A NOVEL METHOD FOR HANDWRITTEN DIGIT

RECOGNITION SYSTEM

Submitted By

TEAM ID: PMT2022TMID20889

TEAM LEADER: PADMAJA S 412419205065
TEAM MEMBER 1:AISHWARYA B 412419205006
TEAM MEMBER 2: PAVITHRA K 412419205066
TEAM MEMBER 3: LAVANYA CB 412419205047

TABLE OF CONTENTS

1. **INTRODUCTION**
    1. Project Overview
    2. Purpose
2. **LITERATURE SURVEY**
    1. Existing problem
    2. References
    3. Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
    1. Empathy Map Canvas
    2. Ideation & Brainstorming
    3. Proposed Solution
    4. Problem Solution fit
4. **REQUIREMENT ANALYSIS**
    1. Functional requirement
    2. Non-Functional requirements
5. **PROJECT DESIGN**
    1. Data Flow Diagrams
    2. Solution & Technical Architecture
    3. User Stories
6. **PROJECT PLANNING & SCHEDULING**
    1. Sprint Planning & Estimation
    2. Sprint Delivery Schedule
    3. Reports from JIRA
7. **CODING & SOLUTIONING**
    1. Feature 1
    2. Feature 2
8. **TESTING**
    1. Test Cases

8.1Test Cases
8.2User Acceptance Testing

9. **RESULTS**
    1. Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**

Source Code

GitHub & Project Demo Link

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

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.

2 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

A Novel Handwritten Digit Classification System Based on Convolutional Neural Network Approach (2021)

The paper makes the following contributions First, the size of the effective receptive field (ERF) is determined after taking domain knowledge into account. They choose a typical filter size with the aid of the ERF calculation, improving the classification accuracy of our CNN. Second, excessive data produces inaccurate results, which has a detrimental impact on classification accuracy. Thirdly, data augmentation has been suggested as a way to reduce training and validation errors. Fourthly, the paper suggests adding an additive white Gaussian noise with a threshold of 0.5 to the MNIST dataset in order to imitate the natural factors that can affect image quality in the real world. With a recognition accuracy of 99.98% and 99.40% with 50% noise.

Novel Deep Neural Network Model for Handwritten Digit Classification and Recognition(2021)

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. 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.

A Novel Method For Hand Written Digit Recognition Using Deep Learning(2019)

This compares the results of some of the most widely used Machine Learning Algorithms like CNNconvolution neural networks and with Deep Learning algorithm like multilayer CNN using Keras with Theano and TensorFlow. MNIST is a dataset which is widely used for handwritten digit recognition. The dataset consists of 60,000 training images and 10,000 test images. The artificial neural networks plays main role in image processing field.

Handwritten Character Recognition using Neural Network and TensorFlow(2019)

The 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 The objective is to create software that is extremely accurate and that has a minimum level of spatial and temporal complexity. The feed forward model in neural networks is the back-propagation algorithm that was primarily used to classify the characters. The paper will describe the best approach to get

more than 90% accuracy.

Handwritten Digits Recognition with Artificial Neural Network(2017)

A multi-layer fully connected neural network with one hidden layer for handwritten digits recognition is implemented. They have used digit images pixels as features vector and ANN as classifiers for handwritten digits recognition. The testing has been conducted from publicly available MNIST handwritten database. From the MNIST database, they extracted 28,000 digits images for training and 14,000 digits images for performing the test. Our multilayer artificial neural network has an accuracy of 99.60% with test performance.

