PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION

submitted by

PNT2022TMID39478

VIJAYAKUMAR P	510319106306
LAVANYA V	510319106002
AGILANDEESVAEI M	510319106301
SORNAMALAYA S	510319106305
SATHISH R	510319106304

TABLE OFCONTENTS

- SPRINT DELIVERY SCHEDULE

- **CODING & SOLUTIONING**

- **TESTING**
  - TEST CASES
  - USER ACCEPTANCE TESTING
    - DEFECT ANALYSIS
    - TEST CASE ANALYSIS

- **RESULTS**
  - PERFORMANCE METRICS

- **ADVANTAGES & DISADVANTAGES**

A

D

V

A

N

T

A

G

E

S

D

I

S

A

D

V

A

N

T

A

G

E

S

- **CONCLUSION**
- **FUTURE SCOPE**
- **APPENDIX**

S

O

U

R

C

E

C

O

D

E

G

I

T

H

U

B

P

R

O

J

E

C

T

D

E

M

O

CHAPTER 1

INTRODUCTION

- **PROJECT OVERVIEW**

Machine learning and deep learning play an important role in computer technology and artificial intelligence. With the use of deep learning and machine learning, human effort can be reduced in recognizing, learning, predictions and in many more areas.

Handwritten Digit Recognition is the ability of computer systems to recognise 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 tasksin recognizing digits.

- **PURPOSE**

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 soon.

CHAPTER 2

LITERATURE SURVEY

- ## EXISTING PROBLEM

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

- ## REFERENCES

Improved Handwritten Digit Recognition Using Convolutional Neural Networks(CNN) (2020)

Ahlawat, Savita and Choudhary, Amit and Nayyar, Anand and Singh, Saurabh and Yoon, Byungun

This paper's primary goal was to enhance handwritten digit recognition ability. To avoid difficult pre-processing, expensive feature extraction, and a complex ensemble (classifier combination) method of a standard recognition system, they examined different convolutional neural network variations. Their current work makes suggestions on the function of several hyper-parameters through thorough

evaluation utilizing an MNIST dataset. They also confirmed that optimizing

hyper-parameters is crucial for enhancing CNN architecture performance. With the Adam optimizer for the MNIST database, they were able to surpass many previously published results with a recognition rate of 99.89%. Through the trials, it is made

abundantly evident how the performance of handwritten digit recognition is affected by the number of convolutional layers in CNN architecture. According to the paper, evolutionary algorithms can be explored for optimizing convolutional filter kernel sizes, CNN learning parameters, and the quantity of layers and learning rates.

An Efficient And Improved Scheme For Handwritten Digit Recognition Based On Convolutional Neural Network (2019)

Ali, Saqib and Shaukat, Zeeshan and Azeem, Muhammad and Sakhawat, Zareen and Mahmood, Tariq and others

This study uses rectified linear units (ReLU) activation and a convolutional neural network (CNN) that incorporates the Deeplearning4j (DL4J) architecture to recognize handwritten digits. The proposed CNN framework has all the necessary parameters for a high level of MNIST digit classification accuracy. The system's training takes into account the time

factor as well. The system is also tested by altering the number of CNN layers for additional accuracy verification. It is important to note that the CNN architecture consists of two convolutional layers, the first with 32 filters and a 5x5 window size and the second with 64 filters and a 7x7 window size. In comparison to earlier proposed systems, the experimental findings show that the proposed CNN architecture for the MNIST dataset demonstrates great performance in terms of time and accuracy. As a result, handwritten numbers are detected with a recognition rate of 99.89% and high precision (99.21%) in a short amount of time.

Improved Handwritten Digit Recognition Using Quantum K-Nearest NeighborAlgorithm (2019)

Wang, Yuxiang and Wang, Ruijin and Li, Dongfen and Adu-Gyamfi, Daniel and Tian, Kaibin and Zhu, Yixin

The KNN classical machine learning technique is used in this research to enable quantum parallel computing and superposition. They used the KNN algorithm with quantum acceleration to enhance handwritten digit recognition. When dealing with more complicated and sizable handwritten digital data sets, their suggested method considerably lowered the computational time complexity of the traditional KNN algorithm. The paper offered a theoretical investigation of how quantum concepts can be applied to machine learning. Finally, they established a fundamental operational concept and procedure for machine learning with quantum acceleration.

Handwritten Digit Recognition Using Machine And Deep Learning Algorithms(2021)

Pashine, Samay and Dixit, Ritik and Kushwah, Rishika

In this study, they developed three deep and machine learning-based models for handwritten digit recognition using MNIST datasets. To determine which model was the most accurate, they compared them based on their individual properties.

