PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

Team ID: PNT2022TMID46059

TEAM MEMBERS:

SHAFEENA BEGUM MS
RAGAVI G
YAZHINI P
PREETHI K

Abstract

Hand Written Digit recognition is also remarkable an important issue. As handwritten digits are not a same size, thickness, position and direction, in this case by the way, various difficulties should be considered find the handwritten digital recognition problem. I unique and a variety of creative styles for different people moreover have an influence on the model as well the presence of digits. It is a strategy to see again edit written digits. It has a wide variety of applications, for example, scheduled bank checks, post offices and tax documents and so on. The purpose of this project is to use the classification algorithm to identify handwritten digits. Background results are probably the most widely used Machine Learning Algorithms such as SVM, KNN and RFC and in-depth reading calculations like CNN multilayer using Keras and Theano and Tensorflow. Using these, 98.70% accuracy was used by CNN (Keras + Theano) compared to 97.91% using SVM, 96.67% using KNN, 96.89% using RFC was obtained.

INTRODUCTION

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. 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. The aim of a handwriting recognition system is to convert handwritten characters into machine readable formats. Handwritten digit recognition has not only professional and commercial applications, but also has practical application in our daily life and can be of great help to the visually impaired. It also helps us to solve complex problems easily thus making our lives easier. Handwritten digit recognition has gained so much popularity from the aspiring beginner of machine learning and deep learning to an expert who has been practicing for years.

Purpose:

Nowadays the whole world is a shift in the digital world. Images of handwritten digits as 10 digits (09). Handwritten digits from the MNIST database are already famous among the community for many recent decades now, as decreasing the error rate with different classifiers and parameters. Digit recognition system is the working of a machine to train itself or 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 (say tax forms) and so on. The handwritten digits are not always of the same size, width, orientation and justified to margins as they differ from writing of person to person, so the general problem would be while classifying the digits due to the similarity between digits such as 1 and 7, 5 and 6, 3 and 8, 2 and 5, 2 and 7, etc. This problem is faced more when many people write a single digit with a variety of different handwritings.

2. LITERATURE SURVEY

Paper 1: Pre-processing techniques involved in the character recognition

Publication Year: 2013

Author: K. Gaurav, Bhatia P. K

Handwritten character recognition (HCR) is the process of conversion of handwritten text into machine readable form. The major problem in HCR system is the variation of the handwriting styles, which can be completely different for different writers. In this, different preprocessing techniques like Skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, Normalization and segmentation, morphological processing techniques are discussed. It was concluded that using a single technique for preprocessing, we can't completely process the Image. However, even after applying all the said techniques might not possible to achieve the full accuracy in a Preprocessing system.

Paper 2: Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Model

Publication Year: 2011

Author: Salvador Espana-Boquera

In this paper, Hybrid HMM/ANN models compute the emission probabilities for the HMMs with a neural network instead of the commonly used Gaussian mixtures. This work is the first successful attempt, to the best of our knowledge, to use hybrid HMM/ANN models in unconstrained offline handwritten text recognition. 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. The key features of this recognition system were to develop a system having high accuracy in preprocessing and recognition, which are both based on resulting in scaled likelihoods which are used as emission probabilities in the HMMs.

Paper 3: Optimizing Feature Selection for Recognizing Handwritten Arabic Characters

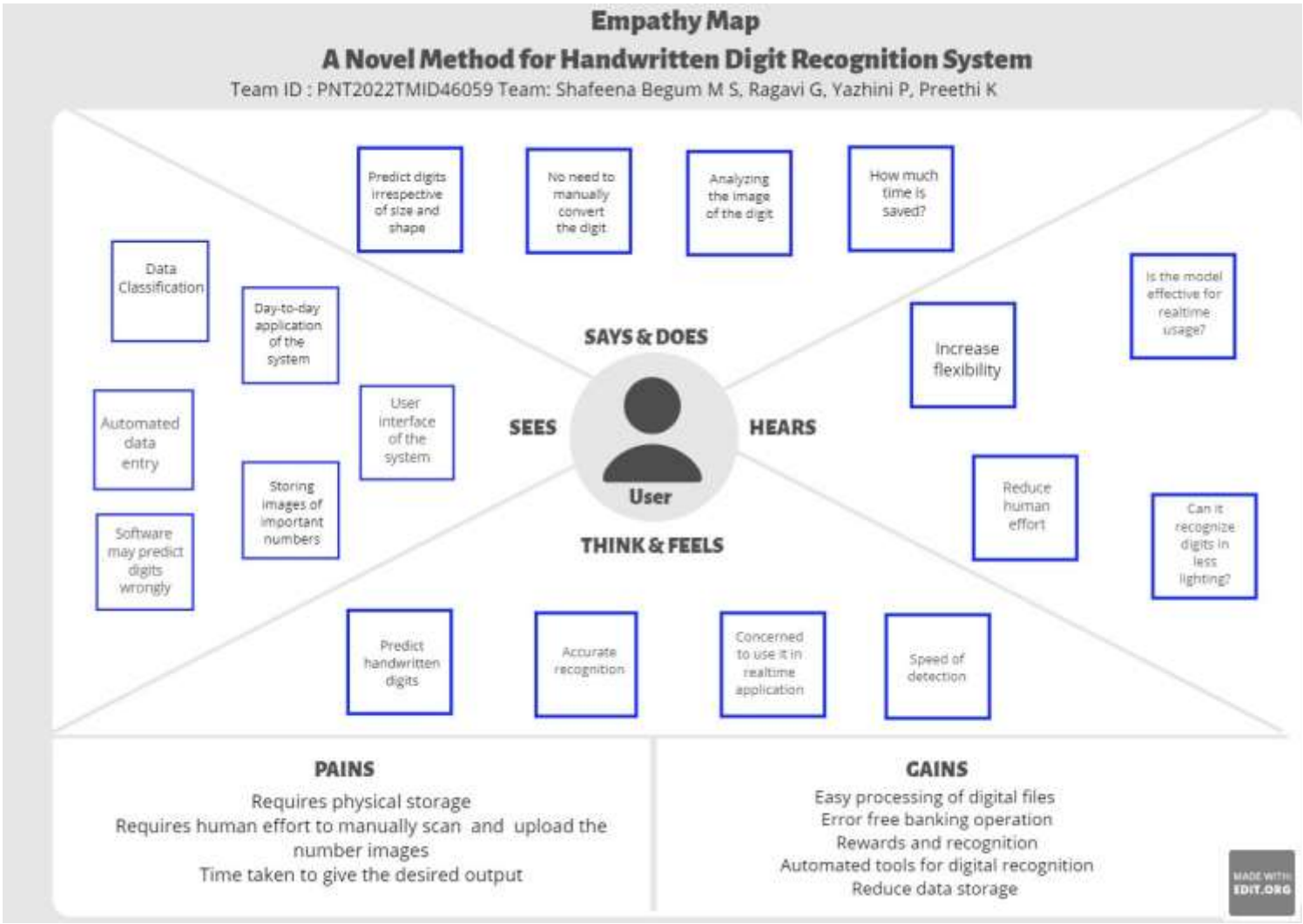
Publication Year: 2005

Author: Mohammed Z. Khedher, Gheith A. Abandah, and Ahmed M. Al Khawaldeh

This paper describes thatRecognition is the process of evaluating the extracted features of an unknown character and comparing them with the features of the set of possible characters. An off-line recognition system based on the selected features was built. The system was trained and tested with realistic samples of handwritten Arabic characters. Feature Extraction is the next stage. Extracted features should contain the useful information carried by the character image. The complexity of the handwritten cursive Arabic text requires using many features to make recognition possible. The recognition based on the selected features give average accuracies of 88% and 70% for the numbers and letters, respectively. Further improvements are achieved by using feature weights based on insights gained from the accuracies of individual features.