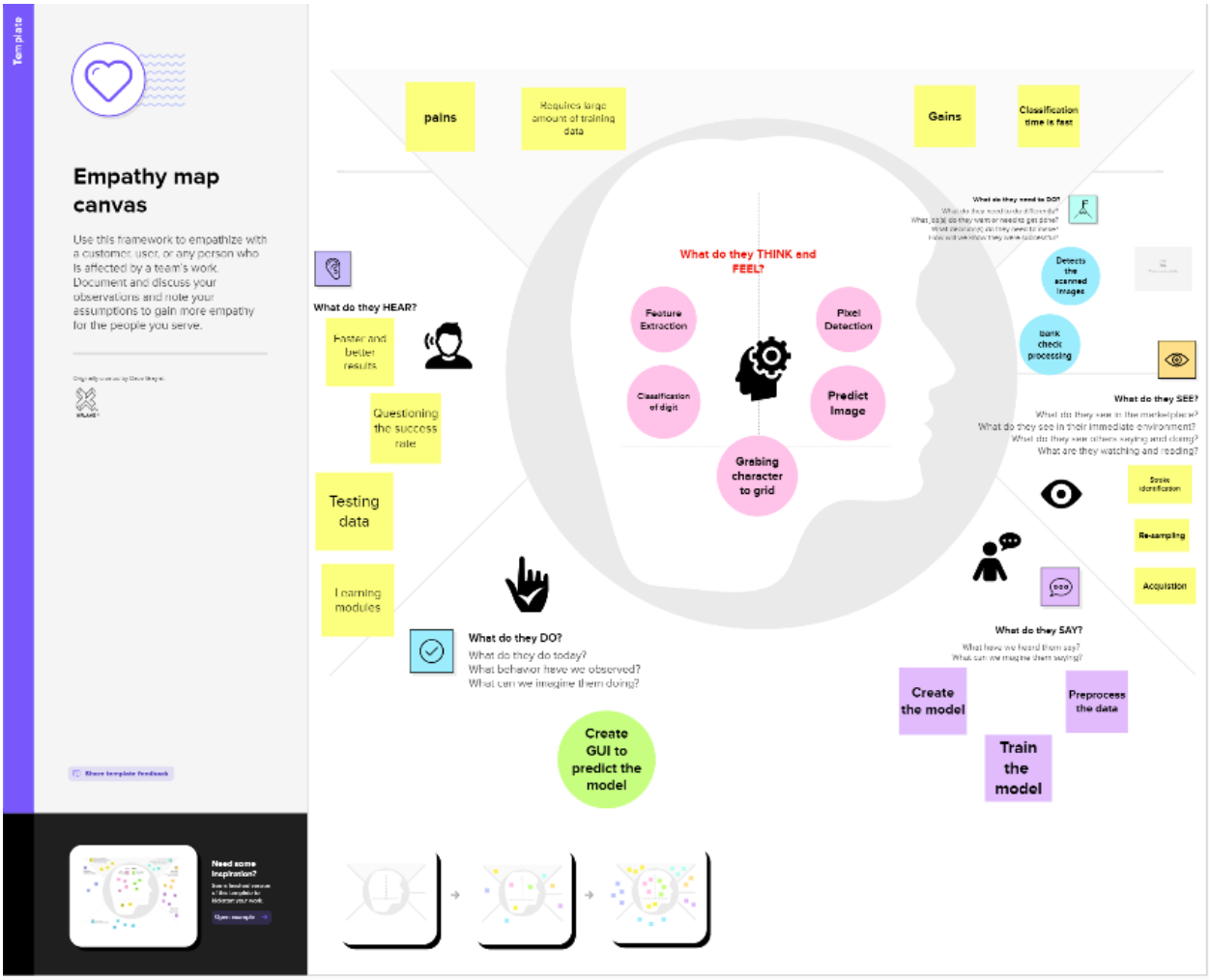
2.3. PROBLEM STATEMENT DEFINITION

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 digitized to reduce human effort. Hence, there comes a need for handwritten digit recognition in many realtime applications. The 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

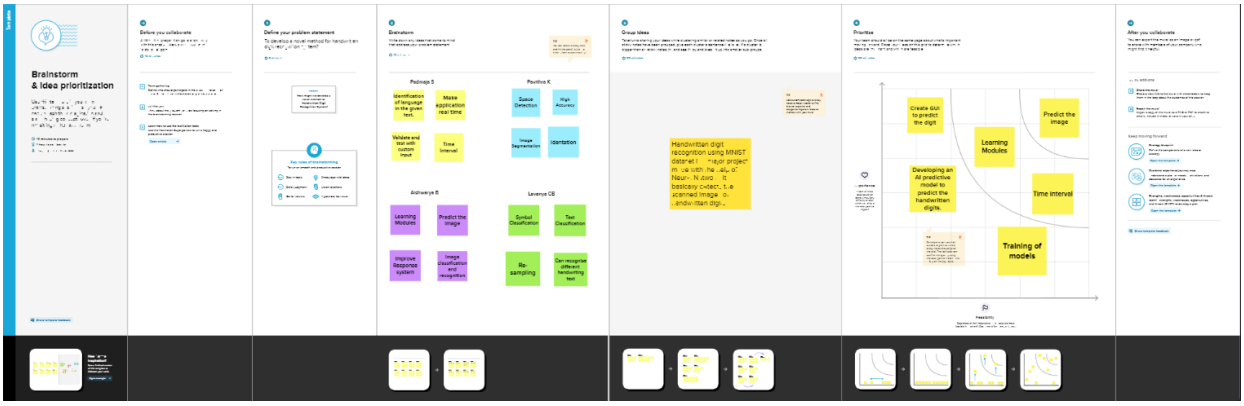
analyzed by the model and the detected result is returned to the UI. MNIST (“Modified National Institute of Standards and Technology”) is considered an unofficial computer vision “hello-world” dataset. This is a collection of thousands of handwritten pictures used to train classification models using Machine Learning techniques.

