

**Project Report**

**A NOVEL METHOD FOR HANDWRITTEN DIGIT  
RECOGNITION**

Submitted by

**PNT2022TMID28883**

Hemapriya K (411719104017)

Ashwini P (411719104006)

Charulatha I (411719104009)

Gokila B (411719104014)

## TABLE OF CONTENTS

## PAGE NO

<b>1. INTRODUCTION</b>	
1.1 Project Overview	1
1.2 Purpose	1
<b>2. LITERATURE SURVEY</b>	
2.1 Existing problem	2
2.2 References	2
2.3 Problem Statement Definition	3
<b>3. IDEATION &amp; PROPOSED SOLUTION</b>	
3.1 Empathy Map Canvas	4
3.2 Ideation & Brainstorming	5
3.3 Proposed Solution	5
3.4 Problem Solution fit	7
<b>4. REQUIREMENT ANALYSIS</b>	
4.1 Functional requirement	8
4.2 Non-Functional requirements	8
<b>5. PROJECT DESIGN</b>	
5.1 Data Flow Diagrams	10
5.2 Solution & Technical Architecture	10
<b>6. PROJECT PLANNING &amp; SCHEDULING</b>	
6.1 Sprint Planning & Estimation	14
6.2 Sprint Delivery Schedule	15
<b>7. CODING &amp; SOLUTIONING</b>	
7.1 Feature 1	18
7.2 Feature 2	19
<b>8. TESTING</b>	
8.1 Test Cases	21
8.2 User Acceptance Testing	21
<b>9. RESULTS</b>	
9.1 Performance Metrics	22
<b>10. ADVANTAGES &amp; DISADVANTAGES</b>	23
<b>11. CONCLUSION</b>	24
<b>12. FUTURE SCOPE</b>	25

## **13. APPENDIX**

Source Code	26
GitHub	36
Project Demo Link	36

## **CHAPTER 1**

### **INTRODUCTION**

#### **1.1.PROJECT OVERVIEW**

Machine Learning provides various methods through which human efforts can be reduced in recognizing the manually written digits. Deep Learning is a machine learning method that trains computers to do what easily falls into place for people: learning through examples. With the utilization of deep learning methods, human attempts can be diminished in perceiving, learning, recognizing and in a lot more regions.

Handwritten Digit Recognition is the ability of computer systems to recognize handwritten digits from various sources, such as images, documents, and so on. This project aims to let users take advantage of machine learning to reduce manual tasks in recognizing digits.

#### **1.2.PURPOSE**

Handwritten digit recognition systems are capable of recognizing the digits from different sources like emails, bank cheque, papers, images, etc. and in different real-world scenarios for online handwriting recognition on computer tablets or system, recognize number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (tax forms) and so on.

## **CHAPTER 2**

### **LITERATURE SURVEY**

#### **2.1 EXISTING PROBLEM**

The fundamental problem with handwritten digit recognition is that handwritten digits do not always have the same size, shape, width, orientation, and margins since they vary from person to person. In addition there would be issues with identifying the numbers because of similarities between numerals like 1 and 7, 5 and 6, 3 and 8, 2 and 5, 2 and 7. Finally depending individual's handwriting the structure and appearance of the digits vary from one person to another.

#### **2.2.REFERENCES**

**[1] Ishani Patel, Virag Jagtap ,Ompriya Kale ,“A Survey on Feature Extraction Methods for Handwritten Digits Recognition”, IJCA(0975 – 8887), Volume 107 – No 12, Dec (2015).**

The proposed neural system was trained and tested on a dataset achieved from MNIST. Their proposed method utilized the image pixels for its feature extraction process. ANN carried out the classification, and the overall classification accuracy is 99.60 percentage. The recognition system is broadly divided into 2 parts, first part is feature extraction from handwritten images and the second one is classification of feature vector into digits. We propose descriptors for handwritten digit recognition based on Histogram of Oriented Gradient (HOG) feature .It is one of the widely used feature vector for object detection in computer vision. For classification of features, linear Proximal Support Vector Machine Classifier is proposed. This is a binary class classifier which is further converted to a 10 class classifier by means of One against all algorithm. Due to small training time, PSVM classifier is preferable over standard Support Vector Machine (SVM) Classifier. The handwritten images both for training and testing are taken from MNIST database. The performance of the system is measured in terms of Sensitivity, Accuracy, Positive Predictively and Specificity.

**[2] K. T. Islam, G. Mujtaba, R. G. Raj and H. F. Nweke, "Handwritten digits recognition with artificial neural network," 2017 International Conference on Engineering Technology and Technopreneurship (ICE2T), Kuala Lumpur, 2017**

In research done on handwritten digits Recognition the model was implemented with an ANN which can identify handwritten digits from 0 to 9. The proposed neural system was trained and tested on a dataset achieved from MNIST. Their proposed method utilized the image pixels for its feature extraction process. ANN carried out the classification, and the overall classification accuracy is 99.60 percentage

**[3] Fathma Siddique, Shadman Sakib, M.A.B,” Recognition of Handwritten Digit using Convolutional Neural Network in Python with Tensorflow and Comparison of Performance for Various Hidden Layers”. Preprints, Issue 2019.**

The handwritten digits recognition model with CNN implemented using different numbers of hidden layers and epochs found that we can reach ideal accuracy with respect to the number of epochs and hidden layers. It is difficult to get a good performance as more parameters are needed for the large-scale neural network. In research, it is discovered that deep nets perform better when they are prepared by basic backpropagation. Their architecture brings about the most minimal error rate on MNIST contrast with NORB and CIFAR10.