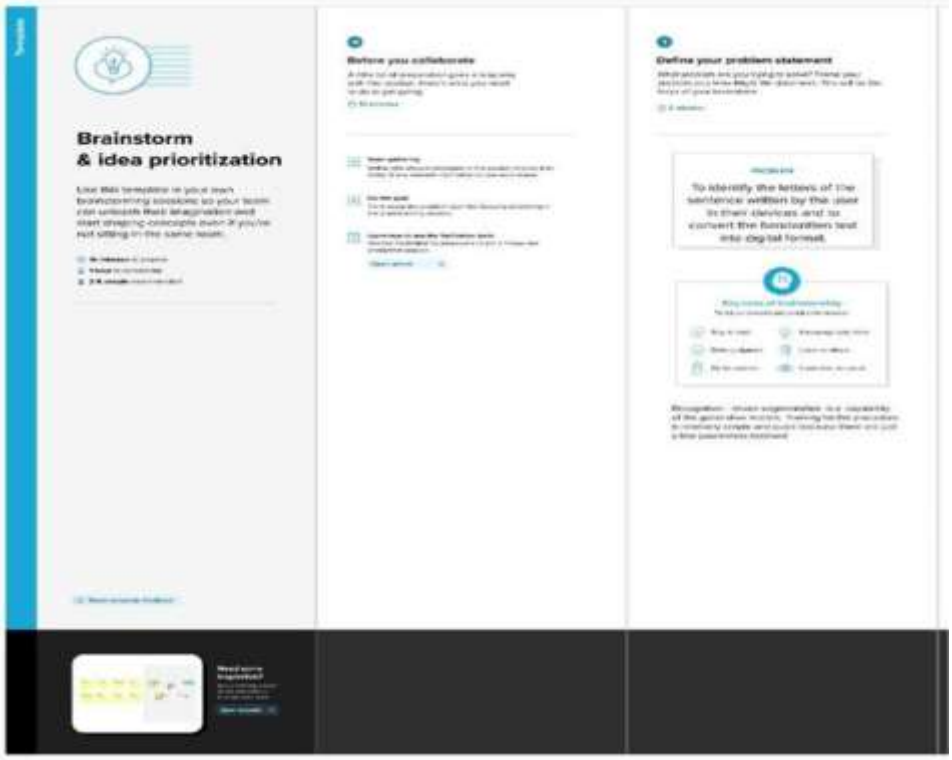
3. IDEATION & PROPOSED SOLUTION

Empathy Map Canvas:

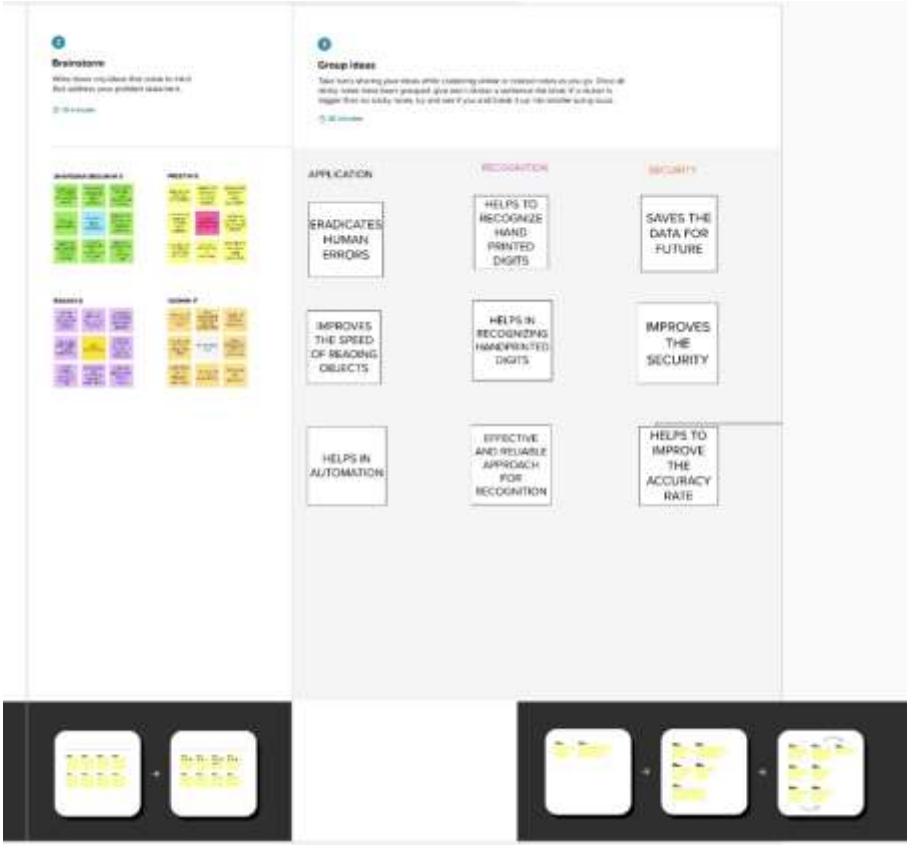


Ideation & Brainstorming:

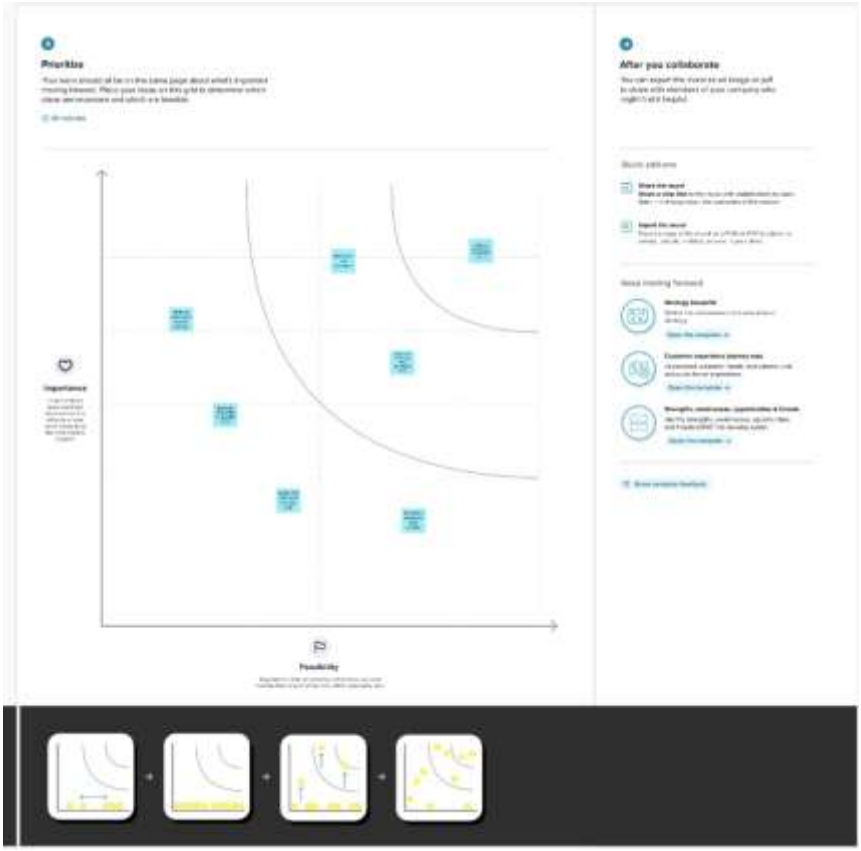
Step-1:TeamGathering,CollaborationandSelecttheProblemStatement



Step 2: Ideation



Step 3: Idea Prioritization



Team Member 1:

- Focus on one feature at a time
- Clean and Simple UI
- No bulk commits in GitHub
- Analyze existing solutions
- Compare with requirements more often
- Clean customer navigations

Team Member 2:

- Stick together as a team - frequent discussions
- Break into pieces - discuss, code, test and deploy
- Each phase - working model to beta test
- Use project management tools
- Analyse the data source and explore other sources for future use
- Bring drag and drop options to upload images + preview

Team Member 3:

- Discussion before every phase and writing code
- Get feedback from other friends as a user
- Develop while learning approach
- Analyze user requirements and thoughts
- Use collaborative todo apps
- Proper project structure - develop layout and logic subsequently

