PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION

Submitted by

PNT2022TMID43340

PREM KUMAR I M - 715519104035

SATHYA VARSHINI R D - 715519104045

SUSMETA A - 715519104053

VISHNUDARSHAN S - 715519104306

YASHWANT C M - 715519104060

INDEX

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. Because everyone in the world has a unique writing style, handwriting identification is one of the fascinating research projects now being conducted. It is the ability of a computer to automatically recognise and comprehend handwritten numbers or letters.

Everything is being digitised to minimise human labour as a result of advancements in science and technology. Thus, handwritten digit recognition is required in many real-time applications. The MNIST data collection, which contains 70000 handwritten digits, is frequently utilised for this recognition method. In order to train these photos and create a deep learning model, we use artificial neural networks. A web application is developed that allows users to upload pictures of handwritten numbers. The model examines this picture and the detected result is returned to the UI.

1.2 PURPOSE

Digit recognition systems can recognize digits from a variety of sources. It is the classification ability of a computer to detect human handwritten digits from various sources such as photographs, papers, touch screens and classify them among one of the digits from 0-9.

- Handwritten digit recognition system can be used in processing bank check amounts, numeric entries in forms filled up by hand.
- It can be used to handles formatting, performs correct segmentation into characters, and finds the most plausible words.
- It can be used by postal services to identify postal codes, recognize zip codes on mail for postal mail sorting.
- It can be used to read form data.
- It can be used to help visually impaired people.

LITERATURE SURVEY

2.1 PROBLEM STATEMENT

The main issue with handwritten digit recognition is that because handwriting varies from person to person, handwritten digits do not always have the same size, width, orientation, and margins. In addition, it would be difficult to distinguish the numbers due to similarities between the numerals, such as 1 and 7, 5 and 6, 3 and 8, 2 and 5, and 2 and 7. Finally, the uniqueness and diversity of each person's handwriting affects the digits' structure and look.

2.2 REFERENCES

S.NO.	Journal Paper Title	Author's Name	Source	Finding
1.	Handwritten Digit Recognition using Machine Learning Algorithms	S M Shamim, Mohammad Badrul Alam Miah, Angona Sarker, Masud Rana & Abdullah Al Jobair.	Global Journal of Computer Science and Technology: D Neural & Artificial Intelligence	Its a pattern recognition applications. The heart of the problem lies within the ability to develop an efficient algorithm that can recognize hand written digits and which is submitted by users by the way of a scanner, tablet, and other digital devices. Several machines learning algorithm namely, Multilayer Perceptron, Support Vector Machine, Naïve Bayes, Bayes Net, Random Forest, J48 and Random.

2.	Recognition of Handwritten Digit using Convolutional Neural Network (CNN)	Md. Anwar Hossain & Md. Mohon Ali	Global Journal of Computer Science and Technology: D Neural & Artificial Intelligence	Computer vision works on enabling computers to see and process images in the same way that human vision does. The goal of the work is to create a model that will be able to identify and determine the handwritten digit from its image with better accuracy. The aim is to complete this by using the concepts of Convolutional Neural Network and MNIST dataset.
3.	Review on Deep Learning Handwritten Digit Recognition using Convolutional Neural Network	Akanksha Gupta, Ravindra Pratap Narwaria, Madhav Singh	International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878 (Online), Volume-9 Issue-5, January 2021	In this paper, it is discussed that classifiers like KNN, SVM, CNN are used for HDR. The scanned document is passed through four different stages for recognition where image is preprocessed, segmented and then recognized by classifier. MNIST dataset is used for training purpose. Complete CNN classifier is discussed in this paper. It is found that CNN is very accurate.
4.	A Comparison of Three Classification Algorithms for Handwritten Digit Recognition	Maiwan Bahjat Abdulrazzaq, Jwan Najeeb Saeed 2019	International Conference on Advanced Science and Engineering (ICOASE)	Recognizing the numeral handwriting of a person is a hard task because each individual has a unique handwriting way. This paper presents a comparison of three classification algorithms namely Naive Bayes (NB), Multilayer Perceptron (MLP) and K_Star algorithm based on correlation features selection (CFS) using NIST handwritten dataset. The results show that K_Star algorithm gives better recognition rate than NB and MLPas it reached the accuracy of 82.36%.

5.	Comparison of learning algorithms for handwritten digit recognition	Y. Lecun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Mūller, E. Sāckinger, P. Simard, and V. Vapnik bell laboratories, holmdel, NJ	Conference: international conference on artificial neural networks	The comparison of the relative merits of several classification algorithms developed at bell laboratories and elsewhere for the purpose of recognizing handwritten digits. It is an excellent benchmark for comparing shape recognition methods. Not only raw accuracy, but also rejection, training time, recognition time, and memory requirements are also considererd.
6.	Handwritten digit recognition by neural networks with single-layer training	S. Knerr; L. Personnaz; G. Dreyfus.	Conference: international conference on artificial neural networks.	The STEPNET procedure, which decomposes the problem into simpler subproblems which can be solved by linear separators, is introduced. Provided appropriate data representations and learning rules are used, performance comparable to that obtained by more complex networks can be achieved.

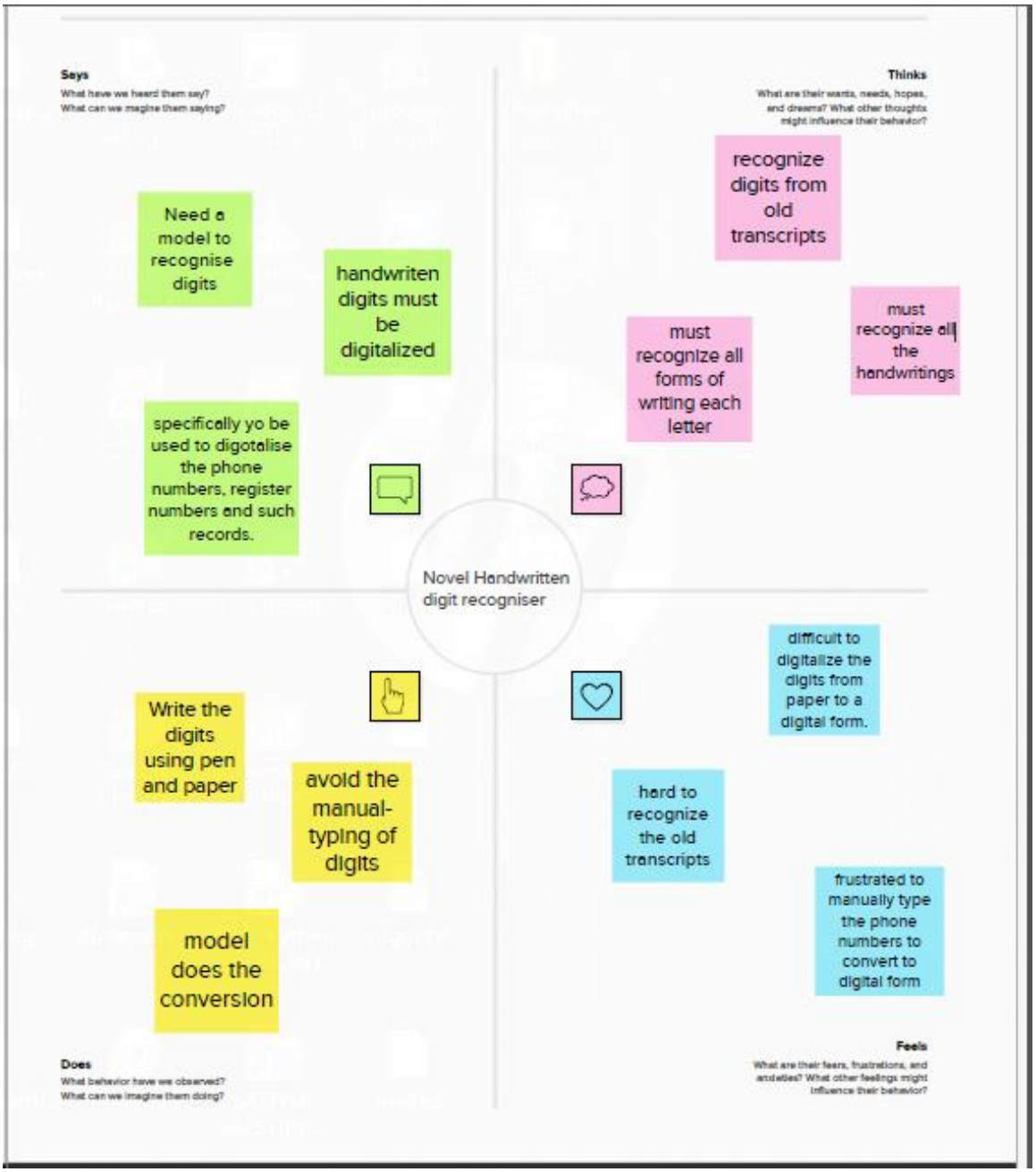
2.3 PROBLEM STATEMENT DEFINITION

The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes. The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image. Convolutional Neural Network model created using PyTorch library over the MNIST dataset to recognize handwritten digits .

