

Project Documentation

A Novel method for handwritten digit recognition

Team id : PNT2022TMID53257

1) Introduction

1.1) Project Overview

The user interacts with the UI (User Interface) to upload the image as input.

The uploaded image is analyzed by the model which is integrated.

Once the model analyses the uploaded image, the prediction is showcased on the UI.

1.2) Purpose

Humans with the help of their brain can recognize the things that they see. Similarly, deep neural networks are developed for the computers to recognize what they see through the User Interface(UI). Handwritten digit recognition is the ability of a computer to receive and interpret intelligible handwritten digit input from sources such as paper documents, photographs, touch-screens and other devices. The applications of digit recognition includes postal mail sorting, bank check processing, form data entry, etc. The heart of the problem lies within the ability to develop an efficient algorithm that can recognize handwritten digits and which is submitted by users by way of a scanner, tablet, and other digital devices.

2) Literature Survey

2.1) Existing Problem

The different architectures of CNN, hybrid CNN, CNN - RNN and CNNHMM models, and domain - specific recognition system, are not thoroughly inquired and evolutionary algorithms are not clearly explored for optimizing CNN learning parameters ,the number of layers, learning rate and kernel sizes of convolutional filters. The fluctuation of

accuracies for handwritten digits was observed for 15 epochs by varying the hidden layers. There is no clear explanation given for observing variation in the overall classification accuracy by varying the number of hidden layers and batch size.

2.2) References

S.NO	Author Name	Paper Title	Journal/Conference title	Page No/Volume No	Year of Publication	Description
1	Savita Ahlawat, Amit Choudhary, Anand Nayyar, Saurabh Singh and Byungun Yoon	Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN)	IEEE Sensors Journal		2020	In this paper, with the aim of improving the performance of handwritten digit recognition, they evaluated variants of a convolutional neural network to avoid complex preprocessing, costly feature extraction and a complex ensemble (classifier combination) approach of a traditional recognition system

2	Vijayalaxmi R Rudraswamimath, Bhavanishankar and Channasandra.	Handwritten Digit Recognition using CNN	International Journal of Innovative Science and Research Technology	Volume-4 Issue-6	2019	In this paper, the most widely used Machine learning algorithms, KNN, SVM, RFC and CNN have been trained and tested on the same data in order to acquire the comparison between the classifiers
3	Akanksha Gupta, Ravindra Pratap Narwaria and Madhav Singh.	Review on Deep Learning Handwritten Digit Recognition using Convolutional Neural Network	International Journal of Recent Technology and Engineering (IJRTE)	Volume-9 Issue-5	2021	In this paper, Object Character Recognition (OCR) is used on printed or documented letters to convert them into text. The database has training image database of 60,000 images and testing image database of

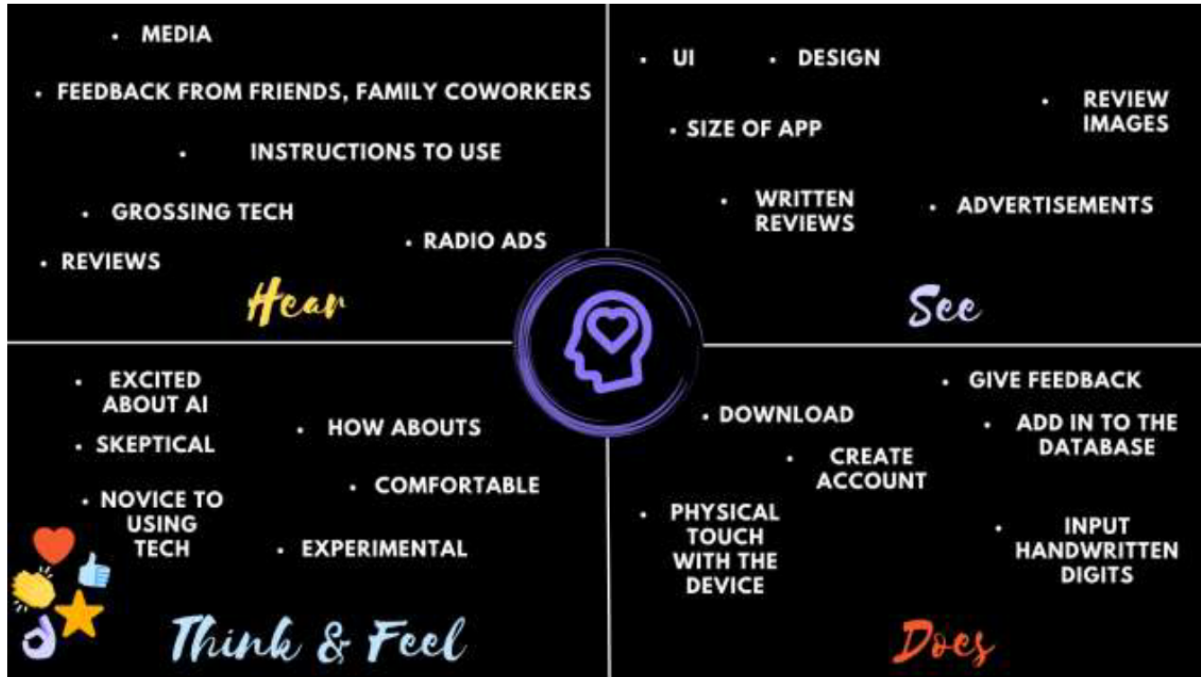
						10,000 images. The KNN algorithm describes categorical value by making use of majority of votes of K - nearest neighbors, the K value used to differ here.
--	--	--	--	--	--	---

2.3) Problem Statement Definition

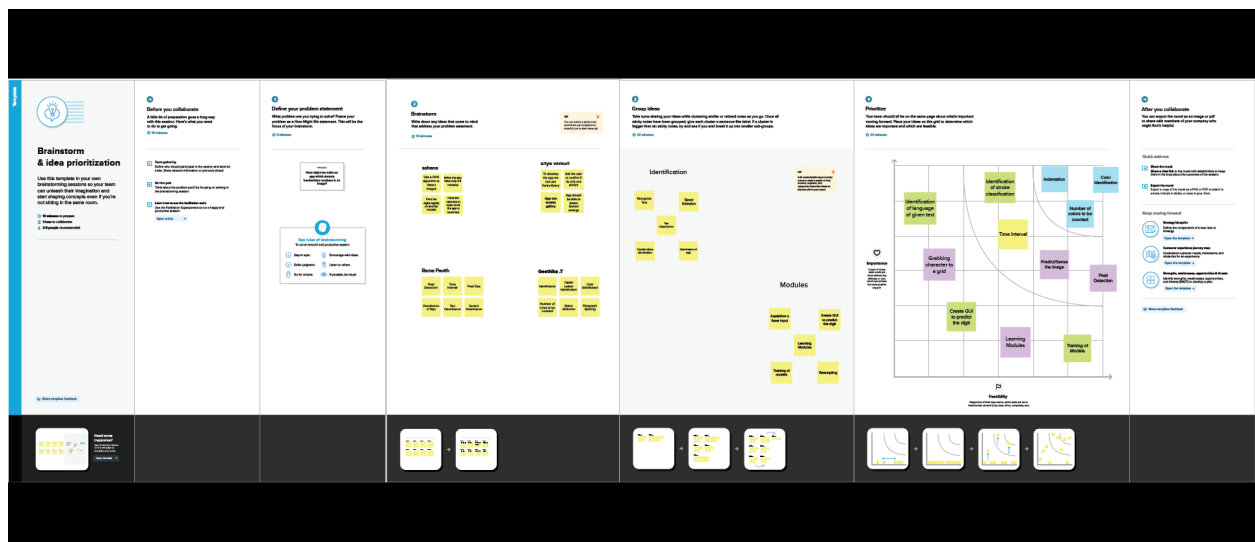
Handwriting recognition is one of the compelling research works going on because every individual in this world has their own style of writing. It is the capability of the computer to identify and understand handwritten digits or characters automatically. Because of the progress in the field of science and technology, everything is being digitalized to reduce human effort. Hence, there comes a need for handwritten digit recognition in many realtime applications. MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. We use Artificial neural networks to train these images and build a deep learning model. Web application is created where the user can upload an image of a handwritten digit. this image is analyzed by the model and the detected result is returned to UI(User Interface).

3) Ideation and Proposed Solution

3.1) Empathy map Canvas



3.2) Ideation & Brain Storming



3.3) Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To design an application that identifies numbers from written script.
2.	Idea / Solution description	Using the concepts of neural networks, an application will be created. From an image the user gives as input the application will identify the numbers in it and display it.
3.	Novelty / Uniqueness	<p>We will provide the options to give mathematical operations from which they can choose. Accordingly the result will be displayed.</p> <p>Our application will only recognize digits and not all text like OCR.</p>
4.	Social Impact / Customer Satisfaction	To correctly and accurately identify handwritten scriptures. This is to ease people when they don't understand something that's written somewhere. For example a prescription given by a doctor.

