utils.np_utils import to_categorical

        print(keras.__version__)
        print(tf.__version__)

        2.10.0
        2.10.0
```

Here, we are importing required libraries like numpy, pandas, tensorflow and keras.

Data Preprocessing:



DATA PREPROCESSING

```
In [13]: X_train = X_train/255.0
         X_test = X_test/255.0

         X_train = X_train.values.reshape(-1, 28, 28, 1)
         X_test = X_test.values.reshape(-1, 28, 28, 1)


         fig, ax = plt.subplots(figsize=(2,2))
         plt.imshow(X_train[random.randint(0,len(X_train))], cmap='Greys')
         plt.show()
```

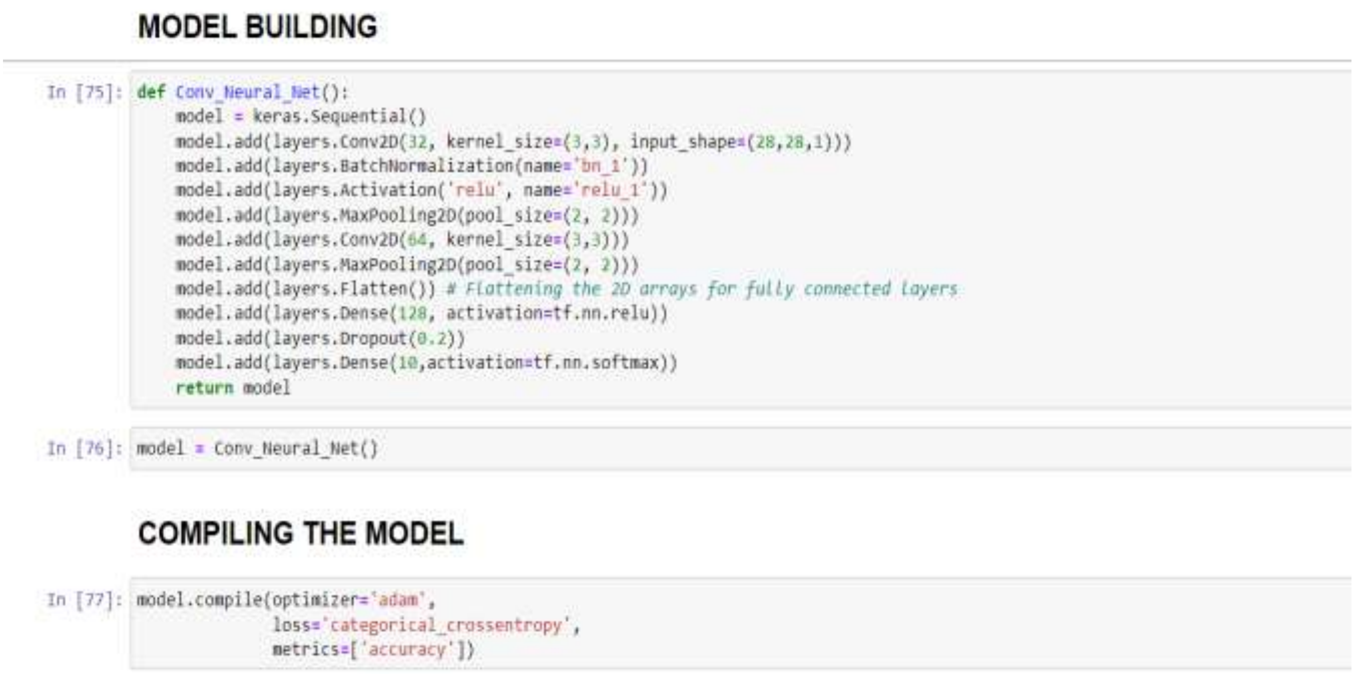
TRAIN AND TEST SPLIT

```
In [14]: validation_size = 0.2
         X_train, X_val, Y_train, Y_val = train_test_split(X_train, Y_train, test_size = validation_size)
```

ONE HOT ENCODING