Support vector machines are among the simplest classifiers, making them faster than other algorithms and providing the highest training accuracy rate in this situation. However, due to their simplicity, SVMs cannot categorize complicated and ambiguous images as accurately as MLP and CNN algorithms can. In their research, they discovered that CNN produced the most precise outcomes for handwritten digit recognition. This led them to the conclusion that CNN is the most effective

solution for all types of prediction issues, including those using picture data. Next, by comparing the execution times of the algorithms, they determined that increasing the number of epochs without changing the configuration of the algorithm is pointless due to the limitation of a certain model, and they discovered that beyond a certain number of epochs, the model begins over-fitting the dataset and provides biased predictions.

- **PROBLEM STATEMENT DEFINITION**

Human handwritten digits can be recognized by using deep learning and machine learning by handwritten digit recognition. This identifies the image and recognizes the digit from the whole dataset of all the digits with maximum accuracy based on Convolution al Neural Networks. MLP helps in better classification of images compared to other algorithms since a machine cannot classify handwritten digits. Final output predicts the accurate digit for the handwritten digit to be of a fairly good efficiency.

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

- **EMPATHY MAP CANVAS**

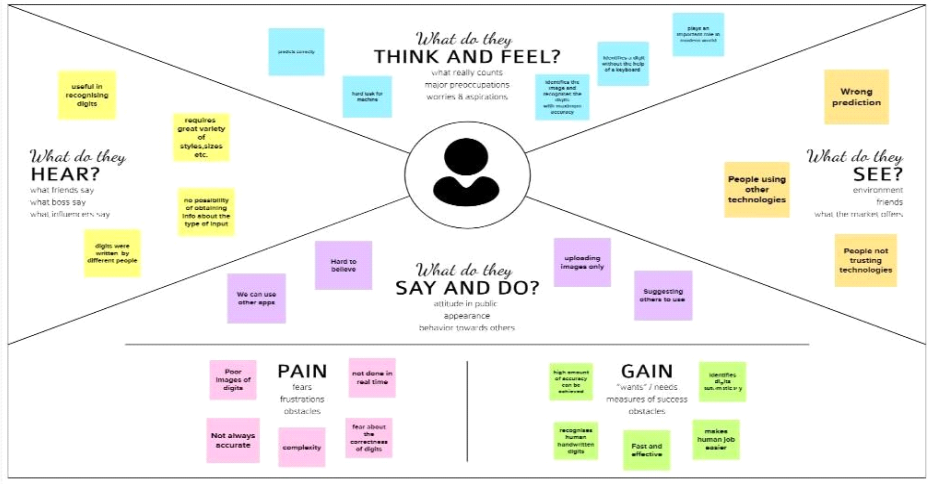
Edit this template
Right-click to unlock

Empathy Map Canvas

Gain insight and understanding on solving customer problems.