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



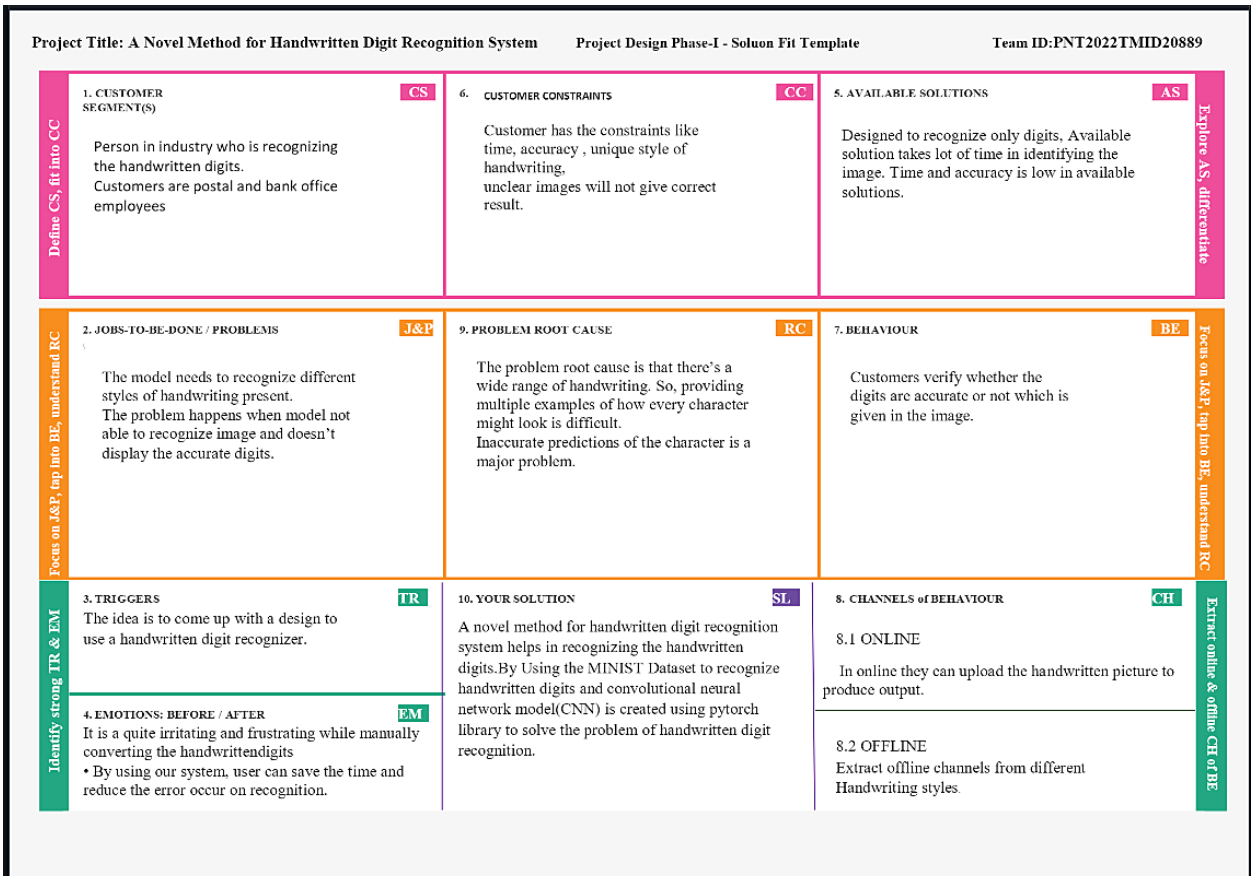
3.2 IDEATION & BRAINSTORMING



3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Handwritten digit 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 computer applications to recognize the human handwritten digits. With the progress in the field of science and technology, everything is being digitized to reduce the human effort. Here, it comes the need of handwritten digit recognition in real time applications.
2.	Idea / Solution description	We need to create a model that will be able to recognize and determine the handwritten digits from its image by using the concepts of Convolution Neural Network (CNN). MNIST is a dataset which is widely used for handwritten digit recognition. The dataset consists of 60,000 training images and 10,000 test images.
3.	Novelty / Uniqueness	Uses advanced digital techniques compared to conventional techniques is high. It accurately recognise the digits rather than recognising all the characters like OCR.
4.	Social Impact / Customer Satisfaction	It has impact on physically impaired people and helps them in terms of safety. There are many benefits associated with handwritten digit recognition system. Example: Reading postal addresses and reading forms. As a result, this system fulfils the customer expectations as it is novel method for recognising handwritten digits.
5.	Business Model (Revenue Model)	This system can be integrated with traffic surveillance cameras to recognise the vehicle's number plates for effective traffic management. It is used in cybersecurity applications and it can also be used for blind people by using sound output.