```
In [15]: Y_train = keras.utils.np_utils.to_categorical(Y_train, num_classes = 10)
         Y_val = keras.utils.np_utils.to_categorical(Y_val, num_classes = 10)
```

Model Building:



MODEL BUILDING

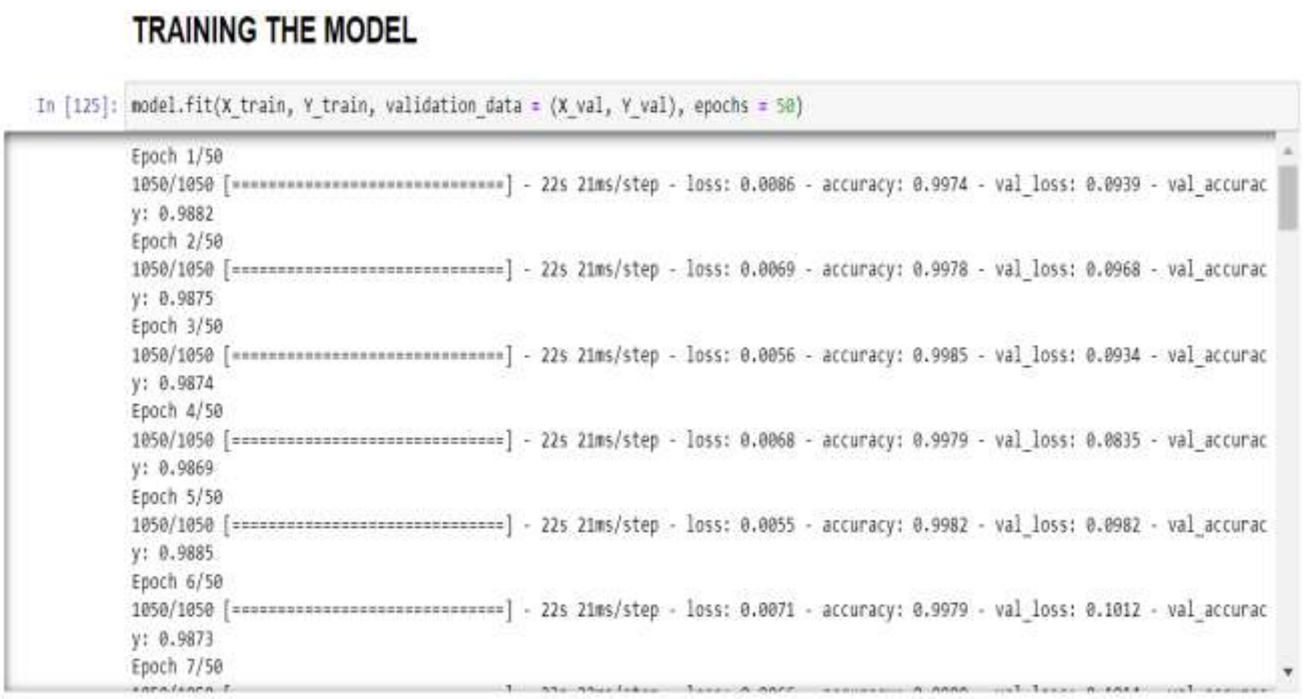
```
In [75]: def Conv_Neural_Net():
             model = keras.Sequential()
             model.add(layers.Conv2D(32, kernel_size=(3,3), input_shape=(28,28,1)))
             model.add(layers.BatchNormalization(name='bn_1'))
             model.add(layers.Activation('relu', name='relu_1'))
             model.add(layers.MaxPooling2D(pool_size=(2, 2)))
             model.add(layers.Conv2D(64, kernel_size=(3,3)))
             model.add(layers.MaxPooling2D(pool_size=(2, 2)))
             model.add(layers.Flatten()) # Flattening the 2D arrays for fully connected layers
             model.add(layers.Dense(128, activation=tf.nn.relu))
             model.add(layers.Dropout(0.2))
             model.add(layers.Dense(10,activation=tf.nn.softmax))
             return model

