NAALAIYA THIRAN PROJECT - 2022
19ECI01-PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP

A NOVEL METHOD USING HANDWRITTEN DIGIT RECOGNITION SYSTEM A PROJECT REPORT

Submitted by

GUNALRAJ K	**420719104010**
SARABUDHEEN A	**420719104032**
RANGANATH N	**420719104029**
RANGANATHAN C	**420719104030**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CK COLLEGE OF ENGINEERING AND TECHNOLOGY,

CUDDALORE– 607003

(Government Aided Autonomous Institution affiliated to Anna University)

ANNA UNIVERSITY: CHENNAI 600025
NOVEMBER 2022

PROJECT CALENDER

Phase	Phase Description	Week	Dates	Activity Details
1	Preparation Phase (Pre-requisites, Registrations, Environment Setup,etc.)	2	22 - 27 Aug 2022	Creation GitHub account & collaborate with Project repository in project workspace
2	Ideation Phase (Literature Survey, Empathize, DefiningProblem Statement, Ideation)	2	29 Aug – 3rd Sept 2022	Literature survey (Aim, objective, problem statement and need for the project)
		3	5 - 10th Sept 2022	Preparing Empathy Map Canvas tocapture the user Pains & Gains
		4	12 - 17 Sept 2022	Listing of the ideas using brainstormingsession
3	Project Design Phase -I (Proposed Solution, Problem-Solution Fit, Solution Architecture)	5	19 - 24 Sept 2022	Preparing document the proposed solution
		6	26 Sept - 01 Oct 2022	Preparing problem - solution fit document &Solution Architecture
4	Project Design Phase -II(Requirement Analysis, Customer Journey, DataFlow Diagrams, Technology Architecture)	7	3 - 8 Oct 2022	Preparing the customer journey maps
		8	10 - 15 Oct 2022	Preparing the Functional Requirement Document & Data-Flow Diagrams and Technology Architecture
5	Project Planning Phase (Milestones & Tasks, Sprint Schedules)	9	17 - 22 Oct 2022	Preparing Milestone & Activity List, Sprint Delivery Plan
6	Project Development Phase (Coding & Solutioning, acceptanceTesting, Performance Testing)	10	24 - 29 Oct 2022	Preparing Project Development Delivery ofSprint-1
		11	31 Oct - 5 Nov 2022	Preparing Project Development - Delivery ofSprint-2
		12	7 - 12 Nov 2022	Preparing Project Development - Delivery ofSprint-3
		13	14 - 19 Nov 2022	Preparing Project Development Delivery ofSprint-4

TABLE OF CONTENTS

SOURCE CODE

GITHUB AND DEMO LINK

LIST OF FIGURES

LIST OF TABLE

CHAPTER 01

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 recognize 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 likeemails, 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 (taxforms) and so on.

CHAPTER 02

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 recognize accurately.

The fundamental problem with handwritten digit recognition is that handwritten digit 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

Handwritten Digit Recognition System(2021)

Shubham Mendapara, Krish Pabani, Yash Paneliya

The Handwritten Digit Recognition using Deep learning methods has been implemented. The most widely used Machine learning algorithms CNN has been trained and tested on the MNIST dataset. With extensive testing using the MNIST data, the current function suggests the role of various hyper parameters. We also confirmed that a good adjustment of hyper parameters is important in improving the performance of Convolutional Neural Network. Utilizing this deep learning technique, a high amount of accuracy can be obtained. This model is able to achieve a recognition rate of 98.85% accuracy and is significantly identifying real world images as well. The effect of increasing the number of convolutional layers on CNN structure in the performance of handwritten digital recognition is clearly demonstrated by Experiments.

Recognition of Handwritten Digit using Convolutional Neural Network (CNN)(2019)

Md.Anwar Hossain, Md.Mohon Ali

Here they demonstrate a model which can recognize handwritten digit. Later it can be extended for character recognition and real-time person's handwriting. Handwritten digit recognition is the first step to the vast field of Artificial Intelligence and Computer Vision. As seen from the results of the experiment, CNN proves to be far better than other classifiers. The results can be made more accurate with more convolution layers and more number of hidden neurons. It can completely abolish the need for typing. Digit recognition is

an excellent prototype problem for learning about neural networks and it gives a great way to develop more advanced techniques of deep learning.

Machine Learning for Handwriting Recognition(2020)

Preetha S, Afrid I M , Karthik Hebbar P , Nishchay S K.

Among these methods, highest accuracy is achieved from Convolutional Neural Network (CNN) and the least accuracy is achieved from Slope and Slant Correction method. When the images are trained with CNN, they will achieve good accuracy and this isone of the successful method for hand writing recognition and only disadvantage with this method is that training time of the model is too high because lot of image samples are included. In Zoning method, if zones which are achieved after dividing input image and if the count of these zones are lesser then accuracy will decrease. Main disadvantage of this method is that developers will face lot of problems while segmentation process but this method is too simple for hand writing recognition. This method only sees the Lat and which makes it simple. Hand writing recognition is very challenging because all the individuals have different hand writing and it becomes more complex to detect when these are compared to that of computer.

Review on handwritten digit recognition(2017)

Priya, Rajendra singh, dr. Soni changlani.

The paper discusses in detail all advances in the area of handwritten character recognition. The most accurate solution provided in this area directly or indirectly depends upon the quality as well as the nature of the material to be read Various techniques have been described in this paper for character recognition in handwriting recognition system. A sort comparison is shown between the different methods proposed. This thesis HOG-PSVM handwritten digit recognition system is presented. The images of handwritten digits are described in terms of 81 dimensions HOG feature descriptor.