## **2.3 PROBLEM STATEMENT DEFINITION**

Handwritten digit recognition tends to have problems when it comes to accuracy. People can struggle to read others' handwriting. The issue is that there's a wide range of handwriting – good and bad. This makes it tricky for programmers to provide enough examples of how every character might look.

In postal system there is a difficulty in recognizing handwritten digit pin codes by the user and also in traffic surveillance cameras face difficulties in recognizing vehicle number plates.

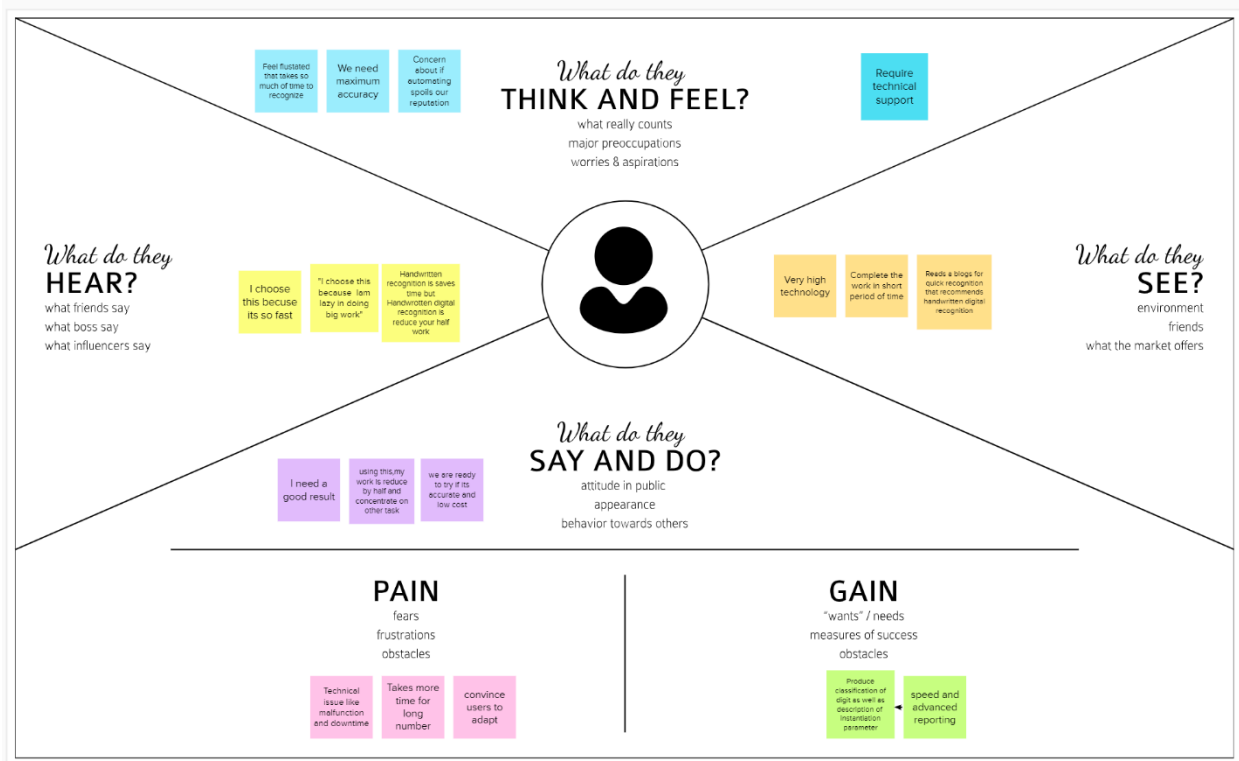
The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image. Convolutional Neural Network model created using Python library over the MNIST dataset to recognize handwritten digits.

The goal is to take an image of a handwritten digit and determine what that digit is.

## CHAPTER 3

### IDEATION AND PROPOSED SOLUTION

#### 3.1 EMPATHY MAP CANVAS



## 3.2 IDEATION AND BRAINSTORMING

**1 Define your problem statement**  
Handwritten digits need to be recognized by the system. The system must identify the digits and tell the report correctly.  
⌚ 5 minutes

**2 Brainstorm**  
Write down any ideas that come to mind that address your problem statement.  
⌚ 10 minutes

Technique 1	Technique 2	Technique 3	Technique 4
Technique 1	Technique 2	Technique 3	Technique 4
Technique 1	Technique 2	Technique 3	Technique 4
Technique 1	Technique 2	Technique 3	Technique 4
Technique 1	Technique 2	Technique 3	Technique 4

**3 Group ideas**  
Take turns sharing your ideas while clustering similar or related notes as you go. Once all ideas have been grouped, give each cluster a central title note. If a cluster is bigger than an sticky note, try to break it up into smaller subgroups.  
⌚ 10 minutes

**4 Prioritize**  
Your team should sit on the same page about what's important. Moving forward, share your ideas on the grid to determine which ideas are important and which are feasible.  
⌚ 10 minutes

**Feasibility**  
How easy is it to build the system?  
How much time and money does it take to build the system?

**Importance**  
How important is the system?  
How much time and money does it take to build the system?

## 3.3 PROPOSED SOLUTION

S.NO	PARAMETER	DESCRIPTION
1.	Problem Statement	The problem statement aims at developing a novel handwritten recognition system using ML .The handwritten digit recognition system is a way to tackle the problem which uses the image of a digit and recognizes the digit present in the image .
2.	Idea / Solution description	Developing an AI predictive model to predict



		the handwritten digits and to construct a neural network with hidden layers and train to detect the digits.
3.	Novelty / Uniqueness	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.
4.	Social Impact / Customer Satisfaction	Handwritten digits can be recognized easily without any strenuous efforts. This reduces time and improves productivity for people.
5.	Business Model	It is used in the detection of vehicle numbers, banks for reading cheques, post offices for arranging letters, and many other tasks.
6..	Scalability of the Solution	To attain higher performances in the domain of character recognition and pattern recognition

