**PROJECT REPORT**

**A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM**

**Submitted By**

***PNT2022TMID28098***

Kailash G - 312419104067

Jude Anton G - 312419104064

Jenwin Amos S - 312419104057

Jelson Prakash M - 312419104058

**TABLE OF CONTENTS**

# 1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

# 2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

# 3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

# 4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

# 5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

# 6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

# 7. CODING & SOLUTIONING (Explain the features added in the project along with code) 8. TESTING

8.1 Test Cases

8.2 User Acceptance Testing

1. **RESULTS**

9.1 Performance Metrics

# 10. ADVANTAGES & DISADVANTAGES

# 11. CONCLUSION

# 12. FUTURE SCOPE

# 13. APPENDIX

Source Code

GitHub & Project Demo Link

# 1. INTRODUCTION

## 1.1 Project Overview

Machine learning is a branch of Artificial Intelligence and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

The handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image.

## 1.2 Purpose

Hand writing recognition of characters has been around since the 1980s.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.

# 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 Publicati on** | **Description** |
| 1. | Amit Choudh  ary,  Anand  Nayyar,  Saurabh Singh and  Byungu  n Yoon. | Improved  Handwritten  Digit  Recognition  Using  Convolutiona l Neural Networks  (CNN) | IEEE Sensors Journal |  | 2021 | In this paper, with the aim of  improving the performance of handwritten digit recognition, they valuated variants of a convolution al neural network to avoid complex preprocessin g, costly feature extraction and a complex ensemble  (classifier  combination ) approach |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | of a traditional recognition system. |
| 2. | Rudras wamima  th,  Bhavani shankar and  Channas andra. | Handwritten  Digit  Recognition using CNN | International  Journal of  Innovative  Science and Research  Technology | Volume  -4 Issue6 | 2018 | 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 acquire the comparison between the classifiers |
| 3. | ,  Shadma n Sakib and Md. Abu  Bakr Siddiqu  e. | Recognition of  Handwritten  Digit using Convolutiona l Neural Network in  Python with Tensorflow and  Comparison  of  Performance for Various  Hidden  Layers | 5th  International Conference on Advances in Electrical  Engineering  (ICAEE) |  | 2017 | In this paper, they observed the variation of accuracies  of CNN to classify handwritten digits for 15 epochs using various numbers of hidden layers and epochs and |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | to make the comparison between the accuracies. For this performance evaluation of CNN, they  performed the experiment using Modified  National  Institute of Standards and  Technology( MN IST) dataset. |
| 4. | Gupta, Ravindr a Pratap Narwari a and Madhav  Singh | Review on  Deep  Learning  Handwritten  Digit Recognition using Convolutiona l Neural Network | International  Journal of  Recent Technology and  Engineering  (IJRTE) | Volume  -9 Issue5 | 2022 | 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. |
| 5. | Anwar Hossain and Md.  Mohon  Ali | Recognition of  Handwritten  Digit using Convolutiona l Neural Network  (CNN) | Global  Journal of  Computer Science and Technology:  D Neural &  Artificial  Intelligence | Volume  19  Issue2 | 2020 | The goal of this work will be to create a model that will be able to identify and determine the handwritten digit from its image with better accuracy using using the concepts of  Convolution al Neural Network and MNIST |
|  |  |  |  |  |  | dataset. Later it can be extended for character recognition and realtime person’s handwriting. The results can be made more accurate with more convolution  layers and more number of hidden neurons. |

## 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 real time 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 on to UI(User Interface).