3.4 PROBLEM SOLUTION FIT



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

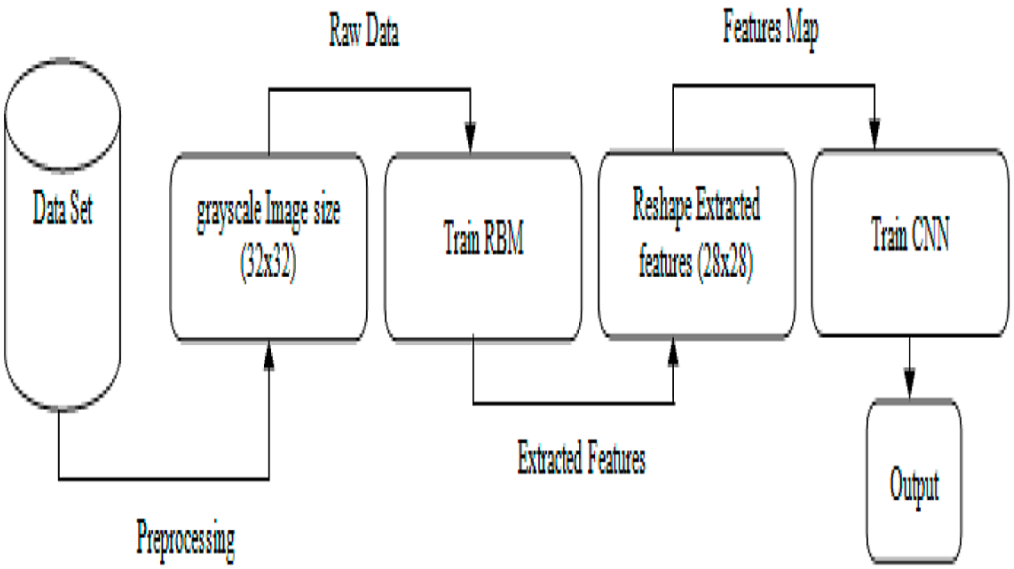
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	MNIST Dataset	A training set of 60,000 instances and a test set are included in the modified National Institute of Standards and Technology dataset (MNIST) database of handwritten digits.
FR-2	Website	Web hosting makes the code, graphics, and other items that make up a website accessible online. A server hosts every website you have 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	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 network.
FR-4	GUI	Allows for the digitalization of the numbers and the user to enter a handwritten image. meant to make virtualization easier.
FR-5	Digit Classifier Model	Train a neural network to predict a digit from an image using the MNIST collection of handwritten digits. Gather the training and validation data first.
FR-6	Evaluation	Make that the model recognises the digit correctly and generates the correct result.

4. 2 Non-Functional requirements

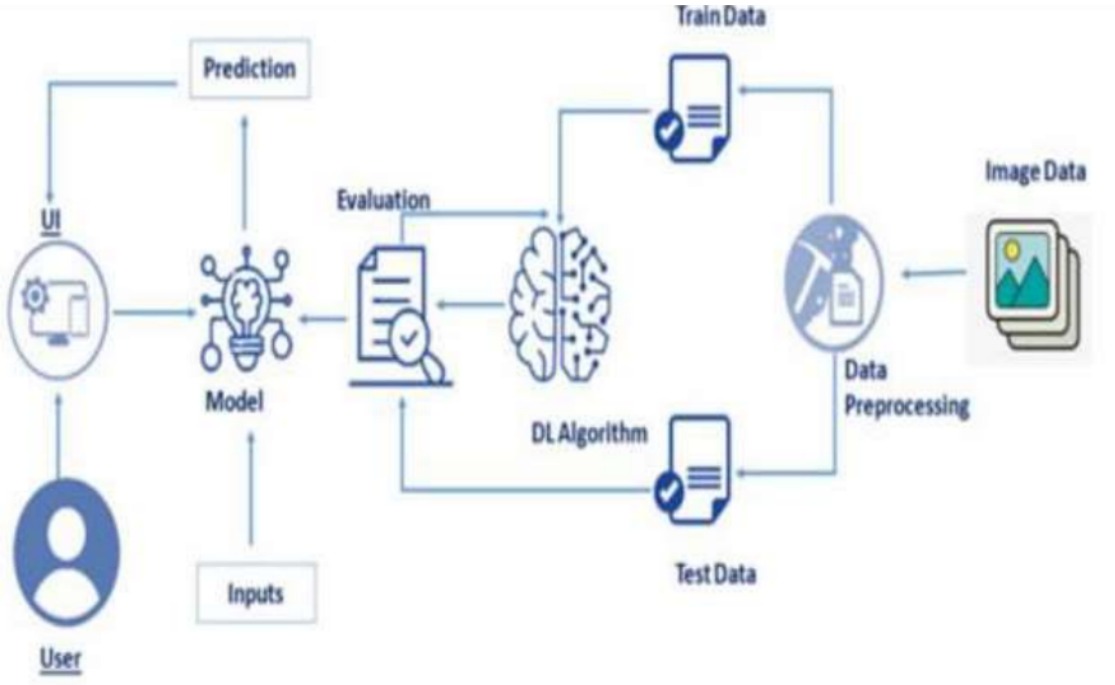
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The problem in pattern recognition application is the recognition of handwritten characters. Application for handwritten digit recognition include filling out forms, processing bank checks and sorting mail.
NFR-2	**Security**	The generative models are capable of segmentation driven by recognition. The system generates a description of the instantiation parameters which might reveal information like the writing style in addition to the categorization of the digit.
NFR-3	**Reliability**	The samples are used by the neural network to automatically generate rules for deciphering handwritten digits. The network may learn more about handwriting as a result of additional training cases, which will also increase its accuracy.
NFR-4	**Performance**	Performance is high because artificial neural networks used in deep learning are trained on the training set of images.
NFR-5	**Availability**	Through a web application, anyone may easily access the system, making it incredibly accessible for desktop and mobile browsers.
NFR-6	**Scalability**	Works with numerous additional datasets with distinct linguistic and writing types