Handwritten digit recognition(2022)

Dhruv Sharma, Ishaan Singh, Upendra Pandey

The Handwritten number Recognition using Deep learning algorithm has been enforced. The most extensively used Machine learning algorithms, KNN, CNN have been trained and tested on the same data in order acquire the comparison between the classifiers. Utilising these deep learning ways, a high quantum of delicacy can be attained. Compared to other exploration styles, this system focuses on which classifier works more by perfecting the delicacy of bracket models by further than 99. Using Keras as back end and Tensor

flow as the software, a CNN model is suitable to give delicacy of about 98.72 Problematic dyads of integers have analogous profile as can be seen from results table; ‘ 4 ’ and ‘ 1 ’, ‘ 7 ’ and ‘ 1 ’,‘ 6 ’ and ‘ 8 ’.Writing styles matter(weird 7's or deficient bottoms).

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 real- time 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 andbuild 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 resultis returned to the UI. MNIST (“Modified National Institute of Standards and Technology”) is considered an unofficial computer vision “hello-world” data set. This is a collection of thousands of handwritten pictures used to train classification models using Machine Learning techniques.

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

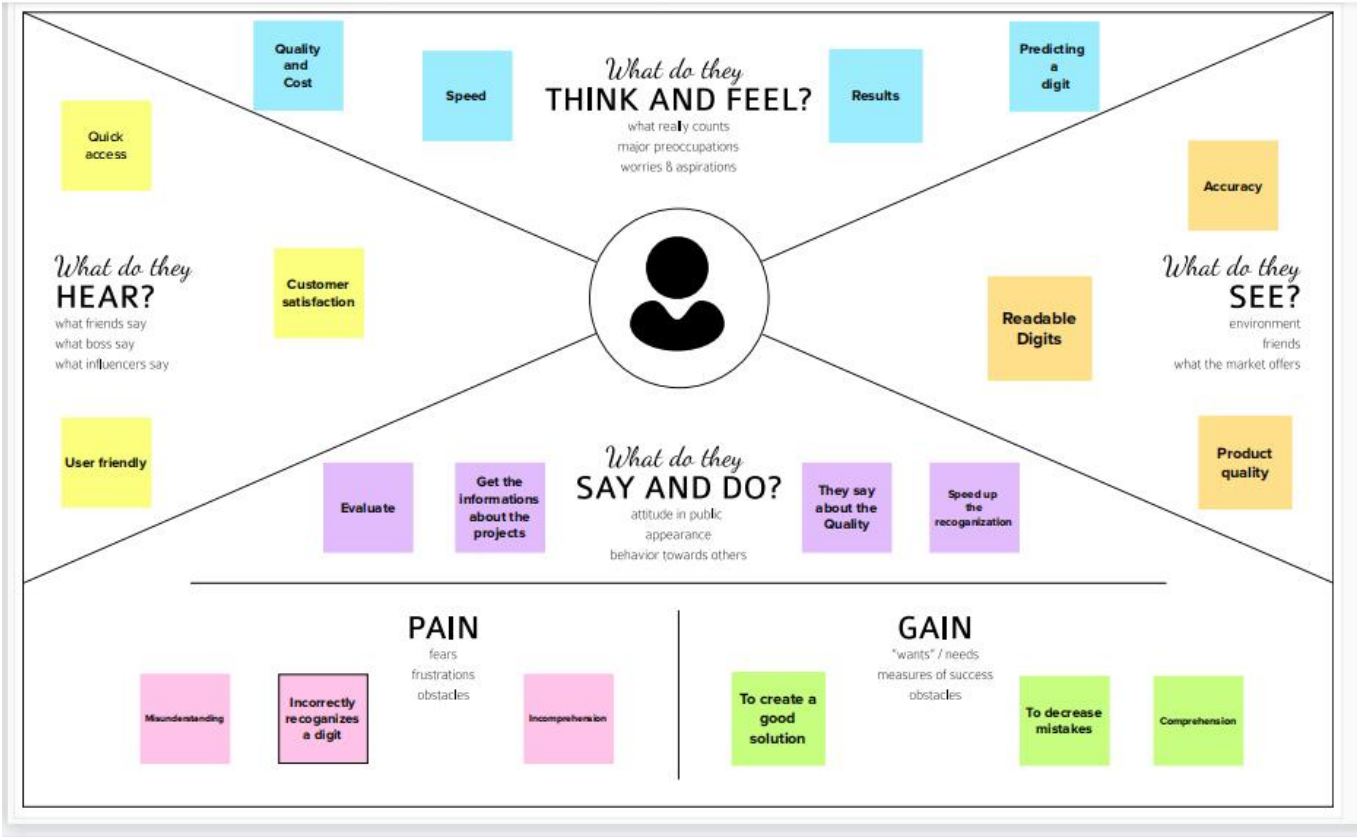


Fig 3.1 Empathy map

3.2 IDEATION AND BRAINSTORMING

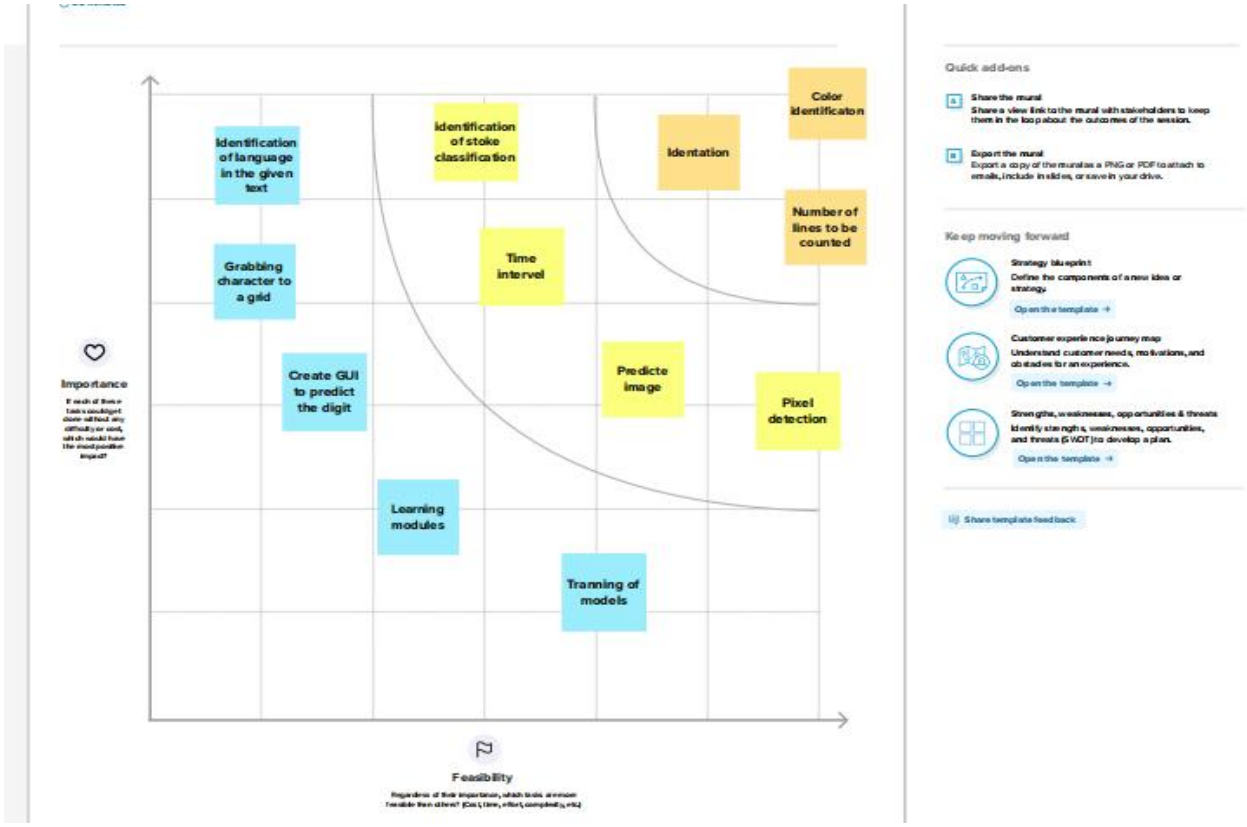


Fig 3.2 Ideation and Brainstorming

3.3 PROPOSED SOLUTION

The proposed solution for a novel method using handwritten digit recognition is shown in table 3.3

S.No.	Parameter	Description
1.	**Problem Statement (Problem tobe solved)**	**Statement**-The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. **Description**: It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and Size.
2.	**Idea / Solution description**	• It is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers, touch defences. • It allows user to translate all those signature and notes into electronic words in a text document format and this data only requires far less physical space than the storage of the physical copies.
3.	**Novelty / Uniqueness**	Accurately recognize the digits rather than recognizing all the characters like OCR.