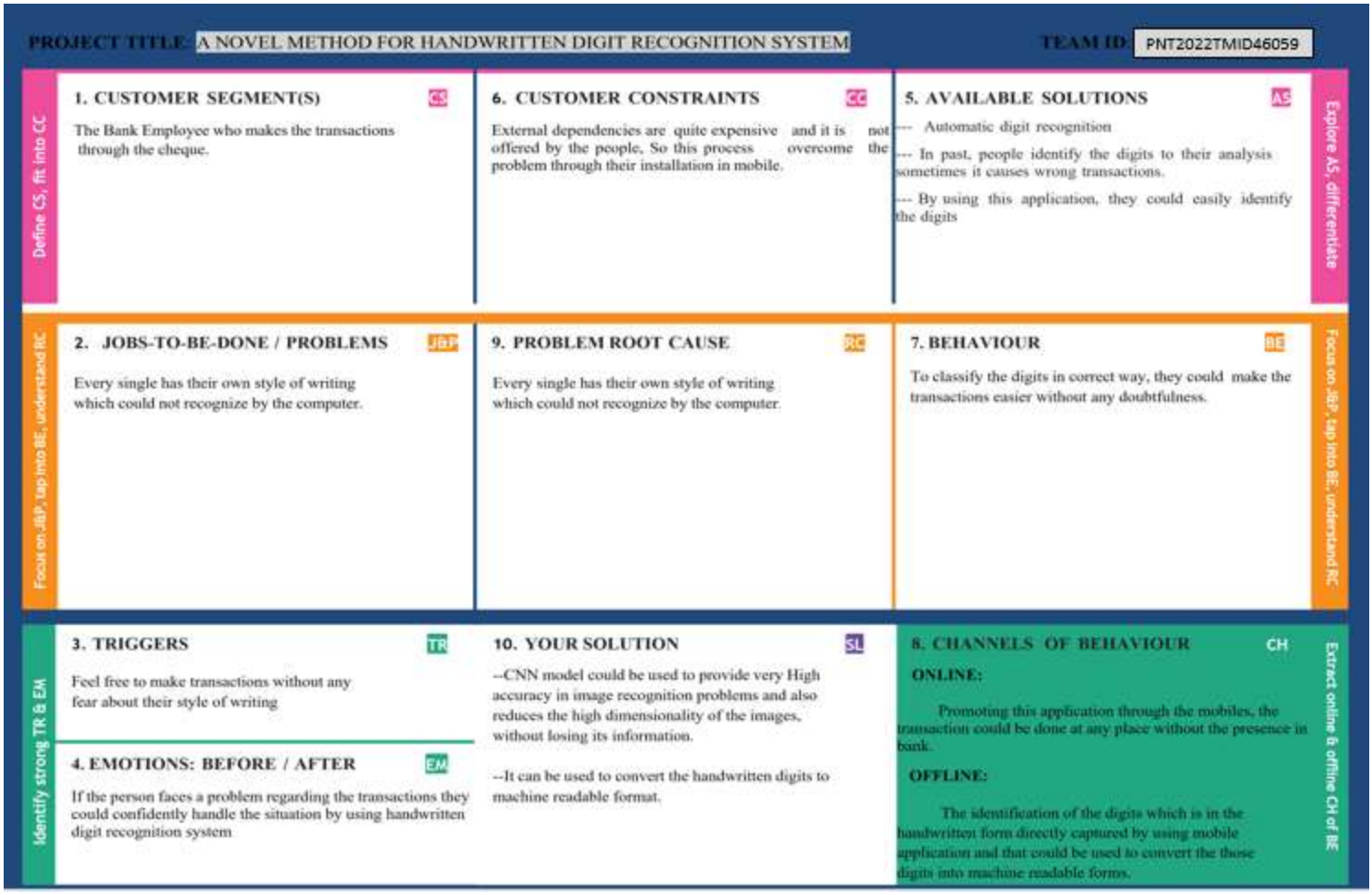
Team Member 4:

- Think like a user - then design
- No unwanted features - quality first
- clean and commented code - avoid inline comments
- Work in branches, and give pull request
- Consider scalability and continuous development right from the start
- Instead of developing all at once and deploying, do continuous development and deployment

Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	A Novel Method for Handwritten Digit Recognition System
2.	Idea / Solution description	The proposed solution is to classify the digits which is in handwritten format by using CNN based model and this model can be trained by using MNIST database which contains 60,000 training samples and 10,000 test samples.
3.	Novelty / Uniqueness	To classify the image datasets by using CNN, which provides efficient solution compare to other methods. Here ANN algorithm is used for voice recognition which helps blind people.
4.	Social Impact / Customer Satisfaction	Users no need to use external dependencies or devices to recognize the digits, this process can be done through our mobile phones.
5.	Business Model (Revenue Model)	⊓ Input module Image processing module Segmentation module ⊓ Feature extraction module Data set training module Classification module
6.	Scalability of the Solution	The accuracy of the result for the training data set is **99.98%**, and 99.40% with 50% noise by using MNIST. Even we can improve this model to achieve the better results by training different types of datasets.

Problem Solution fit:



4. REQUIREMENT ANALYSIS

Functional requirement:

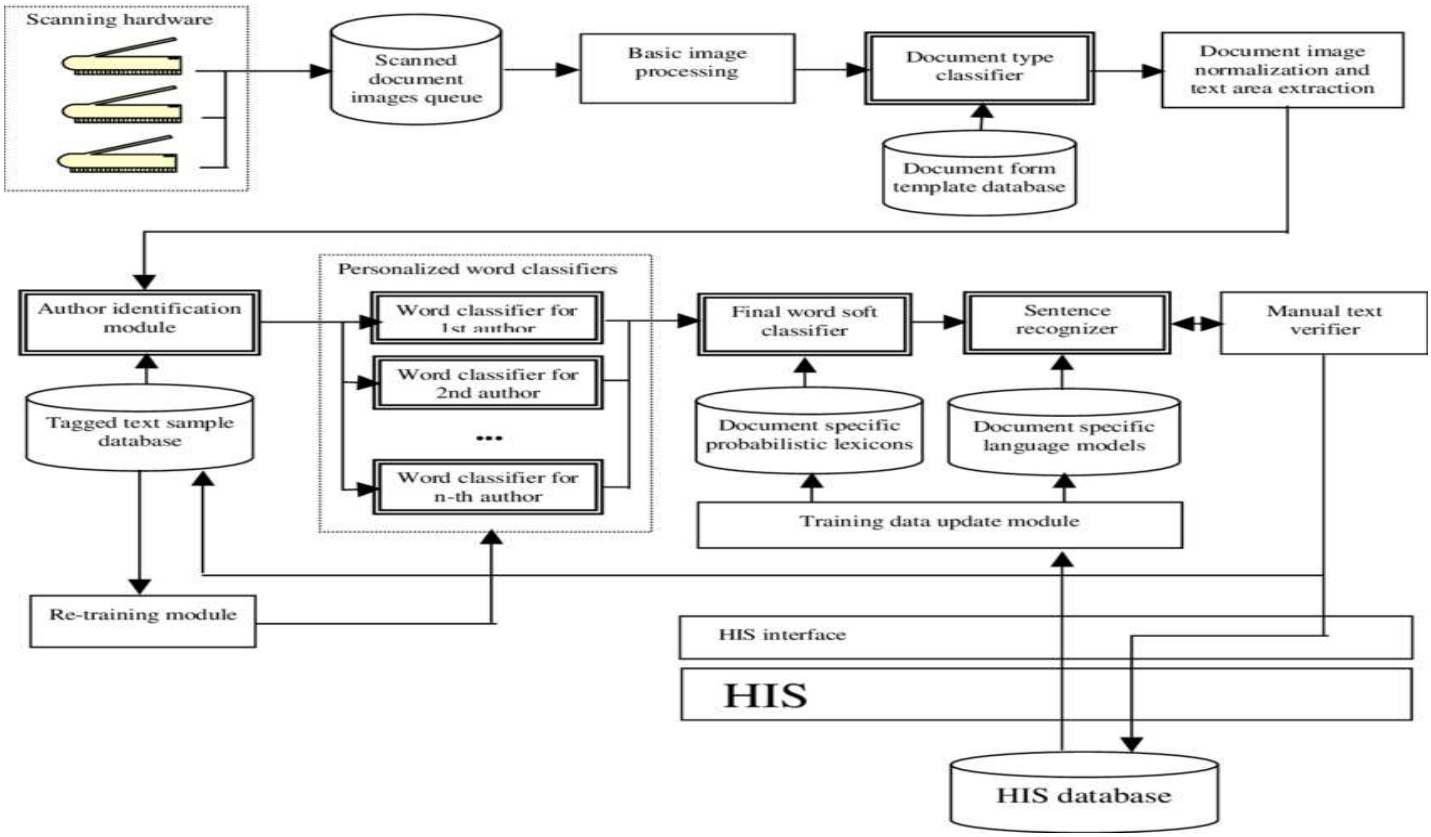
FR No.	Sub Requirement (Story / Sub-Task)
FR-1	Image Data: Handwritten digit recognition refers to a computer's capacity to identify human handwritten digits from a variety of sources, such as photographs, documents, touch screens, etc., and categorise them into ten established classifications (0-9). 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.