5.	Business Model (Revenue Model)	<p>This application can be used in the world as well in places like</p> <p>i. Traffic surveillance - To make note of the license plate of vehicles.</p> <p>This application can be improved upon to solve complicated mathematical equations.</p>
6.	Scalability of the Solution	It will be a very scalable application as it will be generic and easily adaptable.

3.4) Problem Solution fit

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS The Bank Employee who makes the transactions through the cheque.	6. CUSTOMER CONSTRAINTS CC External dependencies are quite expensive and it is not offered by the people, So this process overcome the problem through their installation in mobile.	5. AVAILABLE SOLUTIONS AS --- Automatic digit recognition --- In past, people identify the digits to their analysis sometimes it causes wrong transactions. --- By using this application, they could easily identify the digits	Explore AS, differentiate
Focus on J&P, tap into BE, understand RC	2. JOBS-TO-BE-DONE / PROBLEMS J&P Every single has their own style of writing which could not recognize by the computer.	9. PROBLEM ROOT CAUSE RC Every single has their own style of writing which could not recognize by the computer.	7. BEHAVIOUR BE To classify the digits in correct way, they could make the transactions easier without any doubtfulness.	Focus on J&P, tap into BE, understand RC
Identify strong TR & EM	3. TRIGGERS TR Feel free to make transactions without any fear about their style of writing	10. YOUR SOLUTION SL --CNN model could be used to provide very High accuracy in image recognition problems and also reduces the high dimensionality of the images, without losing its information. --It can be used to convert the handwritten digits to machine readable format.	8. CHANNELS OF BEHAVIOUR CH ONLINE: Promoting this application through the mobiles, the transaction could be done at any place without the presence in bank. OFFLINE: The identification of the digits which is in the handwritten form directly captured by using mobile application and that could be used to convert the those digits into machine readable forms.	Extract online & offline CH of BE
	4. EMOTIONS: BEFORE / AFTER EM If the person faces a problem regarding the transactions they could confidently handle the situation by using handwritten digit recognition system			

4) Requirement analysis

Functional and Non functional requirements

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	The product essentially converts handwritten digits to digital form.	The user is first asked to draw a number on the canvas, and the model that is built is then utilised to compare the data and provide an output in digitalized form.
FR-2	Recognizing the handwritten digit and displaying.	Recognizing the handwritten digit and displaying.
FR-3	Import dataset file directly to the program from a command that will download the dataset from its website. Save the dataset file	Installing packages and applications.

	in the same directory as the program	
FR-4	Build a Neural Network with a number of nodes in the input layer equal to the number of pixels in the arrays	Nil
FR-5	Activating the Neural Network	Packages – tensorflow

4.2) Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	System design should be easily understood and user friendly to users. Furthermore, users of all skill levels of users should be able to navigate it without problems.
NFR-2	Security	The system should automatically be able to authenticate all users with their unique username and password

NFR-3	Performance	Should reduce the delay in information when hundreds of requests are given.
NFR-4	Availability	Information is restricted to each users limited access
NFR-5	Scalability	the system should be able to handle 10000 users accessing the site at the same time

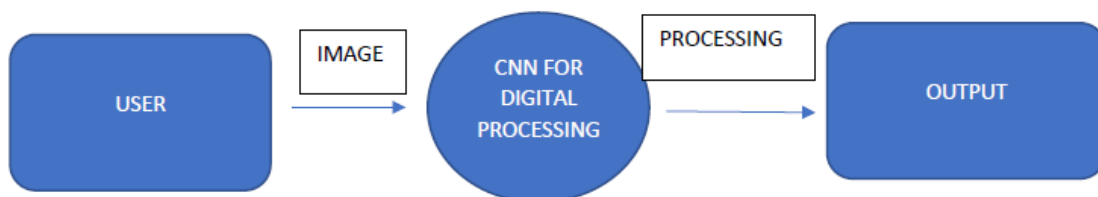
5) Project Design

5.1) Data flow diagrams

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.

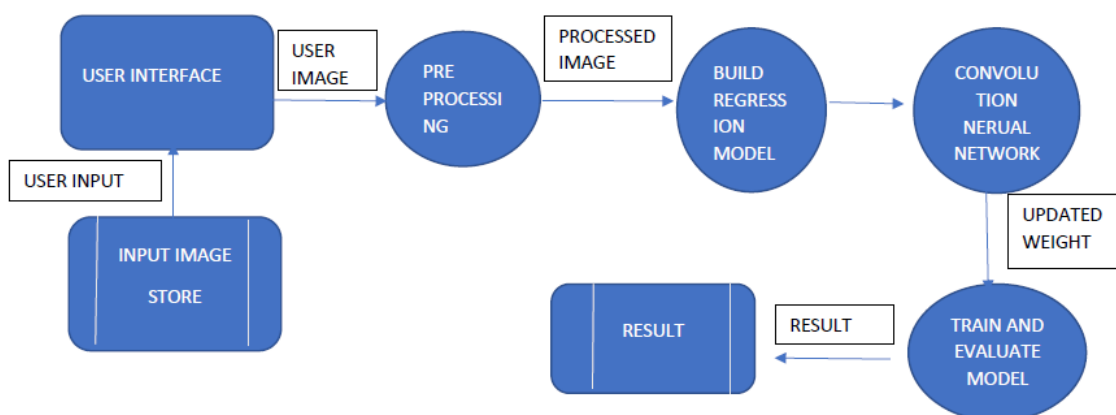
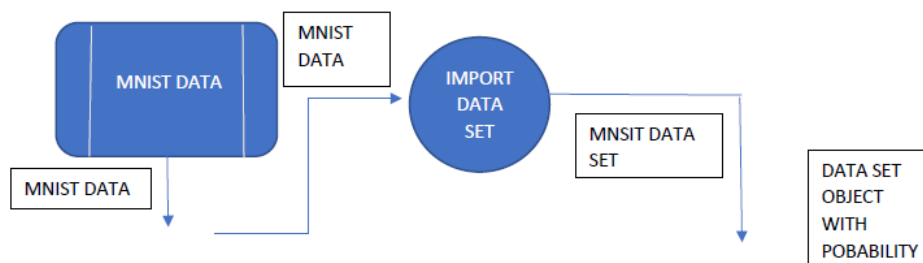
DFD Level-0

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.



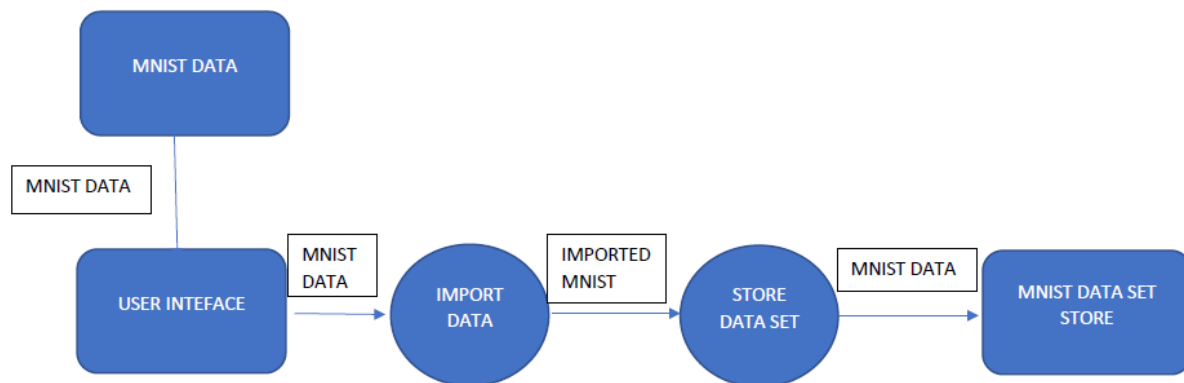
DFD Level-1

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.