4.	**Social Impact / Customer Satisfaction**	• Artificial Intelligence developed the app called Handwritten digit Recognizer. • It converts the written word into digital approximations and utilizes complex algorithms to identify characters before churning out a digital approximation.
5.	**Business Model (Revenue Model)**	•This system can be integrated with traffic surveillance cameras to recognize the vehicle's number plates for effective traffic management. •Can be integrated with Postal system to identify and recognize the pin-code details easily
6.	**Scalability of the Solution**	• Ability to recognize digits in more noisy environments. • There is no limit in the number of digits it can be recognized.

Table 3.3 Proposed Solution

3.1 PROBLEM SOLUTION FIT

Project Title: A Novel Method for Handwritten Digit Recognition System

Project Design Phase-I - Solution Fit Template

Team ID: PNT2022TMID38743

Define CS, fit into CC

1. CUSTOMER SEGMENT(S)

The Customers who deal with handwritten digits like Banking sectors, schools, colleges, railways, firms, etc.

6. CUSTOMER CONSTRAINTS

They believe that the alternatives will result in errors and faults and will be inconvenient.

5. AVAILABLE SOLUTIONS

There are no widely used software's to detect handwriting; instead, they check with other people to affirm what number it is.

Explore AS, differentiate

Focus on J&P, tap into BE

2. JOBS-TO-BE-DONE / PROBLEM S:

Handwritten digits can be difficult to understand and interpret at times. It may cause errors when dealing with rough handwriting.

9. PROBLEM ROOT CAUSE

We face numerous challenges in handwritten number recognition. because of different people's jotting styles and the lack of Optic character Recognition This investigation offers an in-depth comparison of various machine literacy and deep literacy

7. BEHAVIOUR BE

Finding the best software for detecting accurate digits in a more efficient manner.

Focus on J&P, tap into BE

3. TRIGGERS TR

To obtain the numbers accurately and quickly.

4. EMOTIONS: BEFORE / AFTER EM

Feels frustrated and sad when numbers are not entered.

10. YOUR SOLUTION SL

A solution to this problem is the Handwritten digit recognition system, which uses a picture of a digit and recognizes the digit present in the image. Convolutional Neural Network model built with PyTorch and applied to the MNIST data set to recognize handwritten digits..