Non-Functional requirements:

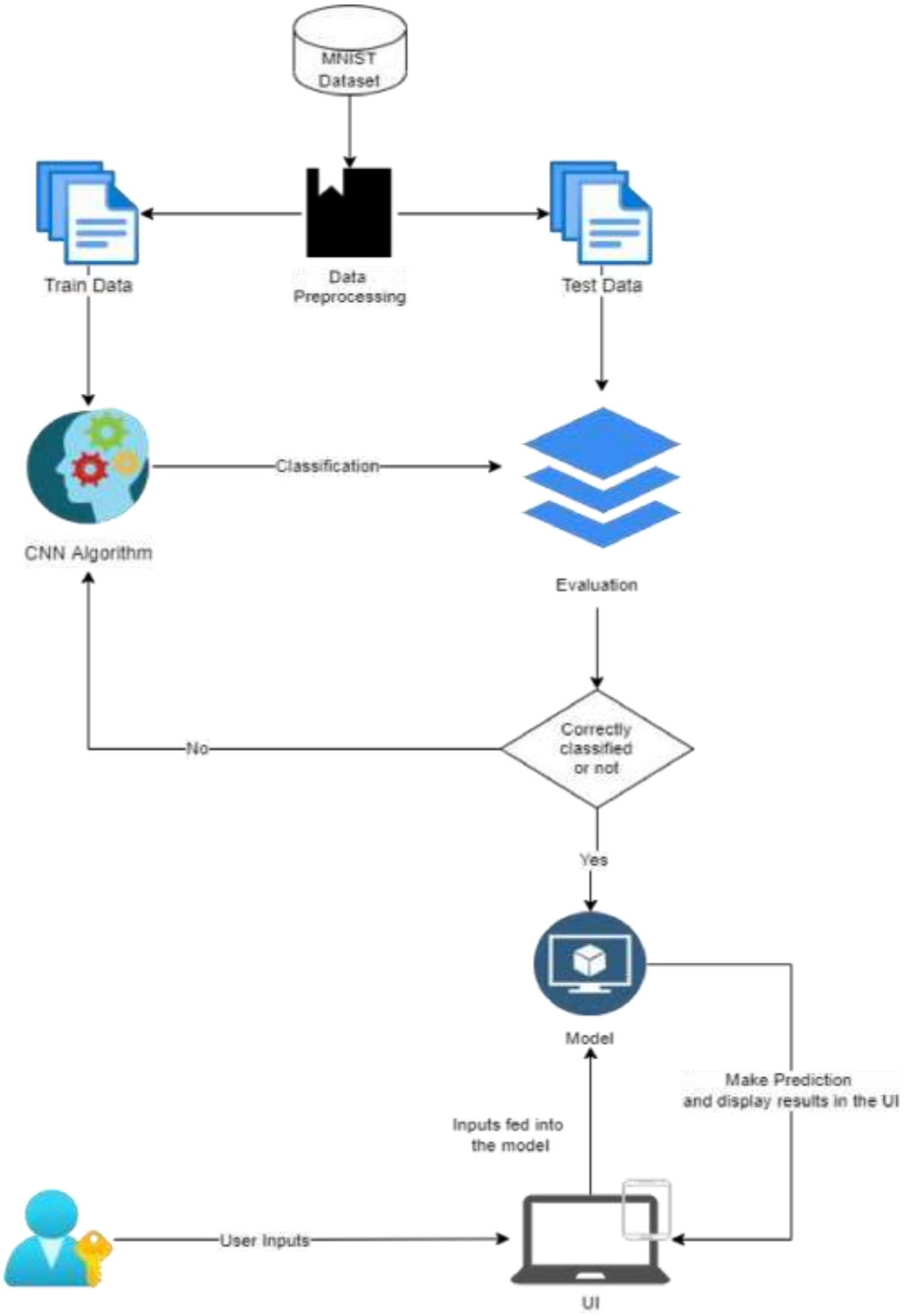
NFR-1	One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Applications for digit recognition include filling out forms, processing bank checks, and sorting mail.
NFR-2	1) The system generates a thorough description of the instantiation parameters, which might
NFR-3	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-4	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-5	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.

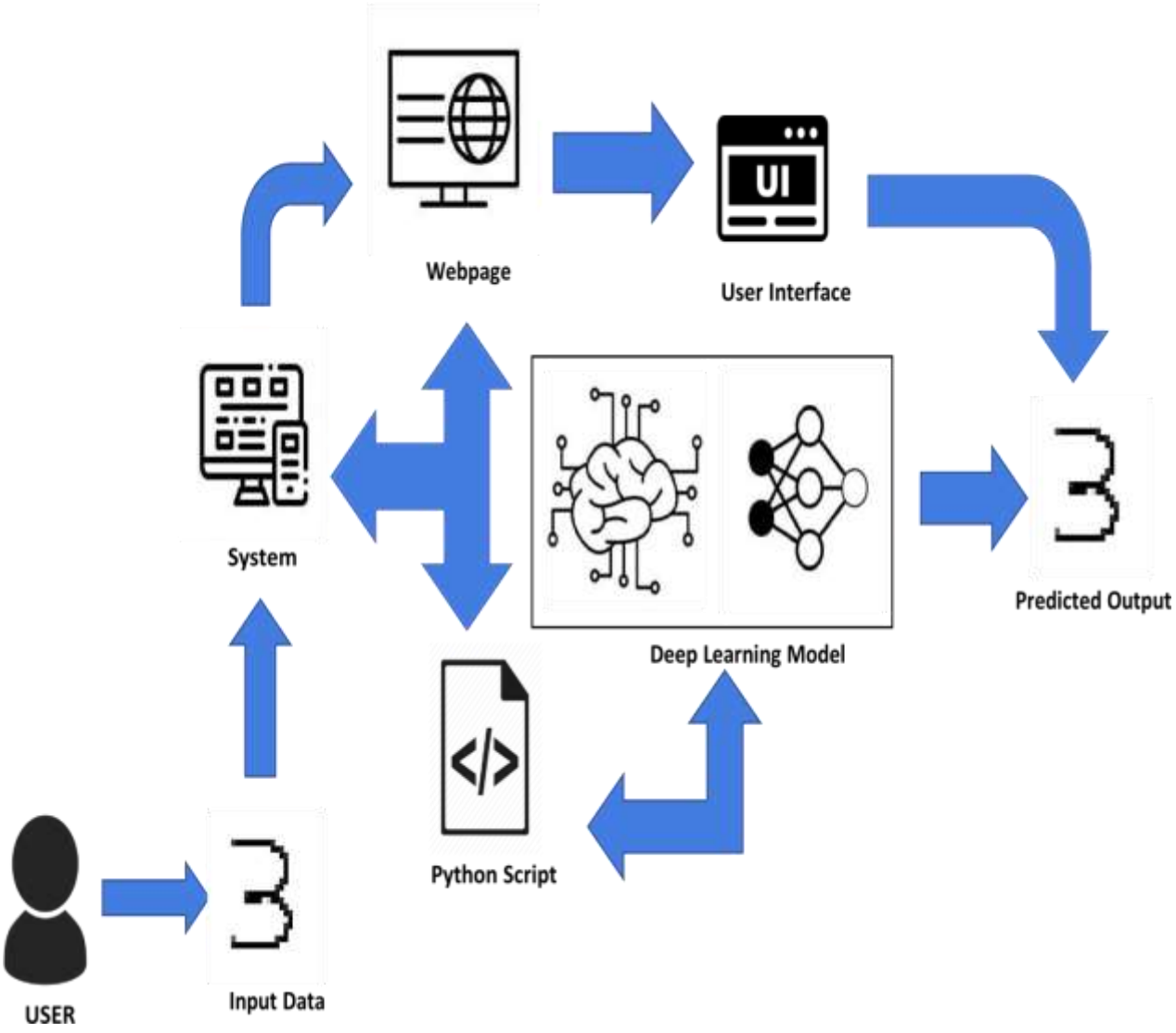
5. PROJECT DESIGN

Data Flow Diagrams:



Solution & Technical Architecture:




Webpage
UI
User Interface
System
Deep Learning Model
Predicted Output
Python Script
USER
Input Data

Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g., Mobile Application	HTML, CSS, JavaScript / AngularJS / Node Red.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on AI	IBM DB2.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc
9.	IoT Model	Purpose of AI Model is for integrating the sensors with a user interface.	IBM AI Platform
10.	Infrastructure (Server / AI)	Application Deployment on Local System / AI Local Server Configuration AI Server Configuration	Local, Kubernetes, etc.