In [76]: model = Conv_Neural_Net()
```

COMPILING THE MODEL

```
In [77]: model.compile(optimizer='adam',
                       loss='categorical_crossentropy',
                       metrics=['accuracy'])
```

7.2 FEATURE 2:

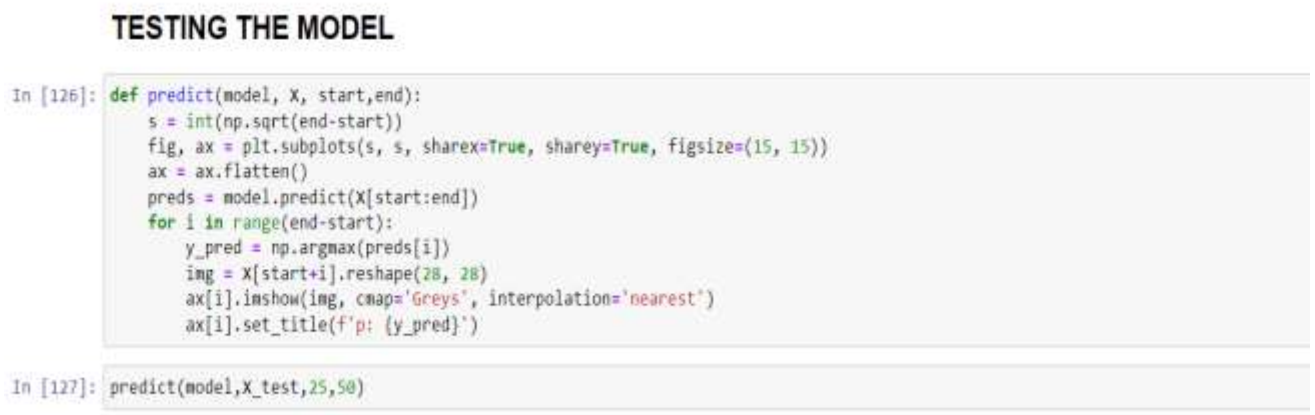
Training the Model:



TRAINING THE MODEL

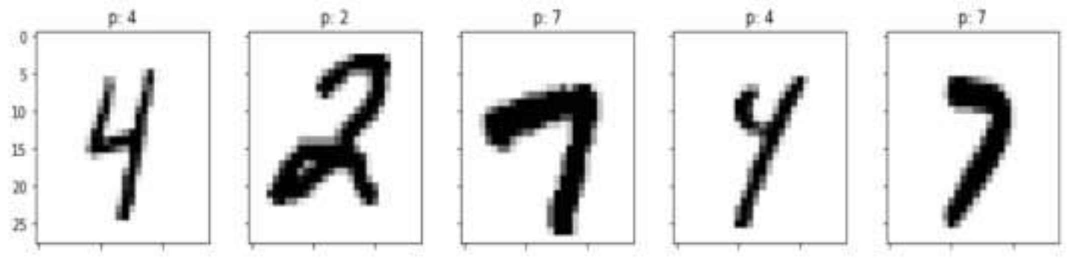
```
In [125]: model.fit(X_train, Y_train, validation_data = (X_val, Y_val), epochs = 50)