8.CHANNELS of BEHAVIOUR CH

Using software that is available on the internet. Obtaining assistance from those nearby in order to recognize the digits written by their customers.

Fig 3.4 Problem Solution fit

CHAPTER 4

REQUIREMENT ANALYSIS

Requirements analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and Non-functional requirements.

4.1 FUNCTIONAL REQUIREMENTS

FR No.	Sub Requirement (Story/ Sub Task)
FR-1	Image Data: Handwritten digit recognition refers to 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	Modified National Institute of Standards and Technology dataset: The abbreviation MNIST stands for the MNIST 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.

Table 4.1 Functional Requirements for the A novel method using handwritten digit recognition System

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 mightreveal 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 recognize handwritten numbers.
NFR- 4	**Accuracy**	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.

Table 4.2 Non-Functional Requirements for the A novel method using handwritten digit recognition System

CHAPTER 05

PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

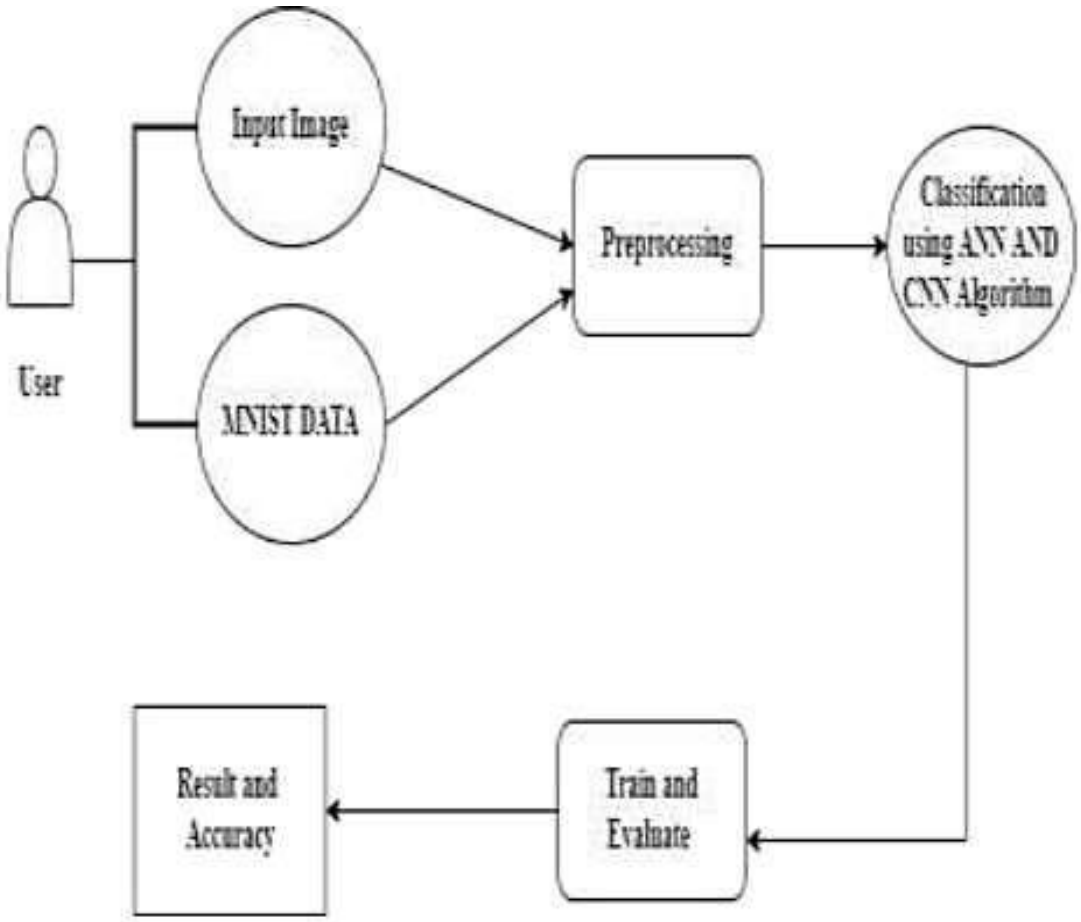


Fig 5.1: Data Flow Diagram for the A novel method using handwritten digit recognition system.