		<p>,due to its excellent feature extraction and working as best classifier characteristics.</p> <p>There is no limit in the number of digits that can be recognized.</p>
--	--	--

### 3.4 PROBLEM SOLUTION FIT

Problem-Solution fit canvas 2.0

Purpose / Vision

Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> <i>One who wants to extract digits from handwritten text images</i>	<b>6. CUSTOMER CONSTRAINTS</b> <span>CC</span> <i>Unclear image will not give accurate results.</i>	<b>5. AVAILABLE SOLUTIONS</b> <span></span> <i>Traditional systems of handwriting recognition have relied on handcrafted feature and a large amount of prior knowledge.</i>	Explore AS, differentiate
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span> <i>People can struggle to read others' handwriting. The handwritten digits are not always of the same size, width, orientation as they differ from writing of person to person, so the general problem would be while classifying the digits.</i>	<b>9. PROBLEM ROOT CAUSE</b> <span>RC</span> <i>The issue is that there's a wide range of handwriting - good and bad. This makes it tricky for programmers to provide enough examples of how every character might look.</i>	<b>7. BEHAVIOUR</b> <span>BE</span> <i>Customers must try with clear image and neat handwriting to get accuracy in digits</i>	
Identify strong TR & EM	<b>3. TRIGGERS</b> <span>TR</span> <i>When there is need for recognition of handwritten digits</i>	<b>10. YOUR SOLUTION</b> <i>It uses Artificial Neural Network to recognize them. Neural Network is used to train and identify written digits. After training and testing, the accuracy rate reached 99%. This accuracy rate is very high.</i>	<b>8. CHANNELS of BEHAVIOUR</b> <span>CH</span> <b>8.1 ONLINE</b> <i>Extract online channels from behaviour block</i>	Extract online & offline CH of BE
	<b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span> <i>frustration, exhausted &gt; curious, satisfied</i>		<b>8.2 OFFLINE</b> <i>Extract offline channels from different handwriting styles</i>	

Problem-Solution fit canvas is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 license  
Created by Daria Nepriakhina / Amaltama.com

AMALTAMA

## CHAPTER 4

### REQUIREMENT ANALYSIS

#### 4.1 FUNCTIONAL REQUIREMENTS

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Upload image	Image upload via files Image upload via folders Image upload via drive Image upload via web Image upload via scan/camera
FR-4	Spelling support	Identifies handwriting of different styles and fonts Spelling check
FR-5	Translation	Handwritten digits from the image are extracted. Conversion of handwritten digits into machine readable form
FR-6	Log out	Log out / sign out.

#### 4.2 NON FUNCTIONAL REQUIREMENTS

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The proposed system gives good results for images that contain handwritten text written in different styles, different size and alignment with varying background
NFR-2	Security	Only authorized people can access the system data and modify the database.

NFR-3	Reliability	The Database is frequently updated with handwriting of different styles and size and will rollback when any update fails.
NFR-4	Performance	The proposed system is advantageous as it uses fewer features to train the neural network, which results in faster convergence.
NFR-5	Availability	The system functionality and services are available for use with all operations.
NFR-6	Scalability	The website traffic limit must be scalable enough to support 2 lakhs users at a time