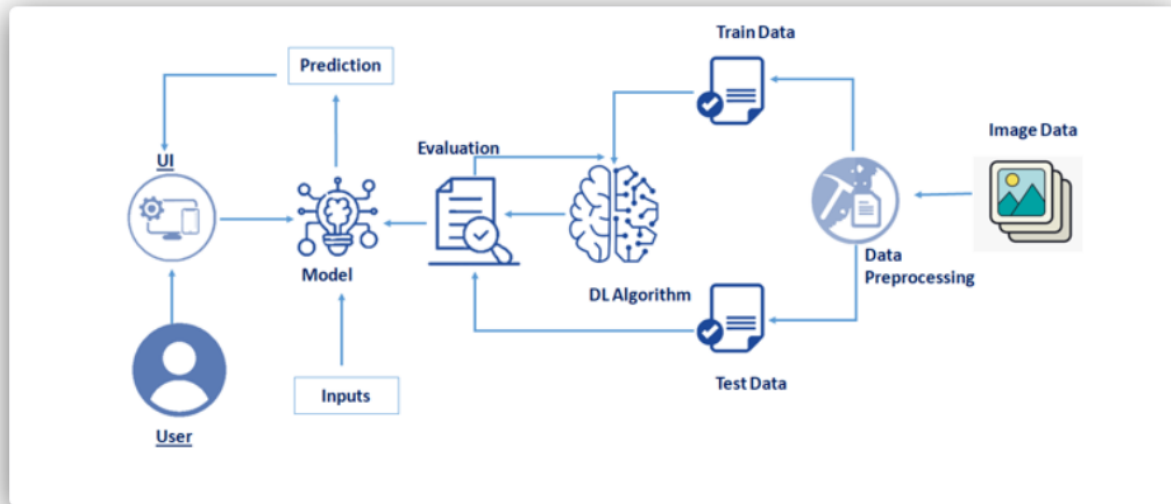
5.2) Solution & Technical architecture

Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Technical Architecture:



5.3) User Stories

USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-2
		USN-3	As a user, I can register for the application through gmail or facebook	I can register & access the dashboard with Facebook Login	Medium	Sprint-2
	Login	USN-4	As a user, I can log into the application by entering email & password	I can login to the application	High	Sprint-1
	Dashboard	USN-5	Go to dashboard and refer the content about our project	I can read instructions also and the home page is user-friendly.	Low	Sprint-1
	Upload Image	USN-6	As a user, I can able to input the images of digital documents to the application	As a user, I can able to input the images of digital documents to the application	High	Sprint-3
	Predict	USN-7	As a user I can able to get the recognised digit as output from the images of digital documents or images	I can access the recognized digits from digital document or images	High	Sprint-3
		USN-8	As a user, I will train and test the input to get the maximum accuracy of output.	I can able to train and test the application until it gets maximum accuracy of the result.	Medium	Sprint-4
Customer (Web user)	Login	USN-9	As a user, I can use the application by entering my email, password.	I can access my account	Medium	Sprint-4
Customer Care Executive	Dashboard	USN-10	upload the image	Recognize and get the output	High	Sprint-1
Administrator	Security	USN-11	updated the features	checking the security	Medium	Sprint-1

6) Project Planning & Scheduling:

6.1) Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Rampavith Kopurapu, HSJ Sahana, Sai Sriya, Thotha Geethika Sree
Sprint-2	Login	USN-2	As a user, I can log into the application by entering email & password	1	High	Rampavith Kopurapu, HSJ Sahana, Sai Sriya, Thotha Geethika Sree
	Dashboard	USN-3	As a user, I can view activities available in home page	2	Medium	Rampavith Kopurapu, HSJ Sahana, Sai Sriya, Thotha Geethika Sree

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Dashboard Customization	USN-4	Further and final work on home page.	2	High	Rampavith Kopurapu, HSJ Sahana, Sai Sriya, Thotha Geethika Sree
	Input image feature	USN-5	As a user, I can input image	2	High	Rampavith Kopurapu, HSJ Sahana, Sai Sriya, Thotha Geethika Sree
Sprint-4	Result Display	USN-6	As a user, I will be able to view the calculated results	2	High	Rampavith Kopurapu, HSJ Sahana, Sai Sriya, Thotha Geethika Sree

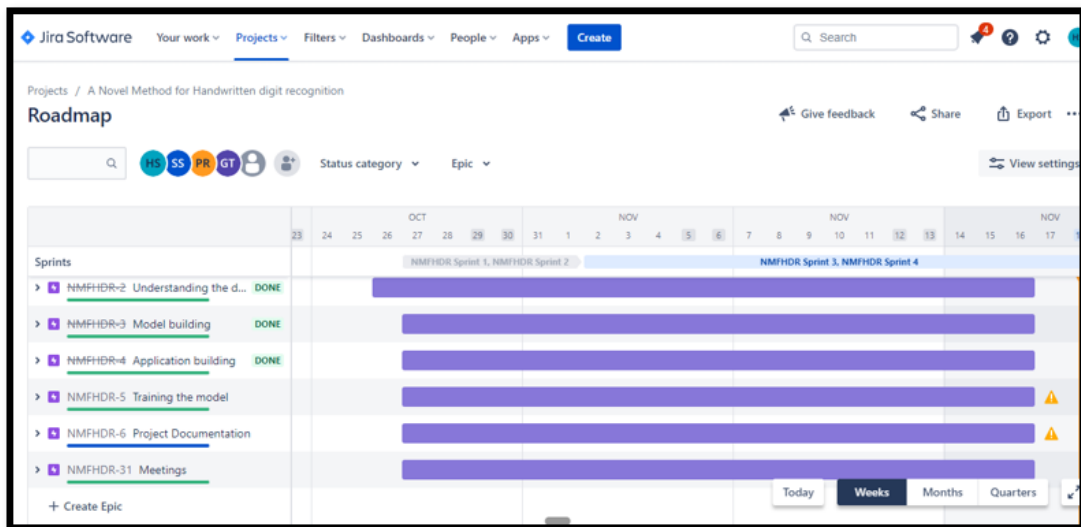
6.2) Sprint Delivery Schedule

Details:

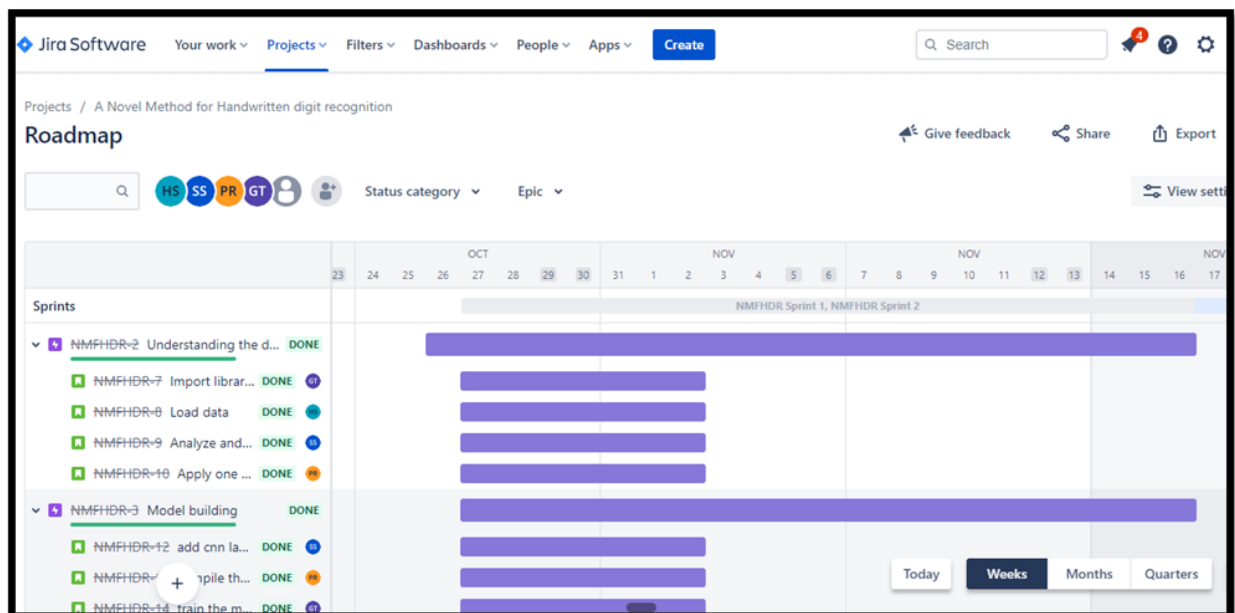
Project Development Start Date: 26/10/2022