5.2 SOLUTION ARCHITECHTURE

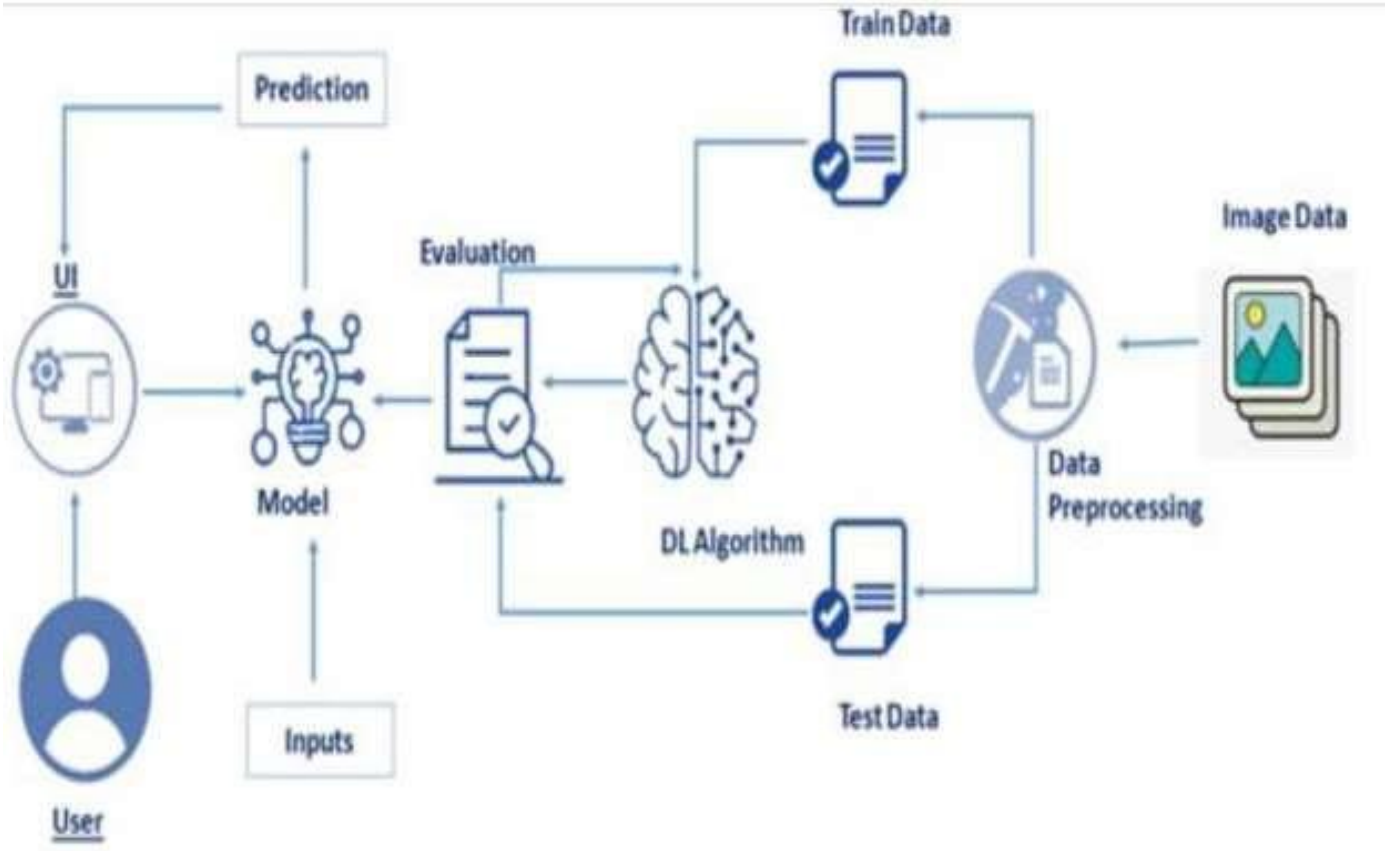


Fig 5.2: Solution architecture Diagram for the A novel method using handwritten digit recognition system.

5.3 TECHNOLOGY ARCHITECHTURE

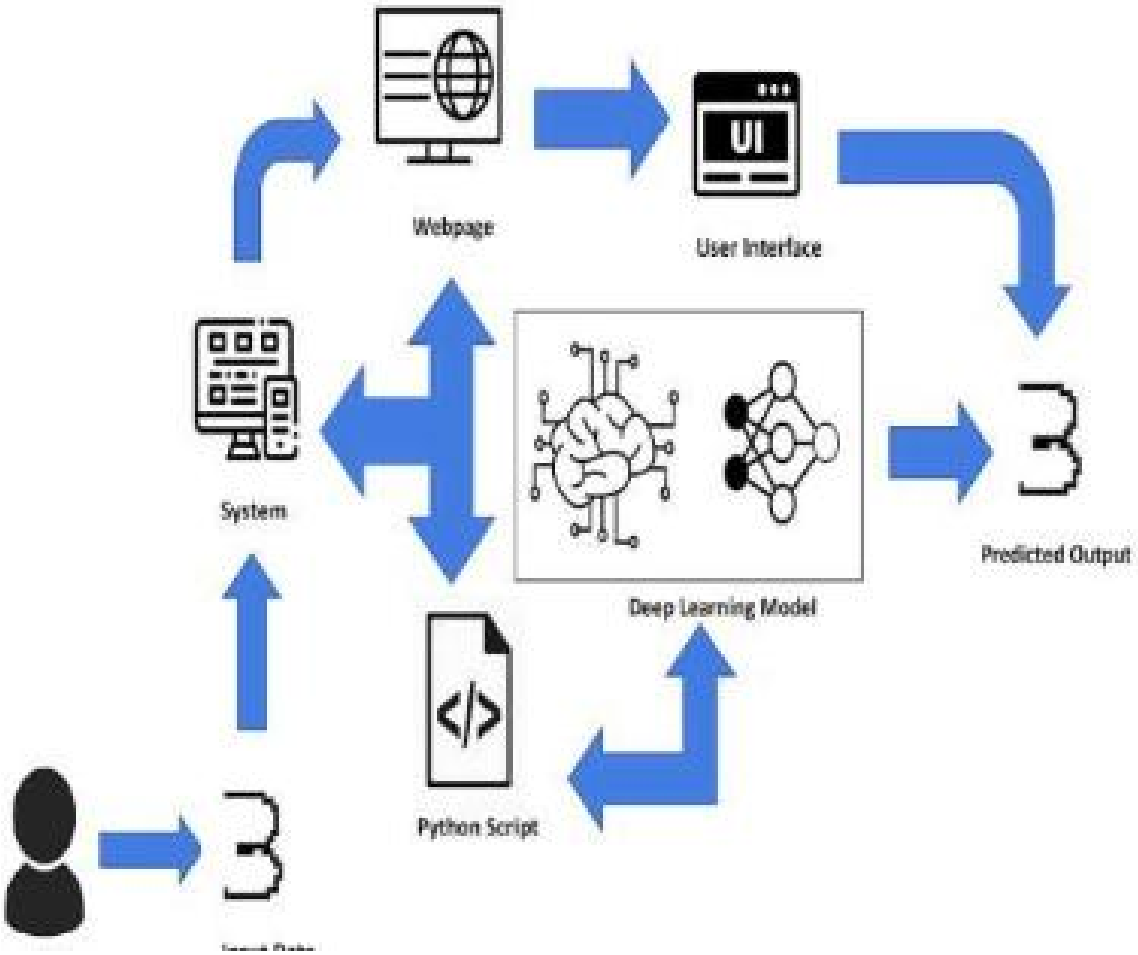


Fig 5.3: Technology architecture for the A novel method using handwritten digit recognition system.