## CHAPTER 5

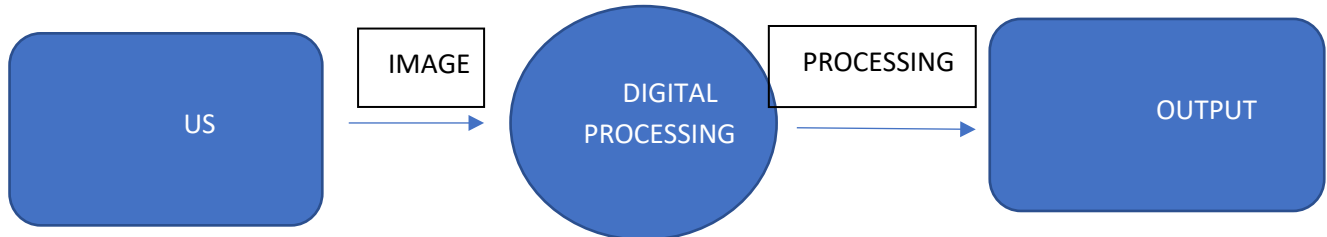
### PROJECT DESIGN

#### 5.1 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

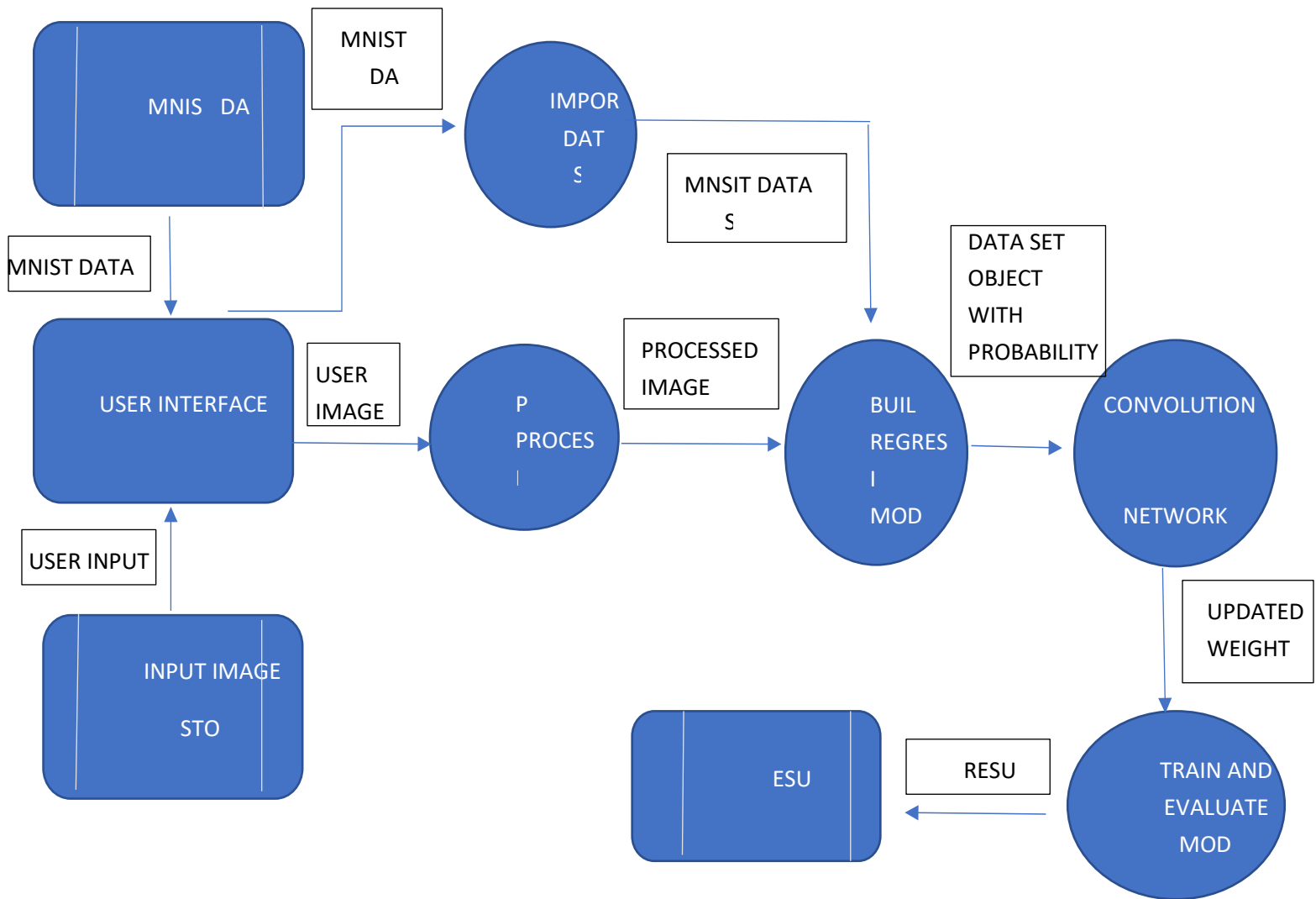
##### *DFD Level-0*

The DFD Level-0 consists of two external entities, the UI and the Output, along with a process, representing the CNN for Digit Recognition .Output is obtained after processing.



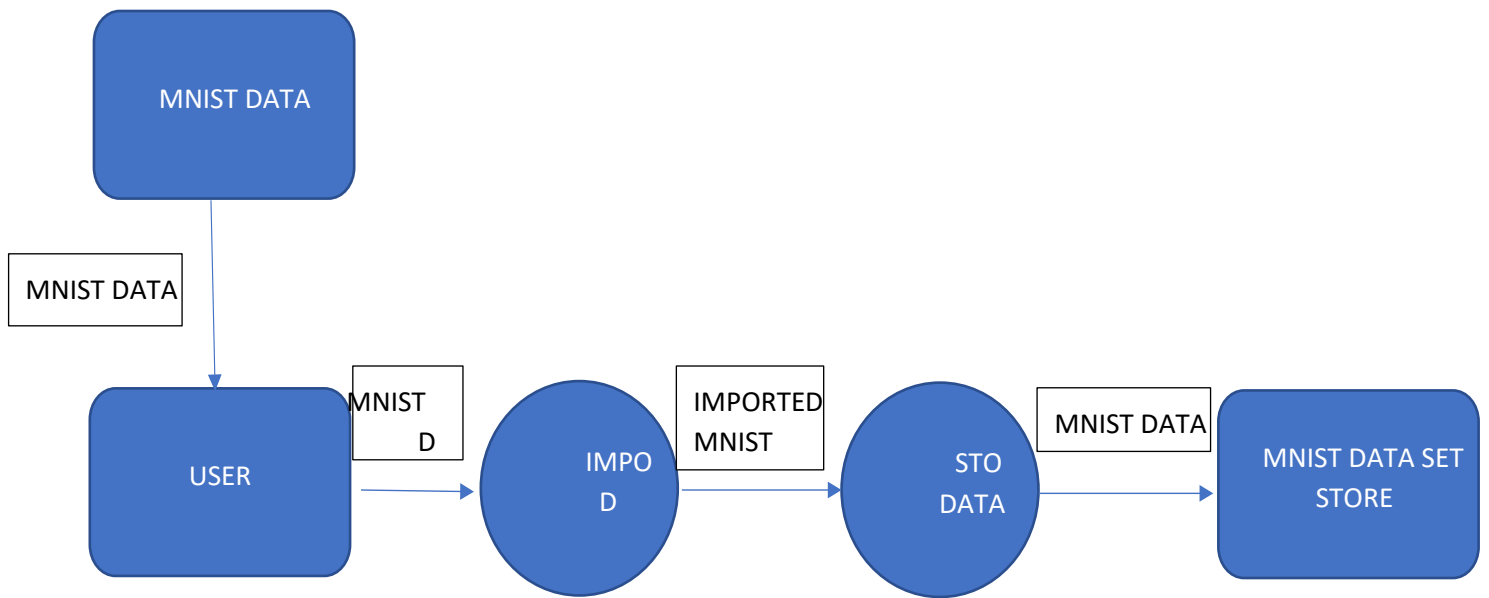
##### *DFD Level-1*

The DFD Level-1 consists of 2 external entities, the GUI and the Output, along with five process blocks and 2 data stores MNIST data and the Input image store, representing the internal workings of the CNN for Digit Recognition System. Process block imports MNIST data from library. Process block imports the image and process it and sends it to block where regression model is built. It sends objects with probabilities to CNN where weights are updated and multiple layers are built. Block trains and evaluates the model to generate output.