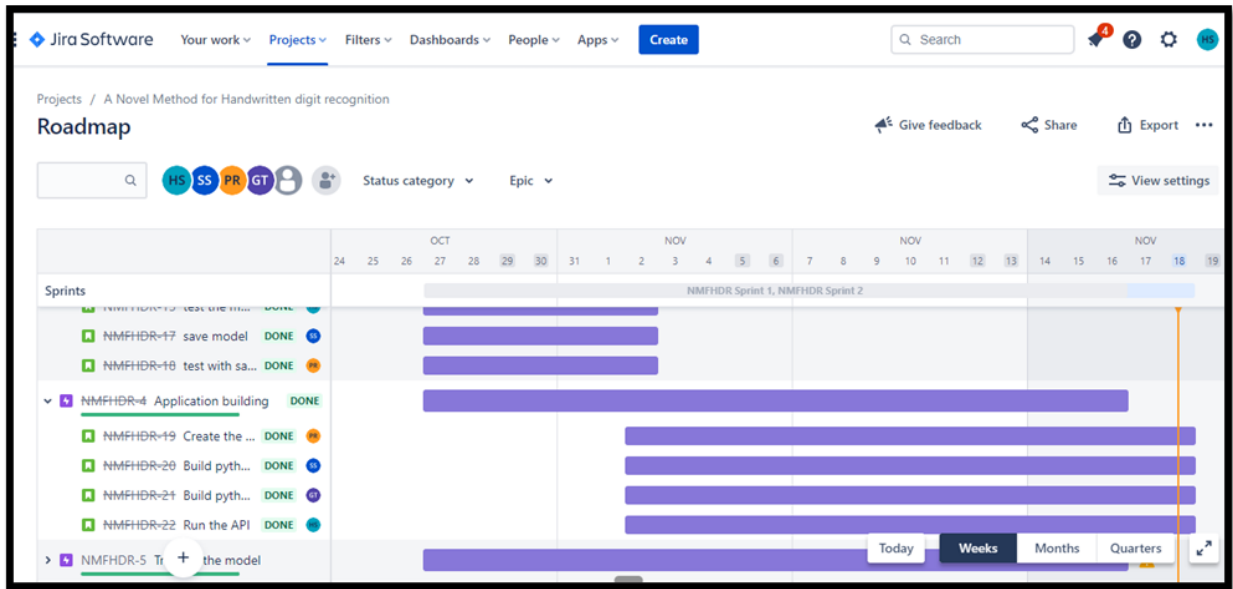
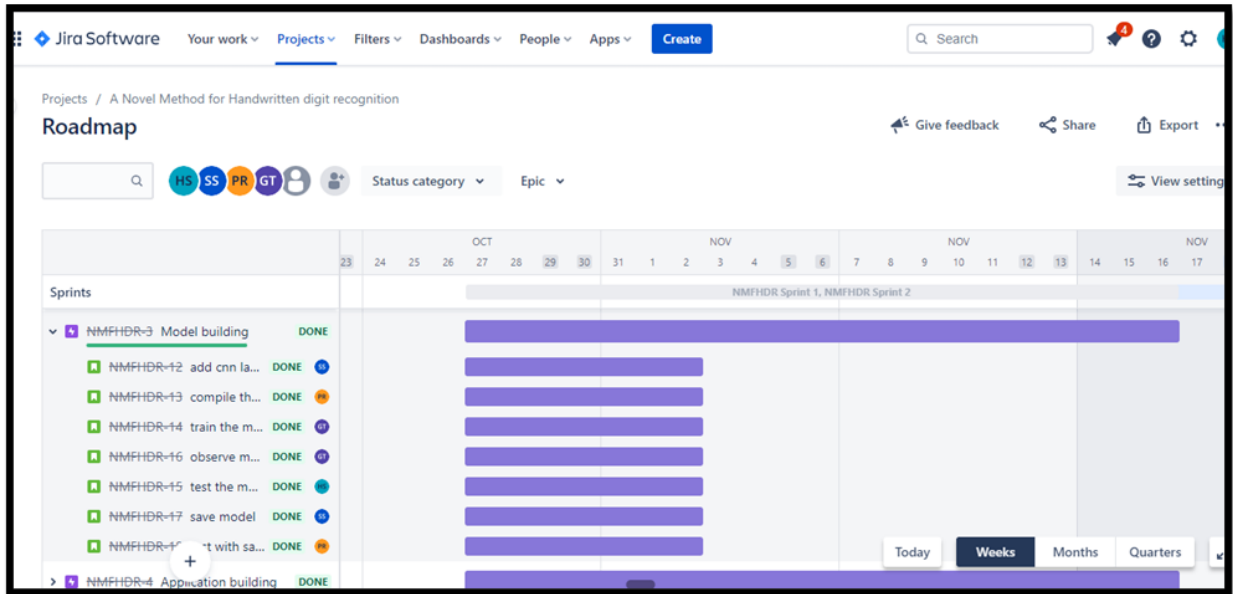
Project Estimated Completion Date: 16/11/2022

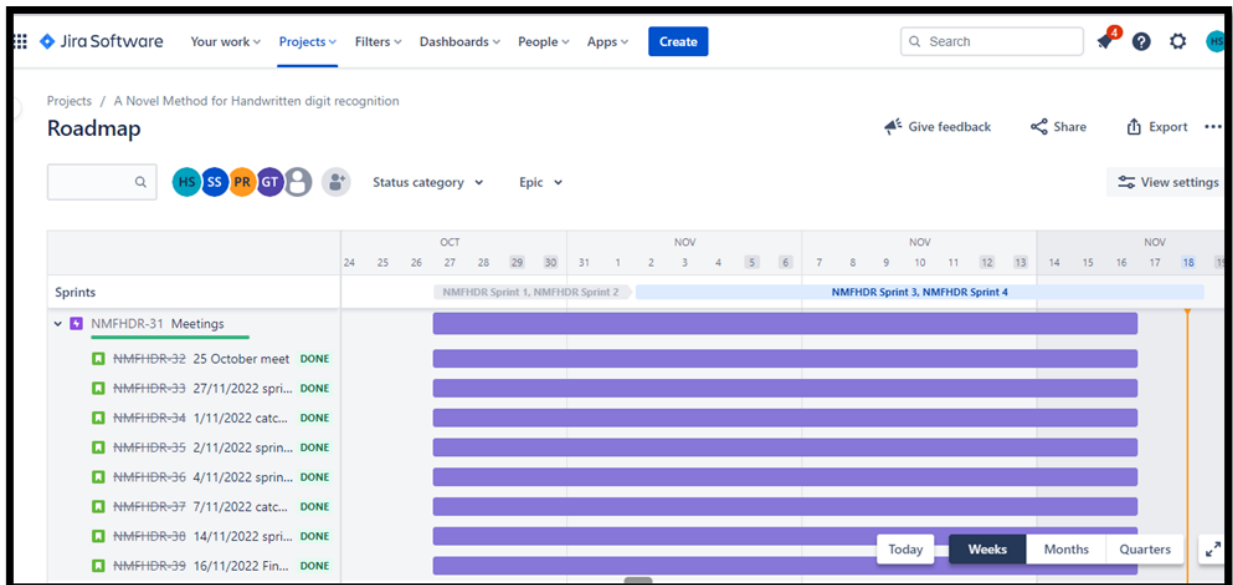
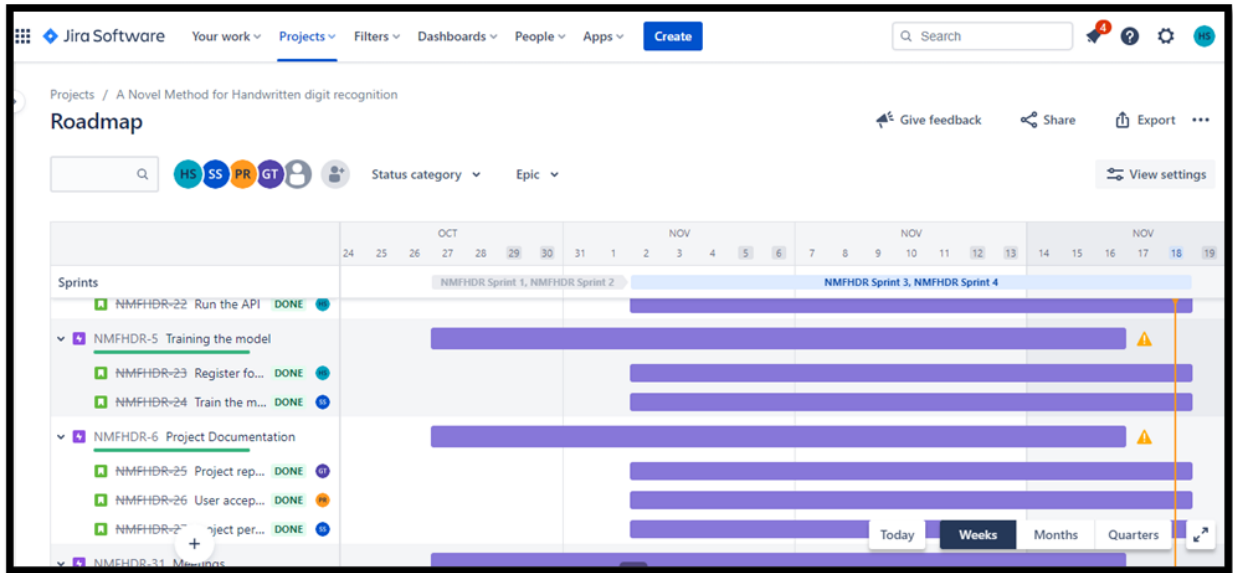
Below is a snapshot of the modules and time allotted for each.