CHAPTER 6
PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

SPRINT	USER STORY / TASK	STORY POINTS	PRIORITY	TEAM MEMBERS
Sprint-1	Get the dataset	3	High	Gunalraj K
	Explore the data	2	Medium	Gunalraj K Sarabudheen A
	Data Pre-Processing	3	High	Ranganath N Ranganathan C
	Prepare training and testing data	3	High	Ranganath N Ranganathan C
Sprint-2	Create the model	3	High	Ranganath N
	Train the model	3	High	Ranganathan C
	Test the model	3	High	Sarabudheen A
Sprint-3	Improve the model	2	Medium	Ranganath N Ranganathan C
	Save the model	3	High	Gunalraj K
	Build the Home Page	3	High	Sarabudheen A Gunalraj K
	Setup a database to store input images	2	Medium	Ranganath N
Sprint-4	Build the results page	3	High	Sarabudheen A Gunalraj K
	Integrate the model with the application	3	High	sarabudheen A Ranganath N

Table 6.1: Sprint Planning and estimation for A novel method using handwritten digit recognition system.

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	11	6 Days	24 Oct 2022	29 Oct 2022	11	29 Oct 2022
Sprint-2	9	6 Days	31 Oct 2022	05 Nov 2022	9	05 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	12 Nov 2022	10	12 Nov 2022
Sprint-4	9	6 Days	14 Nov 2022	19 Nov 2022	9	19 Nov 2022

Table 6.2: Sprint Planning done for A novel method using handwritten digit recognition system.

CHAPTER 7

CODING & SOLUTIONING

```
import pandas as pd
import seaborn as sns
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline
```

```
import csv
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
STOPWORDS = set(stopwords.words('english'))
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense,Flatten,MaxPooling2D
from tensorflow.keras.layers import Conv2D
from keras.optimizers import Adam
from keras.utils import np_utils
from tensorflow.keras.models import load_model
```

FEATURE 1

```
reverse_word_index = dict([(value, key) for (key, value) in word_index.items()])

def decode_article(text):
    return ' '.join([reverse_word_index.get(i, '?') for i in text])
print(decode_article(train_padded[10]))
print('---')
print(train_articles[10])
```

FEATURE 2

```
vocab_size = 5000
embedding_dim = 64
max_length = 200
trunc_type = 'post'
padding_type = 'post'
oov_tok = ''
training_portion = .8
```

```
articles = []
labels = []

with open("spam.csv", 'r', encoding = "ISO-8859-1") as dataset:
    reader = csv.reader(dataset, delimiter=',')
    next(reader)
    for row in reader:
        labels.append(row[0])
        article = row[1]
        for word in STOPWORDS:
            token = ' ' + word + ' '
            article = article.replace(token, ' ')
            article = article.replace('  ', ' ')
        articles.append(article)
print(len(labels))
print(len(articles))
```

```
train_size = int(len(articles) * training_portion)

train_articles = articles[0: train_size]
train_labels = labels[0: train_size]

validation_articles = articles[train_size:]
validation_labels = labels[train_size:]

print(train_size)
print(len(train_articles))
print(len(train_labels))
print(len(validation_articles))
print(len(validation_labels))
```

```
train_padded = pad_sequences(train_sequences, maxlen=max_length, padding=padding_type, truncating=trunc_type)
print(len(train_sequences[0]))
print(len(train_padded[0]))

print(len(train_sequences[1]))
print(len(train_padded[1]))

print(len(train_sequences[10]))
print(len(train_padded[10]))
```

```
model.add(Conv2D(64,(3,3),input_shape=(28,28,1),activation="relu"))
model.add(Conv2D(32,(3,3),activation="relu"))
model.add(MaxPooling2D((2,2)))
model.add(Flatten())
model.add(Dense(number_of_classes,activation="softmax"))
```

CHAPTER 8

TESTING

8.1 TEST CASES

st case ID	Feature Type	Component	Test Scenario	Expected Result	ActualResult	Status
HP_TC_001	UI	Home Page	Verify UI elements in the Home Page	The Home page must bedisplayed 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 bedisplayed properly in all sizes	The UI is not displayed properly in screen size 2353 x 1651 and 758 x 630	FAIL

HP_TC_003	Functional	Home Page	Check if usercan upload their file	The input image shouldbe uploaded to the application successfully	Working as expected	PASS
HP_TC_004	Functional	Home Page	Check if user cannot upload unsupporte d files	The applicatio nshould notallow userto select a non imagefile	User is ableto 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

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
AC_TC_001	Functional	Accuracy	check if the model can provide the output with accuracy	The model should predict the image with accuracy from the dataset.	working as expected	PASS