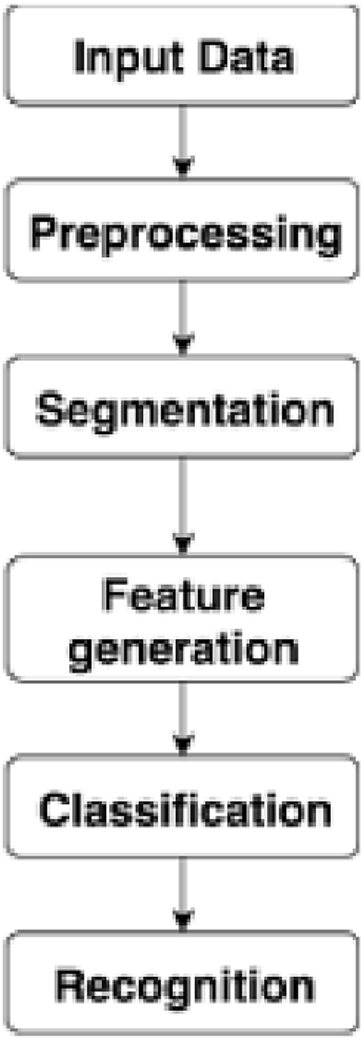
5.PROJECT DESIGN

5. 1 DATA FLOW DIAGRAM



5.2 SOLUTION AND TECHNICAL ARCHITECTURE





5. 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

6. PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Product Backlog, Sprint Schedule, and Estimation

Backlog and Sprint schedule

Sprint	Functional Requirement	Task
Sprint-1	Image Data	As a User need to collect the Image Data of Handly WrittenImages to train the model.
Sprint-2	Dash Board or Website	We using Python Flask Framework to create a dynamicWebpage to host our model (UI).
Sprint-3	Classifier Model	Using CNN Model for Image Classification.
Sprint-4	Cloud	Hosting the Organized appication in Cloud platform.

6.2 SPRINT DELIVERY SCHEDULE

Sprint and Duration Chart

Sprint	Duration	Sprint Start Date	Sprint End Date
Sprint-1	6 Days	25 Oct 2022	29 Oct 2022
Sprint-2	6 Days	31 Oct 2022	05 Nov 2022
Sprint-3	6 Days	07 Nov 2022	12 Nov 2022

7. CODING AND SOLUTIONING

7.1 FEATURE 1

```
import numpy #used for numerical analysis
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.datasets import mnist #mnist dataser
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A layer consists of a tensor-in tensor-out computation function
from tensorflow.keras.layers import Dense, Flatten #Dense-Dense layer is the regular deeply connected n
#Faltten-used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D #Convolutional layer
from keras.optimizers import Adam #optimizer
from keras.utils import np_utils #used for one-hot encoding
```

Importing the required libraries which are required for the model to run. The dataset for this model is imported from the Keras module.The dataset contains ten classes: Digits from 0-9. Each digit is taken as a class.

Loading The Data

The dataset for this model is imported from the Keras module.

```
(X_train, y_train), (X_test, y_test) = mnist.load_data() #splitting the mnist data into train and test
```

We split the data into train and test. Using the training dataset, we train the model and the testing dataset is used to predict the results.

```
print(X_train.shape)#shape is used for give the dimension values #60000-rows 28x28-pixels
print(X_test.shape)
```

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

We are finding out the shape of X_train and x_test for better understanding. It lists out the dimensions of the data present in it.
in trainset, we have 60000 images, and in the test set we have 10000 images

Analysing The Data

Let's see the Information of an image lying inside the x_train variable

```
X_train[0]#printing the first image
```

```
array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,
         18,  18,  18, 126, 136, 175,  26, 166, 255, 247, 127,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,  30,  36,  94, 154, 170,
        253, 253, 253, 253, 253, 225, 172, 253, 242, 195,  64,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  49, 238, 253, 253, 253, 253,
        253, 253, 253, 253, 251,  93,  82,  82,  56,  39,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  18, 219, 253, 253, 253, 253,
        253, 198, 182, 247, 241,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,  80, 156, 107, 253, 253,
        205,  11,   0,  43, 154,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,  14,   1, 154, 253,
```

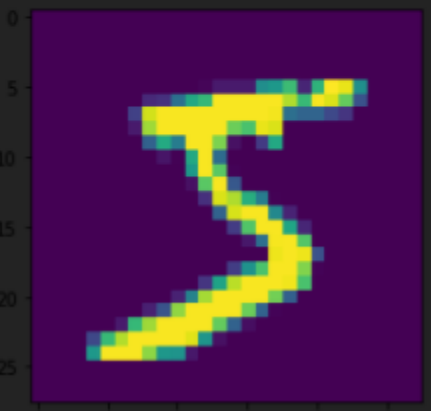
Basically, the pixel values range from 0-255. Here we are printing the first image pixel value which is index [0] of the training data. As you see it is displayed in the output.
With respect to this image, the label of this image will be stored in y_train let's see what is the label of this image by grabbing it from the y_train variable

```
y_train[0]#printing lable of first image

5
```

As we saw in the previous screenshot, we get to know that the pixel values are printed. Now here we are finding to which image the pixel values belong to. From the output displayed we get to know that the image is ‘5’.