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 userfriendly 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
	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

6. PROJECT PLANNING & SCHEDULING

Sprint Planning & Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Data Collection	USN-1	As a user, I can collect the dataset from various resources with different handwritings.	10	Low
Sprint-1	Data Preprocessing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test.	10	Medium
Sprint-2	Model Building	USN-3	As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit.	5	High
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High
Sprint-2	Compiling themodel	USN-5	With both the training data defined and model defined, it's time to configure the learning process.	2	Medium
Sprint-2	Train & test the model	USN-6	As a user, let us train our model withour image dataset.	6	Medium
Sprint-2	Save the model	USN-7	As a user, the model is saved & integrated with an android applicationor web application in order to predict something.	2	Low

Sprint-3	Building UI Application	USN-8	As a user, I will upload the handwritten digit image to the application by clicking a upload button.	5	High
Sprint-3		USN-9	As a user, I can know the details of the fundamental usage of the application.	5	Low
Sprint-3		USN-10	As a user, I can see the predicted / recognized digits in the application.	5	Medium
Sprint-4	Train the model on IBM	USN-11	As a user, I train the model on IBM and integrate flask/Django with scoring end point.	10	High
Sprint-4	Cloud Deployment	USN-12	As a user, I can access the web application and make the use of the product from anywhere.	10	High

7. CODING & SOLUTIONING

Feature 1:

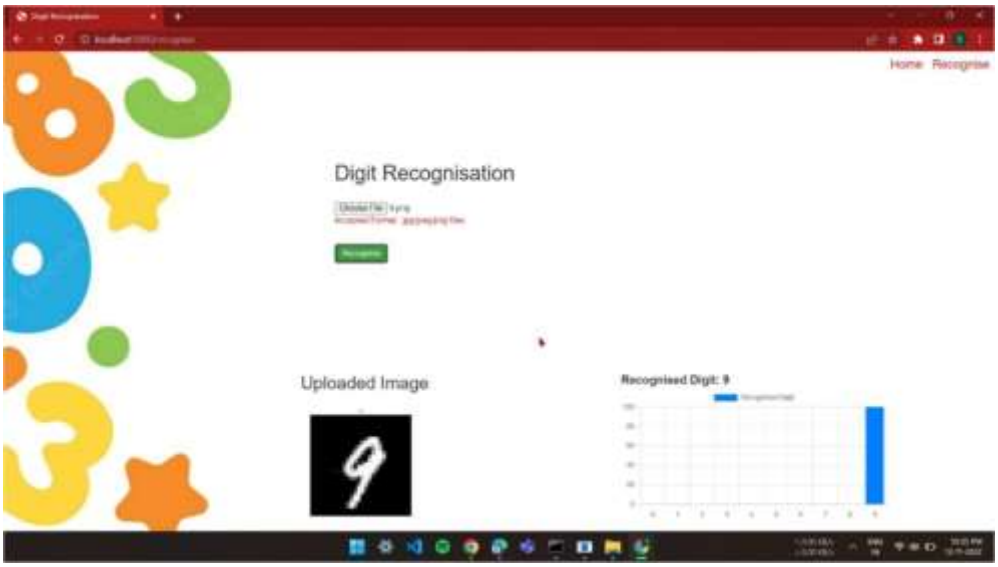
Dynamic Chart Production for Digits Module:

Source Code:

```
d = [0,0,0,0,0,0,0,0,0,0]
var myChart       = null;
var result_number =  document.getElementById('result-digit');
var result = document.getElementById('result1');
var  img_c =  document.getElementById('image1')

function  updateImage(event)
{
 img_c.style.display =  'block';
 var  img  =  document.getElementById('img1');
 img.src = URL.createObjectURL(event.target.files[0])
 img.onload = function(){
 URL.revokeObjectURL(img.src)
 }

}

function Chart1(d) {
 try
 {
    var  chart1  =  document.getElementById('myChart1');
    if(myChart)
    {
      myChart.destroy();
      document.querySelector("#chart-div1").innerHTML = '<canvas id =
"myChart1"></canvas>';
      chart1 = document.getElementById('myChart1');
      console.log('1')
    }
  }
  catch
  {
   console.log("Error")
  }
  const labels = [
     '0',
     '1',
     '2',
     '3',
     '4',
```

```
      '5',
      '6',
      '7',
      '8',
      '9'
    ];
    const data = {
      labels: labels,
      datasets: [{
        label: 'Recognised Digit',
        backgroundColor: 'rgb(0, 128, 255)',
        borderColor: 'rgb(255, 99, 132)',
        data: d,
      }]
    };

    const config = {
      type: 'bar',
      data: data,
      options: {}
    };
  myChart = new Chart(
    chart1,
    config
  )


}

async function predict() {
  url = "http://localhost:5000/predict"
  const formData = new FormData();
  const files = document.getElementById("myfile");
  formData.append("file", files.files[0]);
  const requestOptions = {
    headers: {
      "Content-Type": files.files[0].contentType,
    },
      mode: "no-cors",
      method: "POST",
      files: files.files[0],
      body: formData,
    };

  try{
    const response = await fetch(url,requestOptions);
```

```
        if (response) {
            var data = await response.json();
            var ans = data[1]
            d = [0,0,0,0,0,0,0,0,0,0]
            d[ans] = 100;
            result.style.display = 'block';
            result_number.innerText = ans;

            Chart1(d);


        }
    }
    catch(error) {
        console.log(error);
    }
}
```



Feature 2:

Digit recognition module:

Source Code:

```
from flask import Flask, jsonify, render_template, request
from PIL import Image
import numpy as np
# from tensorflow.keras.models import load_model
import tensorflow as tf
```

```
import numpy as np
import os
@app.route('/')
@app.route('/index')
def index():
    return render_template('home.html')

@app.route('/recognise')
def recognize():
    return render_template('recongise.html')

@app.route('/predict'  , methods=['GET','POST'])
def predict():
    data = request.files
    if request.method == 'POST':
        img = Image.open(data['file'].stream).convert("L")
        img = img.resize((28,28))
        im2arr = np.array(img)
        im2arr = im2arr.reshape(1,28,28,1)
        y_pred =  model.predict(im2arr)
        maxi = np.amax(y_pred)
        classes_x=np.argmax(y_pred,axis=1)
        # print(type(classes_x))
        return jsonify(str(classes_x))
        # for idx, x in np.ndenumerate(y_pred):
        #     if x == maxi:
        #         print(idx)
        #         return jsonify(idx)
        # print(list(y_pred).index(max(y_pred)))

        return render_template('recongise.html')
```

8. TESTING

Test Cases:

Importing the Required Libraries

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

Loading the Data:

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

Reshaping the Data:

```
X_train = X_train.reshape(60000, 28, 28, 1).astype('float32')
X_test = X_test.reshape(10000, 28, 28, 1).astype('float32')
```

Analyzing the Data:

```
print(X_train.shape)
print(X_test.shape)
(60000, 28, 28)
(10000, 28, 28)
X_train[0]
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]
```

Apply One Hot Encoding:

```
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)
```

```
print("After encoding the value 6 of y_test[22] become", y_test[22])
```

```
After encoding the value 6 of y_test[22] become [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
```

User Acceptance Testing:

Unit Testing:

Unit testing verification efforts on the smallest unit of software design, module. This is known as "Module Testing". The modules are tested separately. This testing is carried out during programming stage itself. In these testing steps, each module is found to be working satisfactorily as regard to the expected output from the module.