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

2.USER ACCEPTANCE TESTING

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

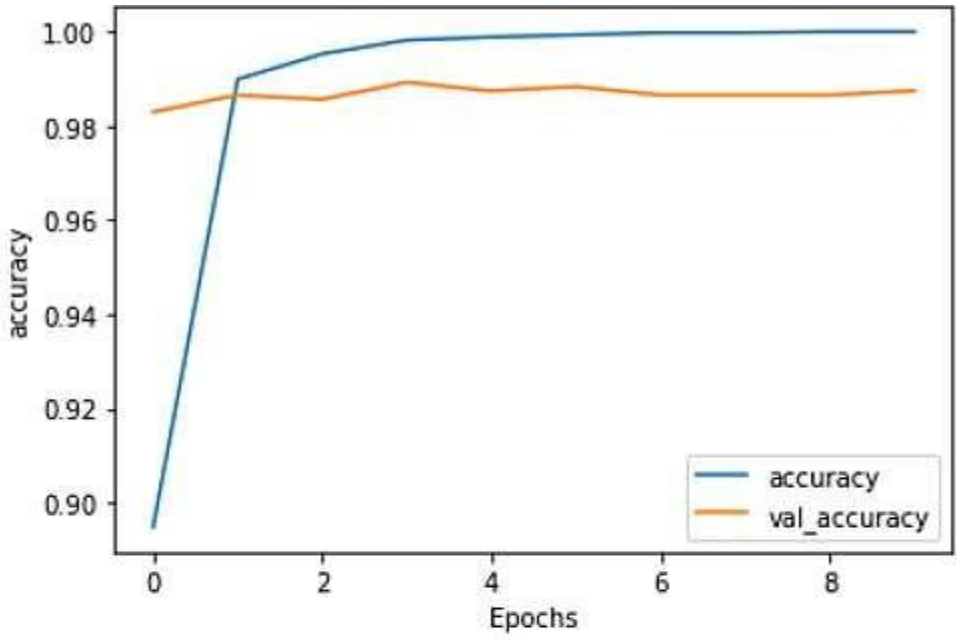
2.2.TEST CASE ANALYSIS

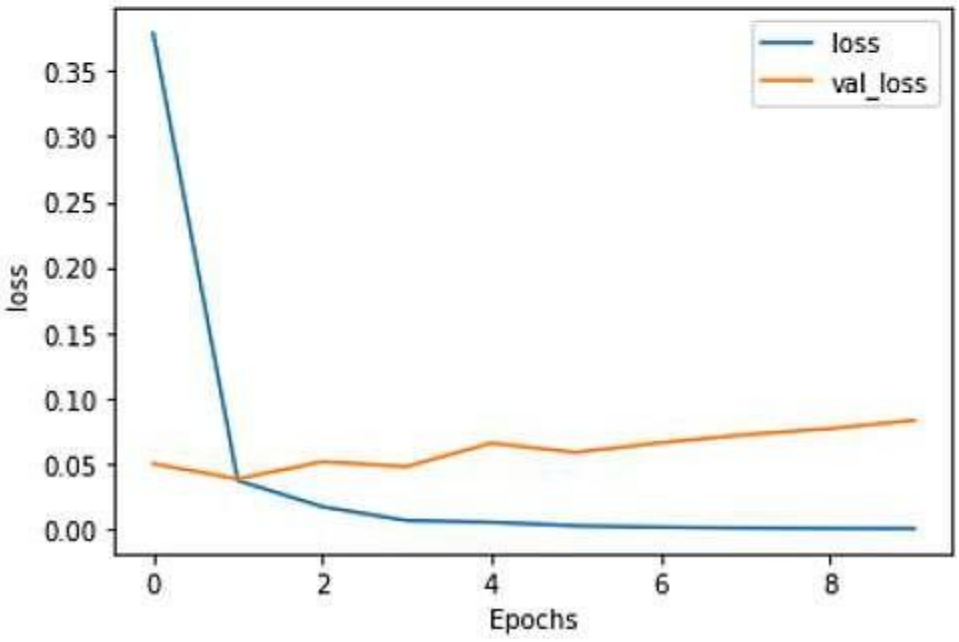
Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	3	7
Security	2	0	1	1
Performance	3	0	1	2
Exception Reporting	2	0	0	2