          Epoch 1/50
          1050/1050 [==============================] - 22s 21ms/step - loss: 0.0086 - accuracy: 0.9974 - val_loss: 0.0939 - val_accurac
          y: 0.9882
          Epoch 2/50
          1050/1050 [==============================] - 22s 21ms/step - loss: 0.0069 - accuracy: 0.9978 - val_loss: 0.0968 - val_accurac
          y: 0.9875
          Epoch 3/50
          1050/1050 [==============================] - 22s 21ms/step - loss: 0.0056 - accuracy: 0.9985 - val_loss: 0.0934 - val_accurac
          y: 0.9874
          Epoch 4/50
          1050/1050 [==============================] - 22s 21ms/step - loss: 0.0068 - accuracy: 0.9979 - val_loss: 0.0835 - val_accurac
          y: 0.9869
          Epoch 5/50
          1050/1050 [==============================] - 22s 21ms/step - loss: 0.0055 - accuracy: 0.9982 - val_loss: 0.0982 - val_accurac
          y: 0.9885
          Epoch 6/50
          1050/1050 [==============================] - 22s 21ms/step - loss: 0.0071 - accuracy: 0.9979 - val_loss: 0.1012 - val_accurac
          y: 0.9873
          Epoch 7/50
```

Evaluating the Model and Predicting:

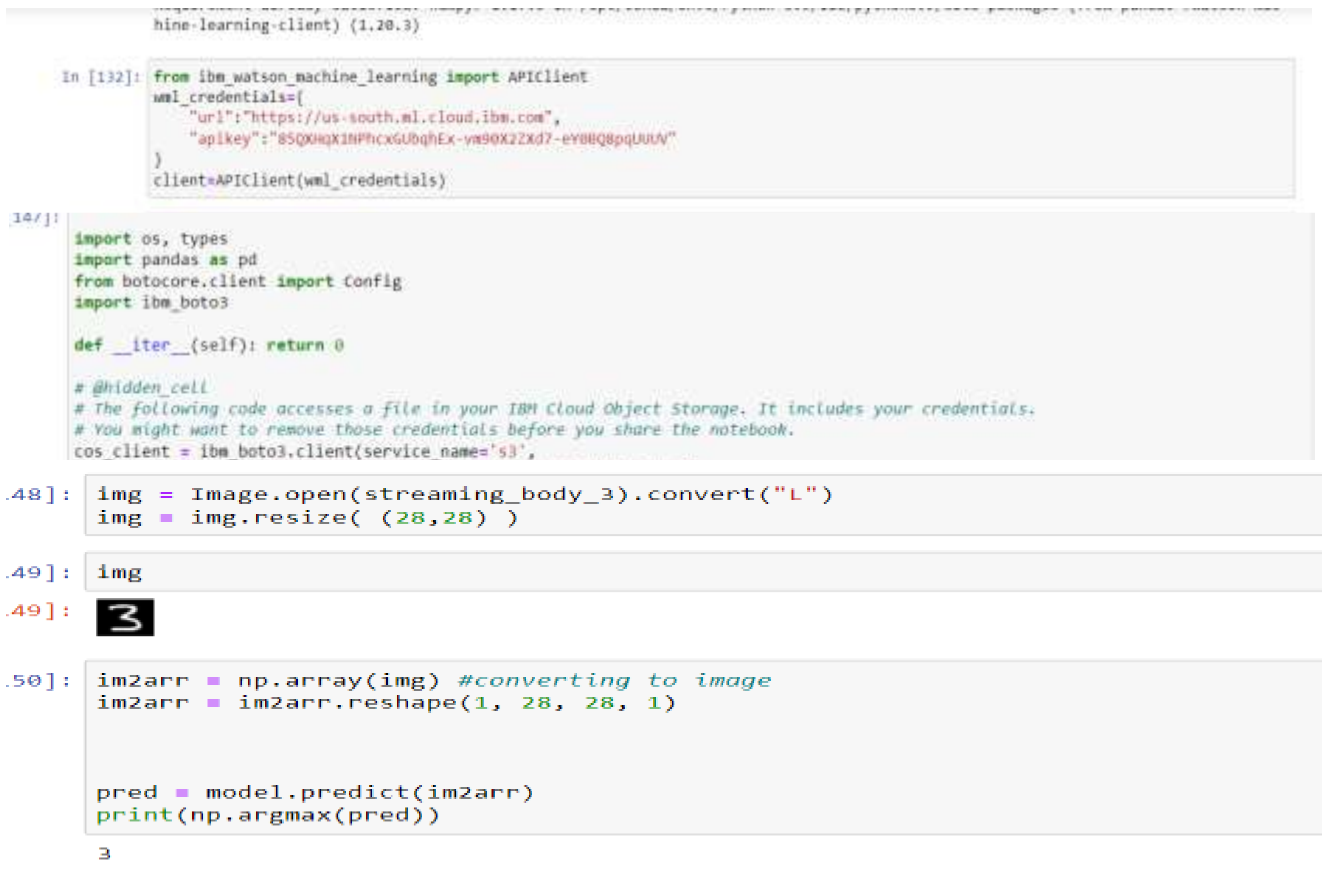


TESTING THE MODEL

```
In [126]: def predict(model, X, start,end):
              s = int(np.sqrt(end-start))
              fig, ax = plt.subplots(s, s, sharex=True, sharey=True, figsize=(15, 15))
              ax = ax.flatten()
              preds = model.predict(X[start:end])
              for i in range(end-start):
                  y_pred = np.argmax(preds[i])
                  img = X[start+i].reshape(28, 28)
                  ax[i].imshow(img, cmap='Greys', interpolation='nearest')
                  ax[i].set_title(f'p: {y_pred}')

In [127]: predict(model,X_test,25,50)
```



Cloud Deployment:



```
hine-learning-client) (1.20.3)

In [132]: from ibm_watson_machine_learning import APIClient
          wml_credentials={
              "url":"https://us-south.ml.cloud.ibm.com",
              "apikey":"85QXHqX1NPhcxGUbqhEx-vm90X2ZXd7-eY8BQ8pqUUUV"
          )
          client=APIClient(wml_credentials)

