

PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

Team ID: PNT2022TMID03362

TEAM MEMBERS:

SANTHOSH KUMAR B – 212219040138

MANOJ KUMAR V – 212219040074

MOHAMED IRFAN P – 212219040077

ROHITCHANDRAN R - 212219040123

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

1.1 Project Overview:

Handwriting recognition is one of the compelling research works going on because every individual in this world has their own style of writing. It is the capability of the computer to identify and understand handwritten digits or characters automatically. 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.

1.2 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

2.1 Existing System:

In most of the existing systems recognition accuracy is heavily dependent on the quality of the input document. In handwritten text adjacent characters tend to be touched or overlapped. Therefore, it is essential to segment a given string correctly into its character components. In most of the existing segmentation algorithms, human writing is evaluated empirically to deduce rules. But there is no guarantee for the optimum results of these heuristic rules in all styles of writing. Moreover, handwriting varies from person to person and even for the same person it varies depending on mood, speed etc. This requires incorporating artificial neural networks, hidden Markov models and statistical classifiers to extract segmentation rules based on numerical data.

2.2 References:

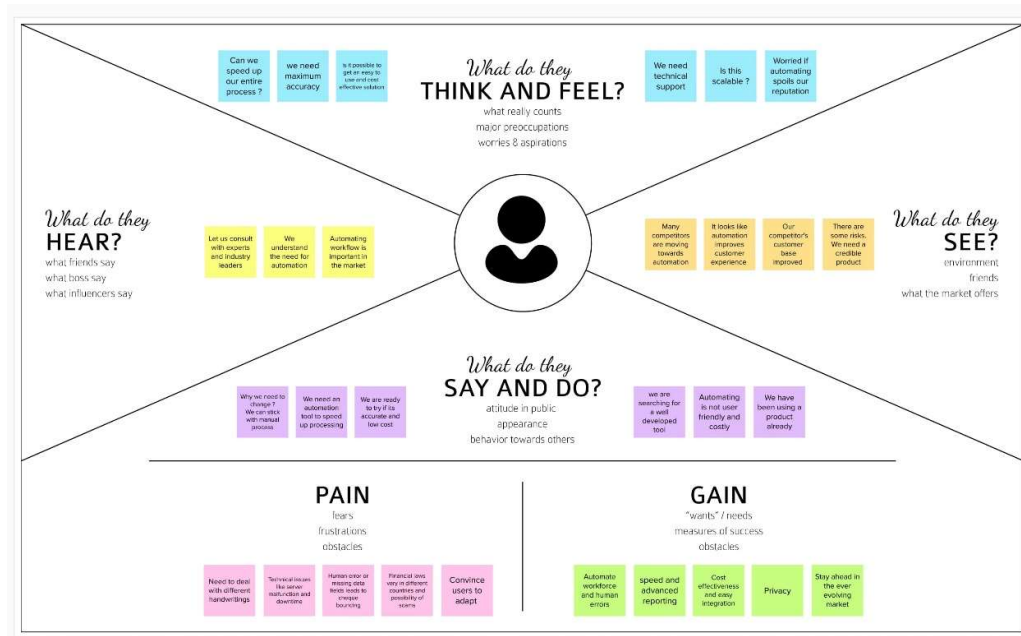
- [1] H. Zeng,(2020)An Off-line Handwriting Recognition Employing Tensorflow. International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE).
 - [2] A. Beltrán and S. Mendoza, "Efficient algorithm for real-time handwritten character recognition in mobile devices ",2011, 8th International Conference on Electrical Engineering, Computing Science and Automatic Control, 2011, pp. 1-6, doi: 10.1109/ICEEE.2011.6106583.
 - [3] H. A. Shiddieqy, T. Adiono and I. Syafalni, \"Mobile Client-Server Approach for Handwriting Digit Recognition\",2019 International Symposium on Electronics and Smart Devices (ISESD), 2019, pp. 1-4, doi: 10.1109/ISESD.2019.8909448.
 - [4] R. Vaidya, D. Trivedi, S. Satra and P. M. Pimpale, \"Handwritten Character Recognition Using DeepLearning\",2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018, pp. 772-775, doi: 10.1109/ICICCT.2018.8473291.
 - [5] H. Du, P. Li, H. Zhou, W. Gong, G. Luo and P. Yang, "Word Recorder: Accurate Acoustic-based Handwriting Recognition Using Deep Learning, "IEEE INFOCOM 2018 - IEEE Conference on Computer Communications, 2018, pp. 1448-1456, doi: 10.1109/INFOCOM.2018.8486285.
- Fig - 7: Image capturing Fig -8: Printed text output © 2021, IRJET | Impact Factor value: 7.529.

2.3 Problem Statement:

The goal of this project is 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. Though the goal is to create a model which can recognize the digits, it can be extended to letters and an individual's handwriting. The major goal of the proposed system is understanding Convolutional Neural Network, and applying it to the handwritten recognition system.

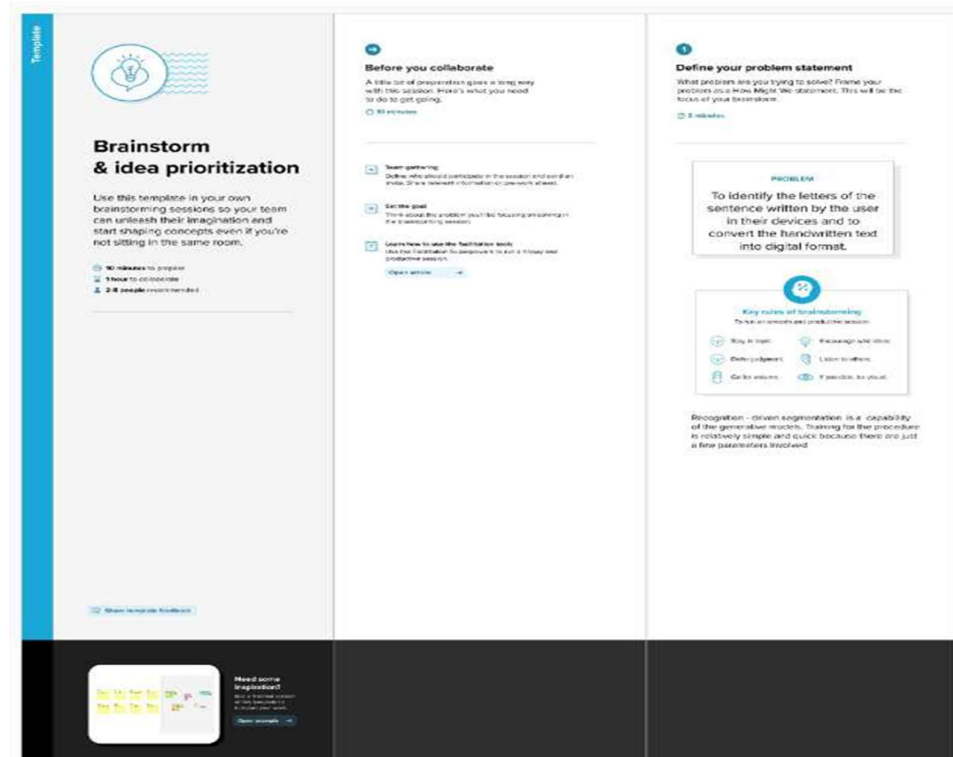
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:

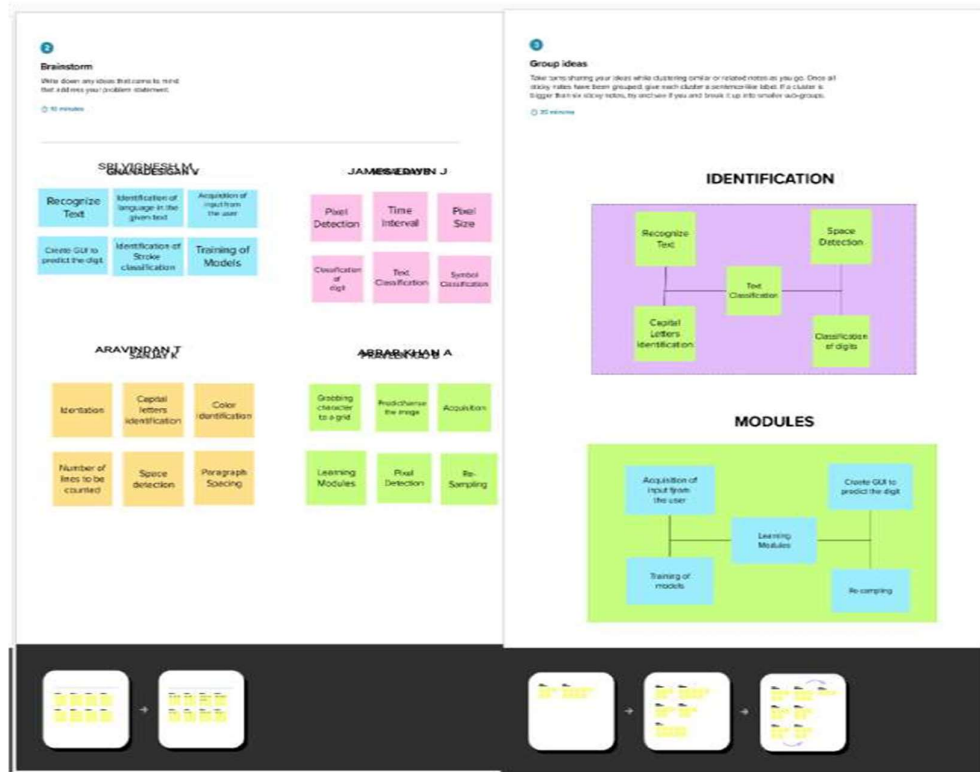


3.2 Ideation & Brainstorming:

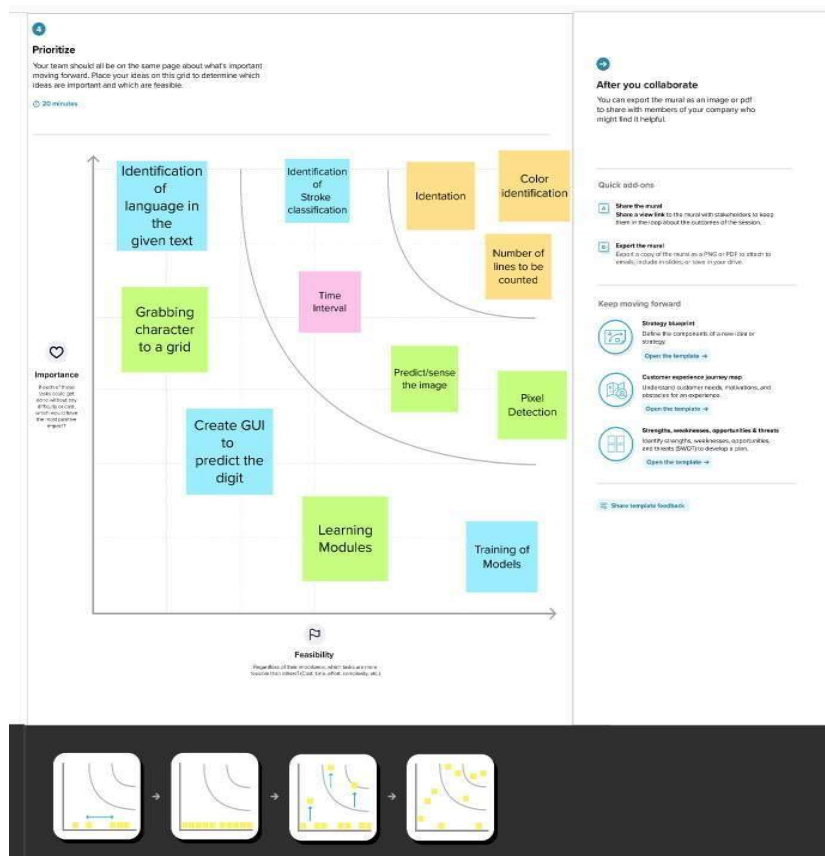
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step 2: Brainstorm, Idea Listing and Grouping



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

3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Computer programmes' ability to detect human-written numbers is known as handwritten digit recognition. Because handwritten figures are not always accurate and can take many various forms and sizes, it is a difficult work for the machine.
2.	Idea / Solution description	Using data from various sources, including images, documents, and touch defences, a computer is able to celebrate the mortal handwritten numbers. It permits users to convert all of their handwritten notes and signatures into text documents in electronic form, using much less physical space than would be needed to store the physical copies of those documents.
3.	Novelty / Uniqueness	Recognize the digits precisely rather than all the characters like OCR.
4.	Social Impact / Customer Satisfaction	The Handwritten Digit Recognizer software was made using artificial intelligence. It approximates the printed word digitally by identifying letters using sophisticated algorithms before producing a digital approximation.
5.	Business Model (Revenue Model)	For efficient traffic control, this technology can be connected with traffic surveillance cameras to read licence plates. Pin-code details can be easily identified and recognised by integrating with the postal system.
6.	Scalability of the Solution	The capacity to recognise numbers in more distracting circumstances. The maximum number of digits that can be recognised is unlimited.