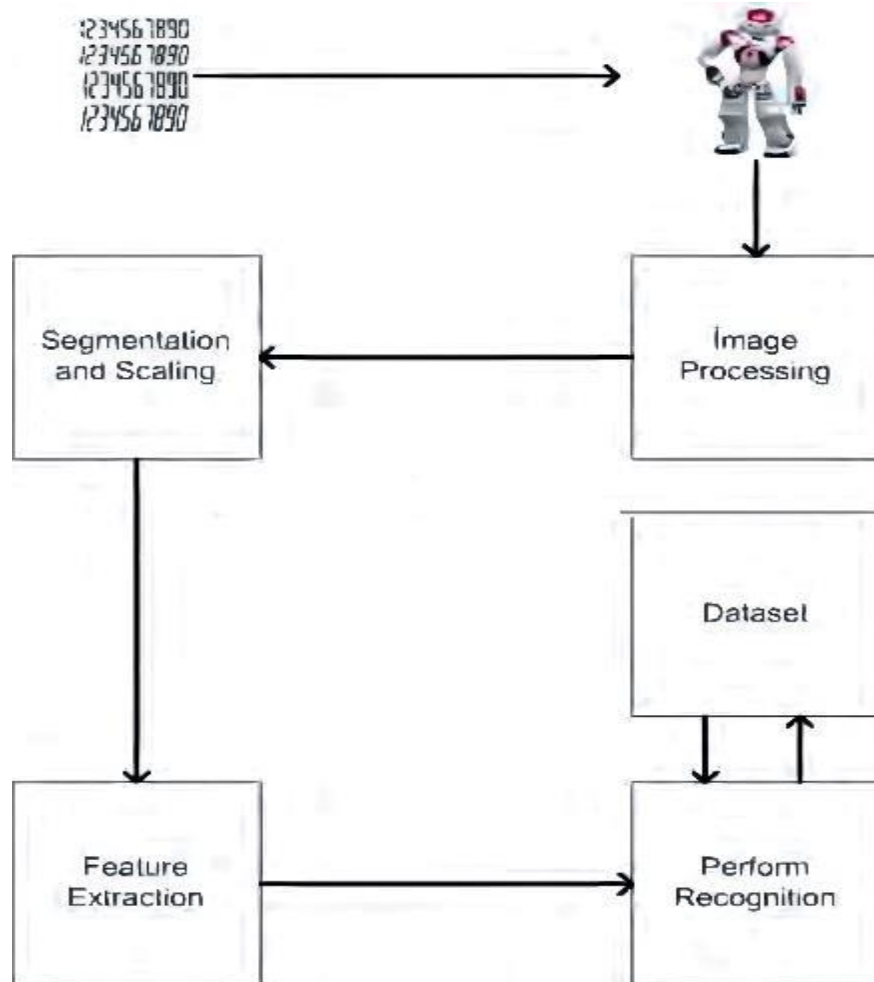


### ***DFD Level-2***

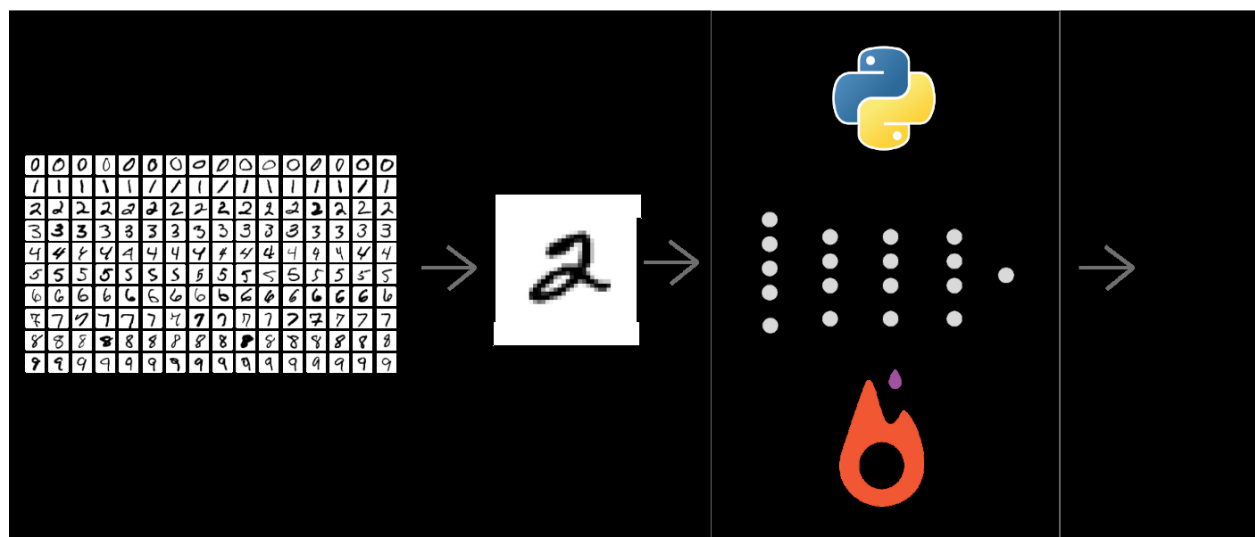
The DFD Level-2 for import data (figure 4) consists of two external data and one entity UI along with three process blocks, representing the three functionalities of the CNN for Digit Recognition System. It imports data from MNIST data store and stores on the system.