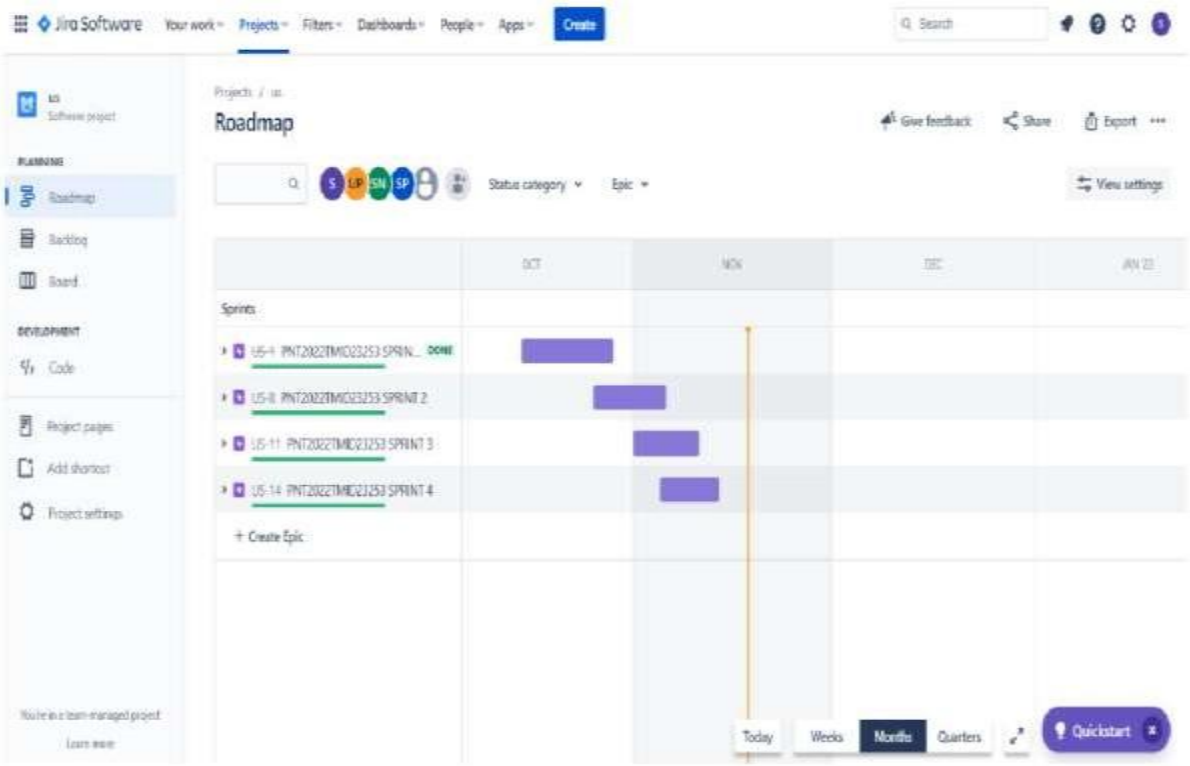
CHAPTER 9

RESULTS

9.1. PERFORMANCE METRICS







CHAPTER 10

ADVANTAGES & DISADVANTACES

10.1 ADVANTAGES

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

10.2 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

CHAPTER 11
CONCLUSION

This project demonstrated a web application that uses machine learning to recognize handwritten numbers. Flask, HTML, CSS, JavaScript, and a few other technologies were used to create this project. The model predicts the handwritten digit using a CNN network. During testing, the model achieved a 99.61% recognition rate. The proposed project is scalable and can easily handle a huge number of users. Since it is a web application, it is compatible with any device that can run a browser. This project is 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 benefifit several industries and reduce the workload on many workers, enhancing overall work effificiency.

APPENDIX

SOURCE CODE

MODEL CREATION

Import necessary package

```
import numpy
import matplotlib.pyplot as plt
from keras.utils import np_utils
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, Dense, Flatten
from tensorflow.keras.optimizers import Adam
```

Load data

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

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

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

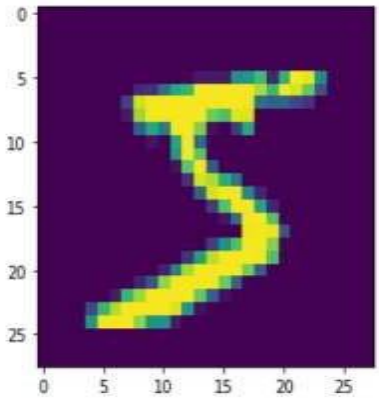
```
(60000, 28, 28)
(10000, 28, 28)
```

In [5]:
```
y_train[0]
```

Out[5]: 5

In [6]:
```
plt.imshow(X_train[0])
```

Out[6]:



Data pre processing

In [7]:
```
X_train = X_train.reshape(60000, 28, 28, 1).astype('float32')
X_test = X_test.reshape(10000, 28, 28, 1).astype('float32')
```

In [8]:
```
number_of_classes = 10
Y_train = np_utils.to_categorical(y_train, number_of_classes)
Y_test = np_utils.to_categorical(y_test, number_of_classes)
```

```
import tensorflow as tf

from keras.models import Sequential
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D

def init():
    num_classes = 10
    img_rows, img_cols = 28, 28
    input_shape = (img_rows, img_cols, 1)
    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'))

    #load weights into new model
    model.load_weights("weights.h5")
    print("Loaded Model from disk")

    #compile and evaluate loaded model
    model.compile(loss=keras.losses.categorical_crossentropy, optimizer=keras.optimizers.Adadelta(), metrics=['accuracy'])
    #loss,accuracy = model.evaluate(X_test,y_test)
    #print('loss:', loss)
    #print('accuracy:', accuracy)
    graph = tf.get_default_graph()

    return model,
```

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'])
```

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">
            <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>
```

TRAIN 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.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')
```

PREDICT PAGE (HTML)

```
<html>
        <head>
                <title>Prediction | Handwritten Digit Recognition</title>
                <link rel="stylesheet" href="{{url_for('static',filename='css/predict.css')}}" />
                <link rel="icon" type="image/svg" sizes="32x32" href="{{url_for('static',filename='images/icon.svg')}}" />
                <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        </head>
        <body>
                <div class="container">
                        <h1>Prediction</h1>
                        <div class="result-wrapper">
                                <div class="input-image-container">
                                        <img src="{{url_for('static',filename='data/')}}{{img_name}}" />
                                </div>
                                <div class="result-container">
                                        <div class="value">{{best.0}}</div>
                                        <div class="accuracy">{{best.1}}%</div>
                                </div>
                        </div>
                        <h1>Other Predictions</h1>
                        <div class="other_predictions">
                                {% for x in others %}
                                <div class="value">
                                        <h2>{{x.0}}</h2>
                                        <div class="accuracy">{{x.1}}%</div>
                                </div>
                                {% endfor %}
                        </div>
                </div>
        </body>
</html>
```

GIT HUB: [IBM-EPBL/IBM-Project-3949-1658673597: A Novel Method for Handwritten Digit Recognition System]