6.3) Reports from JIRA







NMFHDR-2



1



Understanding the data



Done



Done

Description

We have loaded and understood what is in the data. The data consists of images of numbers in the X set and the integer values of the images in the y set.

HS

Add a comment...

Pro tip: press **M** to comment

NMFHDR-2



1



The code is attached:-

Attachments (1)



Understanding T...a.ipynb
16 Nov 2022, 02:16 PM



 NMFHDR-3



Model building



Done ▾

✓ Done

Description

In this step we created the model. We trained the data and saved our model. Using this model that we created we've tested it on random data as well To note its accuracy.

NMFHDR-3



1



saved our model. Using this model that we created we've tested it on random data as well To note its accuracy.

Attachments (1)



A Novel Method... g.ipynb
16 Nov 2022, 02:25 PM



 NMFHDR-4



Application building



Done ▾

✓ Done

Description

The application is built and run.

NMFHDR-4



1




Child issues






Order by




100% Done




- | | | | | | |
|--|---------------------------|------------------------|---|----|--------|
| | NMFHDR-19 | Create the front en... | 3 | PR | DONE ✓ |
| | NMFHDR-20 | Build python code 1 | 4 | SS | DONE ✓ |
| | NMFHDR-21 | Build python code 2 | 4 | GT | DONE ✓ |
| | NMFHDR-22 | Run the API | 4 | HS | DONE ✓ |

 NMFHDR-5

  1   ... 




Training the model






   ...

To Do ▾

Description

Model is trained.

 NMFHDR-5

  1   ... 



Description



Model is trained.

Child issues

Order by ▾ ... +

100% Done

 **NMFHDR-23** Register for IBM cl... 2  **DONE ▾**

 **NMFHDR-24** Train the model o... 4  **DONE ▾**

 NMFHDR-6



1



Project Documentation



In Progress ▾

Description

Data accumulated & documented.

Child issues

Order by ▾



NMFHDR-31

1

Meetings

Done

✓ Done

Description

conduct all meetings and update in jira.

NMFHDR-31

1

Child issues

Order by

100% Done

NMFHDR-32

25 October meet

3

DONE

NMFHDR-33

27/11/2022 sprint ...

3

DONE

NMFHDR-34

1/11/2022 catch u...

3

DONE

NMFHDR-35

2/11/2022 sprint 2...

3

DONE



7) Coding & Solutioning

7.1) Feature 1:

This is the index or the home page of the web application which describes what this web application is for and how to use this web application to predict the numbers. There is an option where we can choose a file and upload the file and click on the 'Predict' option to predict the number.

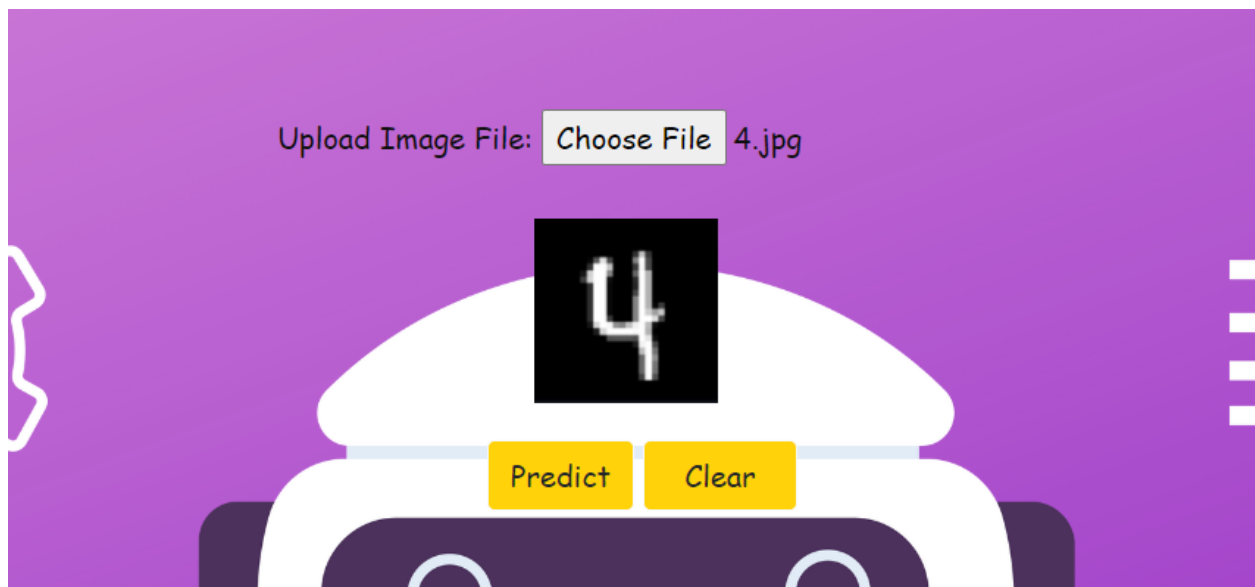
Handwritten Digit Recognition

This application will recognise hand written digits in pictures and print the result. The model used here is Convolutional Neural Networks.

Instructions:

1. Choose your file that is image.
2. Press the predict button
3. The digit will be printed on the next screen!!

There is also one more option 'Clear', next to 'Predict' which can be used to clear the current image and upload a new one.



6) Project Planning and Scheduling:

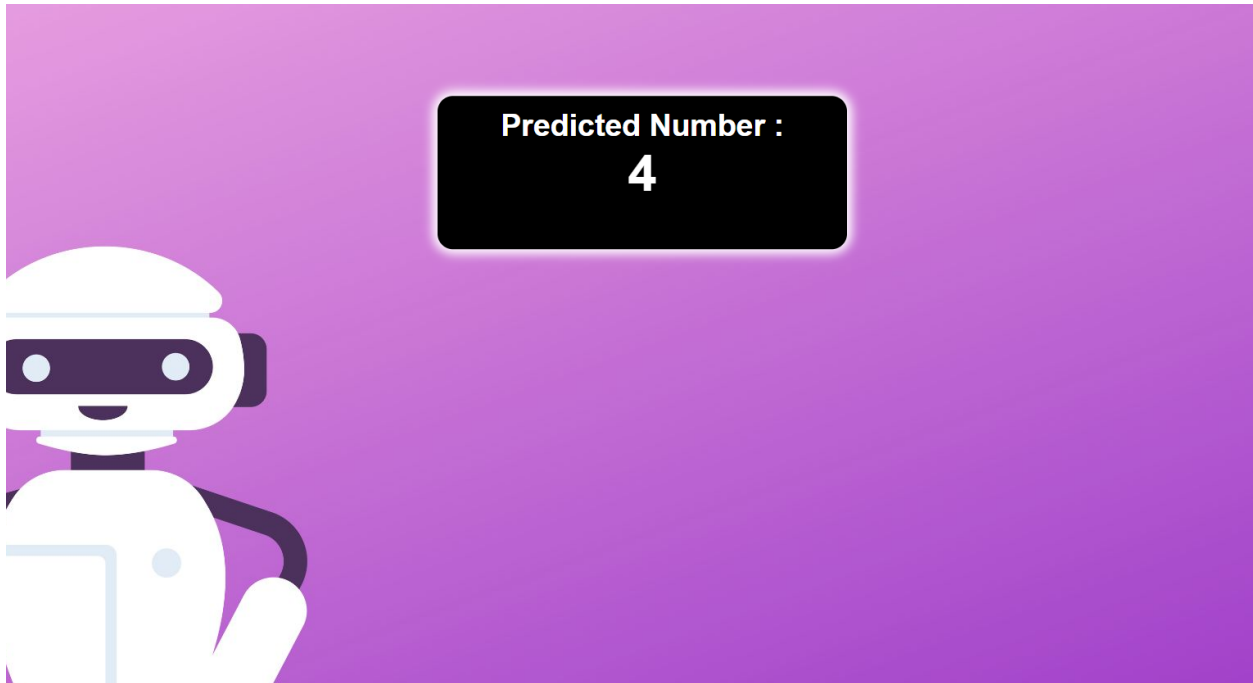
6.1) Sprint Planning and estimation:

6.2) Sprint Delivery Schedule:

6.3) Reports from Jira

7.2) Feature 2:

An image of a handwritten digit is uploaded into the web application and then by clicking on predict, it leads to opening of a new web page where the predicted number by the model is displayed.