3.4 Problem Solution fit:

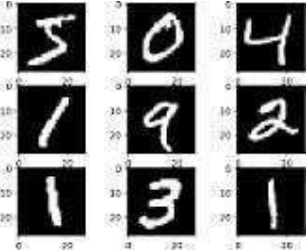
Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS <ul style="list-style-type: none"> Fintech Industries Supply Chain Management Medical data Transcriptions Scientific and Space Research 	2. CUSTOMER CONSTRAINTS CC <ul style="list-style-type: none"> Speed and Accuracy of the system Size of the vocabulary Spatial layout Lack of feedback-based system 	3. AVAILABLE SOLUTIONS AS <ul style="list-style-type: none"> Free OCR API Human centric data feed 	Explore AS, differentiate

Focus on J&P, top into BE, understand RC	4. JOBS-TO-BE-DONE / PROBLEMS J&P <ul style="list-style-type: none"> To design a system that recognizes a wide range of handwriting scripts ML based approach to identify the character quickly and accurately Adaptive learning module to learn from its own instances and gets updated 	5. PROBLEM ROOT CAUSE RC <ul style="list-style-type: none"> In cases where distinct characters look very similar making it hard for a computer to recognize it accurately. Different styles of cursive handwriting is another challenge that requires a support system based on vocabulary 	6. BEHAVIOUR BE <ul style="list-style-type: none"> In handwriting recognition (HWR), the module interprets the user's handwritten script into an appropriate digital format s Provision for real-time handwritten update in case if the application used by fixed and same users Know the market trends and adapt accordingly 	Focus on J&P, top into BE, understand RC

	7. TRIGGERS TR <ul style="list-style-type: none"> Longer and more in scale, the system understood better With its rich vocabulary, it has a support system to autofill the suggestions based on user input 	9. YOUR SOLUTION SoLN <ul style="list-style-type: none"> Deep learning. Intelligent feedback and support system based on neural network making the system more robust 	10. CHANNELS of BEHAVIOUR CH <p>1. ONLINE</p> <ul style="list-style-type: none"> online handwriting recognition consists of interpreting handwriting represented either by the trajectory of the pen or by scanning the script <p>2.OFFLINE</p> <ul style="list-style-type: none"> Offline handwriting recognition consists of

4. REQUIREMENT ANALYSIS

4.1 Functional requirement:

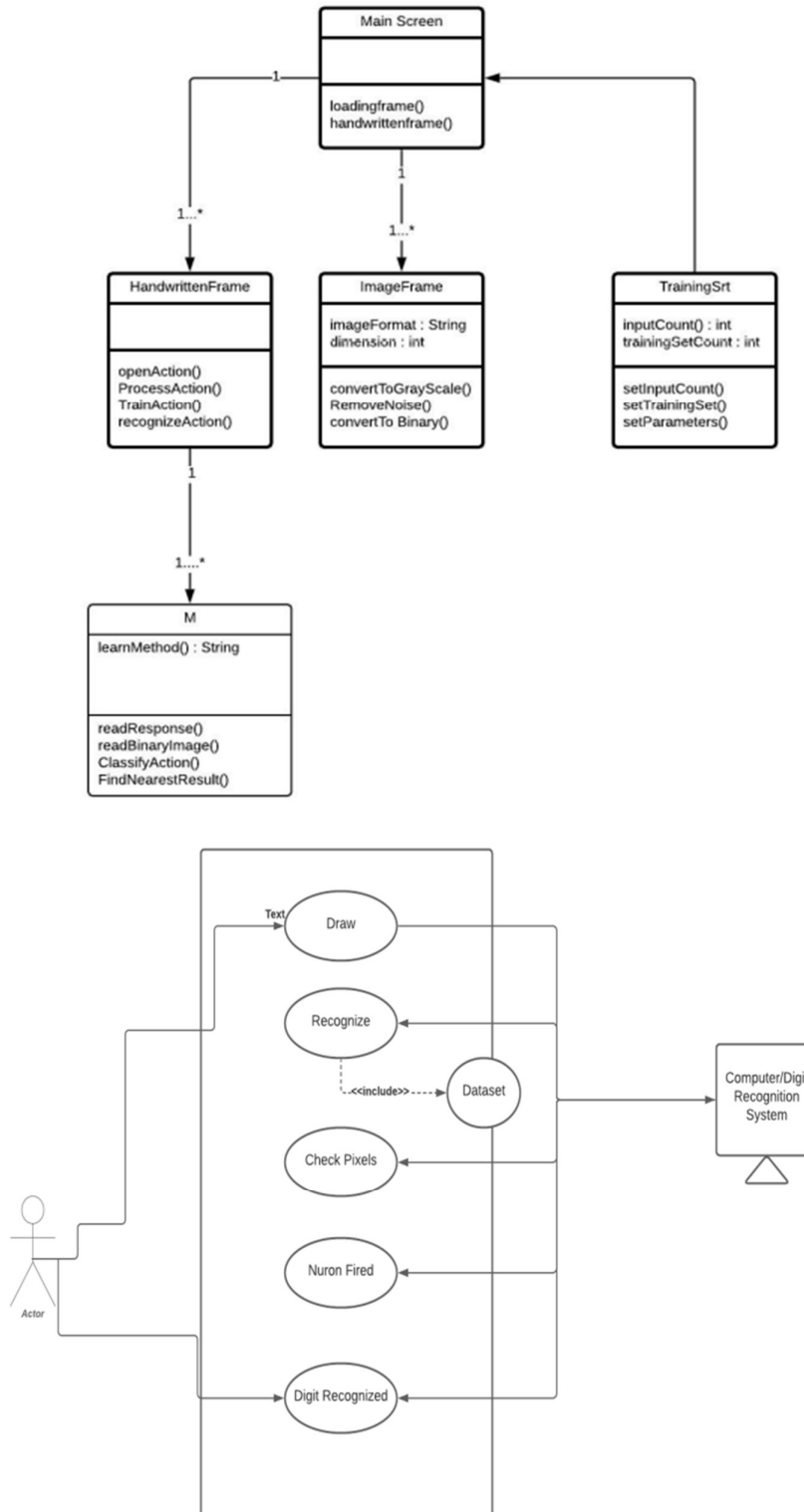
IR No:	Functional Requirement and description:
FR-1	Image Data: Handwritten digit recognition is the ability of a computer to recognize the human handwritten digits from different sources like images, papers, touch screens, etc., and classify them into 10 predefined classes (0-9). this has been a topic of boundless-research in the field of deep learning.
FR-2	Website: Web hosting makes the files that comprise a website (code, images, etc.) available for viewing online. Every website you've ever visited is hosted on a server. The amount of space allocated on a server to a website depends on the type of hosting. the main types of hosting are shared, dedicated, VPS..
FR-3	Digit_Classifier_Model: Use the MNIS database of handwritten digits to train a convolutional network to predict the digit given an image. First obtain the training and validation data.
FR-4	<p>MNIS dataset: the MNIST dataset is an acronym that stands for the Modified National Institute of Standards and Technology dataset. It is a dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9.</p> 
FR-5	Cloud: The cloud provides a number of IT services such as servers,
	databases, software, virtual storage, and networking, among others. In layman's terms, Cloud Computing is defined as a virtual platform that allows you to store and access your data over the internet without any limitations.