Lets Plot the image on a graph using the Matplot library

```
import matplotlib.pyplot as plt #used for data visualization
plt.imshow(X_train[0]) #ploting the index=0 image

<matplotlib.image.AxesImage at 0x1c397af32e0>
```



Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. By using the Matplotlib library we are displaying the number ‘5’ in the form of an image for proper understanding.

Reshaping The Data

As we are using Deep learning neural network, the input for this network to get trained on should be of higher dimensional. Our dataset is having three-dimensional images so we have to reshape them too higher dimensions

```
# Reshaping to format which CNN expects (batch, height, width, channels)
X_train = X_train.reshape(60000, 28, 28, 1).astype('float32')
X_test = X_test.reshape(10000, 28, 28, 1).astype('float32')
```

We are reshaping the dataset because we are building the model using CNN. As CNN needs four attributes batch, height, width, and channels we reshape the data.

Applying One Hot Encoding

If you see our y_train variable contains Labels representing the images containing in x_train. AS these are numbers usually, they can be considered as numerical or continuous data, but with respect to this project these Numbers are representing a set of class so these are to be represented as categorical data, and we need to binaries these categorical data that's why we are applying One Hot encoding for y_train set

```
# one hot encode
number_of_classes = 10 #storing the no. classes in a variable
y_train = np_utils.to_categorical(y_train, number_of_classes) #converts the output in binary format
y_test = np_utils.to_categorical(y_test, number_of_classes)
```

One hot encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction. We apply One-Hot Encoding in order to convert the values into 0's and 1's.

Now let's see how our label 5 is index 0 of y_train is converted

```
y_train[0] #printing the new label

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

As we see the new the label is printed in the form of 0's and 1's and is of type float.

7.2 FEATURE 2

Adding CNN Layers

Creating the model and adding the input, hidden, and output layers to it

```
# create model
model = Sequential()
# adding model layer
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))
model.add(Conv2D(32, (3, 3), activation='relu'))
#model.add(Conv2D(32, (3, 3), activation='relu'))
#flatten the dimension of the image
model.add(Flatten())
#output layer with 10 neurons
model.add(Dense(number_of_classes, activation='softmax'))
```

The Sequential model is a linear stack of layers. You can create a Sequential model by passing a list of layer instances to the constructor:

Compiling The Model

With both the training data defined and model defined, it's time to configure

the learning process. This is accomplished with a call to the compile () method of the Sequential model class. Compilation requires 3 arguments: an optimizer, a loss function, and a list of metrics.

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

In our project, we have 2 classes in the output, so the loss is binary_crossentropy.
If you have more than two classes in output put “loss = categorical_cross entropy”.

Train The Model

Now, let us train our model with our image dataset.
functions used to train a deep learning neural network

```
# Fit the model
model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=5, batch_size=32)
```

```
Epoch 1/5
1875/1875 [==============================] - 184s 98ms/step - loss: 0.2451 - accuracy: 0.9497 - val_loss: 0.0966 - val_accuracy: 0.9
715
Epoch 2/5
1875/1875 [==============================] - 183s 98ms/step - loss: 0.0694 - accuracy: 0.9785 - val_loss: 0.0971 - val_accuracy: 0.9
714
Epoch 3/5
1875/1875 [==============================] - 183s 98ms/step - loss: 0.0487 - accuracy: 0.9850 - val_loss: 0.0829 - val_accuracy: 0.9
782
Epoch 4/5
1875/1875 [==============================] - 177s 94ms/step - loss: 0.0382 - accuracy: 0.9877 - val_loss: 0.0881 - val_accuracy: 0.9
769
```

steps_per_epoch: it specifies the total number of steps taken from the generator as soon as one epoch is finished and the next epoch has started. We can calculate the value of steps_per_epoch as the total number of samples in your dataset divided by the batch size.

Observing The Metrics

We here are printing the metrics which lists out the Test loss and Test accuracy

- Loss value implies how poorly or well a model behaves after each

iteration of optimization.

- An accuracy metric is used to measure the algorithm's performance in an interpretable way.

```
# Final evaluation of the model
metrics = model.evaluate(X_test, y_test, verbose=0)
print("Metrics(Test loss & Test Accuracy): ")
print(metrics)
```

```
Metrics(Test loss & Test Accuracy):
[0.1097492054104805, 0.9753000140190125]
```

Test The Model

Firstly, we are slicing the x_test data until the first four images. In the next step we the printing the predicted output.