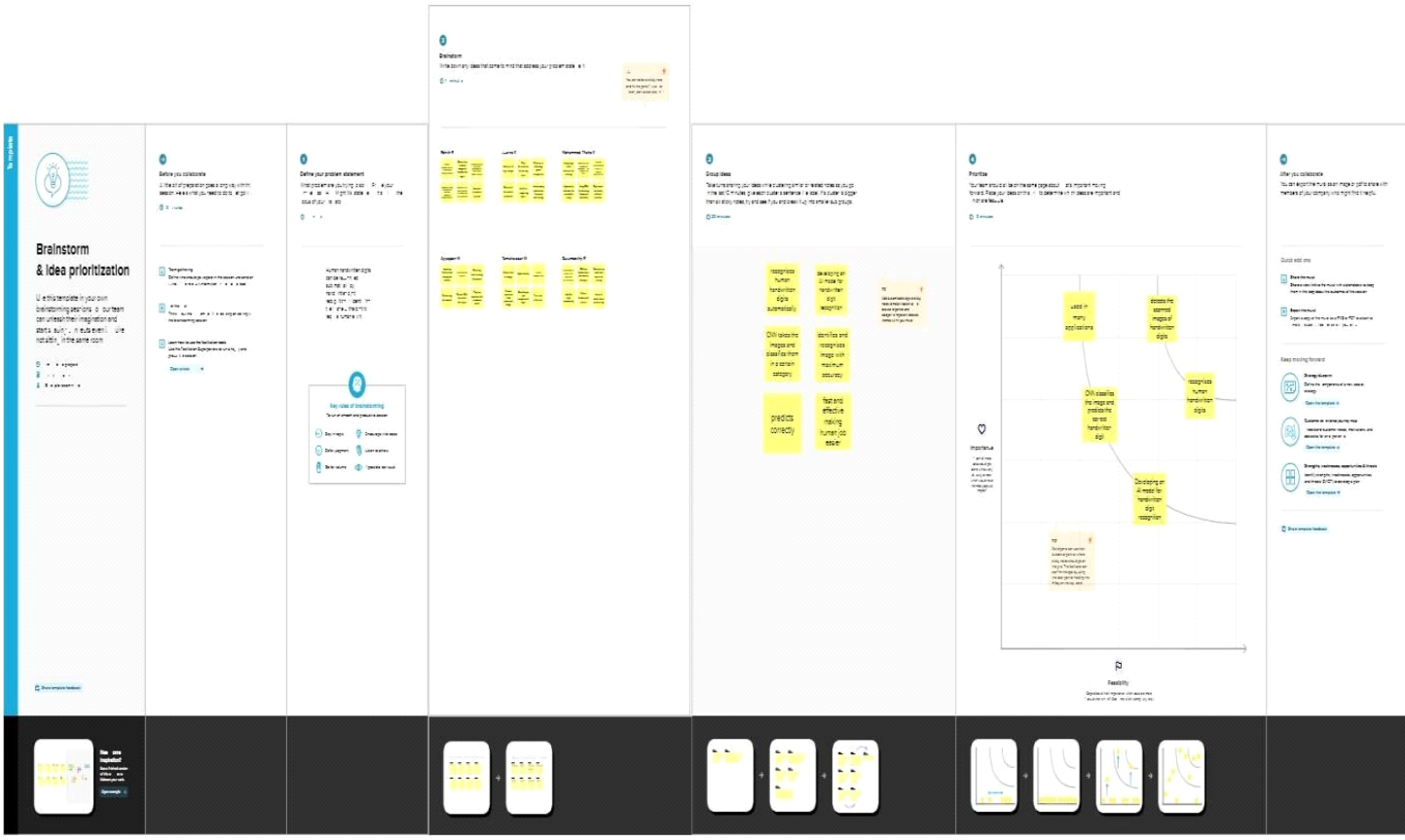
1

Build empathy and keep your focus on the user by putting yourself in their shoes.