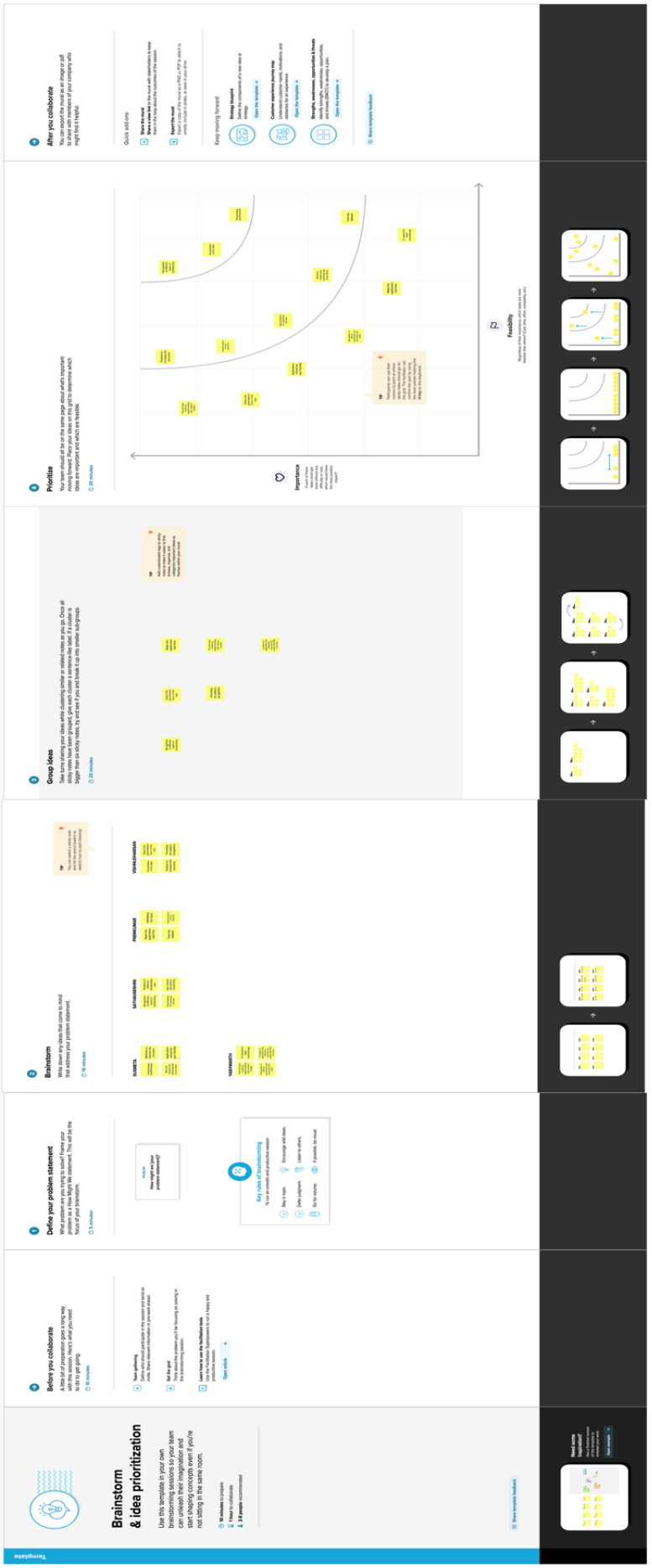
The task is to classify a given image of a handwritten digit into one of 10 classes representing integer values from 0 to 9, inclusively.It is a widely used and deeply understood dataset and, for the most part, is “solved.” Top-performing models are deep learning convolutional neural networks that achieve a classification accuracy of above 99%, with an error rate between 0.4 %and 0.2% on the hold out test dataset.

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP



3.2 IDEATION AND BRAINSTORMING



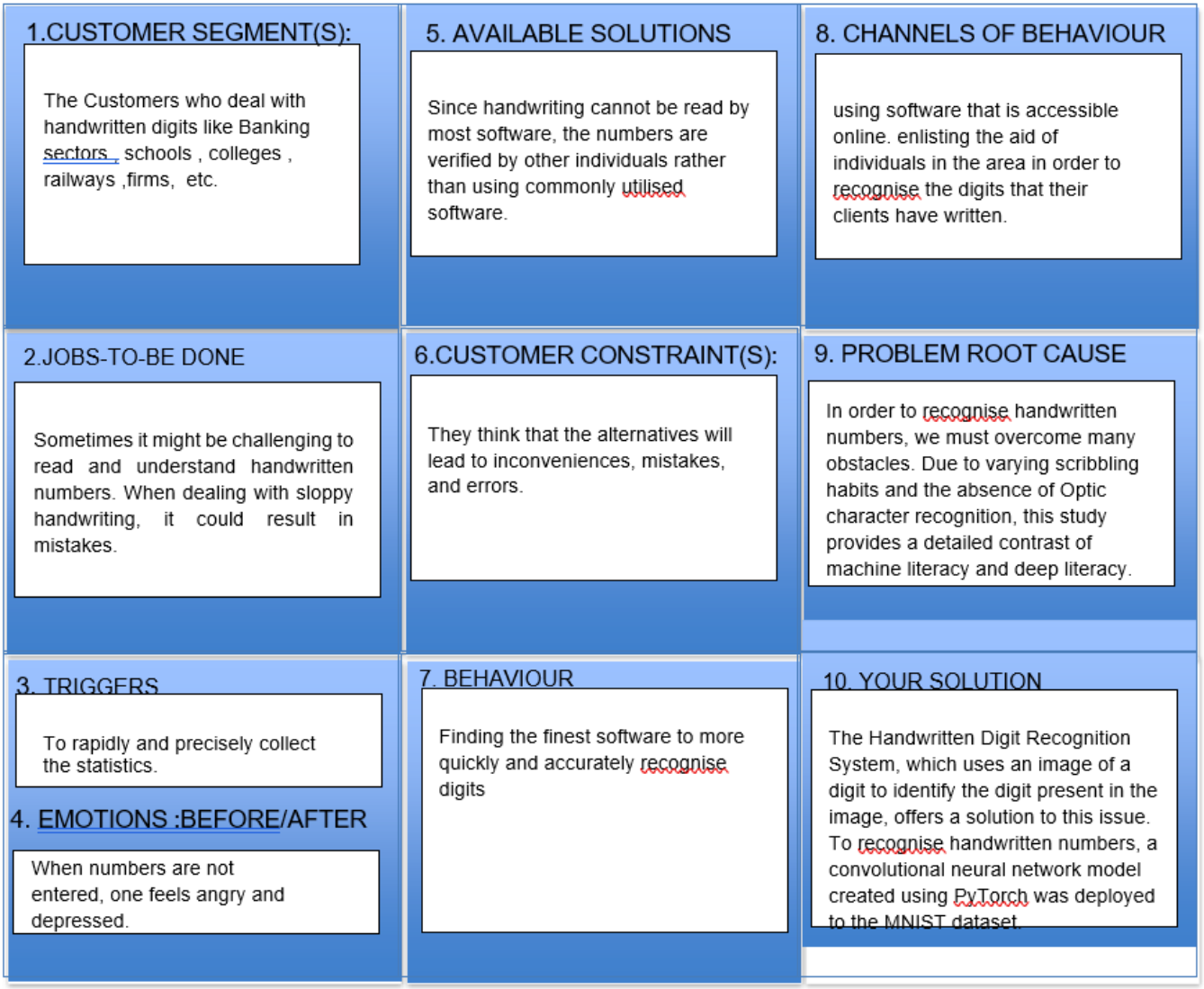
3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	ProblemStatement (Problemtobesolved)	Digit recognition is crucial in the contemporary environment. It can resolve more difficult issues while also simplifying human tasks. A handwritten digit is one instance. HANDWRITTEN Digit recognition refers to a computer system's capacity to identify handwritten inputs, such as numbers from a range of sources, including emails, papers, photographs, letters, etc. Humans can easily do a task accurately by frequently rehearsing it and recalling it for the next time. Images are easily processed and analysed by the human brain. Recognize the various components that each image has.
2.	Idea/Solutiondescription	The MNIST dataset, which includes 10,000 test images and 60,000 training images of handwritten digits from zero to nine, is used to perform handwritten digit recognition. Thus, there are 10 different classes in the MNIST dataset. In this project, we'll put into practice a Convolutional Neural Networks model–trained application for handwritten digit recognition. In the end, a GUI is created in which the user enters a handwritten digit, which is then identified, and the answer is shown right away.
3.	Novelty/Uniqueness	In this project, a practical method for addressing novelty in the field of handwriting visual recognition is introduced. A flawless transcription agent would be able to recogniserecognised and unrecognised characters in a picture as well as any aesthetic differences that might exist within or across texts.