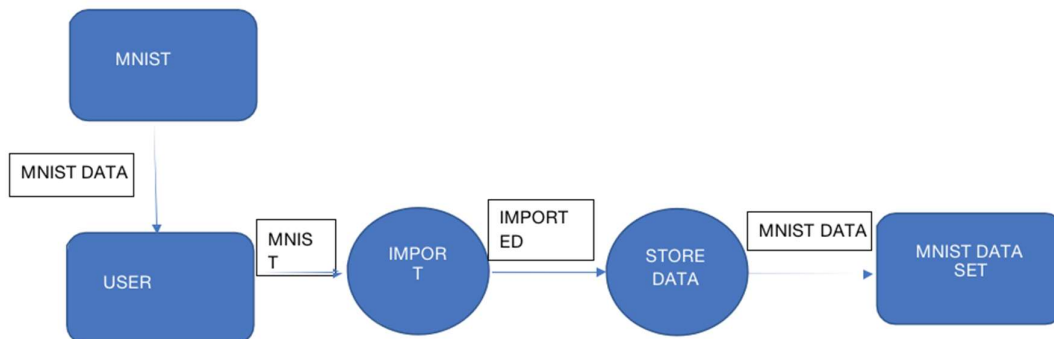
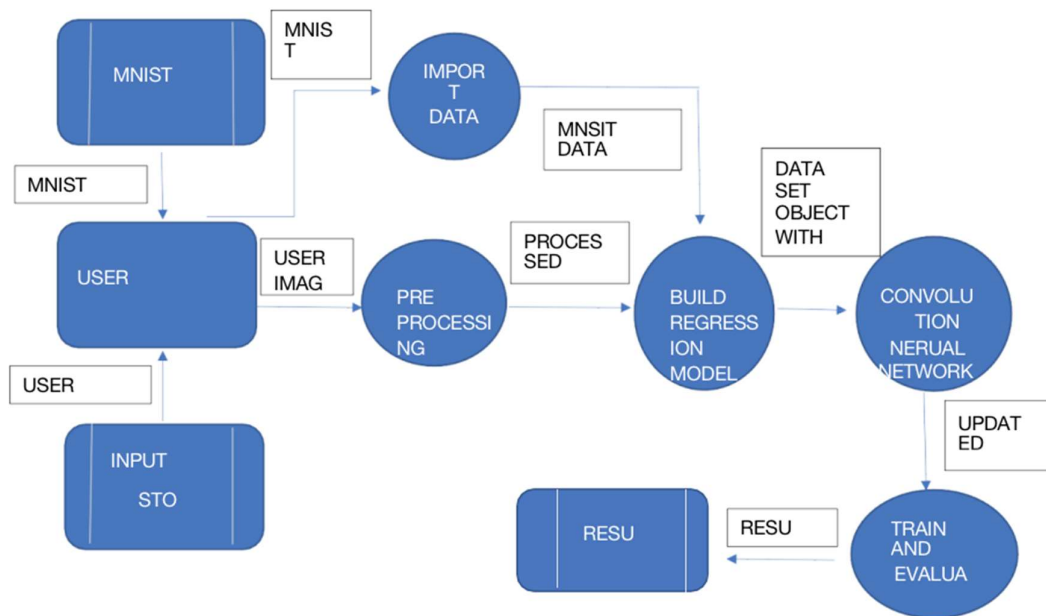
4.2 Non-Functional requirements:

NIR No.	Non-Functional Requirement
NFR-1	Usability: Handwritten character recognition is one of the practically important issues in pattern recognition applications. The applications of digit recognition include in postal mail sorting, bank check processing, form data entry , etc.
NFR-2	Reliability: 1) the system not only produces a classification of the digit but also a rich description of the instantiation parameters which can yield information such as the writing style. 2) the generative models can perform recognition driven segmentation. 3) the method involves a relatively.
NFR-3	Performance: the neural network uses the examples to automatically infer rules for recognizing handwritten digits . Furthermore, by increasing the number of training examples, the network can learn more about handwriting, and so improve its accuracy. there are a number of ways and algorithms to recognize handwritten digits, including Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision trees, Random Forests , etc.
NFR-4	Accuracy: Optical Character Recognition (OCR) technology provides higher than 99% accuracy with typed characters in high- quality images. However, the diversity in human writing types, spacing differences, and inequalities of handwriting causes less accurate character recognition.

5. PROJECT DESIGN

5.1 Data Flow Diagrams:





5.2 Solution & Technical Architecture:

The architectural diagram of the model is as below and the Technology used is shown in table1 & table 2.

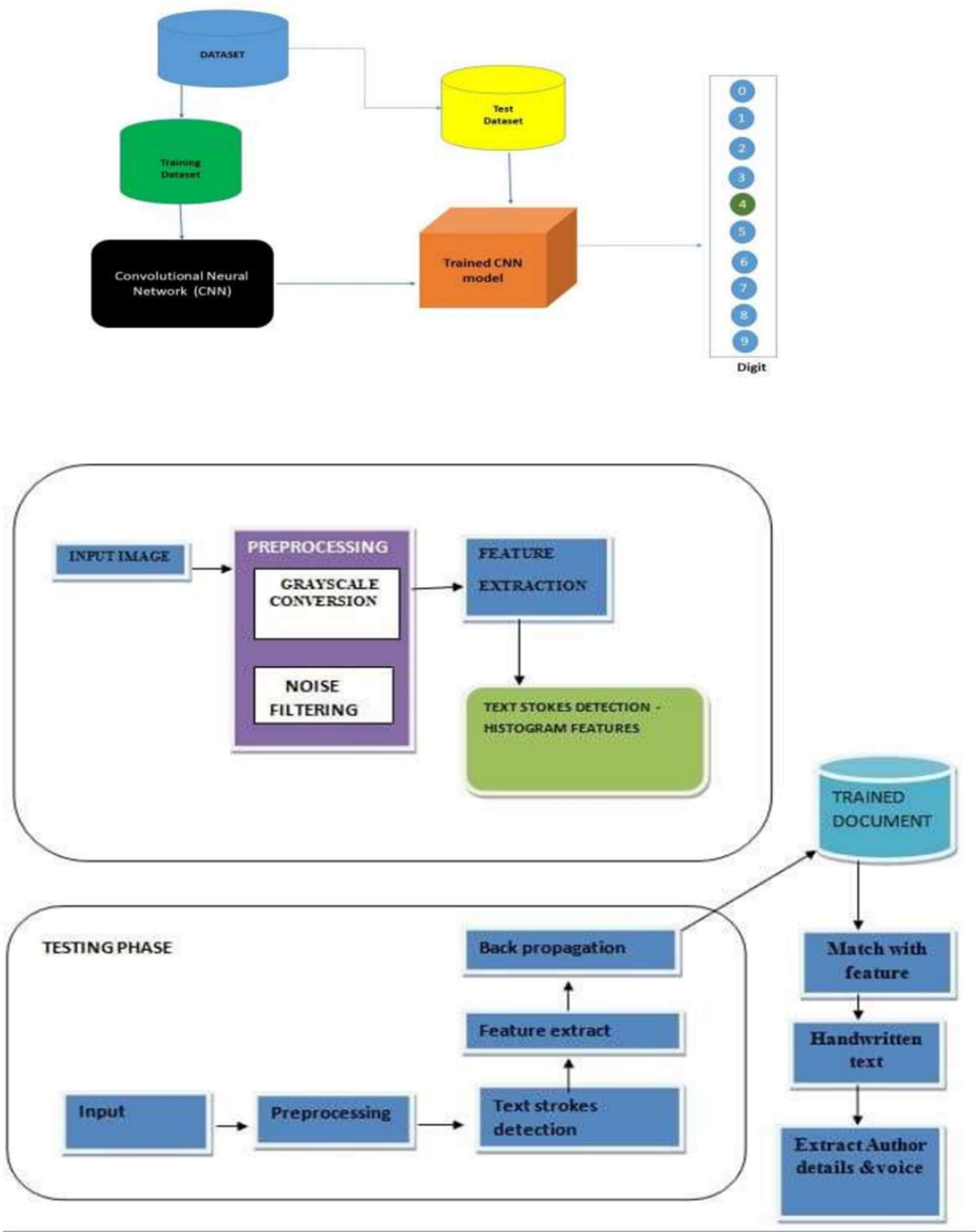


FIG. 1. BLOCK DIAGRAM

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.