## 5.2 SOLUTION AND TECHNICAL ARCHITECTURE



## MNIST dataset processing with python





## CHAPTER 6

### PROJECT PLANNING & SCHEDULING

#### 6.1 SPRINT PLANNING & ESTIMATION

##### Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Collection of dataset from various resources with different handwritings.	10	Low	Hemapriya K Charulatha I
Sprint-1	Data Preprocessing	USN-2	Loading the dataset, handling the missing data, scaling and split data into train and test.	10	Medium	Ashwini P Gokila B
Sprint-2	Model Building	USN-3	Getting an application with ML model which provides high accuracy of recognized handwritten digit.	5	High	Hemapriya K Gokila B Ashwini P
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High	Gokila B Charulatha I

						Ashwini P
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	Ashwini P  Gokila B

<b>Sprint</b>	<b>Functional Requirement (Epic)</b>	<b>User Story Number</b>	<b>User Story / Task</b>	<b>Story Points</b>	<b>Priority</b>	<b>Team Members</b>
Sprint-2	Train & test the Model	USN-6	Train our model with our image dataset.	6	Medium	Hemapriya K Charulatha I
Sprint-2	Save the model	USN-7	The trained model is saved & integrated  with an android application or web  application in order to predict something.	2	Low	Hemapriya K
Sprint-3	Building UI Application	USN-8	Uploading the handwritten digit image to  the application by clicking a upload button.	5	High	Gokila B  Charulatha I Ashwini P
Sprint-3	Knowledge about	USN-9	Knowing the details of the fundamental	5	Low	Charulatha I

	application built		usage of the application.			
Sprint-3	Testing	USN-10	See the predicted / recognized digits in the application.	5	Medium	Hemapriya K Gokila B
Sprint-4	Train the model on IBM	USN-11	Train the model on IBM and integrate flask/Django with scoring endpoint.	10	High	Hemapriya K Gokila B Ashwini P Charulatha I
Sprint-4	Cloud Deployment	USN-12	Accessing the web application and make the use of the product from anywhere.	10	High	Hemapriya K Gokila B Ashwini P Charulatha I

## 6.2 SPRINT DELIVERY SCHEDULE

<b>Sprint</b>	<b>Total Story Points</b>	<b>Duration</b>	<b>Sprint Start Date</b>	<b>Sprint End Date (Planned)</b>	<b>Story Points Completed (as on Planned End Date)</b>	<b>Sprint Release Date (Actual)</b>
Sprint-1	20	6 Days	Oct 2022	9 Oct 2022	20	1 Oct 2022
Sprint-2	20	6 Days	Oct 2022	5 Nov 2022	20	5 Nov 2022
Sprint-3	20	6 Days	Nov 2022	2 Nov 2022	20	2 Nov 2022
Sprint-4	20	6 Days	Nov 2022	9 Nov 2022	20	9 Nov 2022

## CHAPTER 7

### CODING AND SOLUTIONING

#### 7.1 FEATURE 1

##### FLASK APP

```
import numpy as np
import os
from PIL import Image
from flask import Flask, request, render_template, url_for
from werkzeug.utils import secure_filename, redirect
from gevent.pywsgi import WSGIServer
from keras.models import load_model
from keras.preprocessing import image
from flask import send_from_directory

UPLOAD_FOLDER = 'C:\Users\Hp\OneDrive\Desktop\flask app\uploads'

app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

model = load_model("mnistCNN.h5")

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

@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == "POST":
        f = request.files["image"]
        filepath = secure_filename(f.filename)
        f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))

        upload_img = os.path.join(UPLOAD_FOLDER, filepath)
        img = Image.open(upload_img).convert("L") # convert image to monochrome
        img = img.resize((28, 28)) # resizing of input image
```

```

        im2arr = np.array(img) # converting to image
        im2arr = im2arr.reshape(1, 28, 28, 1) # reshaping according to our
requirement

        pred = model.predict(im2arr)

        num = np.argmax(pred, axis=1) # printing our Labels

        return render_template('predict.html', num=str(num[0]))

if __name__ == '__main__':
    app.run(debug=True, threaded=False)

```

## 7.2 FEATURE 2

### Test model in ibm cloud deployment

```

from tensorflow.keras.models import load_model
from keras.preprocessing import image
from PIL import Image
import numpy as np
model=load_model("mnistCNN.h5")

```

```

import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It
includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='80bpeEczl5btEjsCCpo0klid5WLhhYiedCKG6aEviOVK',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),

```

```
endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'handwrittendigitrecognition-donotdelete-pr-x9vvmczanx2bvt'
object_key = '9.png'

streaming_body_2 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the
possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/
```

```
img = Image.open(streaming_body_2).convert("L") # convert image to monochrome
img = img.resize( (28,28) ) # resizing of input image
im2arr = np.array(img) #converting to image
im2arr = im2arr.reshape(1, 28, 28, 1) #reshaping according to our requirement
```

```
pred = model.predict(im2arr)
print(pred)
print(np.argmax(pred, axis=1)) #printing our Labels
```