		Even the most reliable machine learning-based algorithms for these activities have demonstrated to be severely hindered by the presence of novelty. Novelty in handwritten documents can take many different forms, such as a change in the author, character traits, writing skills, or overall document appearance. We think that an integrated agent that can handle well-known characters and innovations simultaneously is a better approach than looking at each aspect separately.
4.	SocialImpact/CustomerSatisfaction	The handwriting recognition system offers a wide range of advantages. It is helpful for reading forms in addition to reading postal addresses and bank check amounts. Additionally, it is employed in the detection of fraud since it makes it simple to compare two texts and identify which is a copy. Because it employs an innovative technique for identifying handwritten digits, this system ensures high accuracy for the model and meets all customer expectations. Users will save a lot of time and effort if the system provides various synonyms for the words recognized. Due to the fact that the users in rural areas will be using their own regional language, this proposed system should be able to detect those digits as well. As the system is being used in socially crowded places such as banks to check amounts, it should be fast and reliable. As it is designed to solve real-world problems, it should be highly reliable and trustworthy in every way, and users throughout the world should be able to use it effectively
5.	BusinessModel(RevenueModel)	A revenue model means understanding how astartupcanmakemoney.Ourmajorrevenueso urcesconsist of ***sales, government funds, and publicdonations.*** The introduction of novel ideas increasesrevenue streams, such as introducing gesture ortouchfeatures,voicereadoutofrecogniseddi gits, etc..

6.	ScalabilityoftheSolution	Making use of cloud-native techniques is one way to scale the handwritten digit recognition system. IBM Cloud, for instance, is one of the cloud-based AI scalability options. Run and manage AI models, as well as optimise decisions at scale across any cloud, with the aid of IBM Cloud Build. The benefit of using the cloud to scale solutions is that we can install our AI programme there.

3.4 PROBLEM SOLUTION FIT



REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	**Website**	The code, graphics, and other components of a websiteare made available online through web hosting. Every website you've ever visited is on a server. The amount of space a website has on a server is determined by the type of hosting. The four primary types of hosting are shared, dedicated, VPS, and reseller.
FR-2	**User Registration**	Registration through Form Registration through Gmail Registration through LinkedIN
FR-3	**User Confirmation**	Confirmation via Email Confirmation via OTP
FR-4	**Digit Classifier Model**	Packages - tensorflow
FR-5	**MNIST Dataset**	MNIST is a handwritten digits dataset which can be used for training various image processing systems. It has 60,000 training and 10,000 testing examples.

NON-FUNCTIONAL REQUIREMENTS:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The recognition of handwritten characters is one of the major issues with pattern recognition applications. The processing of bank checks, filling out forms, and sorting mail are a few uses for digit recognition.
NFR-2	**Security**	The system produces a detailed description. parameters of the instantiation, which may elucidate details like the writing style, in in addition to the digit's classification. Each user should be able to sign in independently to the system using their unique username and password.

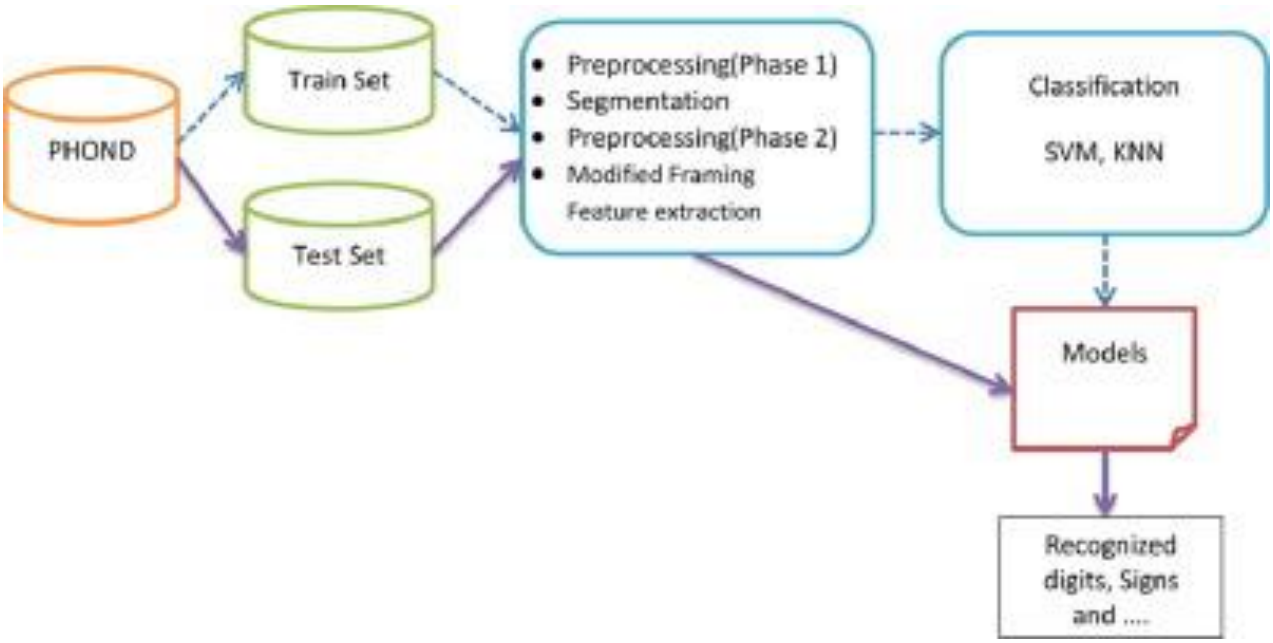
NFR-3	**Reliability**	The neural network uses the data to automatically determine rules for deciphering handwritten numerals. By increasing the number of training instances, the network may also learn more about handwriting and hence improve its accuracy. To recognise handwritten numbers, a variety of methods and algorithms can be employed, including Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests, etc.
NFR-4	**Performance**	The delay in providing the information when hundreds of requests are given should be minimum.
NFR-5	**Availability**	Access to information is restricted to each user.