Codes

Code for the required features is written in the files “index.html”, “predict.html”, “style.css” and “app.py”

The “index.html” file renders the home page which gives the information about prediction and the ‘predict’ and ‘clear’ options.

“index.html” code:

```
<html>
```

```
<head>
```

```
  <title>HDR</title>
```

```
  <meta name="viewport" content="width=device-width">
```

```
  <link rel="stylesheet" href="../static/css/style.css">
```

```
<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=swap" rel="stylesheet">
<link
href="https://fonts.googleapis.com/css?family=Calistoga|Josefin+Sans:400,700|Pacifico&display=swap" rel="stylesheet">

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
<link rel="stylesheet" type="text/css" href="{{
url_for('static',filename='css/style.css') }}">

<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+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>
```

```

    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.
css">
    <script
src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></
script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"
></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle
.min.js"></script>

</head>
<style>
    body{
        background-image: url('../static/images/1.png');
        background-repeat: no-repeat;
        background-size: cover;
    }
</style>

<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>
    <br><br><br><br><br><br><br>
    <div class="container p-3 my-3 text-dark" id="inst">

```


<p>This application will recognise hand written digits in pictures and print the result. The model used here is Convolutional Neural Networks.

</p>

<p>Instructions:

Choose your file that is image.

Press the predict button

The digit will be printed on the next screen!!

</p>

</div>

<section id="content">

<div class="leftside">

<form action="/predict" method="POST"
enctype="multipart/form-data">

<label>Upload Image File: </label>

<input id="image" type="file" name="image" accept="image/png,
image/jpeg" onchange="preview()">

<div class="buttons_div">

<button type="submit" class="btn btn-light"
id="bt1">Predict</button>

<button type="button" class="btn btn-light" id="bt1"> Clear </button>

</div>

</form>

</div>

</section>

</body>

</html>

The “predict.html” file renders the page where the home page leads to when the user clicks the ‘predict’ option. On this page, the predicted number is shown.

“predict.html” code:

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

<style>
  body{
    background-image: url('../static/images/3.jpg');
    background-repeat: no-repeat;
    background-size: cover;
  }

  #rectangle{
    width:400px;
    height:150px;
    background-color: #000000;
    border-radius: 15px;
    position:absolute;
    box-shadow: 0px 0px 10px 5px white;
    top:25%;
    left:50%;
    transform:translate(-50%,-50%);
  }

  #head{
    text-align: center;
    font-size: 30px;
    margin: 0 auto;
    padding: 3% 5%;
    font-family: Arial, Helvetica, sans-serif;
    color: white;
  }

  #num{
    font-size: 50px;
  }
```

```

</style>

<body>

    <div id="rectangle">
        <h1 id="head">Predicted Number : <br><center
id="num">{{num}}</center></h1>
    </div>

</body>
</html>

```

The “style.css” file contains the code for styling the index page and the predict page.

“style.css” code:

```

#clear_button{
    margin-left: 15px;
    font-weight: bold;
    color: rgb(0, 174, 255);
}

#inst{
    color : rgb(172, 167, 167);
    background-color:rgb(250, 214, 250);
    font-family:URW Chancery L, cursive;
    font-weight: bold;
}

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

#content{
    color: rgb(15, 15, 15);
    margin: 0 auto;
    padding: 2% 15%;
}

```

```
padding-bottom: 0;
font-family: URW Chancery L, cursive;
}

.welcome{
    text-align: center;
    position: relative;
    color: rgb(253, 253, 253);
    background-color: skyblue;
    padding-top: 1%;
    padding-bottom: 1%;
    font-weight: bold;
    font-family: 'Bookman', 'URW Bookman L', serif;
}

#team_id{
    text-align: right;
    font-size: 25px;
    padding-right: 3%;
}

#predict_button{
    margin-right: 15px;
    color: rgb(0, 255, 72);
    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: rgb(250, 214, 250);
    border: 1px solid #888;
    margin-top: 20px;
    margin-bottom: 20px;
}

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

#bt1{
    background-color: rgb(255, 210, 10);
}

#bt1:hover{
    background-color: white;
}

.heading{
    font-family: "American Typewriter", 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%; */
}

h1{
    text-align: center;
    color:black;
    padding: 100px 50px 65px 100px;
}

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

```

The app.py contains the backend code and also integrates front end with backend.

“app.py” code:

```

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 = r"C:\Users\Welcome\Desktop\Sem 7\IBM\Project\Application
Building\data"

```