147]:
          import os, types
          import pandas as pd
          from botocore.client import Config
          import ibm_boto3

          def __iter__(self): return 0

          # @hidden_cell
          # The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
          # You might want to remove those credentials before you share the notebook.
          cos_client = ibm_boto3.client(service_name='s3',

.48]:  img = Image.open(streaming_body_3).convert("L")
       img = img.resize( (28,28) )

.49]:  img

.49]:  3

.50]:  im2arr = np.array(img) #converting to image
       im2arr = im2arr.reshape(1, 28, 28, 1)


       pred = model.predict(im2arr)
       print(np.argmax(pred))

       3
```

TESTING

8.1 USER ACCEPTANCE TESTING:

Date	18 November 2022
Team ID	PNT2022TMID35510
Project Name	Project – A novel method for handwritten digit recognition
Maximum Marks	4 Marks

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the **A novel method for handwritten digit recognition** project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	13	4	2	5	24
Duplicate	1	0	0	2	3
External	2	3	1	1	7
Fixed	4	6	4	10	24
Not Reproduced	1	1	1	1	4
Skipped	1	2	0	1	4
Won't Fix	0	5	2	4	11
Totals	22	21	10	24	77

8.2 TEST CASE ANALYSIS:

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Security	2	0	0	2
Performance	3	0	0	3
Exception Reporting	9	0	0	9

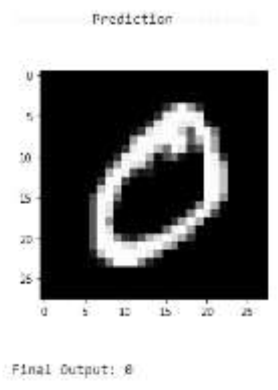
<u>RESULTS</u>

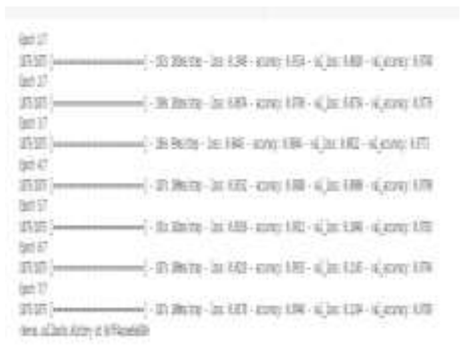
9.1 PERFORMANCE METRICS:

Date	18 November 2022
Team ID	PNT2022TMID35510
Project Name	Project – A novel method for handwritten digit recognition
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Loss: 0.0181 Val_loss:0.1560	 Final Output: 0

2.	Accuracy	Training Accuracy - 97.87 Validation Accuracy -95.48	 [illegible]

ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

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

DISADVANTAGES:

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

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.

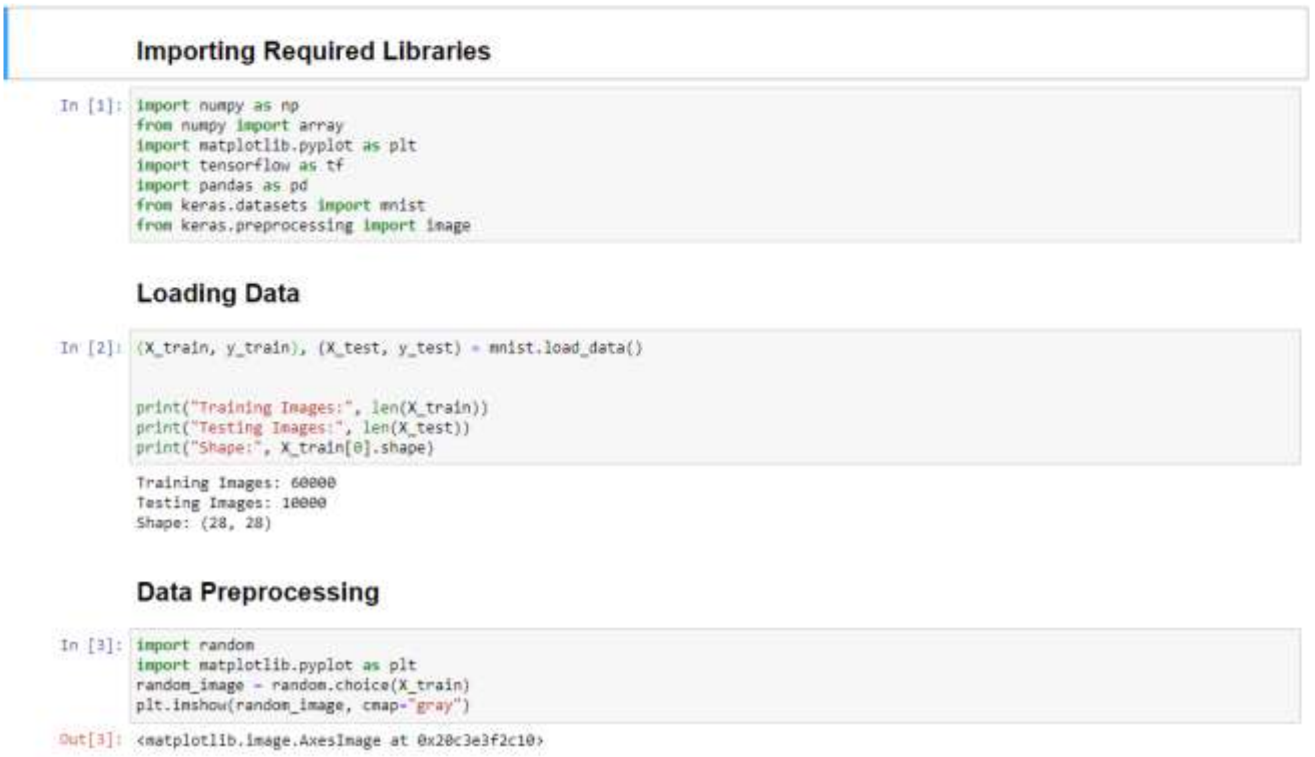
FUTURE SCOPE

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:



Importing Required Libraries

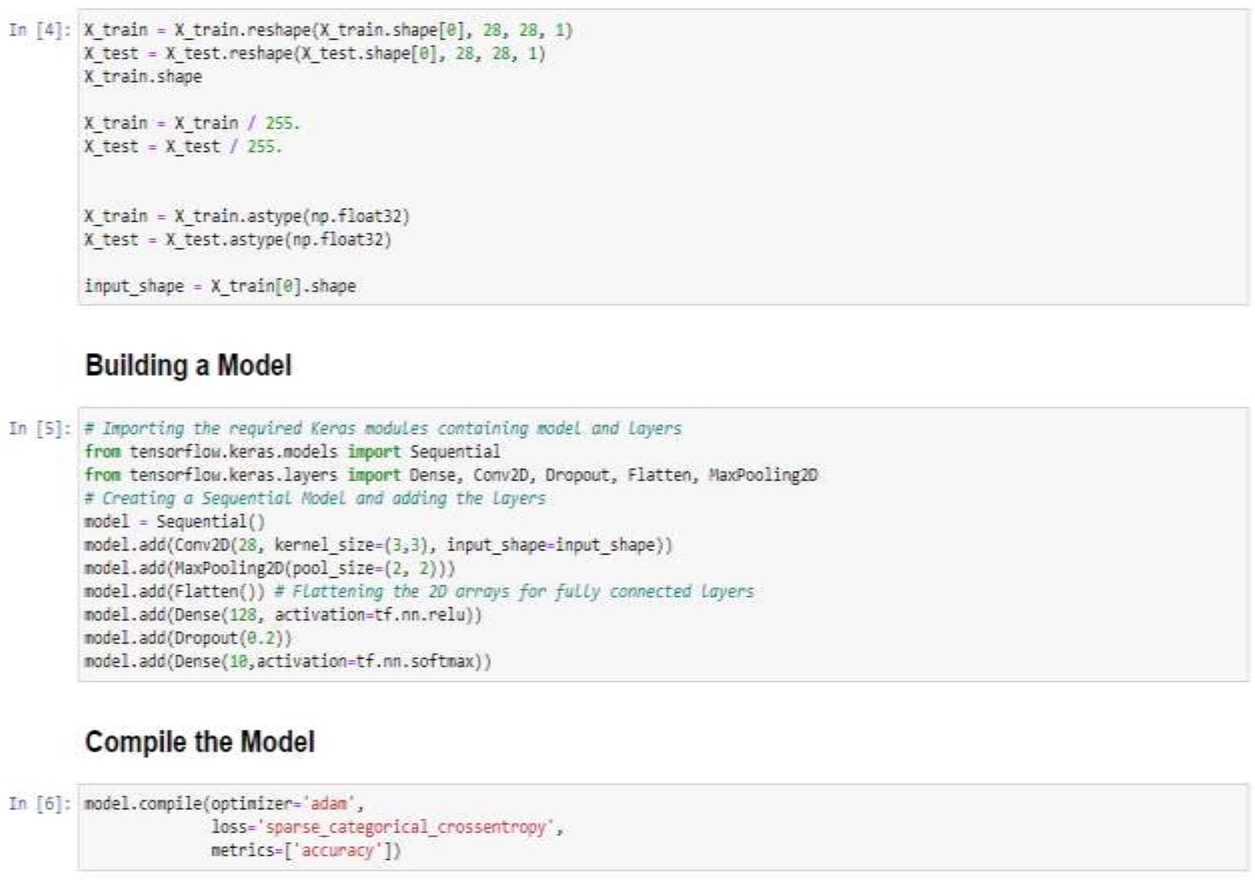
```
In [1]: import numpy as np
        from numpy import array
        import matplotlib.pyplot as plt
        import tensorflow as tf
        import pandas as pd
        from keras.datasets import mnist
        from keras.preprocessing import image
```

Loading Data