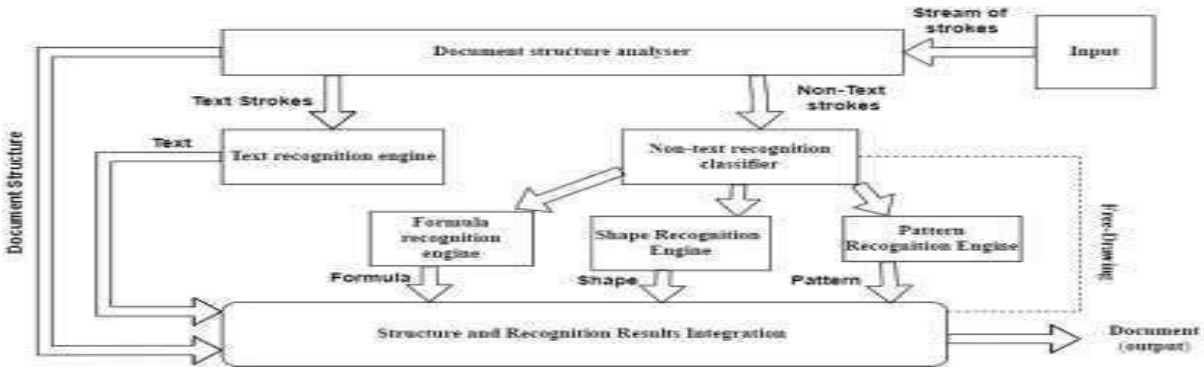
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

Data flow diagrams:

A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method .A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

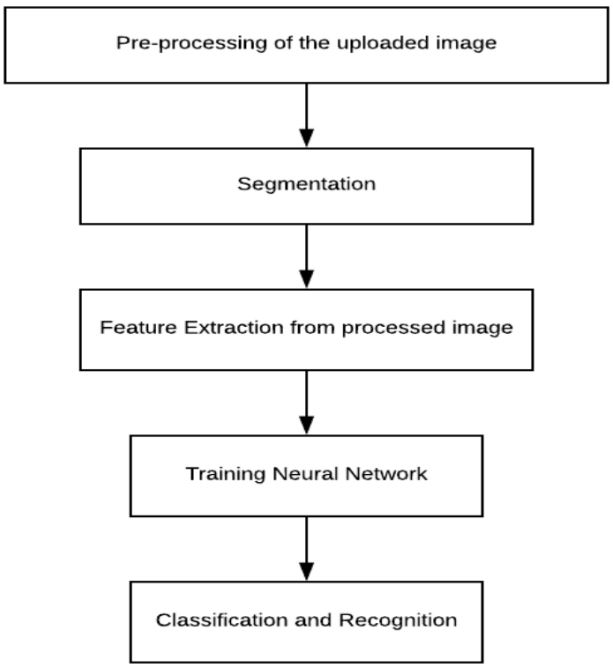


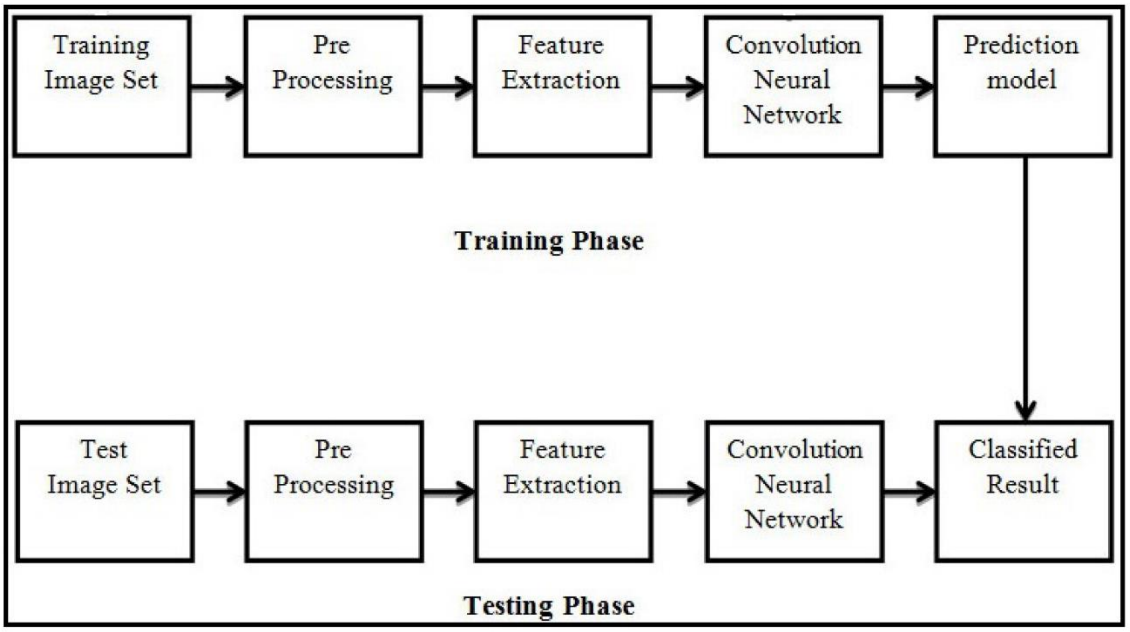


Example: DFDLevel0(Industry Standard):

5.2 SOLUTION AND TECHNICAL ARCHITECTURE

Block Diagram of the model proposed:




Training Image Set
Pre Processing
Feature Extraction
Convolution Neural Network
Prediction model
Training Phase
Test Image Set
Pre Processing
Feature Extraction
Convolution Neural Network
Classified Result
Testing Phase

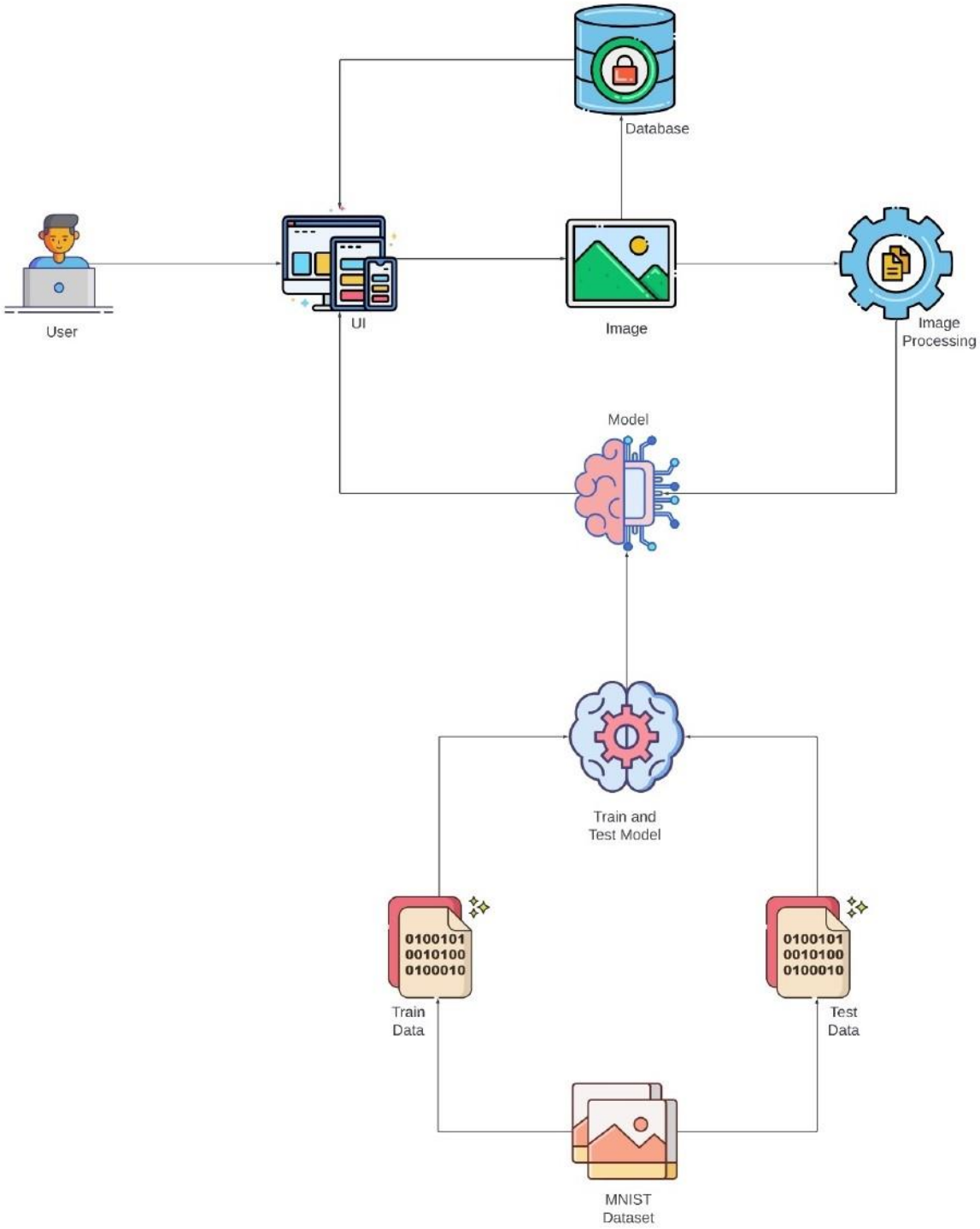

Database
User
UI
Image
Image
Processing
Model
Train and
Test Model
0100101
0010100
0100010
Train
Data
0100101
0010100
0100010
Test
Data
MNIST
Dataset