Share your feedback

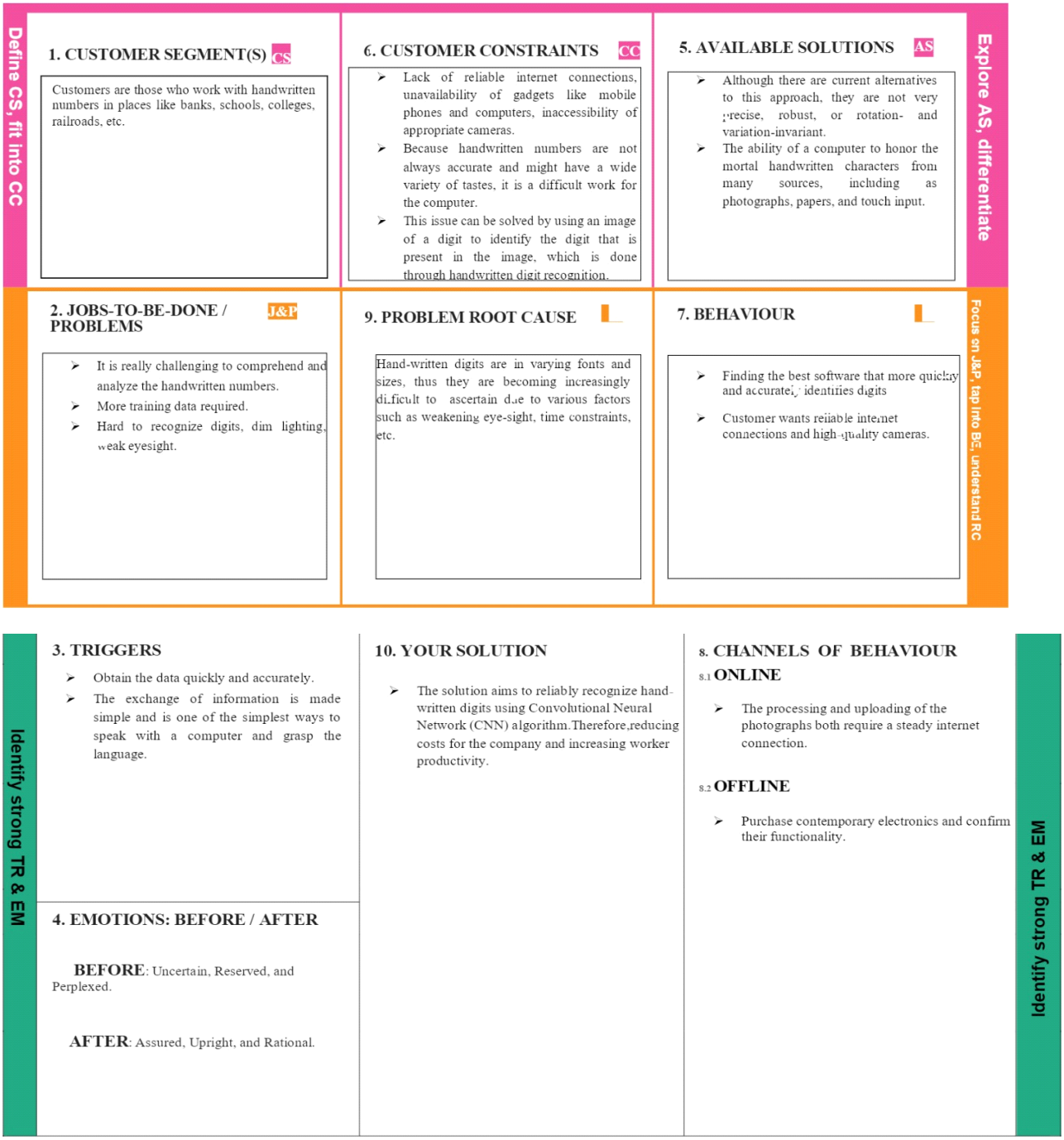
- **IDEATION & BRAINSTORMING**



- **PROPOSED SOLUTION**

S.NO	PARAMETER	DESCRIPTION
1	Problem Statement	The challenge is getting a computer system to read human-written digits. The objective is to accurately identify the digit after uploading a photograph of the handwritten digit.
2	Idea / Solution Description	The Convolution Neural Network algorithm (CNN). By doing this, the trained model will be ready to be used to categorise the digits found in the test data. As a result, the digits in the photos can be categorised as Class 0,1,2,3,4,5,6,7,8,9. A dataset that is frequently used for handwritten digit recognition is MNIST. 10,000 test photos and 60,000 training images make up the dataset.
3	Novelty / Uniqueness	With written characters in high-quality photos, OCR technology offers greater than 99% accuracy. But unlike OCR, it only recognises only the digits, not all the characters. With the aid of a neural network, handwritten digit recognition is performed using the MNIST dataset. It recognises the scanned copies of handwritten numbers. In further step, the handwritten digit recognition system allows users to write their own digits on the screen with the aid of an integrated GUI for recognition in addition to detecting scanned images of handwritten digits.
4	Social Impact / CustomerSatisfaction	Handwritten Digit Recognition has various uses such as less time consumption. It is used in the detection of vehicle numbers, banks for reading cheques, post offices for arranging letters, and other tasks

- **PROBLEM SOLUTIONS**



CHAPTER 4

4.1 FUNCTIONAL REQUIREMENTS

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration is neither needed nor necessary.
FR-2	Input image	Input images must be given by the user. The inputs are handwritten digits from 0 to 9. The model classifies them and convert them into digitalized form.
FR-3	Algorithm	Convolution Neural Network is used to implement handwritten digit recognition.
FR-4	Dataset	Import MNIST dataset. It is a dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9.
FR-5	Website	Build a GUI in which we can draw the digit and recognize it in a straight away.
FR-6	Cloud	Access to files on any device is made possible through the cloud for employees. Anyone wishing to develop memory-demanding, complex Machine Learning/Deep Learning models should consider using cloud services. A cost-effective solution is offered by cloud services.

- **NON-FUNCTIONAL REQUIREMENTS**

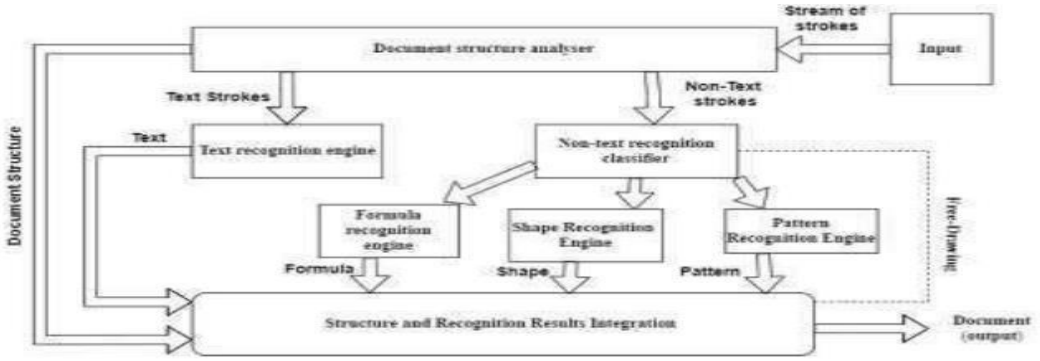
Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	It is user-friendly and it is used in the detection of car numbers, reading of checks at banks, post offices and other tasks.
NFR-2	**Security**	Authentication can be used to verify the user. It also provides security by ensuring that once recognition is complete, the images uploaded for it will not be stored.
NFR-3	**Reliability**	This system will work reliably for low resolution images.
NFR-4	**Performance**	Handwritten digits in the input image will be recognized with an accuracy of 99.99% .