```

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

model = load_model("./models/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)
    app.debug = True
    app.run()

```

8) Testing

8.1) Test Cases

				Date	9-Nov-22								
				Team ID	PNT 2022/H065257								
				Project Name	Project - A Novel Method for Handwritten Digit Recognition								
				Maximum Marks	4 marks								
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
IndexPage_TC_001	Functional	Home Page	Verify user is able to see the options to choose their file and able to press the predict button	Python flask and necessary libraries installed on the system	1.Run application on command prompt 2.Enter 127.0.0.1:5000 as URL 3.Verify if option to Choose Files and predict is given or not	python-m flask run	Choose File and predict buttons should be visible	Working as expected	Pass		Y		Sa Sriya
Index&PredictPage_TC_002	UI	Home Page	Verify the UI elements in Index page and Predict.html page	Python flask and necessary libraries installed on the system	1.Open both the HTML pages 2.Check if the UI is correct and as imagined. 3.Verify with below UI elements a.Heading b.A small description c.Choose file button d.Predict button e. Clear button f. In a box the number should be predicted in predict page	Index.html & predict.html	Application should show below UI elements a.Heading b.A small description c.Choose file button d.Predict button e. Clear button f. In a box the number should be predicted in predict page	Working as expected	Pass	A clean and neat UI	Y		Sahana HJ
IndexPage_TC_003	Functional	Home page	Verify user is able to attach their image to the application with no error	Python flask and necessary libraries installed on the system	1.Enter URL 127.0.0.1:5000 after running on command prompt 2.Click on Choose File 3.Choose picture and press open	1.png	The image should be displayed in the box	No image displayed	Fail	There might be an error in HTML code	N	BUG 01	Geethika Sree
IndexPage_TC_004	Functional	Home page	Verify user is able to attach their image to the application with no error	Python flask and necessary libraries installed on the system	1.Enter URL 127.0.0.1:5000 after running on command prompt 2.Click on Choose File 3.Choose picture and press open	1.png	The image should be displayed in the box	Image displayed	Pass	Bug fixed image is displaying properly	Y		Geethika Sree
IndexPage_TC_005	Functional	Home page	After uploading image verify if predicted value is getting displayed	Python flask and necessary libraries installed on the system	1.Enter URL 127.0.0.1:5000 after running on command prompt 2.Click on Choose File 3.Choose picture and press open 4.Press Predict	1.png	On pressing predict should redirect to predict and display the predicted value	Value displayed	Pass	Model is working properly application is successful	Y		Ramaparth

8.2) User Acceptance testing

Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the A Novel Method for Handwritten Digit Recognition System project at the time of the release to User Acceptance Testing (UAT).

Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	2	4	7	10	23
Duplicate	1	0	1	2	4
External	2	3	0	1	6
Fixed	6	2	4	20	32
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	11	14	16	35	76

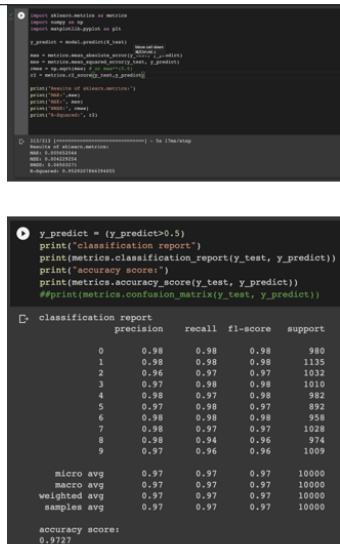
Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	80	0	0	80
Security	2	0	0	2
Outsource Shipping	4	0	0	4
Exception Reporting	10	0	0	10
Final Report Output	6	0	0	6
Version Control	3	0	0	3

9) Results

9.1) Performance Metrics

S.No.	Parameter	Values	Screenshot																																																																																
1.	Metrics	<p>Regression Model: MAE - 0.005652044, MSE - 0.004229254, RMSE - 0.06503271, R2 score - 0.9529207866394055</p> <p>Classification Model: Confusion Matrix - , Accuracy Score- 0.9727 & Classification Report</p>	 <pre># Import relevant metrics to measure import numpy as np import matplotlib.pyplot as plt # Predictions y_predict = model.predict(X_test) # Regression Metrics mse = metrics.mean_squared_error(y_test, y_predict) rmse = metrics.sqrt(mse) mae = metrics.mean_absolute_error(y_test, y_predict) r2 = metrics.r2_score(y_test, y_predict) print('Mean Squared Error: %f' % mse) print('Root Mean Squared Error: %f' % rmse) print('Mean Absolute Error: %f' % mae) print('R-squared: %f' % r2) # Confusion Matrix and Accuracy Score y_test = y_test.astype(int) y_predict = y_predict.astype(int) cm = confusion_matrix(y_test, y_predict) # Print Confusion Matrix print('Confusion Matrix:') print(cm) # Print Accuracy Score accuracy_score = metrics.accuracy_score(y_test, y_predict) print('Accuracy Score: %f' % accuracy_score)</pre> <pre>y_predict = (y_predict>0.5) print("classification report") print(metrics.classification_report(y_test, y_predict)) print("accuracy score:") print(metrics.accuracy_score(y_test, y_predict)) #print(metrics.confusion_matrix(y_test, y_predict))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>0.98</td><td>0.98</td><td>0.98</td><td>980</td></tr><tr><td>1</td><td>0.98</td><td>0.98</td><td>0.98</td><td>1135</td></tr><tr><td>2</td><td>0.96</td><td>0.97</td><td>0.97</td><td>1032</td></tr><tr><td>3</td><td>0.97</td><td>0.98</td><td>0.98</td><td>1010</td></tr><tr><td>4</td><td>0.98</td><td>0.97</td><td>0.98</td><td>982</td></tr><tr><td>5</td><td>0.97</td><td>0.98</td><td>0.97</td><td>892</td></tr><tr><td>6</td><td>0.98</td><td>0.98</td><td>0.98</td><td>958</td></tr><tr><td>7</td><td>0.98</td><td>0.97</td><td>0.97</td><td>1028</td></tr><tr><td>8</td><td>0.98</td><td>0.94</td><td>0.96</td><td>974</td></tr><tr><td>9</td><td>0.97</td><td>0.96</td><td>0.96</td><td>1009</td></tr><tr><td>micro avg</td><td>0.97</td><td>0.97</td><td>0.97</td><td>10000</td></tr><tr><td>macro avg</td><td>0.97</td><td>0.97</td><td>0.97</td><td>10000</td></tr><tr><td>weighted avg</td><td>0.97</td><td>0.97</td><td>0.97</td><td>10000</td></tr><tr><td>samples avg</td><td>0.97</td><td>0.97</td><td>0.97</td><td>10000</td></tr><tr><td>accuracy score:</td><td colspan="4">0.9727</td></tr></tbody></table>		precision	recall	f1-score	support	0	0.98	0.98	0.98	980	1	0.98	0.98	0.98	1135	2	0.96	0.97	0.97	1032	3	0.97	0.98	0.98	1010	4	0.98	0.97	0.98	982	5	0.97	0.98	0.97	892	6	0.98	0.98	0.98	958	7	0.98	0.97	0.97	1028	8	0.98	0.94	0.96	974	9	0.97	0.96	0.96	1009	micro avg	0.97	0.97	0.97	10000	macro avg	0.97	0.97	0.97	10000	weighted avg	0.97	0.97	0.97	10000	samples avg	0.97	0.97	0.97	10000	accuracy score:	0.9727			
	precision	recall	f1-score	support																																																																															
0	0.98	0.98	0.98	980																																																																															
1	0.98	0.98	0.98	1135																																																																															
2	0.96	0.97	0.97	1032																																																																															
3	0.97	0.98	0.98	1010																																																																															
4	0.98	0.97	0.98	982																																																																															
5	0.97	0.98	0.97	892																																																																															
6	0.98	0.98	0.98	958																																																																															
7	0.98	0.97	0.97	1028																																																																															
8	0.98	0.94	0.96	974																																																																															
9	0.97	0.96	0.96	1009																																																																															
micro avg	0.97	0.97	0.97	10000																																																																															
macro avg	0.97	0.97	0.97	10000																																																																															
weighted avg	0.97	0.97	0.97	10000																																																																															
samples avg	0.97	0.97	0.97	10000																																																																															
accuracy score:	0.9727																																																																																		

10) Advantages and 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

11) Conclusion

The performance of the CNN for handwritten recognition performed significantly. The proposed method obtained around 97.27% accuracy and is able to identify real-world images as well; the loss percentage obtained in both training and evaluation is almost negligible. The only difficult part is the noise present in the input canvas image, which needs to be taken care of. The learning rate of the model is much dependent on the number of dense neurons and the cross-validation measure.

12) Future Scope

The task of handwritten digit recognition, using a classifier, has great importance and use such as – online handwriting recognition on computer tablets, recognize zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand (for example - tax forms) and so on.

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.

13) Appendix

Source Code:

Firstly the model has been trained and saved as “mnistCNN.h5” and is used in the “app.py” file where everything is integrated. Following are the codes of all the files for the project.

The model (saved as mnistCNN.h5):

Importing Libraries

```
In [1]: import numpy
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 keras.layers import Convolution2D as Conv2D
from keras.optimizers import Adam
from keras.utils import np_utils
```

Loading Data

```
In [2]: (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
```

```
In [ ]: print(X_train.shape)
print(X_test.shape)

(60000, 28, 28)
(10000, 28, 28)
```

Analyzing the data

```
[3]: X_train[0]
```

```
Out[3]: 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,  0,
 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,  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, 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, 80, 156, 107, 253, 253,
205, 11,  0, 43, 154,  0,  0,  0,  0,  0,  0,  0,  0,
  0,  0]
```

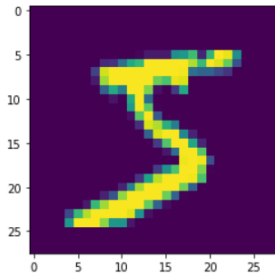
```
0, 0]], dtype=uint8)
```

```
In [4]: y_train[0]
```

```
Out[4]: 5
```

```
In [5]: import matplotlib.pyplot as plt
plt.imshow(X_train[0])
```

```
Out[5]: <matplotlib.image.AxesImage at 0x7f34a90fb3d0>
```



#Reshaping the data

```
In [6]: X_train = X_train.reshape(60000,28,28,1).astype('float32')
X_test = X_test.reshape(10000,28,28,1).astype('float32')
```

One-hot Encoding

```
In [7]: num_of_classes = 10
y_train = np_utils.to_categorical(y_train,num_of_classes)
y_test = np_utils.to_categorical(y_test,num_of_classes)
```

```
In [8]: y_train[0]
```

```
Out[8]: array([0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)
```

Model Building

Creating Model (Add CNN Layers)

```
In [9]: 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(num_of_classes,activation='softmax'))
```

Compiling the model

```
In [10]: model.compile(loss='categorical_crossentropy',optimizer="Adam",metrics=['accuracy'])
```

Fitting the model

```
In [11]: model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=5,batch_size=32)

Epoch 1/5
1875/1875 [=====] - 124s 66ms/step - loss: 0.2578 - accuracy: 0.9482 - val_loss: 0.1180 - val_accu
acy: 0.9652
Epoch 2/5
1875/1875 [=====] - 124s 66ms/step - loss: 0.0778 - accuracy: 0.9769 - val_loss: 0.0907 - val_accu
acy: 0.9739
Epoch 3/5
1875/1875 [=====] - 126s 67ms/step - loss: 0.0549 - accuracy: 0.9829 - val_loss: 0.0921 - val_accu
acy: 0.9734
Epoch 4/5
1875/1875 [=====] - 122s 65ms/step - loss: 0.0421 - accuracy: 0.9865 - val_loss: 0.1090 - val_accu
acy: 0.9707
Epoch 5/5
1875/1875 [=====] - 122s 65ms/step - loss: 0.0315 - accuracy: 0.9898 - val_loss: 0.1161 - val_accu
acy: 0.9731

Out[11]: <keras.callbacks.History at 0x7f34a4958790>
```

Observing the metrics

```
In [28]: project_metrics=model.evaluate(X_test,y_test,verbose=0)
print("Metrics(Loss& Accuracy)")
print(project_metrics)

Metrics(Loss& Accuracy)
[0.11606404930353165, 0.9731000065803528]
```

```
In [27]: import sklearn.metrics as metrics
import numpy as np
import matplotlib.pyplot as plt

y_predict = model.predict(X_test)

mae = metrics.mean_absolute_error(y_test, y_predict)
mse = metrics.mean_squared_error(y_test, y_predict)
rmse = np.sqrt(mse) # or mse**(0.5)
r2 = metrics.r2_score(y_test,y_predict)

print("Results of sklearn.metrics:")
print("MAE:",mae)
print("MSE:", mse)
print("RMSE:", rmse)
print("R-Squared:", r2)