5.3 User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-2
		USN-3	As a user, I can register for the application through gmail or facebook	I can register & access the dashboard with Facebook Login	Medium	Sprint-2
	Login	USN-4	As a user, I can log into the application by entering email & password	I can login to the application	High	Sprint-1
	Dashboard	USN-5	Go to dashboard and refer the content about our project	I can read instructions also and the home page is user-friendly	Low	Sprint-1
	Upload Image	USN-6	As a user, I can able to input the images of digital documents to the application	As a user, I can able to input the images of digital documents to the application	High	Sprint-3

	Predict	USN-7	As a user I can able to get the recognised digit as output from the images of digital documents or images	I can access the recognized digits from digital document or images	High	Sprint-3
		USN-8	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.	Medium	Sprint-4
Customer (Web user)	Login	USN-9	As a user, I can use the application by entering my email, password.	I can access my account	Medium	Sprint-4
Customer Care Executive	Dashboard	USN-10	upload the image	Recognize and get the output	High	Sprint-1
Administrator	Security	USN-11	updated the features	checking the security	Medium	Sprint-1

6. PROJECT PLANNING & SCHEDULING

6.1 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	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-2	Compiling the model	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 with our image dataset.	6	Medium
Sprint-2	Save the model	USN-7	As a user, the model is saved & integrated with an android application or 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

6.2 Sprint Delivery Schedule:

Project Planning Phase Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

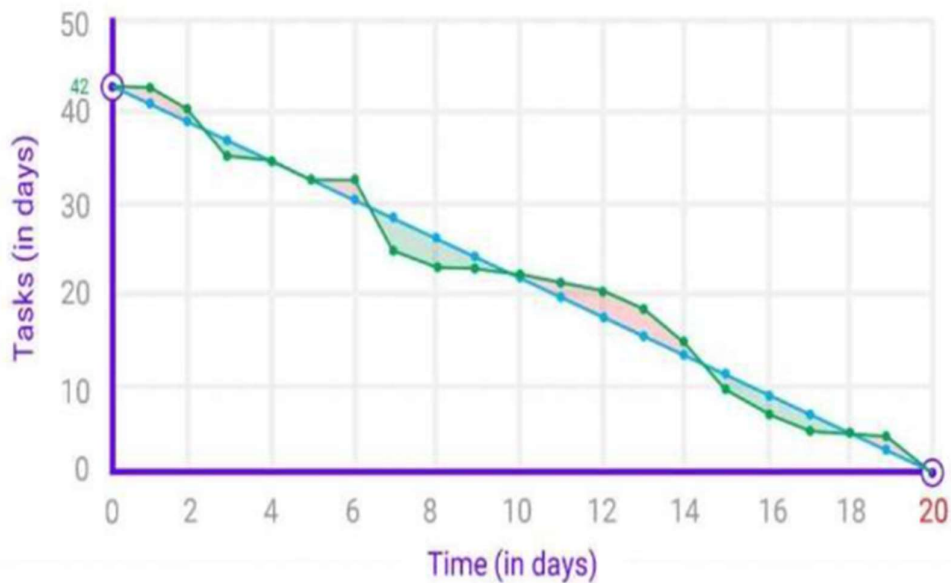
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022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

Velocity:

Imagine we have a 6-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)
 $AV = \text{sprint duration} / \text{Velocity} = 20 / 6 = 3.33$

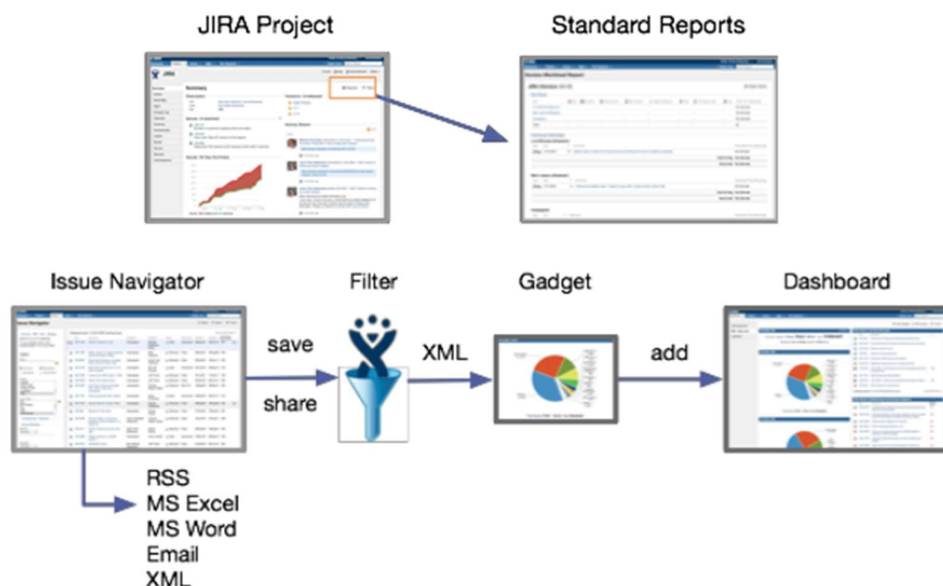
Burndown Chart:

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



6.3 Reports from JIRA:

JIRA offers reporting in a number of different formats. Project reports that are available from the home screen of the selected project, Gadgets that can be added and arranged in Dashboards and for each filter, the issue navigator offers various output formats that can be used in third party reporting software. Additionally, we will mention some advanced methods that customers have been using. In JIRA, a project will automatically offer standard reports available to the user without any necessary configuration. These standard reports comprise a wide range of reporting applications such as time tracking, workload and also abstract reports like Pie Charts that can be used in various ways.