CHAPTER 5 PROJECT DESIGN

- **DATA FLOW**



=

- **SOLUTION & TECHNICAL ARCHITECTURE**

Technical Architecture for Handwritten Digit Recognition System:

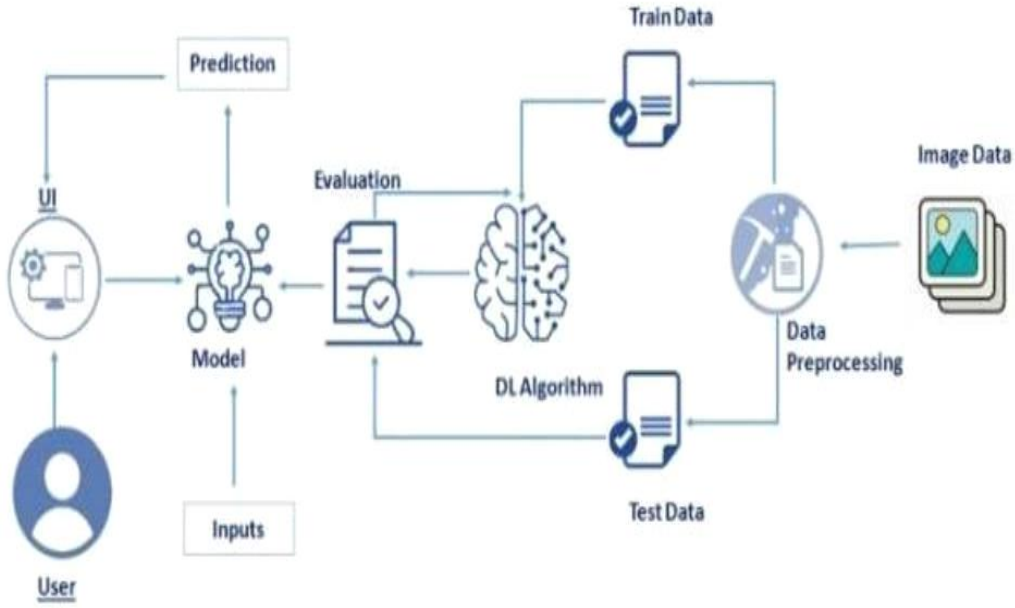


Table-1: Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	User interacts with me application using a web app	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic	Login to access the application	Java / Python
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	Storage of user files of handwritten image	IBM Block Storage or Other Storage Service or Local Filesystem
10.	Machine Learning Model	Machine learning model is used lo identify the nandwr!lten uploaded by users	Object Recognition Model, etc.
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Al Local Server Configuration Al Server Configuration	Local, Cloud Foundry, Kubernetes, etc.

Table•2: Application Characteristics:

S.No	Characteristics	Description	Technology

1.	Upen-Source Frameworks	Machine learning frameworks is used to train a predictive model	PyTorch, Open-cv
2.	Security Implementations	The system should automatically be able to authenticate all users with their unique username and password	Password based login, Authorization
3.	Scalable Architecture	The website traffic limit must be scalable enough to support 2 lakhs users at a time	3-tier
4.	Availability	The system functionality and services are available for use witn all operations.	distributed servers
5.	Performance	The application can give response to requests within 5 sec. It uses*ewer*eatures to train theneural networ , which results in faster convergence.	number of requests per sec

User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Home	USN-1	As a user, I can able to know about the application and read the instruction to usage of mobile app.	I can view the instruction about application.	High	Sprint-1
		USN-2	As a user, I am allowed to view Demo video for using the application.	I can gain Knowledge from Demo Video.	High	Sprint-4
		USN-3	As a user, I can access the MNIST dataset from my Drive Files.	I can access the MNIST dataset to get the output.	Low	Sprint-2
	Upload	USN-4	As a user, I have access to upload the dataset from my Drive Files or from other Files.	I can upload the image from System Storage.	Medium	Sprint-1
	Result	USN-5	As a user, I can able to view the result of uploaded image as my predicted output.	I can able to view the result of uploaded image.	High	Sprint-1
Customer (Web View)	Home	USN-6	As a user, I can read the information about the Web application.	I can read and gain knowledge about the web application.	High	Sprint-1
	Pre-Processing	USN-7	As a user, I will train and test the input.	I can able to train and test the input data	High	Sprint-4
	Recognize	USN-8	As a user, I can recognize how the input is evaluated.	I can able to know the Evolution of input.	Low	Sprint-2
	Predict	USN_9	As a user, I am able to predict the image.	I can able to predict the image.	Medium	Sprint-3
	Accuracy	USN_10	As a user, I can see the accuracy of my input image as output result.	I can able to view the resulted output.	High	Sprint-1

CHAPTER 6

PROJECT PLANNING ANDSCHEDULING

- **SPRINT PLANNING AND ESTIMATION**

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Get the Dataset	USN-1	As a user, I need to collect the data with different handwriting to train the model	6	High	Aparna K Rohith R
Sprint-1	Importing Libraries	USN-2	As a user, I have to implement necessary libraries in python packages.	4	Low	Ayyanpan M Gurumoorthy R
Sprint-1	Data pre-processing	USN-3	As a user, I can load the dataset, handle the missing values, scale and split the data.	10	Medium	Tamizharasan M Mohammed Thaha K Rohith R
Sprint-2	Model building	USN-4	As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit	5	High	Mohammed Thaha K