## CHAPTER 8

### TESTING

#### 8.1 TEST CASES

#### 8.2 USER ACCEPTANCE TESTING

##### 8.2.1 DEFECT ANALYSIS

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	2	0	1	0	3
Duplicate	0	0	0	0	0
External	0	0	0	1	1
Fixed	2	1	0	1	4
Not Reproduced	0	0	1	0	1
Skipped	0	0	0	1	1
Won't Fix	1	0	0	1	2
Totals	5	1	2	4	12

##### 8.2.2 TEST CASE ANALYSIS

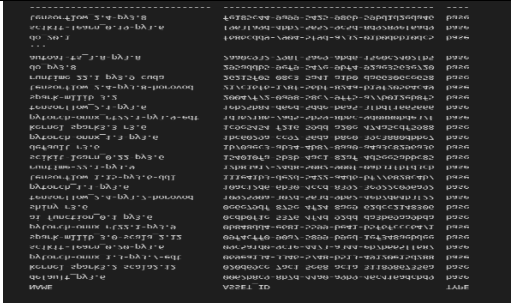
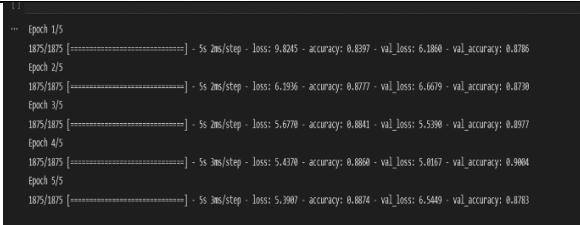
Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	0	10
Exception reporting	3	0	0	3
Final report output	4	0	0	4



CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot
1.	Model  Summary		
2.	Accuracy	Training  Accuracy – 88.74   Validation  Accuracy –87.83	

## **CHAPTER 10**

### **ADVANTAGES & DISADVANTAGES**

#### **ADVANTAGES**

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

#### **DISADVANTAGES**

- Cannot handle complex data
- All the data must be in digital format
- Requires a high performance server for faster predictions
- Prone to occasional errors

## **CHAPTER 11**

### **CONCLUSION**

This project demonstrated a web application that uses machine learning to recognise handwritten numbers. Flask, HTML, CSS, JavaScript, and a few other technologies were used to create this project. The model predicts the handwritten digit using a CNN network. During testing, the model achieved a 99.61% recognition rate. The proposed project is scalable and can easily handle a huge number of users. Since it is a web application, it is compatible with any device that can run a browser. This project is extremely useful in real-world scenarios such as recognizing number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (tax forms) and so on. There is so much room for improvement, which can be implemented in subsequent versions.

## **CHAPTER 12**

### **FUTURE SCOPE**

This project is far from complete and there is a lot of room for improvement. Some of the improvements that can be made to this project are as follows:

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

This project has endless potential and can always be enhanced to become better. Implementing this concept in the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency.

## APPENDIX

### SOURCE CODE

#### MODEL CREATION

```
#loading libraries
import numpy as np
import tensorflow
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.layers import Conv2D
from keras.utils import np_utils
import matplotlib.pyplot as plt
```

```
#loading data
(x_train, y_train), (x_test, y_test)=mnist.load_data ()

#data preprocessing
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 #storing the no of classes in a variable

y_train = np_utils.to_categorical (y_train, number_of_classes) #converts the
output in binary format
y_test = np_utils.to_categorical (y_test, number_of_classes)
```

```
#creating model
model=Sequential ()
#adding model Layer
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))
model.add(Conv2D(32, (3, 3), activation = 'relu'))
model.add(Dense(number_of_classes,activation = 'softmax'))
model.compile(loss= 'categorical_crossentropy', optimizer="Adam",
metrics=['accuracy'])
```

```

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

#evaluate the model
metrics = model.evaluate(x_test, y_test, verbose=0)
print("Metrics (Test loss &Test Accuracy) : ")
print(metrics)

#save the model
model.save("mnistCNN.h5")

#Test the saved model
import numpy as np
print(np.argmax(prediction, axis=1))
np.argmax(y_test[6000:6001])

```

## FLASK APP

```

import numpy as np
import os
from PIL import Image
from flask import Flask, request, render_template, url_for
from werkzeug.utils import secure_filename, redirect
from gevent.pywsgi import WSGIServer
from keras.models import load_model
from keras.preprocessing import image
from flask import send_from_directory

UPLOAD_FOLDER = 'C:\Users\Hp\OneDrive\Desktop\flask app\uploads'

app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

model = load_model("mnistCNN.h5")

```

```

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

@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == "POST":
        f = request.files["image"]
        filepath = secure_filename(f.filename)
        f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))

        upload_img = os.path.join(UPLOAD_FOLDER, filepath)
        img = Image.open(upload_img).convert("L") # convert image to monochrome
        img = img.resize((28, 28)) # resizing of input image

        im2arr = np.array(img) # converting to image
        im2arr = im2arr.reshape(1, 28, 28, 1) # reshaping according to our
requirement

        pred = model.predict(im2arr)

        num = np.argmax(pred, axis=1) # printing our Labels

        return render_template('predict.html', num=str(num[0]))

if __name__ == '__main__':
    app.run(debug=True, threaded=False)

```

## RECOGNIZER

```

from tensorflow.keras.models import load_model
from keras.preprocessing import image
from PIL import Image
import numpy as np
model=load_model("mnistCNN.h5")

```

```

import os, types

```

```

import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It
# includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
                              ibm_api_key_id='80bpeEczl5btEjsCCpo0klid5WLhhYiedCKG6aEviOVK',
                              ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
                              config=Config(signature_version='oauth'),
                              endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'handwrittendigitrecognition-donotdelete-pr-x9vvmczanx2bvt'
object_key = '9.png'

streaming_body_2 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the
# possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/