7. CODING & SOLUTIONING

7.1 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);

```

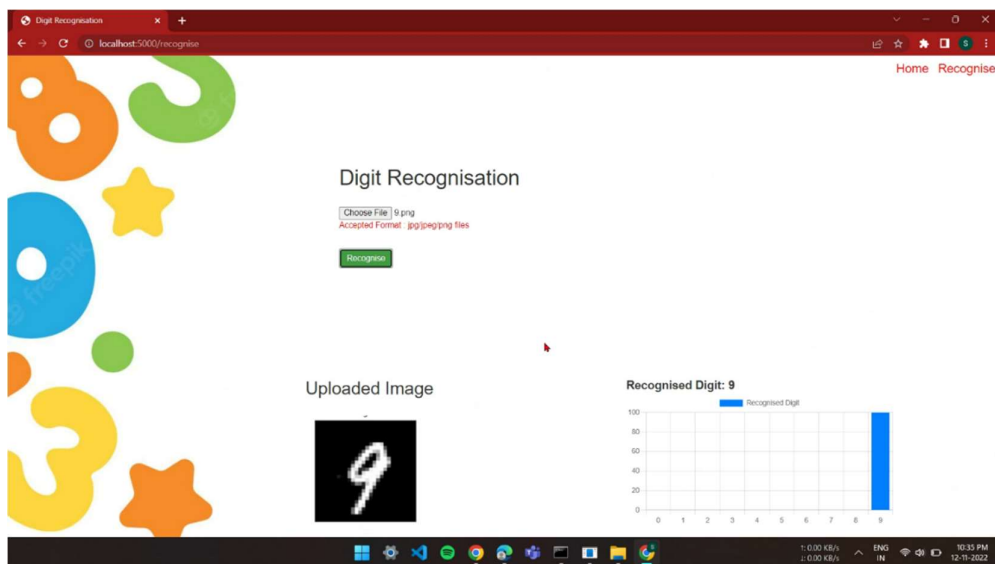
```

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

  Chart1(d);

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

```



7.2 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

8.1 Test Cases:

Importing the Required Libraries

```
import numpy
import matplotlib.pyplot as plt
from keras.utils import np_utils
from tensorflow.keras.datasets import mnist
import tensorflow.keras.models
import tensorflow.keras.layers
import tensorflow.keras.optimizers
import tensorflow.keras.backend as K
import numpy as np
import pandas as pd
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps
```

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.,  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.]
```

8.2 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:

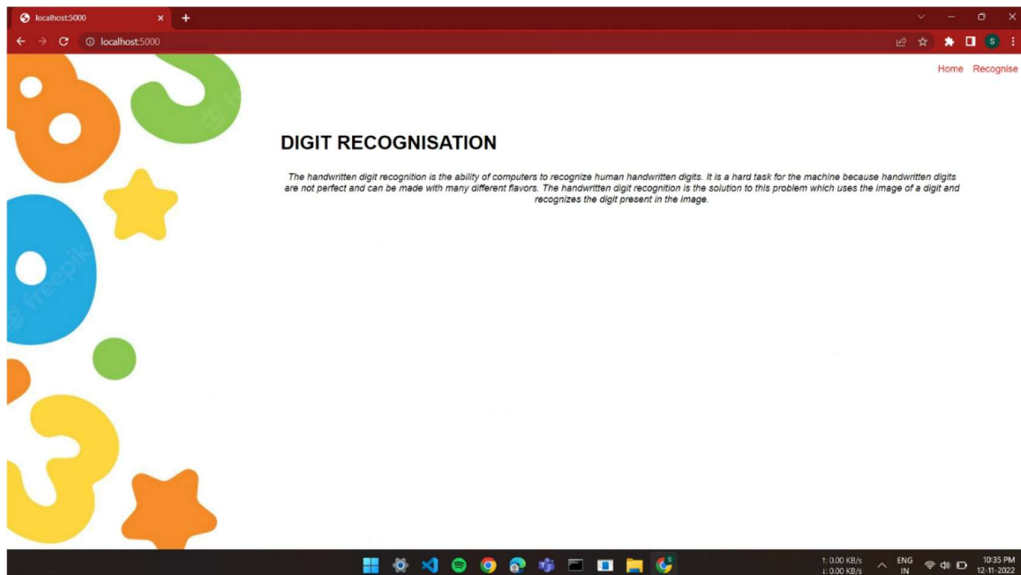
Integration testing is a systematic technique for constructing tests to uncover error associated within the interface. In the project, all the modules are combined and then the entire programmer is tested as a whole. In the integration-testing step, all the error uncovered is corrected for the next testing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

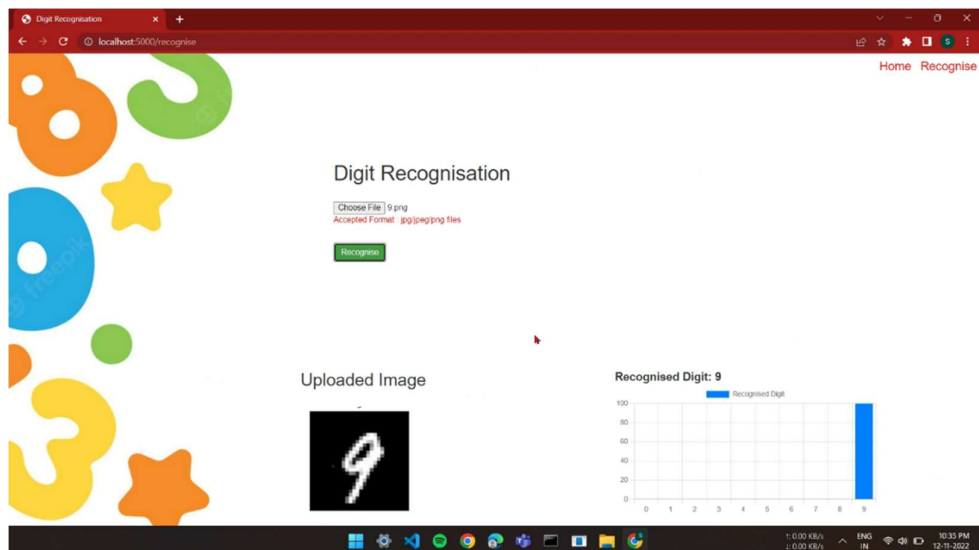
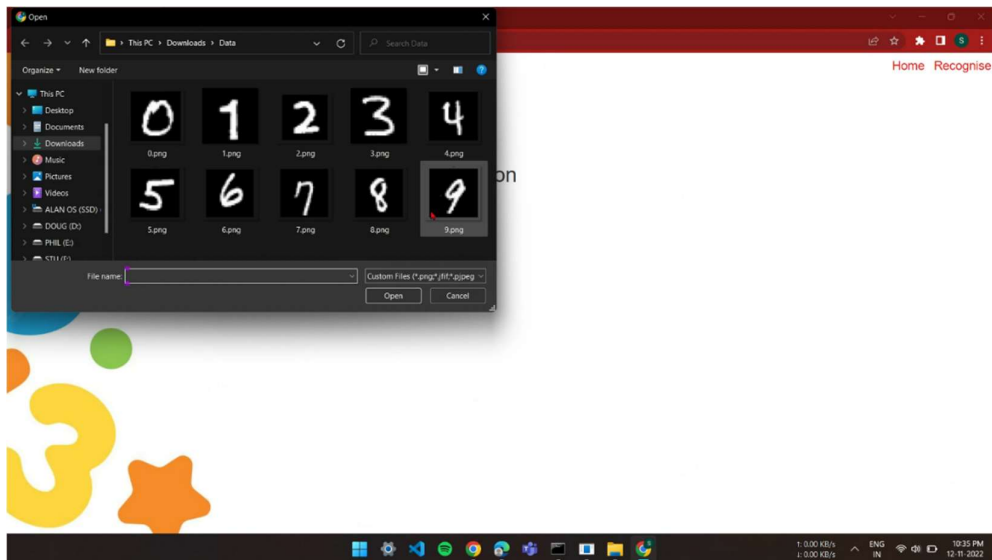
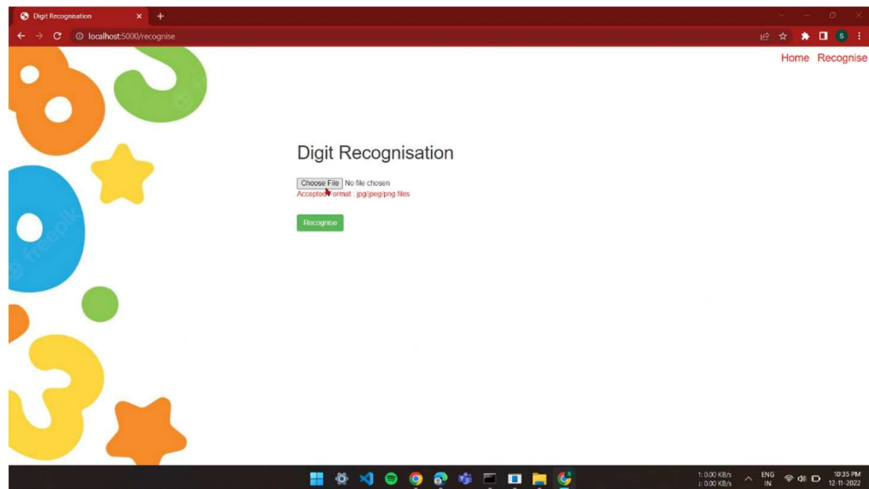
Validation Testing:

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.





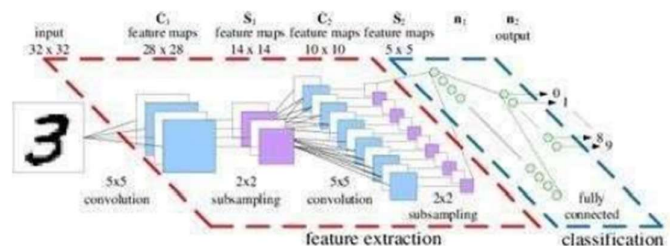
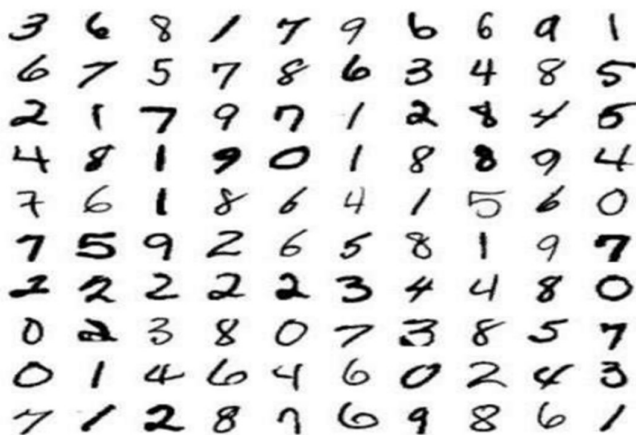
9. RESULTS

9.1 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 save and keep data and documents well. But at times it also has its disadvantages such as that sometimes it 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 to find comparisons between classifiers. Using 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. Using 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 import mnist
import tensorflow.keras.models
import tensorflow.keras.layers
import tensorflow.keras.optimizers
import numpy as np
import pandas as pd
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps
```

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:

[illegible]

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.]`


```

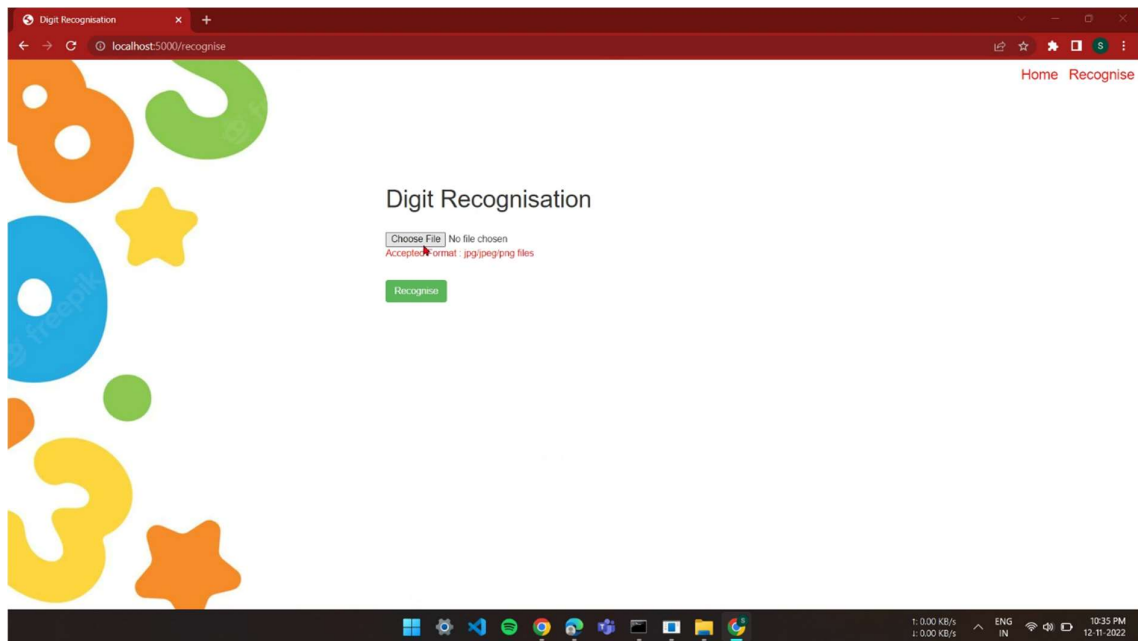
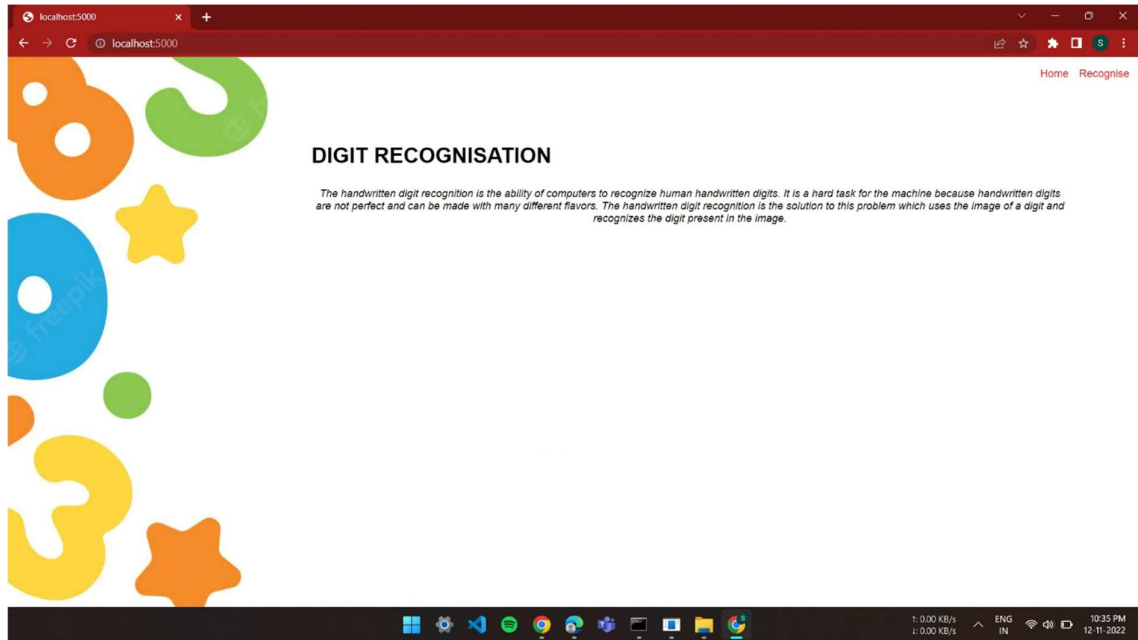
from flask import Flask, jsonify, render_template, request
from PIL import Image
import numpy as np
# from tensorflow.keras.models import load_model
import tensorflow as tf
import numpy as np
import os
@app.route('/')
@app.route('/index')
def index():
    return render_template('home.html')