313/313 [=====] - 5s 17ms/step
Results of sklearn.metrics:
MAE: 0.005652044
MSE: 0.004229254
RMSE: 0.06503271
R-Squared: 0.9529207866394055
```

Predicting the model

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

1/1 [=====] - 0s 66ms/step
[[5.4520988e-12 1.0888041e-21 7.5880036e-13 1.2055289e-11 2.6268963e-19
 3.1448650e-15 1.1666435e-19 1.0000000e+00 2.1508427e-11 2.9282635e-11]
 [4.4279619e-12 1.2889054e-12 1.0000000e+00 2.0362232e-17 5.6934279e-20
 5.3493605e-19 4.8295710e-13 5.0981827e-21 1.0086084e-15 1.9516000e-19]
 [9.2632938e-08 9.9921167e-01 6.2528164e-07 4.8549702e-11 7.8550651e-04
 1.5237079e-07 6.6308399e-09 1.0095750e-09 1.9272488e-06 1.2933888e-12]
 [1.0000000e+00 1.6884626e-16 2.7246841e-10 5.4617701e-15 1.6128288e-14
 1.5063359e-10 5.7584715e-10 9.0131895e-14 7.9959858e-11 2.1980879e-08]]
```

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

```
Metrics(Loss& Accuracy)
[0.09232578426599503, 0.9797999858856201]
```

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

1/1 [=====] - 0s 48ms/step
[[5.4520988e-12 1.0888041e-21 7.5880036e-13 1.2055289e-11 2.6268963e-19
 3.1448650e-15 1.1666435e-19 1.0000000e+00 2.1508427e-11 2.9282635e-11]
 [4.4279619e-12 1.2889054e-12 1.0000000e+00 2.0362232e-17 5.6934279e-20
 5.3493605e-19 4.8295710e-13 5.0981827e-21 1.0086084e-15 1.9516000e-19]
 [9.2632938e-08 9.9921167e-01 6.2528164e-07 4.8549702e-11 7.8550651e-04
 1.5237079e-07 6.6308399e-09 1.0095750e-09 1.9272488e-06 1.2933888e-12]
 [1.0000000e+00 1.6884626e-16 2.7246841e-10 5.4617701e-15 1.6128288e-14
```

```
print(prediction)
```

```
1/1 [=====] - 0s 48ms/step
[[5.4520988e-12 1.0888041e-21 7.5880036e-13 1.2055289e-11 2.6268963e-19
 3.1448650e-15 1.1666435e-19 1.0000000e+00 2.1508427e-11 2.9282635e-11]
 [4.4279619e-12 1.2889054e-12 1.0000000e+00 2.0362232e-17 5.6934279e-20
 5.3493605e-19 4.8295710e-13 5.0981827e-21 1.0086084e-15 1.9516000e-19]
 [9.2632938e-08 9.9921167e-01 6.2528164e-07 4.8549702e-11 7.8550651e-04
 1.5237079e-07 6.6308399e-09 1.0095750e-09 1.9272488e-06 1.2933888e-12]
 [1.0000000e+00 1.6884626e-16 2.7246841e-10 5.4617701e-15 1.6128288e-14
 1.5063359e-10 5.7584715e-10 9.0131895e-14 7.9959858e-11 2.1980879e-08]]
```

```
In [ ]: ▶ import numpy as np
print(np.argmax(prediction,axis=1))
print(y_test[:4])
```

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

Saving the model

```
In [ ]: ▶ model.save('models/mnistCNN.h5')
```

The “index.html” file renders the home page which gives the information about prediction and the ‘predict’ and ‘clear’ options.

“index.html” code:

<html>

```
<head>
  <title>HDR</title>

  <meta name="viewport" content="width=device-width">
  <link rel="stylesheet" href="../../static/css/style.css">
  <link
href="https://fonts.googleapis.com/css2?family=Prompt:wght@600&display=swa
p" 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&di
splay=swap" rel="stylesheet">
  <link
href="https://fonts.googleapis.com/css?family=Calistoga|Josefin+Sans:400,7
00|Pacifico&display=swap" rel="stylesheet">

  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min
.css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoR
xT2MZw1T" crossorigin="anonymous">
  <link rel="stylesheet" type= "text/css" href= "{{
url_for('static',filename='css/style.css') }}">

  <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+8abtTE
lPi6jizo" crossorigin="anonymous"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.mi
n.js"
integrity="sha384-UO2eT0CpHqdsJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86d
IHNDz0W1" crossorigin="anonymous"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.j
s"
```



```

integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">
<script
src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>

</head>
<style>
    body{
        background-image: url('../static/images/1.png');
        background-repeat: no-repeat;
        background-size: cover;
    }
</style>

<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>
  <br><br><br><br><br><br><br>
  <div class="container p-3 my-3 text-dark" id="inst">
    <p>This application will recognise hand written digits in
pictures and print the result. The model used here is Convolutional Neural
Networks.
    </p>
    <p>Instructions:
      <ol>
        <li>Choose your file that is image.</li>
        <li>Press the predict button</li>
        <li>The digit will be printed on the next screen!!</li>
      </ol>
    </p>
  </div>
  <section id="content">

    <div class="leftside">
      <form action="/predict" method="POST"
enctype="multipart/form-data">
        <label>Upload Image File: </label>
        <input id="image" type="file" name="image" accept="image/png,
image/jpeg" onchange="preview()"><br><br>
        <img id="frame" width="100px" height="100px"/>
        <div class="buttons_div">
          <button type="submit" class="btn btn-light"
id="bt1">Predict</button>
          <button type="button" class="btn btn-light" id="bt1">&nbsp;&nbsp;&nbsp;
Clear &nbsp;&nbsp;&nbsp;</button>
        </div>
      </form>
    </div>
  </section>

</body>

</html>

```

The “predict.html” file renders the page where the home page leads to when the user clicks the ‘predict’ option. On this page, the predicted number is shown.

“predict.html” code:

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

<style>
  body{
    background-image: url('../static/images/3.jpg');
    background-repeat: no-repeat;
    background-size: cover;
  }

  #rectangle{
    width:400px;
    height:150px;
    background-color: #000000;
    border-radius: 15px;
    position:absolute;
    box-shadow: 0px 0px 10px 5px white;
    top:25%;
    left:50%;
    transform:translate(-50%,-50%);
  }

  #head{
    text-align: center;
    font-size: 30px;
    margin: 0 auto;
    padding: 3% 5%;
    font-family: Arial, Helvetica, sans-serif;
    color: white;
  }
```

```

        #num{
            font-size: 50px;
        }

</style>

<body>

    <div id="rectangle">
        <h1 id="head">Predicted Number : <br><center
id="num">{{num}}</center></h1>
    </div>

</body>
</html>

```

The “style.css” file contains the code for styling the index page and the predict page.

“style.css” code:

```

#clear_button{
    margin-left: 15px;
    font-weight: bold;
    color: rgb(0, 174, 255);
}

#inst{
    color : rgb(172, 167, 167);
    background-color:rgb(250, 214, 250);
    font-family:URW Chancery L, cursive;
    font-weight: bold;
}

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

```

```
#content{
  color: rgb(15, 15, 15);
  margin: 0 auto;
  padding: 2% 15%;
  padding-bottom: 0;
  font-family: URW Chancery L, cursive;
}

.welcome{
  text-align: center;
  position: relative;
  color: rgb(253, 253, 253);
  background-color: skyblue;
  padding-top: 1%;
  padding-bottom: 1%;
  font-weight: bold;
  font-family: 'Bookman', 'URW Bookman L', serif;
}

#team_id{
  text-align: right;
  font-size: 25px;
  padding-right: 3%;
}

#predict_button{
  margin-right: 15px;
  color: rgb(0, 255, 72);
  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: rgb(250, 214, 250);
  border: 1px solid #888;
  margin-top: 20px;
  margin-bottom: 20px;
}

.buttons_div{
  margin-bottom: 30px;
  margin-right: 80px;
}
#bt1{
  background-color: rgb(255, 210, 10);
}

#bt1:hover{
  background-color: white;
}

.heading{
  font-family: "American Typewriter", 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%; */
}

h1{
    text-align: center;
    color: black;
    padding: 100px 50px 65px 100px;
}

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

```

The app.py contains the backend code and also integrates front end with backend.

“app.py” code:

```

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 = r"C:\Users\Welcome\Desktop\Sem 7\IBM\Project\Application  
Building\data"
```

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

```
model = load_model("./models/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)
app.debug = True
app.run()
```

Github link:

<https://github.com/IBM-EPBL/IBM-Project-16835-1659623726>

Demo link:

<https://github.com/IBM-EPBL/IBM-Project-16835-1659623726/tree/main/Final%20Deliverables/Demo%20Video>