Sprint-2	Add the CNN layers	USN-5	Add input convolutional layer, maxpooling layer, flatten , hidden and output layers to the model.	5	High	Ayyappan M
Sprint- 2	Compile the model	USN-6	As a user, compile the model for trained dataset.	2	Medium	Tamizharasan M
Sprint-2	Train and test the model	USN-7	As a user, train and test the model for the dataset collected and data are validated.	4	High	Rohith R Abarna K
Sprint-2	Save the model	USN-8	As a user, the compiled data are saved and integrated with an android application or web application.	2	Low	Gurumoorthy R
Sprint-3	HTML-Home page	USN-9	As a user upload the input image that contains handwritten digits.	10	Medium	Rohith R Abarna K
Sprint-3	Building UI application	USN-10	As a user, I can provide the fundamental details about the usage of application to customer.	5	Low	Ayyappan M Mohammed Thaha K Tamizharasan M
Sprint-3	Run the application	USN-11	As a user, I can see the predicted or recognized digits in the application.	5	Medium	Gurumoorthy R
Sprint-4	Train the model on IBM	USN-12	As a user train the model in IBM cloud and integrate the results.	10	High	Ayyappan M Tamizharasan M Gurumoorthy R

- **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	31 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	6 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	13 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

CHAPTER 7

CODING & SOLUTIONING

```python
# Import necessary packages
import os
import random
import string
from pathlib import Path
import numpy as np
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps
```

```python
def random_name_generator(n: int) -> str:
    """
    Generates a random file name.

    Args:
        n (int): Length the of the file name.

    Returns:
        str: The file name.
    """
    return ''.join(random.choices(string.ascii_uppercase + string.digits, k=n))
```

```python
def recognize(image: bytes) -> tuple:
    """
    Predicts the digit in the image.

    Args:
        image (bytes): The image data.

    Returns:
        tuple: The best prediction, other predictions and file name
    """

    model=load_model(Path("./model/model.h5"))

    img = Image.open(image).convert("L")

    # Generate a random name to save the image file.
    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}"))

    # Convert the Image to Grayscale, Invert it and Resize to get better prediction.
    img = ImageOps.grayscale(img)
    img = ImageOps.invert(img)
    img = img.resize((28, 28))

    # Convert the image to an array and reshape the data to make prediction.
    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]

    # Get all the predictions and it's respective accuracy.
    pred = list(map(lambda x: round(x*100, 2), results[0]))

    values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
    others = list(zip(values, pred))

    # Get the value with the highest accuracy
    best = others.pop(best)

    return best, others, img_name
```

CHAPTER 8

TESTING

- **TEST CASES**

Test caseID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
HP_TC_001	UI	Home Page	Verify UI elements in the Home Page	The Home page must be displayed properly	Working as expected	PASS
HP_TC_002	UI	Home Page	Check if the UIelements are displayed properly in different screen sizes	The Home page must be displayed properly in all sizes	The UI is not displayed properly in screen size 2560 x 1801 and 768 x 630	FAIL
HP_TC_003	Functional	Home Page	Check if usercan upload their file	The input image should be uploaded to the application successfully	Working as expected	PASS
HP_TC_004	Functional	Home Page	Check if usercannot upload unsupported files	The application should not allow user to select a non image file	User is able to upload any file	FAIL
HP_TC_005	Functional	Home Page	Check if the page redirectsto the result page oncethe	The page should redirect to the results page	Working as expected	PASS

			input is given			

BE_TC_001	Functional	Backend	Check if all theroutes are working properly	All the routes should properly work	Working as expected	PASS
M_TC_001	Functional	Model	Check if the model can handle various image sizes	The model should rescale the imageand predict the results	Working as expected	PASS
M_TC_002	Functional	Model	Checkifthe model predicts thedigit	The model should predict the number	Working as expected	PASS
M_TC_003	Functional	Model	Check if the model can handle complex inputimage	The model should predict the number in the complex image	The model failsto identify the digit since themodel is not built to handle such data	FAIL
RP_TC_001	UI	Result Page	Verify UI elements in the Result Page	The Result page must be displayed properly	Working as expected	PASS
RP_TC_002	UI	Result Page	Check if the input image isdisplayed properly	The input image should be displayed	The size of theinput image exceeds the display	FAIL

				properly	container	
RP_TC_003	UI	Result Page	Check if the result is displayed properly	The result shouldbe displayed properly	Working as expecte d	PASS
RP_TC_004	UI	Result Page		The other predictions shouldbe displayed properly	Working as expecte d	PASS

- **USER ACCEPTANCE TESTING**
- **DEFECT ANALYSIS**

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Total
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won't Fix	1	0	1	0	2
Total	6	1	4	3	14