Integration Testing:

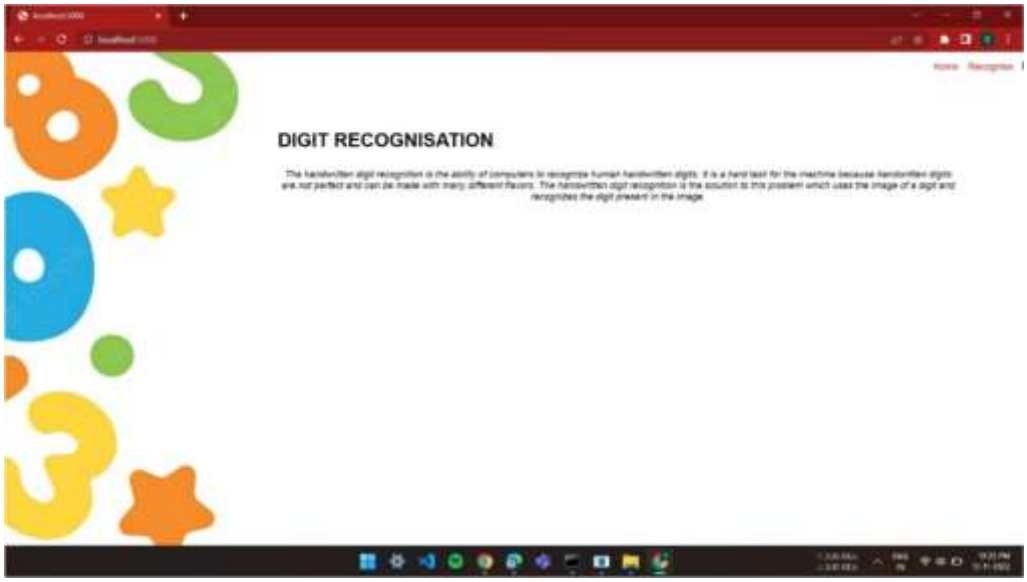
Integration testing is a systematic technique for constructing tests to uncover error associated within the interface. In the project, all the modules are combined and then the entire programmer is tested as a whole. In the integration-testing step, all the error uncovered is corrected for the next testing steps.

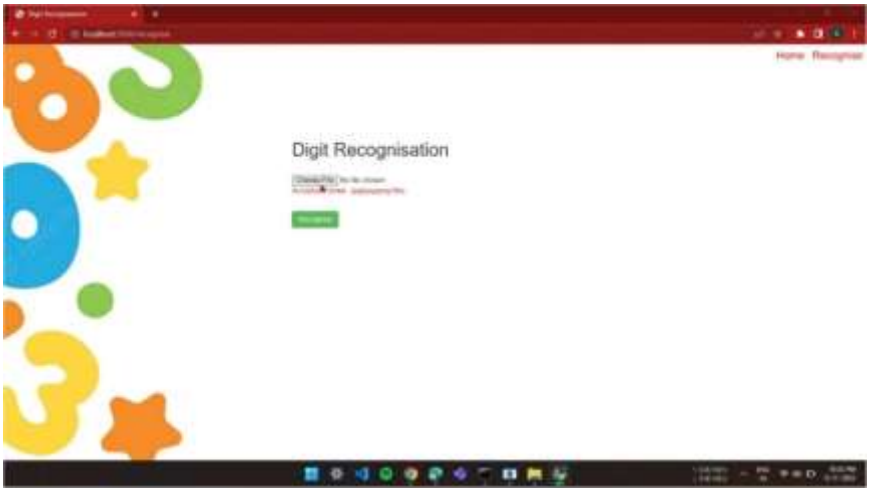
Validation Testing:

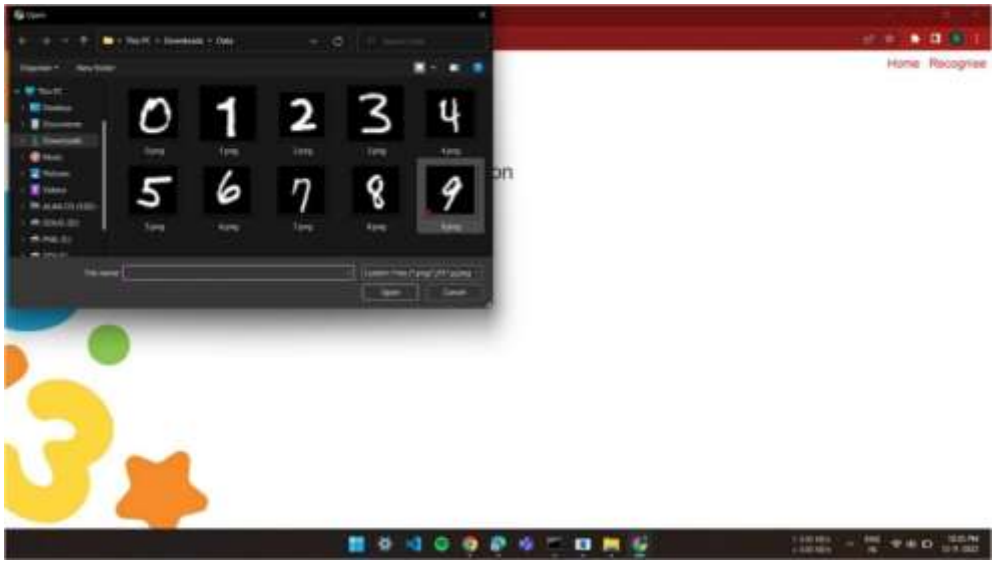
To uncover functional errors, that is, to check whether functional characteristics confirm to specification or not specified.

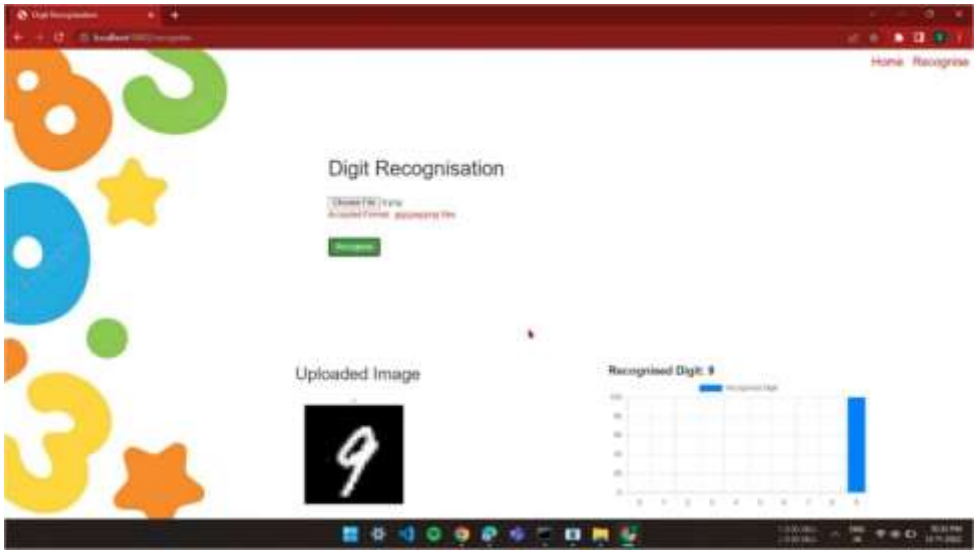
System Testing:

Once individual module testing completed, modules are assembled to perform as a system. Then the top down testing, which begins from upper level to lower level module testing, has to be done to check whether the entire system is performing satisfactorily.