```

```

img = Image.open(streaming_body_2).convert("L") # convert image to monochrome
img = img.resize( (28,28) ) # resizing of input image
im2arr = np.array(img) #converting to image
im2arr = im2arr.reshape(1, 28, 28, 1) #reshaping according to our requirement

```

```

pred = model.predict(im2arr)
print(pred)
print(np.argmax(pred, axis=1)) #printing our Labels

```

INDEX.HTML



```

<html>

<head>
  <title>Digit Recognition WebApp</title>

  <meta name="viewport" content="width=device-width">
  <!-- GoogleFont -->
  <link
href="https://fonts.googleapis.com/css2?family=Prompt:wght@600&display=swap"
rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Varela+Round&display=swap"
rel="stylesheet">
  <link
href="https://fonts.googleapis.com/css2?family=Source+Code+Pro:wght@500&display=s
wap" rel="stylesheet">
  <link
href="https://fonts.googleapis.com/css?family=Calistoga|Josefin+Sans:400,700|Paci
fico&display=swap" rel="stylesheet">
  <!-- bootstrap -->
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
gg0yR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQU0hcWr7x9JVoRxT2MZw1T"
crossorigin="anonymous">
  <link rel="stylesheet" type= "text/css" href=url(flask app/static/style.css)>
  <!-- fontawesome -->
  <script src="https://kit.fontawesome.com/b3aed9cb07.js"
crossorigin="anonymous"></script>

  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-
U02eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1c1HTMga3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy60RQ6VrjIEEaFf/nJGzIxFDs4x0xIM+B07jRM"
crossorigin="anonymous"></script>
  <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>

```

```

</head>

<script>
    function preview() {
        frame.src=URL.createObjectURL(event.target.files[0]);
    }

    $(document).ready(function() {
        $('#clear_button').on('click', function() {
            $('#image').val('');
            $('#frame').attr('src','');
        });
    });
</script>

<body>

    <h1 class="welcome">IBM PROJECT
    </h1>
    <section id="title">
        <h4 class="heading">Handwritten Digit Recognition </h4>
        <br><br>
        <p><h2>
            The website is designed to predict the handwritten digit.</h2>
        </p>

    </section>

    <section id="content">

        <div class="leftside">
            <form action="/predict" method="POST" enctype="multipart/form-data">
                <label>Select a image:</label>
                <input id="image" type="file" name="image" accept="image/png, image/jpeg"
onchange="preview()"><br><br>
                <img id="frame" src="" width="100px" height="100px"/>
                <div class="buttons_div">
                    <button type="submit" class="btn btn-dark"
id="predict_button">Predict</a></button>
                    <button type="button" class="btn btn-dark" id="clear_button">&nbsp;
Clear &nbsp;</button>
                </div>
            </form>

```

```
        </div>

    </section>

</body>
</html>
```

## STYLE.CSS

```
#clear_button{
    margin-left: 15px;
    font-weight: bold;
    color: blue;
}

#confidence{
    font-family: 'Josefin Sans', sans-serif;
    margin-top: 7.5%;
}

#content{
    margin: 0 auto;
    padding: 2% 15%;
    padding-bottom: 0;
}

.welcome{
    text-align: center;
    position: relative;
    color: honeydew;
    background-color: greenyellow;
    padding-top: 1%;
    padding-bottom: 1%;
    font-weight: bold;
    font-family: 'Prompt', sans-serif;
}

#team_id{
    text-align: right;
    font-size: 25px;
}
```

```

padding-right: 3%;
}

#predict_button{
margin-right: 15px;
color: blue;
font-weight: bold;
}

#prediction_heading{
font-family: 'Josefin Sans', sans-serif;
margin-top: 7.5%;
}

#result{
font-size: 5rem;
}

#title{
padding: 1.5% 15%;
margin: 0 auto;
text-align: center;
}

.btn {
font-size: 15px;
padding: 10px;
-webkit-appearance: none;
background: #eee;
border: 1px solid #888;
margin-top: 20px;
margin-bottom: 20px;
}

.buttons_div{
margin-bottom: 30px;
margin-right: 80px;
}

.heading{
font-family: 'Varela Round', sans-serif;
font-weight: 700;
font-size: 2rem;
display: inline;
}

```

```

.leftside{
  text-align: center;
  margin: 0 auto;
  margin-top: 2%;
  /* padding-left: 10%; */
}

#frame{
  margin-right: 10%;
}

.predicted_answer{
  text-align: center;
  margin: 0 auto;
  padding: 3% 5%;
  padding-top: 0;
  /* padding-left: 10%; */
}

p{
  font-family: 'Source Code Pro', monospace,sans-serif;
  margin-top: 1%;
}

@media (min-width: 720px) {
  .leftside{
    padding-left: 10%;
  }
}

```

PREDICT.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Prediction</title>
</head>

<style>

  #rectangle{

```

```

width:500px;
height:250px;
background-color: blue;
border-radius: 100px;
position:absolute;
top:25%;
left:50%;
transform:translate(-50%,-50%);
}

#ans{
text-align: center;
font-size: 40px;
margin: 0 auto;
padding: 3% 5%;
padding-top: 15%;
color: white;
}

</style>
<body bgcolor="lightblue">
  <div id="rectangle">
    <h1 id="ans">Predicted Number :{num}</h1>
  </div>
</body>
</html>

```

## **GITHUB**

<https://github.com/IBM-EPBL/IBM-Project-11883-1659352204>

## **PROJECT DEMO**

<https://photos.app.goo.gl/TcgLZQr7yrLvRtmj8>