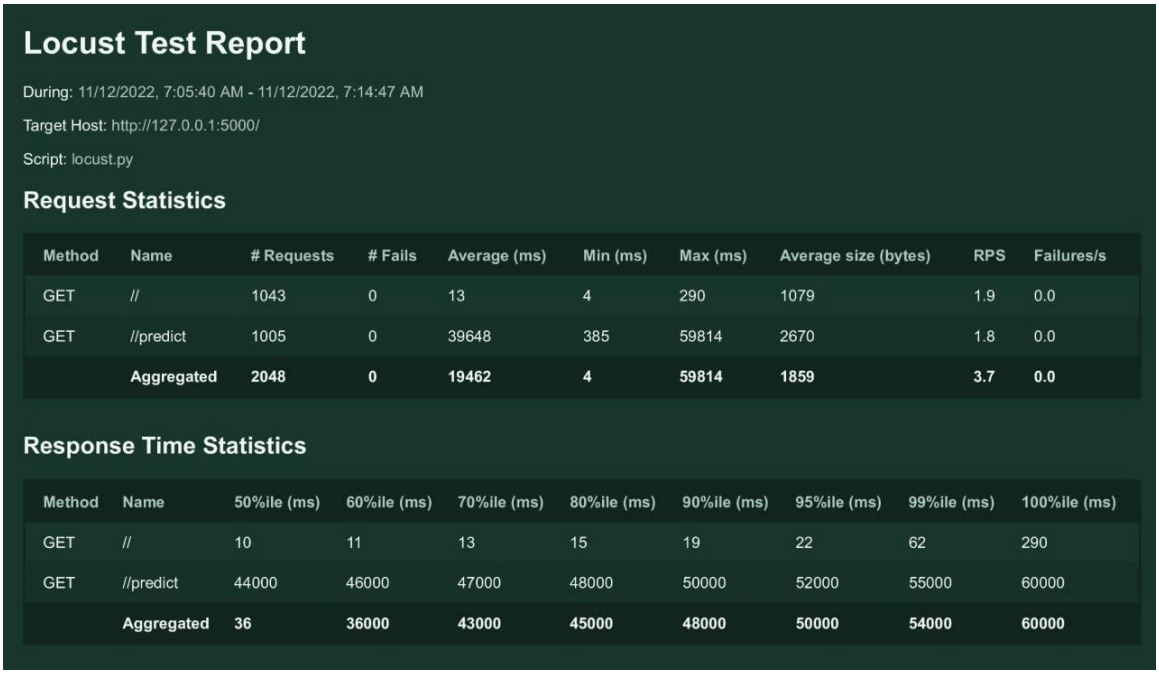
- **TEST CASE ANALYSIS**

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

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS



Locust Test Report

During: 11/12/2022, 7:05:40 AM - 11/12/2022, 7:14:47 AM

Target Host: http://127.0.0.1:5000/

Script: locust.py

Request Statistics

Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	//	1043	0	13	4	290	1079	1.9	0.0
GET	//predict	1005	0	39648	385	59814	2670	1.8	0.0
	Aggregated	**2048**	**0**	**19462**	**4**	**59814**	**1859**	**3.7**	**0.0**

Response Time Statistics

Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	//	10	11	13	15	19	22	62	290
GET	//predict	44000	46000	47000	48000	50000	52000	55000	60000
	Aggregated	**36**	**36000**	**43000**	**45000**	**48000**	**50000**	**54000**	**60000**

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

CHAPTER 12
FUTURE SCOPE

This project is far from complete and there is a lot of room for improvement.

Some of the improvements that can be made to this project are as follows:

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

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

APPENDIX

SOURCE CODE

```
# Load the 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

# Load the data
(X_train, y_train), (X_test, y_test) = mnist.load_data()

# 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)
```

```
# Create 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(Flatten())
model.add(Dense(number_of_classes, activation="softmax"))

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

# Train the model
model.fit(X_train, Y_train, batch_size=32, epochs=5, validation_data=(X_test,Y_test))

# 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("model.h5")
```

MODEL CREATION

```
# Test the saved model
model=load_model("model.h5")

img = Image.open("sample.png").convert("L")
img = img.resize((28, 28))
img2arr = np.array(img)
img2arr = img2arr.reshape(1, 28, 28, 1)
results  = model.predict(img2arr)
results = np.argmax(results,axis = 1)
results = pd.Series(results,name="Label")
print(results)
```

```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()
```

FLASK APP

```python
# Import necessary packages
import os
import random
import string
from pathlib import Path
import numpy as np
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps
```