```
prediction=model.predict(X_test[:4])
print(prediction)
```

```
[[5.50544734e-15 7.41999492e-20 5.00876077e-12 1.26642463e-09
  3.52252804e-21 1.54133163e-17 3.15550259e-21 1.00000000e+00
  1.32678888e-13 6.44072333e-14]
 [1.51885260e-08 8.02883537e-09 1.00000000e+00 6.44802788e-13
  6.37117113e-16 3.40490114e-15 2.15804121e-08 2.18907611e-19
  3.38496564e-10 2.07915498e-20]
 [3.14093924e-08 9.99941349e-01 2.01593957e-06 1.45100779e-10
  5.25237965e-06 1.59223120e-07 3.15299786e-08 1.53995302e-07
  5.09846941e-05 1.14552066e-07]
 [1.00000000e+00 1.35018288e-14 2.28308122e-10 1.79766094e-16
  1.28767550e-14 7.12401882e-12 2.92727509e-11 3.52439052e-13
  2.56207252e-12 2.32345068e-12]]
```

```
import numpy as np
print(np.argmax(prediction,axis=1)) #printing our labels from first 4 images
print(y_test[:4]) #printing the actual labels
```

```
[7 2 1 0]
[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

As we already predicted the input from the x_test. According to that by using argmax function here we are printing the labels with high prediction values

Test With Saved Model

The model is saved with .h5 extension as follows:
An H5 file is a data file saved in the Hierarchical Data Format (HDF). It contains multidimensional arrays of scientific data.

```
# Importing the Keras libraries and packages
from tensorflow.keras.models import load_model
model = load_model(r'C:/Users/DELL/Hand written recognition System/models/mnistCNN.h5')
from PIL import Image#used for manipulating image uploaded by the user.
import numpy as np#used for numerical analysis
for index in range(4):
    img = Image.open('data/' + str(index) + '.png').convert("L")# convert image to monochrome
    img = img.resize((28,28))# resizing of input image
    im2arr = np.array(img) #converting to image
    im2arr = im2arr.reshape(1,28,28,1) #reshaping according to our requirement
    # Predicting the Test set results
    y_pred = model.predict(im2arr) #predicting the results
    print(y_pred)
```

```
[[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
[[0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]]
[[0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]]
[[0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]]
```

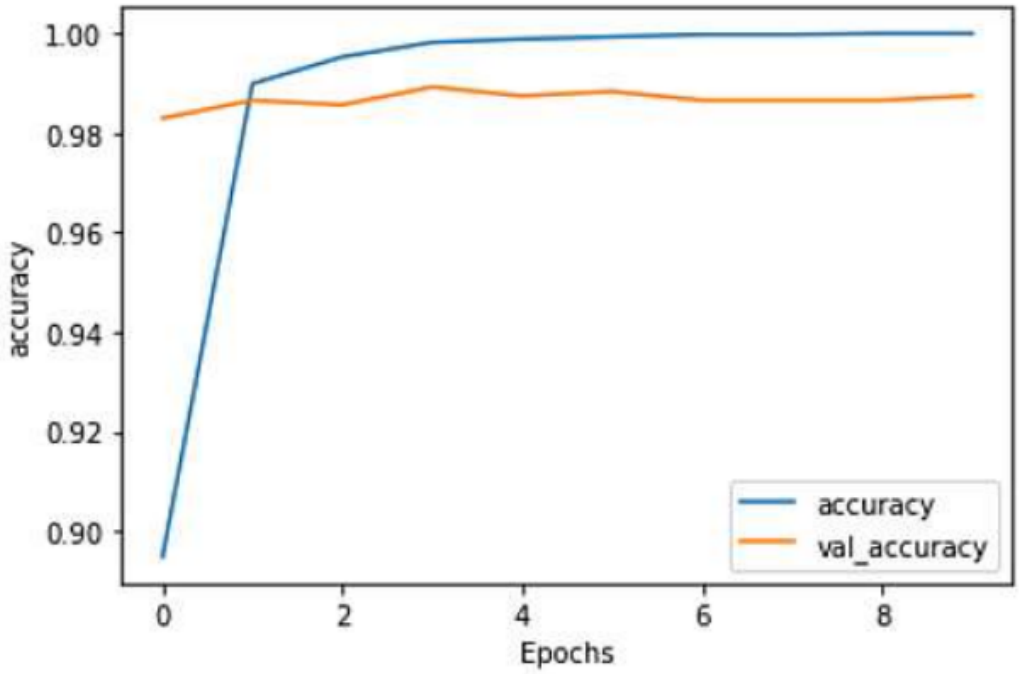
Firstly, we are loading the model which was built. Then we are applying for a loop for the first four images and converting the image to the required format. Then we are resizing the input image, converting the image as per the CNN model and we are reshaping it according to the requirement. At last, we are predicting the result.
You can use predict_classes for just predicting the class of an image