Table-1 : Components, technologies and description:

S.No	Component	Description	Technology
1.	User Interface	How the user interacts with applications e.g. Web UI	HTML, CSS, Flask
2.	Application Logic-1	Logic for the model that is being built	Python
3.	Application Logic-2	IBM model is trained using the Python code developed	IBM Watson service
4.	Deep Learning Model	Purpose of this model is to recognise the digits from the image uploaded by the user	CNN model
5.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration	IBM cloud

Table-2: Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Deep learning frameworks can help you upload data and train a deep learning model that would lead to accurate and intuitive predictive analysis.	Tensorflow, Visual code or pycharm
2.	Scalable Architecture	The system should be able to handle 10000 users accessing the site at the same time.	Python with IBM cloud model
3.	Availability	Information is restricted to each user's limited access.	IBM cloud
4.	Performance	Should reduce the delay in information when hundreds of requests are given. The system should be fast.	Python and flask

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-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register the application with Gmail	Medium	Sprint-2
	Login	USN-5	As a user, I can log into the application by entering email & password	I can login to the application	High	Sprint-1
	Home	USN-6	As a user, I can view the application's home page where I can read the instructions to use this application	I can read instructions also and the home page is user-friendly.	Low	Sprint-1
	Upload Image	USN-7	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

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection & pre processing	USN-1	As a user, I can upload any kind of image that has gone through the pre-processing step.	10	High	Sathya Varshini R D ,Vishnudarshan S
Sprint-1		USN-2	As a user, I can upload the image in any resolution.	10	High	Yashwant C M , Prem Kumar I M
Sprint-2	Building the Machine learning model	USN-3	As a user, I will receive a web application with a machine learning model that is highly accurate at recognizing digits written by hand.	5	Medium	Susmeta A, Prem Kumar I M
Sprint-2		USN-4	As a user, I can show the image of the handwritten number for them to recognize.	5	Medium	Sathya Varshini R D
Sprint-2		USN-5	As a user, I am able to obtain the ideal recognized digit.	10	High	Yashwant C M , Prem Kumar I M
Sprint-3	Building User Interface for Web Application	USN-6	As a user, I will use an upload button to add the image of the handwritten digits to the web page.	5	Medium	Susmeta A Vishnudarshan S

Sprint-3		USN-7	As a user, I can be aware of the web page's fundamental usage in detail.	3	Low	Susmeta A
Sprint-3		USN-8	As a user, in the web page, I may see the anticipated or recognised digits.	5	Medium	Yashwant C , Vishnudarshan S
Sprint-4	Train and deployment of model in IBM Cloud	USN-9	I can utilize the product, and access the web application from anywhere.	10	High	Vishnudarshan S

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	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

CODING AND SOLUTIONING

DEVELOPMENT OF MODEL

Import required library

```
[ ] import numpy as np
    import pandas as pd

    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
    from tensorflow.keras.models import load_model

    from PIL import Image, ImageOps
```

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 [==============================] - 1s 0us/step
```

Data Analysis

```
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,  30,  36,  94, 154, 170,
        253, 253, 253, 253, 253, 225, 172, 253, 242, 195,  64,   0,   0,
          0,   0,   0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  49, 238, 253, 253, 253, 253,
        253, 253, 253, 253, 251,  93,  82,  82,  56,  39,   0,   0,   0,
          0,   0,   0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  18, 219, 253, 253, 253, 253,
        253, 198, 182, 247, 241,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,  80, 156, 107, 253, 253,
        205,  11,   0,  43, 154,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,  14,   1, 154, 253,
         90,   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, 139, 253,
        190,   2,   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,  11, 190,
        253,  70,   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,  35,
        241, 225, 160, 108,   1,   0,   0,   0,   0,   0,   0,   0,   0,
```

Data Pre-Processing

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

```
[ ] 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)
```

```
[ ] Y_train[0]

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

Model Building

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

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

Model Building

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

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

Training the model

```
[ ] model.fit(X_train, Y_train, batch_size=32, epochs=5, validation_data=(X_test,Y_test))

    Epoch 1/5
    1875/1875 [==============================] - 195s 103ms/step - loss: 0.2911 - accuracy: 0.9459 - val_loss: 0.1052 - val_accuracy: 0.9673
    Epoch 2/5
    1875/1875 [==============================] - 193s 103ms/step - loss: 0.0777 - accuracy: 0.9761 - val_loss: 0.1166 - val_accuracy: 0.9696
    Epoch 3/5
    1875/1875 [==============================] - 193s 103ms/step - loss: 0.0495 - accuracy: 0.9846 - val_loss: 0.1010 - val_accuracy: 0.9711
    Epoch 4/5
    1875/1875 [==============================] - 192s 103ms/step - loss: 0.0373 - accuracy: 0.9882 - val_loss: 0.0997 - val_accuracy: 0.9758
    Epoch 5/5
    1875/1875 [==============================] - 192s 102ms/step - loss: 0.0297 - accuracy: 0.9908 - val_loss: 0.1043 - val_accuracy: 0.9763
    <keras.callbacks.History at 0x7fde8b335050>
```

Testing the model

```
[ ] metrics = model.evaluate(X_test, Y_test, verbose=0)
    print("Metrics (Test Loss & Test Accuracy): ")
    print(metrics)
```

```
Metrics (Test Loss & Test Accuracy):
[0.10433795303106308, 0.9763000011444092]
```

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

```
1/1 [==============================] - 0s 89ms/step
[[2.37035357e-11 4.02153565e-20 2.39775906e-15 1.33595852e-13
  4.97287439e-19 3.56922955e-16 9.66210090e-23 1.00000000e+00
  1.84621118e-09 8.92001588e-14]
 [1.69142582e-11 2.28400740e-15 1.00000000e+00 5.66886626e-15
  1.45082682e-18 1.42091100e-17 1.38232705e-08 9.43912082e-20
  1.45648115e-13 2.08561429e-19]
 [1.63044900e-09 9.99997020e-01 6.79160728e-09 2.41297824e-12
  2.91080778e-06 2.19643432e-08 1.91560967e-09 3.77489325e-11
  4.48562343e-09 3.18847444e-12]
 [1.00000000e+00 1.94177089e-16 4.16544577e-10 4.38118068e-14
  1.02700932e-12 4.44104649e-12 3.01250780e-09 3.23853561e-15
  6.20657914e-10 2.53168642e-09]]
```

```
[ ] print(np.argmax(prediction, axis=1))
    print(Y_test[:4])
```