Home Recognise
Digit Recognisation


Home Recognise


Home Recognise
Digit Recognisation
Uploaded Image
Recognised Digit: 9

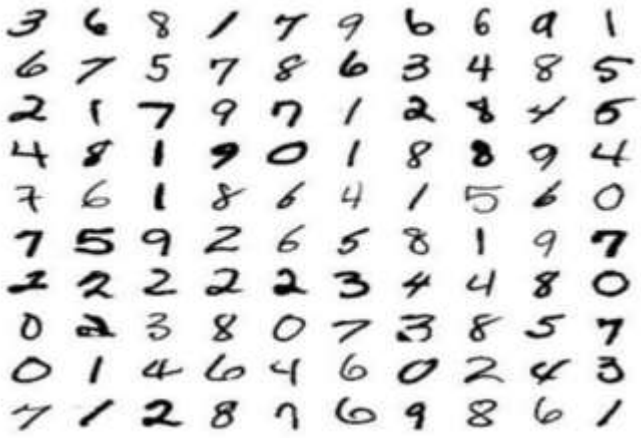
9. RESULTS

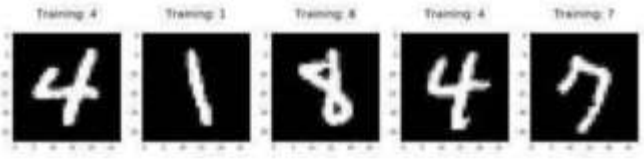
Performance Metrics:

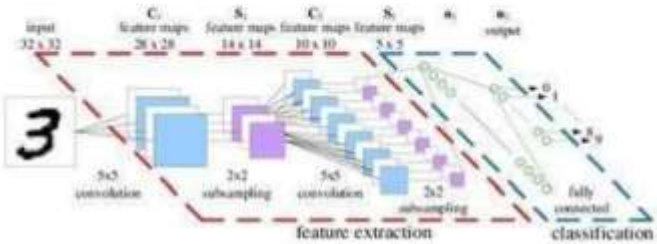
Handwritten digit recognition is one of the important problems in computer vision these days. There is a great interest in this field because of many potential applications, most importantly where a large number of documents must be dealt such as post mail sorting, bank cheque analysis, handwritten form processing etc. So a system should be designed in such a way that it is capable of reading handwritten digits and providing appropriate results. We propose a solution on neural network approaches to recognize handwritten digits.

Classification: Convolutional neural network that is very popular for computer vision tasks like image classification, object detection, image segmentation and a lot more. Image classification is one of the most needed techniques in today's era, it is used in various domains like healthcare, business, and a lot more.

Tensor flow: TensorFlow is an open-source machine learning library for research and production. TensorFlow offers APIs for beginners and experts to develop for desktop, mobile, web, and cloud. See the sections below to get started. By scanning the numerical digit and converting it into png format using the python3 command in the terminal we can get text output and sound output.







10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- **Image Segmentation:**
  Image segmentation is the process of partitioning an image into multiple segments. Image classification is the process of predicting a specific class, or label, for something that is defined by a set of data points
- **Character Recognition:**
  The handwritten character image is converted into printed text by sending the image to the model. In this model, each character is recognized after preprocessing and they are converted into readable text.
- Less Complicated.
- Easy to process
- Accuracy is more
- User friendly
- Consistent Response and Availability

DISADVANTAGES:

- Provides inappropriate result when handwritten image is not clear
- Technology issues
- Data should be some what clear for prediction

11. CONCLUSION

Handwriting recognition is very useful software that really helps to safe and keeps data and documents well. But at time it also has its disadvantage such as that sometimes if fails to read certain people's handwriting and due to this many people do not prefer to use the handwriting recognition software that much. Even though handwriting recognition has its disadvantages but still it is growing rapidly in the technology world. Handwriting recognition is used when there are certain people who prefer writing on the screens rather than writing it on a paper. In this project, Handwritten Digital Recognition is used In-depth learning strategies have been developed. Many widely used machine learning algorithms, RFC and CNN trained and tested on the same data into find comparisons between dividers. Use these are deeper learning methods, the higher the level of accuracy can be found. Compared to other research methods, this method focuses on which category works best for developing more than 99% separation accuracy models. Use Keras as backend and Tensorflow as software, CNN The model is able to provide about 98.72% accuracy. In the first one test, CNN provides 98.72% accuracy, while KNN provides 96.67% accuracy.

12. FUTURE SCOPE

Handwriting recognition is undoubtedly one of the most challenging areas of the pattern recognition. The goal of the project is to classify numeric samples which are mostly saved as digital images. Several pattern recognitions approaches have been applied to both online and off line handwriting recognition on the basis of unique patterns. The process of recognition consists of several steps such features extraction and recognition with voice alert. Python has a special toolbox, called neural network toolbox which makes the implementation less difficult but the knowledge of theory is needed. We can train these networks with preferred parameters. Artificial Neural Network approach for character recognition is now gaining importance because of CNN's high fault tolerance and parallel architecture. Comparing the performance of the proposed system with the eminent results of existing techniques, it is concluded that the new CNN based proposed system has achieved comparable performance to most of the existing techniques as well as it has exceeded the performance of others. Hence our results are comparable to state-of-the-art and this research will contribute positively to the research effort in the field of handwritten digit recognition. In future, other feature extraction methods and classification schemes will be considered for the digit recognition system.

13. APPENDIX

Source Code:

Importing the Required Libraries

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

Loading the Data:

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

Reshaping the Data:

```
X_train = X_train.reshape(60000, 28, 28, 1).astype('float32')
X_test = X_test.reshape(10000, 28, 28, 1).astype('float32')
```

Analyzing the Data:

```
print(X_train.shape)
print(X_test.shape)
(60000, 28, 28)
(10000, 28, 28)
X_train[0]
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]
```

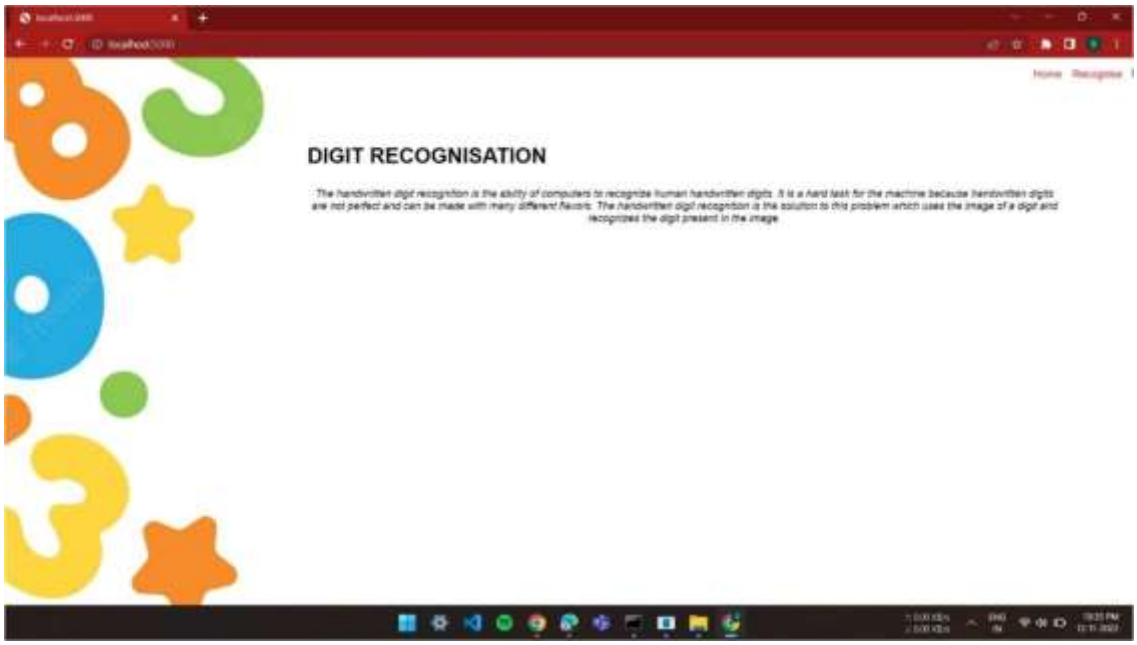
Apply One Hot Encoding:

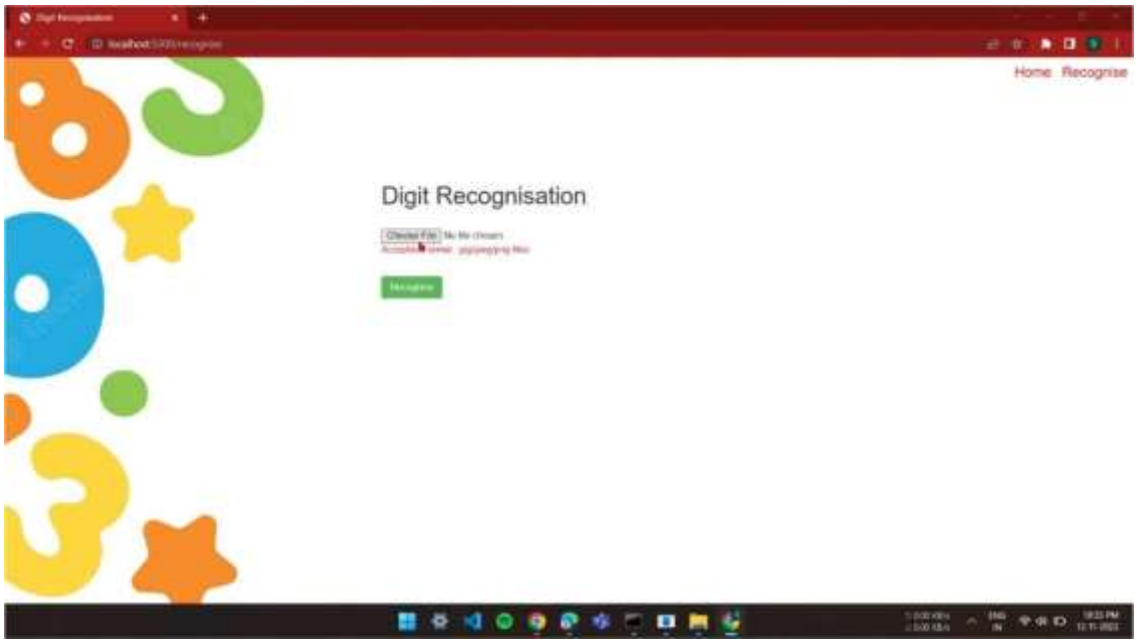
```
number_of_classes= 10
y_train = np_utils.to_categorical(y_train,number_of_classes)
y_test = np_utils.to_categorical(y_test,number_of_classes)
```

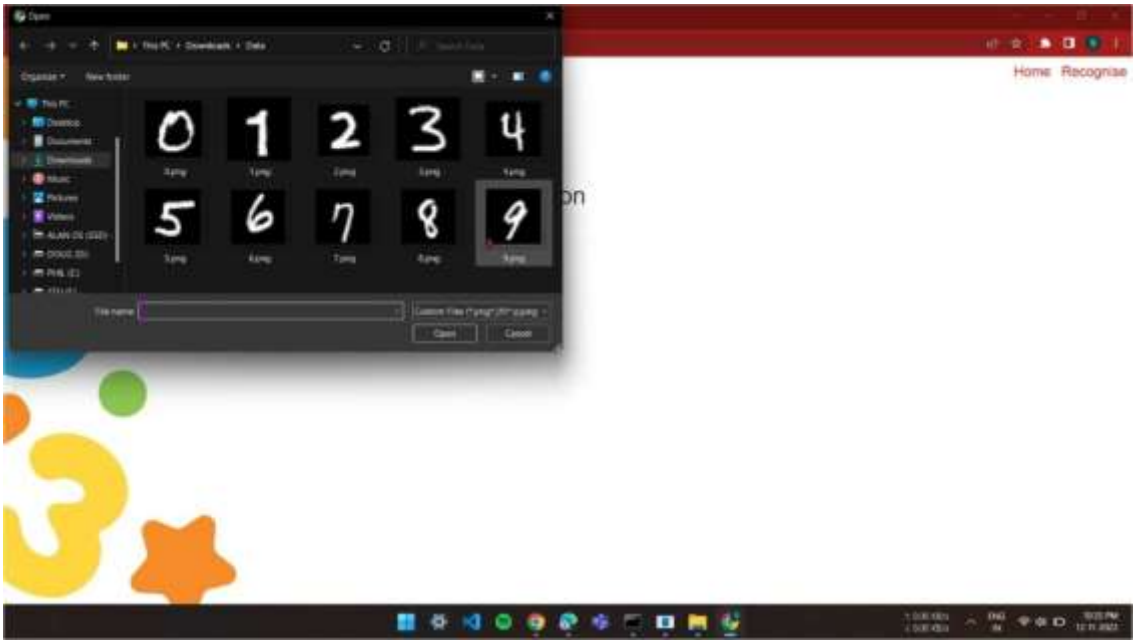
```
print("After encoding the value 6 of y_test[22] become", y_test[22])
```

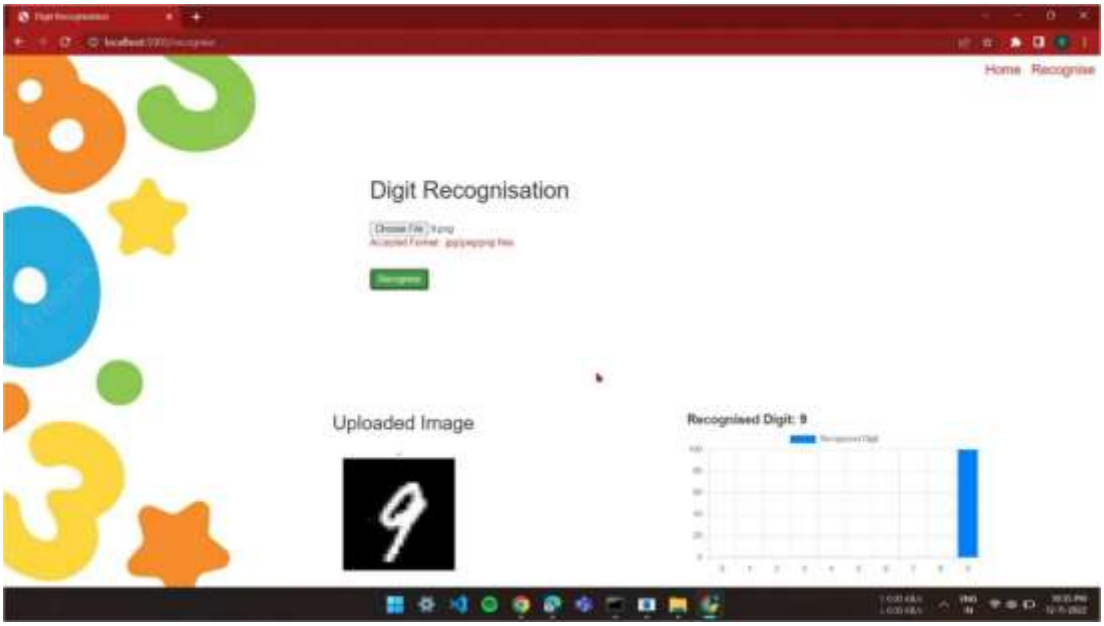
```
After encoding the value 6 of y_test[22] become [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
```

```
from flask import Flask, jsonify, render_template, request
from PIL import Image
import numpy as np
# from tensorflow.keras.models import load_model
import tensorflow as tf
import numpy as np
import os
@app.route('/')
@app.route('/index')
def index():
    return render_template('home.html')

@app.route('/recognise')
def recognize():
    return render_template('recongise.html')

@app.route('/predict'  , methods=['GET','POST'])
def predict():
    data = request.files
    if request.method == 'POST':
        img = Image.open(data['file'].stream).convert("L")
        img = img.resize((28,28))
        im2arr = np.array(img)
        im2arr = im2arr.reshape(1,28,28,1)
        y_pred =  model.predict(im2arr)
        maxi = np.amax(y_pred)
        classes_x=np.argmax(y_pred,axis=1)
        # print(type(classes_x))
        return jsonify(str(classes_x))
        # for idx, x in np.ndenumerate(y_pred):
        #     if x == maxi:
        #         print(idx)
        #         return jsonify(idx)
        # print(list(y_pred).index(max(y_pred)))

        return render_template('recongise.html')
```


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


Home Recognise
Digit Recognisation





GitHub Link:

https://github.com/IBM-EPBL/IBM-Project-49694-1660835299

Demo Link: https://drive.google.com/file/d/1UTcc7pGeYs4NrWdEsAzW-7_Oir6_vuOF/view?usp=share_link