```
In [2]: (X_train, y_train), (X_test, y_test) = mnist.load_data()


        print("Training Images:", len(X_train))
        print("Testing Images:", len(X_test))
        print("Shape:", X_train[0].shape)

        Training Images: 60000
        Testing Images: 10000
        Shape: (28, 28)
```

Data Preprocessing

```
In [3]: import random
        import matplotlib.pyplot as plt
        random_image = random.choice(X_train)
        plt.imshow(random_image, cmap="gray")

Out[3]: <matplotlib.image.AxesImage at 0x20c3e3f2c10>
```



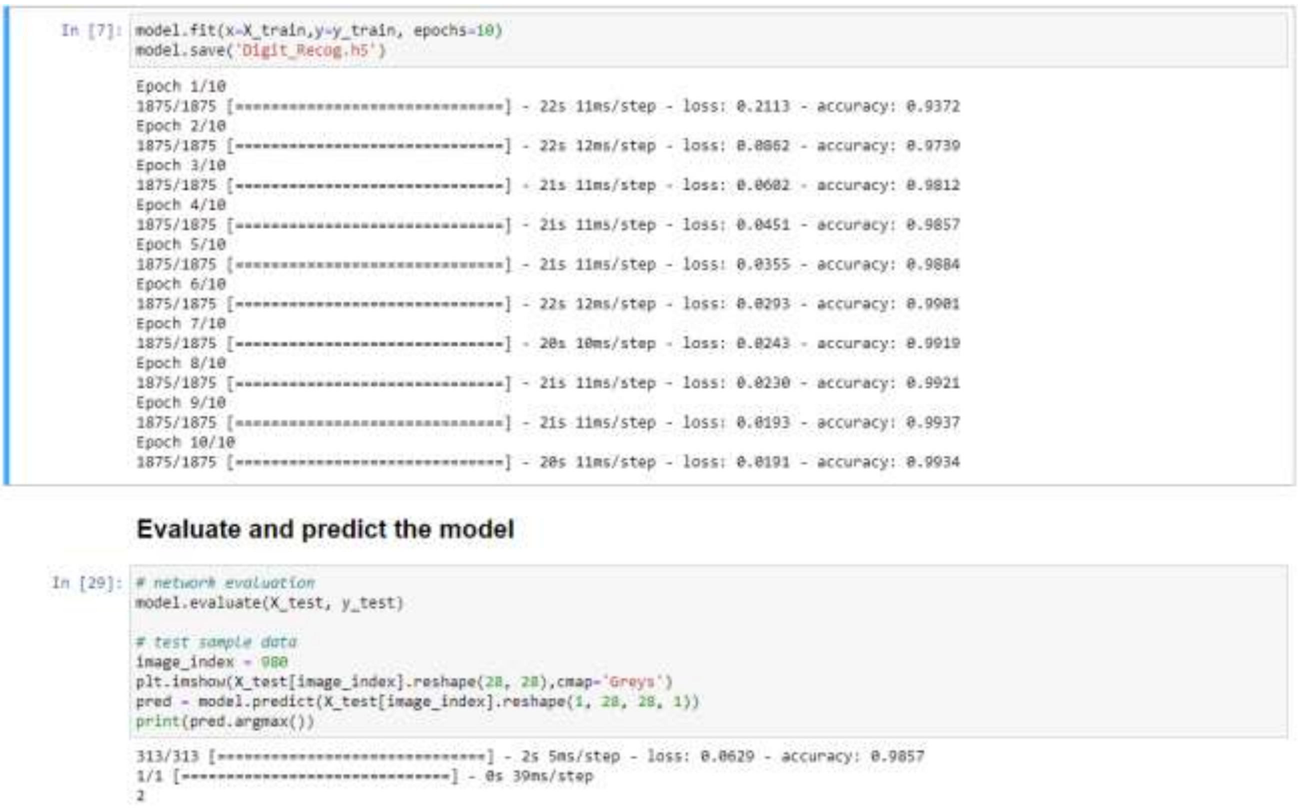
```
In [4]: X_train = X_train.reshape(X_train.shape[0], 28, 28, 1)
        X_test = X_test.reshape(X_test.shape[0], 28, 28, 1)
        X_train.shape