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

Save model

```
[ ] model.save("model.h5")
```

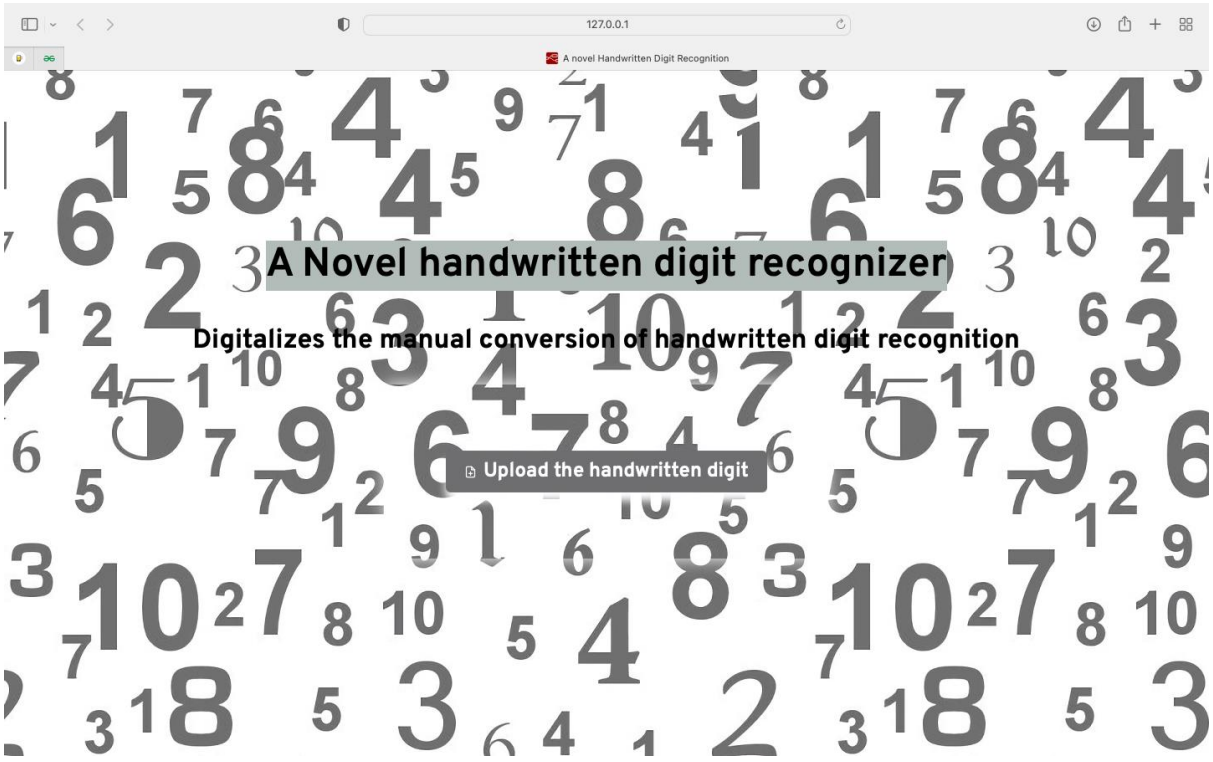
Testing

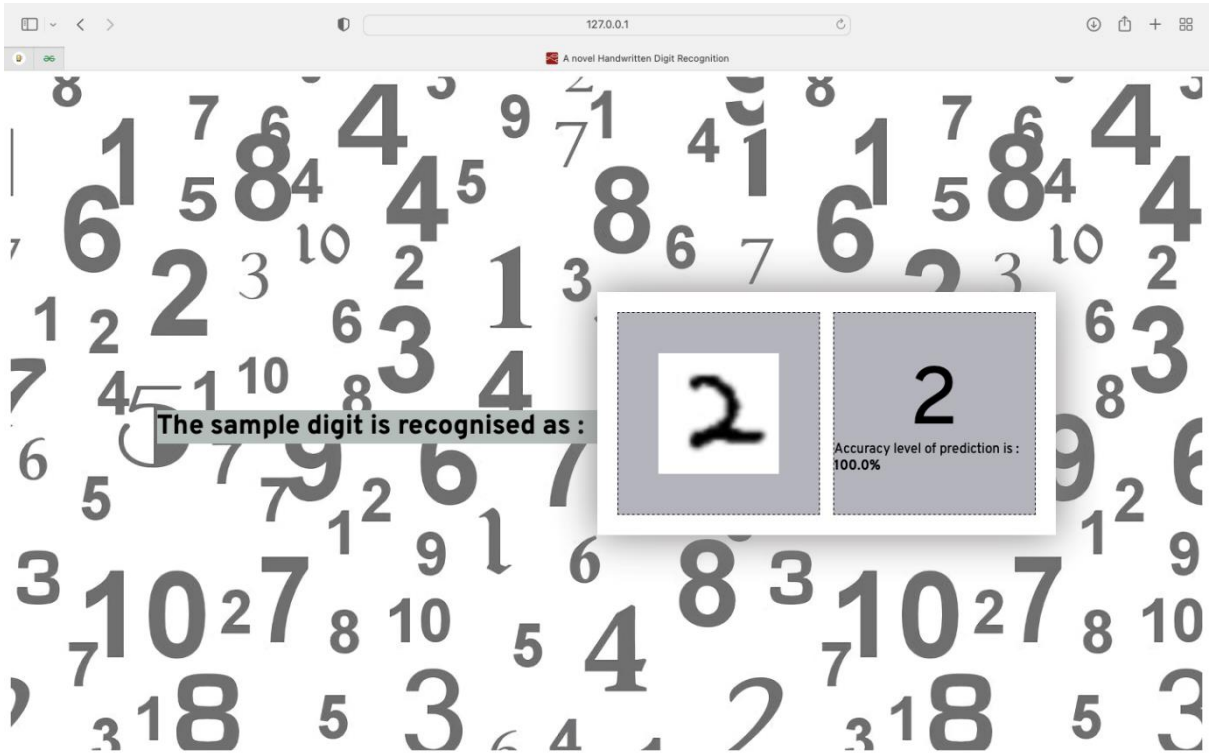
```
[ ] model=load_model("/content/model.h5")
```

```
[ ] img = Image.open("/content/three.PNG").convert("L")
    img = img.resize((28, 28))
    img_arr = np.array(img)
    img_arr = img_arr.reshape(1, 28, 28, 1)
    predict_value  = model.predict(img_arr)
    predict_value = np.argmax(predict_value,axis = 1)
    predict_value = pd.Series(predict_value,name="Label")
    print(predict_value)
```

```
1/1 [==============================] - 0s 20ms/step
0    3
Name: Label, dtype: int64
```

OUTPUT


127.0.0.1
A novel Handwritten Digit Recognition
A Novel handwritten digit recognizer
Digitalizes the manual conversion of handwritten digit recognition
Upload the handwritten digit


127.0.0.1
A novel Handwritten Digit Recognition
The sample digit is recognised as :
2
Accuracy level of prediction is :
100.0%

ADVANTAGES AND DISADVANTAGES

Any system of its own has both advantages and disadvantages. The advantages and disadvantages of this model are as follows.

ADVANTAGES

Using this system has many advantages. Some of them are:

- Ability to handle a lot of data
- Less manual labor
- Better accuracy than an average person
- Can handle arbitrary scalings, translations and a limited degree of image rotation.

DISADVANTAGES

The disadvantages of this model are:

- All data must be digital
- For quicker predictions, a server with high performance is required.
- Prone to occasional errors
- Unable to handle complex data
- It is not done in real time as a person writes and therefore not appropriate for immediate text input
- Requires much more computation than more standard OCR techniques

CONCLUSION

This project showcased a web application that uses machine learning to recognise handwritten numerals. Flask, HTML, CSS, and a few other technologies were used to construct this project. The model forecasts the handwritten digit using a CNN network. The suggested project is scalable and reliable.

There are many real world applications for this project, which includes reading bank cheques for amounts, processing license plate recognition data, and manually inputting data on forms like tax returns. There is a great deal of room for development that can be included into later iterations.

Accuracy can alter as it depends on the splitting of training and testing data, and this can further be improved if the number of training and testing data is provided. There is always a chance to improve accuracy if the size of data increases. Every classifier has its own accuracy and time consumption. We can also include the fact that if the power of CPU changes to GPU, the classifier can perform with better accuracy and less time and better results can be observed.

The performance of the classifier can be measured in terms of ability to identify a condition properly (sensitivity), the proportion of true results (accuracy), number of positive results from the procedure of classification as false positives (positive predictions) and ability to exclude condition correctly (specificity).

FUTURE SCOPE

There is a tonne of room for improvement in this project. Some ways this project could be enhanced include the following:

1. Provide capabilities for saving the results of repeated digit image detection.

2. Include support to identify different digits.

3. Improve the model so it can spot numbers in a range of photos.

4. Adding support for additional languages will help users everywhere.

This project has limitless potential and is always up for improvement. This strategy's implementation will help many people and various sectors because it will lighten workloads and increase overall job effectiveness. The potential for this project is endless, and it can always be improved. Implementing this strategy will benefit numerous industries, as well as many people, as it will reduce workloads and improve overall job efficiency.