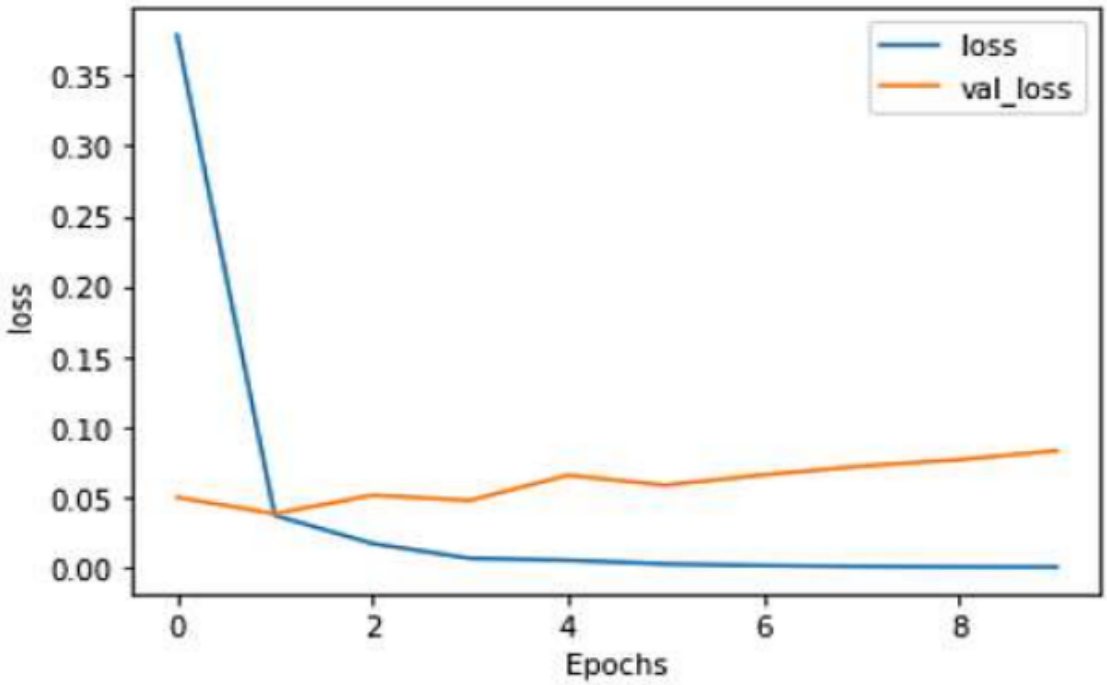
8.TESTING

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

9. RESULTS

9.1 PERFORMANCE METRICS





10.ADVANTAGES & DISADVANTAGES

ADVANTAGES

- ☆ Reduces manual work
- ☆ Recognises different type of handwriting
- ☆ 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
- ☆ high performance server for faster predictions is required.

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 subesquent versions.

12. FUTURE SCOPE

In our project we will be doing the improvements like

- ◊ Adding support to recognising from digits multiple images and save the results.
- ◊ Add support to detect multiple digits
- ◊ Improve model to recognising digits from complex images.
- ◊ Add support to different languages to help users from all over the world

This project has many benefits. Implementing this project inn the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency.It helps

in recognising the images and predicting them faster.

13. APPENDIX

FLASK APP

```
from flask import Flask, render_template, request
from scipy.misc import imsave, imread, imresize
import numpy as np
import keras.models
import re
import base64

import sys
import os
sys.path.append(os.path.abspath("./model"))
from load import *

app = Flask(__name__)
global model, graph
model, graph = init()

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

@app.route('/predict/', methods=['GET','POST'])
def predict():
    # get data from drawing canvas and save as image
    parseImage(request.get_data())

    # read parsed image back in 8-bit, black and white mode (L)
    x = imread('output.png', mode='L')
    x = np.invert(x)
    x = imresize(x,(28,28))

    # reshape image data for use in neural network
    x = x.reshape(1,28,28,1)
    with graph.as_default():
        out = model.predict(x)
```

HOME PAGE

```
<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>
```

TRAINING THE MODEL

```
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras import backend as K
import json

batch_size = 32
num_classes = 10
epochs = 60

# input image dimensions
img_rows, img_cols = 28, 28

# the data, shuffled and split between train and test sets
(x_train, y_train), (x_test, y_test) = mnist.load_data()

if K.image_data_format() == 'channels_first':
    x_train = x_train.reshape(x_train.shape[0], 1, img_rows, img_cols)
    x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
    input_shape = (1, img_rows, img_cols)
else:
    x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
    x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
    input_shape = (img_rows, img_cols, 1)

x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
print('x_train shape:', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')
```

```
# convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)

model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])


with open('model.json', 'w') as outfile:
    json.dump(model.to_json(), outfile)
model.save_weights('weights2.h5')
```

PREDICTION PAGE

```
<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>
```

GitHub & Project Demo Link

Github link

https://github.com/IBM-EPBL/IBM-Project-6485-1658829989

Project Demo

https://github.com/IBM-EPBL/IBM-Project-6485-1658829989/tree/main/Final%20Deliverables