        X_train = X_train / 255.
        X_test = X_test / 255.


        X_train = X_train.astype(np.float32)
        X_test = X_test.astype(np.float32)

        input_shape = X_train[0].shape
```

Building a Model

```
In [5]: # Importing the required Keras modules containing model and layers
        from tensorflow.keras.models import Sequential
        from tensorflow.keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D
        # Creating a Sequential Model and adding the layers
        model = Sequential()
        model.add(Conv2D(28, kernel_size=(3,3), input_shape=input_shape))
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Flatten()) # Flattening the 2D arrays for fully connected layers
        model.add(Dense(128, activation=tf.nn.relu))
        model.add(Dropout(0.2))
        model.add(Dense(10,activation=tf.nn.softmax))
```

Compile the Model

```
In [6]: model.compile(optimizer='adam',
                      loss='sparse_categorical_crossentropy',
                      metrics=['accuracy'])
```



```
In [7]: model.fit(x=X_train,y=y_train, epochs=10)
        model.save('Digit_Recog.h5')

Epoch 1/10
1875/1875 [==============================] - 22s 11ms/step - loss: 0.2113 - accuracy: 0.9372
Epoch 2/10
1875/1875 [==============================] - 22s 12ms/step - loss: 0.0862 - accuracy: 0.9739
Epoch 3/10
1875/1875 [==============================] - 21s 11ms/step - loss: 0.0602 - accuracy: 0.9812
Epoch 4/10
1875/1875 [==============================] - 21s 11ms/step - loss: 0.0451 - accuracy: 0.9857
Epoch 5/10
1875/1875 [==============================] - 21s 11ms/step - loss: 0.0355 - accuracy: 0.9884
Epoch 6/10
1875/1875 [==============================] - 22s 12ms/step - loss: 0.0293 - accuracy: 0.9901
Epoch 7/10
1875/1875 [==============================] - 20s 10ms/step - loss: 0.0243 - accuracy: 0.9919
Epoch 8/10
1875/1875 [==============================] - 21s 11ms/step - loss: 0.0230 - accuracy: 0.9921
Epoch 9/10
1875/1875 [==============================] - 21s 11ms/step - loss: 0.0193 - accuracy: 0.9937
Epoch 10/10
1875/1875 [==============================] - 20s 11ms/step - loss: 0.0191 - accuracy: 0.9934
```

Evaluate and predict the model

```
In [29]: # network evaluation
         model.evaluate(X_test, y_test)

         # test sample data
         image_index = 980
         plt.imshow(X_test[image_index].reshape(28, 28),cmap='Greys')
         pred = model.predict(X_test[image_index].reshape(1, 28, 28, 1))
         print(pred.argmax())

313/313 [==============================] - 2s 5ms/step - loss: 0.0629 - accuracy: 0.9857
1/1 [==============================] - 0s 39ms/step
2
```

GITHUB:

https://github.com/IBM-EPBL/IBM-Project-6377-1658827484

PROJECT DEMO LINK:

https://drive.google.com/drive/folders/1zH1MZw10hlSbfMLQAQ094LbeemxxMD5G?usp=share_link