It is impossible to foresee the future of handwriting recognition, but one unlikely scenario predicts that as keyboard use rises, handwriting abilities are beginning to deteriorate. However, this trend will most likely develop gradually, and handwriting will probably continue to hold some significance for the foreseeable future.

APPENDIX

SOURCE CODE - MODEL

Importing necessary packages

```
[ ] import numpy as np
    import pandas as pd
    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
    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()

    Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
    11490434/11490434 [==============================] - 2s 0us/step
```

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,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,
         18,  18,  18, 126, 136, 175,  26, 166, 255, 247, 127,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,  30,  36,  94, 154, 170,
        253, 253, 253, 253, 253, 225, 172, 253, 242, 195,  64,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  49, 238, 253, 253, 253, 253,
        253, 253, 253, 253, 251,  93,  82,  82,  56,  39,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  18, 219, 253, 253, 253, 253,
        253, 198, 182, 247, 241,   0,   0,   0,   0,   0,   0,   0,   0,
```

Data Pre-processing

```
[ ] x_train = x_train.reshape(60000, 28, 28, 1).astype('float32')
    x_test = x_test.reshape(10000, 28, 28, 1).astype('float32')
```

```
[ ] 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)
```

```
y_train[0]
```

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

Creating the model

```
[ ] model = Sequential()
    model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation="relu"))
    model.add(Conv2D(32, (3, 3), activation="relu"))
    model.add(Conv2D(16, (3, 3), activation="relu"))
    model.add(Flatten())
    model.add(Dense(number_of_classes, activation="softmax"))
```

Training the model

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

```
model.fit(x_train, y_train, batch_size=32, epochs=5, validation_data=(x_test,y_test))
```

```
Epoch 1/5
1875/1875 [==============================] - 19s 4ms/step - loss: 0.1559 - accuracy: 0.9560 - val_loss: 0.0648 - val_accuracy: 0.9784
Epoch 2/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0591 - accuracy: 0.9815 - val_loss: 0.0625 - val_accuracy: 0.9811
Epoch 3/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0430 - accuracy: 0.9865 - val_loss: 0.0718 - val_accuracy: 0.9791
Epoch 4/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0325 - accuracy: 0.9893 - val_loss: 0.0692 - val_accuracy: 0.9813
Epoch 5/5
1875/1875 [==============================] - 14s 8ms/step - loss: 0.0255 - accuracy: 0.9918 - val_loss: 0.0776 - val_accuracy: 0.9802
<keras.callbacks.History at 0x7f3130360290>
```

Saving the model

```
[ ] model.save("model.h5")
```

```
[ ] # Saving in tar

    !tar -zcvf Handwritten_Digit_Recognition.tgz model.h5
```

```
model.h5
```

IBM Deployment

```
[ ] !pip install watson-machine-learning-client
```

```
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (2.23.0)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (1.24.3)
Requirement already satisfied: tabulate in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (0.8.10)
Collecting lomond
  Downloading lomond-0.3.3-py2.py3-none-any.whl (35 kB)
Collecting ibm-cos-sdk
  Downloading ibm-cos-sdk-2.12.0.tar.gz (55 kB)
     |████████████████████████████████| 55 kB 4.0 MB/s
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (4.64.1)
Collecting boto3
  Downloading boto3-1.26.12-py3-none-any.whl (132 kB)
     |████████████████████████████████| 132 kB 67.4 MB/s
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (2022.9.24)
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages (from watson-machine-learning-client) (1.3.5)
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
     |████████████████████████████████| 79 kB 10.2 MB/s
Collecting botocore<1.30.0,>=1.29.12
  Downloading botocore-1.29.12-py3-none-any.whl (9.9 MB)
     |████████████████████████████████| 9.9 MB 69.1 MB/s
Collecting urllib3
  Downloading urllib3-1.26.12-py2.py3-none-any.whl (140 kB)
     |████████████████████████████████| 140 kB 69.5 MB/s
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/local/lib/python3.7/dist-packages (from botocore<1.30.0,>=1.29.12->boto3->watson-machine
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.30.0,>=1.29.12->boto3->wats
Collecting ibm-cos-sdk-core==2.12.0
  Downloading ibm-cos-sdk-core-2.12.0.tar.gz (956 kB)
     |████████████████████████████████| 956 kB 61.1 MB/s
Collecting ibm-cos-sdk-s3transfer==2.12.0
  Downloading ibm-cos-sdk-s3transfer-2.12.0.tar.gz (135 kB)
```

```
!pip install ibm_watson_machine_learning
```

```
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (21.3)
Collecting ibm-cos-sdk==2.7.*
  Downloading ibm-cos-sdk-2.7.0.tar.gz (51 kB)
     |████████████████████████████████| 51 kB 748 kB/s
Requirement already satisfied: tabulate in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (0.8.10)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (1.3.5)
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (2022.9.24)
Requirement already satisfied: lomond in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (4.13.0)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (from ibm_watson_machine_learning) (1.26.12)
Collecting ibm-cos-sdk-core==2.7.0
  Downloading ibm-cos-sdk-core-2.7.0.tar.gz (824 kB)
     |████████████████████████████████| 824 kB 55.9 MB/s
Collecting ibm-cos-sdk-s3transfer==2.7.0
  Downloading ibm-cos-sdk-s3transfer-2.7.0.tar.gz (133 kB)
     |████████████████████████████████| 133 kB 69.1 MB/s
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /usr/local/lib/python3.7/dist-packages (from ibm-cos-sdk==2.7.*->ibm_watson_machine_learning) (0.1
Collecting docutils<0.16,>=0.10
  Downloading docutils-0.15.2-py3-none-any.whl (547 kB)
     |████████████████████████████████| 547 kB 66.3 MB/s
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/local/lib/python3.7/dist-packages (from ibm-cos-sdk-core==2.7.0->ibm-cos-sdk==2.7.*->ibm
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.7/dist-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (1.21.6)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (2022.6)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil<3.0.0,>=2.1->ibm-cos-sdk-core==2.7.0->ibm-cos-sdk==
Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.7/dist-packages (from requests->ibm_watson_machine_learning) (2.1.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->ibm_watson_machine_learning) (2.10)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata->ibm_watson_machine_learning) (3.10.0)
Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata->ibm_watson_machine_learning) (4
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging->ibm_watson_machine_learning) (3.0.9)
Building wheels for collected packages: ibm-cos-sdk, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer
  Building wheel for ibm-cos-sdk (setup.py) ... done
  Created wheel for ibm-cos-sdk: filename=ibm_cos_sdk-2.7.0-py2.py3-none-any.whl size=72563 sha256=e1bbaf75bdaed62c3037dce6608a560822f94bc46800be36a1ae4675
  Stored in directory: /root/.cache/pip/wheels/47/22/bf/e1154ff0f5de93cc477acd0ca69abfbb8b799c5b28a66b44c2
  Building wheel for ibm-cos-sdk-core (setup.py) ... done
  Created wheel for ibm-cos-sdk-core: filename=ibm_cos_sdk_core-2.7.0-py2.py3-none-any.whl size=501013 sha256=befd1c288c9b63d576415c080bb2f9357adb696e02556
```

```
[ ] from ibm_watson_machine_learning import APIClient

    wml_credentials = {
        "url": "https://eu-gb.ml.cloud.ibm.com",
        "apikey": "lYkfZuIGZU4tlgcOmkgFHANeC5hCY0H565dIFrDefvPh"
    }

    client = APIClient(wml_credentials)
    client
```

```
Python 3.7 and 3.8 frameworks are deprecated and will be removed in a future release. Use Python 3.9 framework instead.
<ibm_watson_machine_learning.client.APIClient at 0x7f30332f1a10>
```