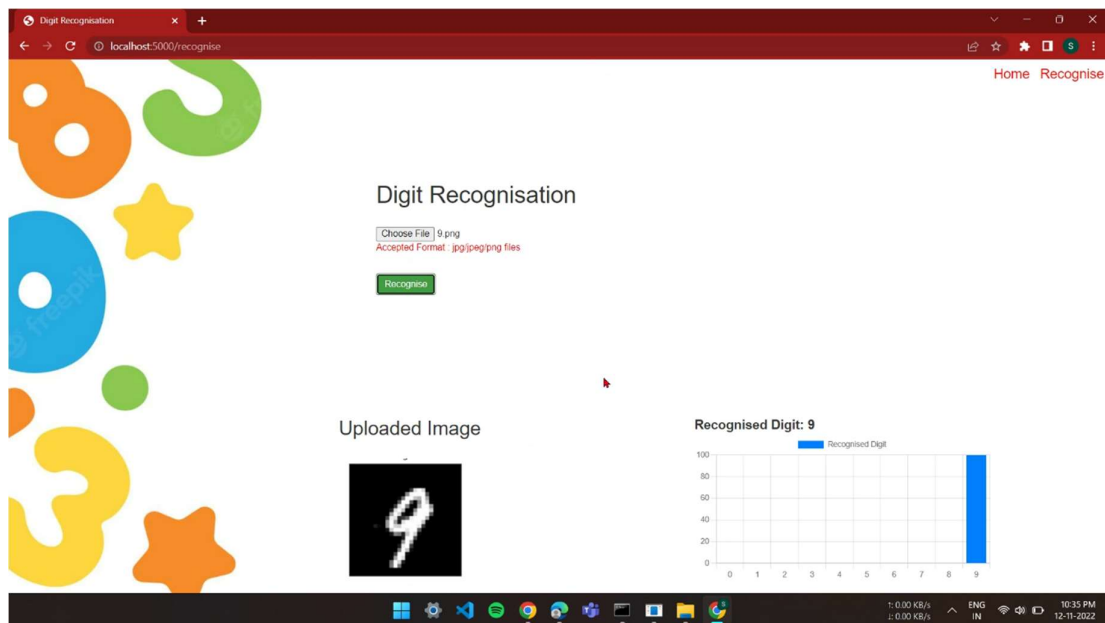
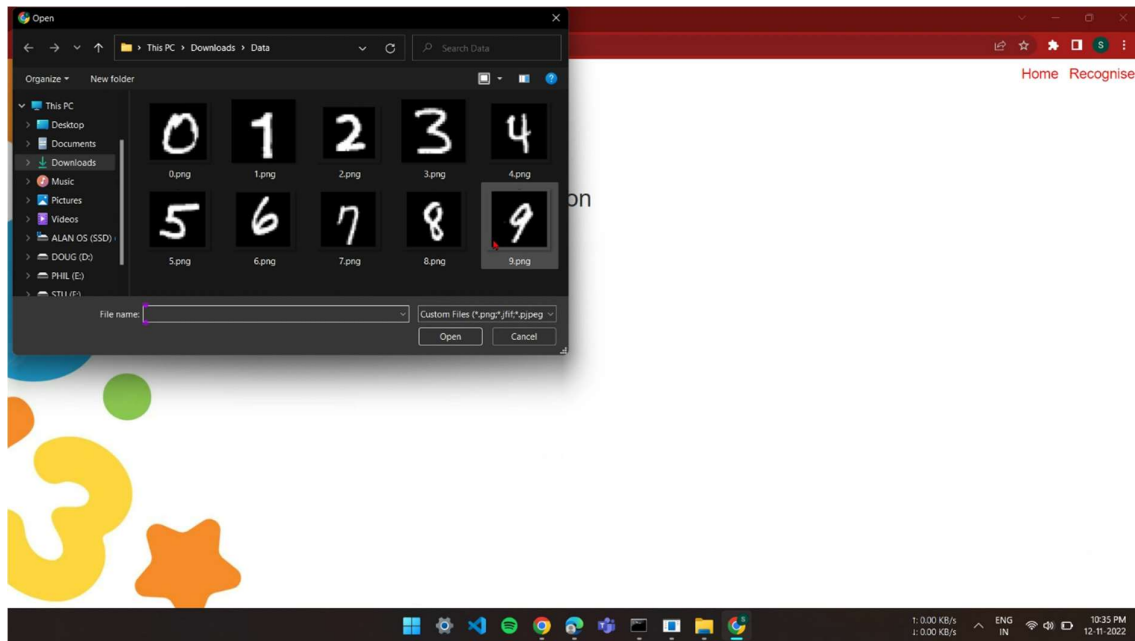
@app.route('/recognise')
def recognize():
    return render_template('recongise.html')

@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')

```





GitHub Link:

<https://github.com/IBM-EPBL/IBM-Project-14988-1659593048>

Demo Link:

https://drive.google.com/file/d/1UTcc7pGeYs4NrWdEsAzW-7_Oir6_yuOF/view