RECOGNIZER

```python
def random_name_generator(n: int) -> str:
    """
    Generates a random file name.

    Args:
        n (int): Length the of the file name.

    Returns:
        str: The file name.
    """
    return ''.join(random.choices(string.ascii_uppercase + string.digits, k=n))
```

```python
def recognize(image: bytes) -> tuple:
    """
    Predicts the digit in the image.

    Args:
        image (bytes): The image data.

    Returns:
        tuple: The best prediction, other predictions and file name
    """

    model=load_model(Path("./model/model.h5"))

    img = Image.open(image).convert("L")

    # Generate a random name to save the image file.
    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}"))

    # Convert the Image to Grayscale, Invert it and Resize to get better prediction.
    img = ImageOps.grayscale(img)
    img = ImageOps.invert(img)
    img = img.resize((28, 28))

    # Convert the image to an array and reshape the data to make prediction.
    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]

    # Get all the predictions and it's respective accuracy.
    pred = list(map(lambda x: round(x*100, 2), results[0]))

    values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
    others = list(zip(values, pred))

    # Get the value with the highest accuracy
    best = others.pop(best)

    return best, others, img_name
```


HOME PAGE (HTML)

```css
.container {
    width: 100%;
    height: 100%;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    background-color: white;
}

.heading {
    margin-top: -2rem;
    padding-bottom: 2rem;
    width: fit-content;
    text-align: center;
}

.heading .heading__main {
    font-size: 3rem;
    font-weight: 550;
}

.heading .heading__sub {
    font-size: 1rem;
    color: rgb(90, 88, 88);
}

.upload-container {
    box-shadow: 0 0 20px rgb(172, 170, 170);
    width: 40rem;
    height: 25rem;
    padding: 1.5rem;
}

.form-wrapper {
    background-color: rgba(190, 190, 190, 0.5);
    width: 100%;
    height: 100%;
    display: flex;
    border: 1px dashed black;
    justify-content: center;
    align-items: center;
}

.form-wrapper #loading {
    display: none;
    position: absolute;
}
```

```css
.form-wrapper .upload {
    display: flex;
    justify-content: center;
    align-items: center;
    width: 8rem;
    height: -webkit-fit-content;
    height: -moz-fit-content;
    height: fit-content;
    border-radius: 6px;
    color: white;
    background-color: rgb(114, 96, 182);
    box-shadow: 0 5px 10px rgb(146, 135, 247);
}

.form-wrapper .upload #up_btn {
    display: none;
}

.form-wrapper .upload label {
    font-size: 1rem;
    font-weight: 600;
    color: white;
    height: 100%;
    width: 100%;
    padding: 10px;
    display: block;
}

.form-wrapper .upload svg {
    height: 15px;
    width: auto;
    padding-right: 8px;
    margin-bottom: -2px;
}

@media screen and (max-width: 700px) {
    .upload-container {
        height: 20rem;
        width: 18rem;
        margin-top: 3.5rem;
        margin-bottom: -8rem;
    }

    .heading .heading__main {
        margin-top: -6rem;
        font-size: 2rem;
        padding-bottom: 1rem;
    }
}
```

HOME PAGE (JS)

```
feather.replace(); // Load feather icons

form = document.querySelector('.upload')
loading = document.querySelector("#Loading")
select = document.querySelector("#upload-image");

select.addEventListener("change", (e) => {
    e.preventDefault();

    form.submit()
    form.style.visibility = "hidden";
    loading.style.display = 'flex';
});
```

PREDICT PAGE (HTML)

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

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

body {
    color: black;
    font-family: "Overpass", sans-serif;
}

h1 {
    padding-top: 2rem;
}

.container {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
}

.result-wrapper {
    width: -webkit-fit-content;
    width: -moz-fit-content;
    width: fit-content;
    height: -webkit-fit-content;
    height: -moz-fit-content;
    height: fit-content;
    box-shadow: 0 0 10px rgb(126, 125, 125);
    padding: 1.5rem;
    display: flex;
    justify-content: center;
    align-items: center;
    -moz-column-gap: 1rem;
    column-gap: 1rem;
}

.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: rgb(209, 206, 206);
}
```

```css
.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;
}

.other_predictions {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-wrap: wrap;
    column-gap: 1rem;
    row-gap: 1rem;
    font-weight: 700;
}

.other_predictions .value {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
    width: 5rem;
    height: 5rem;
    box-shadow: 0 0 7px rgb(158, 157, 157);
}

.other_predictions .value div {
    margin-top: -1.2rem;
}

@media screen and (max-width: 700px) {
    h1 {
        font-size: 2.3rem;
    }

    .result-wrapper .input-image-container,
    .result-wrapper .result-container {
        width: 7rem;
        height: 7rem;
    }

    .result-wrapper .result-container .value {
        font-size: 4rem;
    }
}
```

GITHUB

h HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"t HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"t HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"p HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"s: HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"// HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"g HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"i HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"t HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"hub. HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"c HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"o HYPERLINK

"https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"m HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"/I HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"B HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"M HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"- HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"E HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"P HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"B HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"L HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"/I HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"B HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"M HYPERLINK

"https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"- HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"P HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"r HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"o HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"j HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"e HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"c HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"t HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"- HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"12275 HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-12275-1659445819"- 1659445819

PROJECT DEMO

https://drive.google.com/file/d/1s8AhclYgCb5rwebvyXDgTdlEtoP4pkRV/view?usp=share_link