```
client.spaces.get_details()
```

```
{'resources': [{'entity': {'compute': [{'crn': 'crn:v1:bluemix:public:pm-20:eu-gb:a/0099690998344e7b8208cc1fb401bd5f:c31e6fd7-61ee-4e85-9bdc-
44700d3ddc4f::',
      'guid': 'c31e6fd7-61ee-4e85-9bdc-44700d3ddc4f',
      'name': 'Watson Machine Learning-3l',
      'type': 'machine_learning'}],
    'description': '',
    'name': 'Handwritten Digit Recognition',
    'scope': {'bss_account_id': '0099690998344e7b8208cc1fb401bd5f'},
    'stage': {'production': False},
    'status': {'state': 'active'},
    'storage': {'properties': {'bucket_name': '7cbf8c51-e5a5-4794-bad7-ae1b5a3c9efd',
      'bucket_region': 'eu-gb-standard',
      'credentials': {'admin': {'access_key_id': '326d32df820e491d9e94e19150a6cf5c',
        'api_key': 'AEZQRPmnnShF4qfqgfVMPXirM1iTWUWRpWccOtN32sPu',
        'secret_access_key': '9590302866ae28e2bbaf3e191c377f0ea14df6853b5e7018',
        'service_id': 'ServiceId-3400c39e-0533-4562-a39b-522555656950'},
       'editor': {'access_key_id': '4e0e9d18dbb2478f9f318b397e4d6ac1',
        'api_key': 'qzSzAiodIUcCLYzdBygCayw81JkZwbzyDkzmKEeQfKkD',
        'resource_key_crn': 'crn:v1:bluemix:public:cloud-object-storage:global:a/0099690998344e7b8208cc1fb401bd5f:4e03828b-095a-4263-80f7-e9ef1b3ef5ee::',
        'secret_access_key': '96835df666183ade820cd88a7cdd26edccbc20510df038b2',
        'service_id': 'ServiceId-13b3483a-a407-49c7-a139-6c0534d0767f'},
       'viewer': {'access_key_id': '73f8fe1163fe4d69b2a5edb562994f56',
```

```
[ ] def guid_space_name(client, Handwritten_Digit_Recognition):
      space = client.spaces.get_details()
      return(next(item for item in space['resources'] if item['entity']['name']==Handwritten_Digit_Recognition)['metadata']['id'])

    space_uid = guid_space_name(client, 'Handwritten Digit Recognition')
    space_uid
```

```
'b6e4d602-5eb0-43ce-ada0-3f0dcd95cb43'
```

```
[ ] client.set.default_space(space_uid)
```

```
'SUCCESS'
```

```
client.software_specifications.list()
```

```
------------------------------  ------------------------------------  ----
NAME                            ASSET_ID                              TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12       020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9       0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6               10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl       111e41b3-de2d-5422-a4d6-bf776828c4b7  base
autoai-kb_rt22.2-py3.10         125b6d9a-5b1f-5e8d-972a-b251688ccf40  base
runtime-22.1-py3.9              12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6         154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                    1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6          1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
kernel-spark3.3-r3.6            1c9e5454-f216-59dd-a20e-474a5cdf5988  base
pytorch-onnx_rt22.1-py3.9-edt   1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
```

```
[ ] software_space_uid = client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9')
    software_space_uid
```

```
'acd9c798-6974-5d2f-a657-ce06e986df4d'
```

```
[ ] model_details = client.repository.store_model(model='Handwritten_Digit_Recognition.tgz', meta_props={
        client.repository.ModelMetaNames.NAME:'Handwritten Digit Recognition',
        client.repository.ModelMetaNames.TYPE: "tensorflow_2.7",
        client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_space_uid
    })
```

```
model_details
```

```
{'entity': {'hybrid_pipeline_software_specs': [],
  'software_spec': {'id': 'acd9c798-6974-5d2f-a657-ce06e986df4d',
   'name': 'tensorflow_rt22.1-py3.9'},
  'type': 'tensorflow_2.7'},
 'metadata': {'created_at': '2022-11-18T00:41:59.387Z',
  'id': 'b5a61208-a7dc-4cce-b8e9-acec041f4561',
  'modified_at': '2022-11-18T00:42:02.666Z',
  'name': 'Handwritten Digit Recognition',
  'owner': 'IBMid-667000FO18',
  'resource_key': '10345caa-76bd-49c8-a928-41655db97de8',
  'space_id': 'b6e4d602-5eb0-43ce-ada0-3f0dcd95cb43'},
 'system': {'warnings': []}}
```

```
model_id = client.repository.get_model_id(model_details)
model_id
```

```
'b5a61208-a7dc-4cce-b8e9-acec041f4561'
```

+ Code + Text

```
[ ] client.repository.download(model_id, 'IBMHandwrittenDigitRecognition.tar.gb')
```

```
Successfully saved model content to file: 'IBMHandwrittenDigitRecognition.tar.gb'
'/content/IBMHandwrittenDigitRecognition.tar.gb'
```

UI BUILDING

FLASK APPLICATION

app.py

```python
from flask import Flask,render_template,request
from recognizer import recognize

app=Flask(__name__)

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


@app.route('/predict',methods=['POST'])
def predict():
    if request.method=='POST':
      image = request.files.get('photo', '')
      best, others, img_name = recognize(image)
      return render_template("predict.html", best=best, others=others, img_name=img_name)


if __name__=="__main__":
    app.run()
```

recognizer.py

```python
def recognize(image):
  model=load_model(Path("./model/mnistCNN.h5"))

  img = Image.open(image).convert("L")
  img_name = random_name_generator(10) + '.jpg'

  if not os.path.exists(f"./static/data/"):
    os.mkdir(os.path.join('./static/', 'data'))
  img.save(Path(f"./static/data/{img_name}"))

  img = ImageOps.grayscale(img)
  img = ImageOps.invert(img)
  img = img.resize((28, 28))

  img2arr = np.array(img)
  img2arr = img2arr / 255.0
  img2arr = img2arr.reshape(1, 28, 28, 1)

  results  = model.predict(img2arr)
  best = np.argmax(results,axis = 1)[0]
```

TEMPLATES - HTML FILES

home.html

templates > home.html > html

```
<html>
    <head>
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <title>A novel Handwritten Digit Recognition</title>
        <link rel="icon" type="image/svg" sizes="32x32" href="{{url_for('static',
        filename='images/icon2.jpg')}}" />
        <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>
        <style>
            mark {
              background-color: #b4bcba;
              color: black;
            }
            </style>

    </head>

    <body>
        <div class="container">
            <div class="heading">
                <h1 class="heading__main"><mark>A Novel handwritten digit recognizer</mark></h1>
                <br><br>
                <h2 class="heading__sub"> <b>Digitalizes the manual conversion of handwritten
                digit recognition </b>    </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"></
```

home.html

templates > home.html > html

```
            </style>

    </head>

    <body>
        <div class="container">
            <div class="heading">
                <h1 class="heading__main"><mark>A Novel handwritten digit recognizer</mark></h1>
                <br><br>
                <h2 class="heading__sub"> <b>Digitalizes the manual conversion of handwritten
                digit recognition </b>    </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>Upload the handwritten digit</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>
```

predict.html

templates > predict.html > html > body > div.container > div.result-wrapper

```
<html>
  <head>
    <title>A novel 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/
    icon2.jpg')}}" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <style>
            mark {
              background-color: #b4bcba;
              color: black;
            }
            </style>

  </head>
  <body>
    <div class="container">
      <h1><mark>The sample digit is recognised as :  </mark></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">Accuracy level of prediction is : <b>{{best.1}}% </b></div>
        </div>
      </div>
    </div>
  </body>
</html>
```

STATIC - CSS FILES

predict.html main.css

static > css > main.css > ...

```css
@import url("https://fonts.googleapis.com/css2?family=Overpass:wght@200;300;400;500;600;700;900&display=swap");

* {
  padding: 0;
  margin: 0;
}

body {

  color: #384b8f;
  font-family: "Overpass", sans-serif;
}

.container {
  background-image: url('../images/bg_image.png');


  width: 100%;
  height: 100%;
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  background-color: #d0d9f7;

}
```

predict.html predict.css

static > css > predict.css > ...

```css
.result-wrapper .input-image-container,
.result-wrapper .result-container {
  width: 15rem;
  height: 15rem;
  border: 1px dashed black;
  justify-content: center;
  display: flex;
  align-items: center;
  flex-direction: column;
  background-color: #b4b4bc;
}

.result-wrapper .input-image-container img {
  width: 60%;
  height: 60%;
  background-color: aqua;
  background-size: contain;

}

.result-wrapper .result-container .value {
  font-size: 6rem;
}

.result-wrapper .result-container .accuracy {
  margin-top: -1rem;
}
```

GITHUB LINK

DEMO LINK