# 3. IDEATION & PROPOSED SOLUTION

## 3.1 Empathy Map Canvas



## 3.2 Ideation & Brainstorming

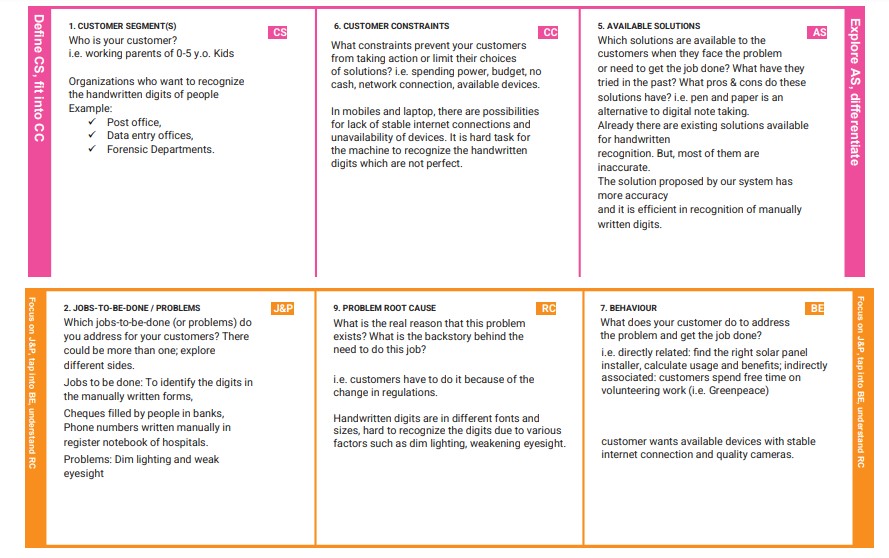


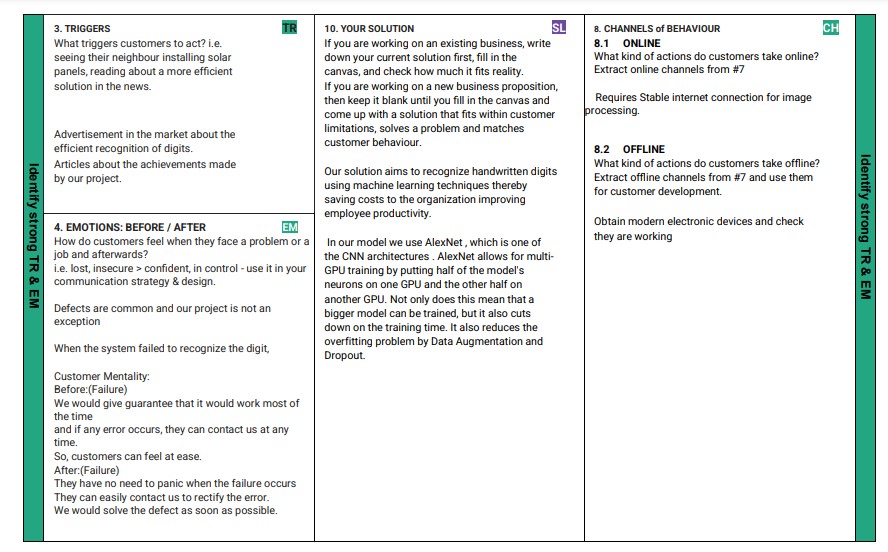
## 3.3 Proposed Solution

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** |  | **Parameter** | **Description** |
|  | 1 | Problem Statement (Problem to be solved) | Statement-The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits.  Description: It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes. |
|  | 2 | Idea / Solution description | 1. It is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers, touch defences. 2. It allows user to translate all those signature and notes into electronic words in a text document format and this data only requires far less physical space than the storage of the physical copies. |

|  |  |  |
| --- | --- | --- |
| 3 | Novelty / Uniqueness | Accurately recognize the digits rather than recognizing all the characters like OCR Handwritten Digit Recognition is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers,touch defenses, etc. And classify them into 10 predefined classes(09).This is the existing method along with this we add some features to make our project unique among them. |
| 4 | Social Impact / Customer Satisfaction | 1.Artificial Intelligence developed the app called Handwritten digit Recognizer.  2. It converts the written word into digital approximations and utilizes complex algorithms to identify characters before churning out a digital approximation. |
| 5 | Business Model (Revenue Model) | • This system can be integrated with traffic surveillance cameras to recognize the vehicle’s number plates for effective traffic management.  • Can be integrated with Postal system to identify and recognize the pin-code details easily. |
|  |  | necessarily because of sloppy handwriting, but because people write all over the envelope.We have hard time segmenting handwritten addresses into their components, such as ZIP code or street address, because very few people print addresses neatly in a prescribed format. So, this problem can be solved using Handwritten digit recognition system. |
| 6 | Scalability of the Solution | In our model, AlexNet significantly outperformed as it is trained on a GTX 580 GPU with only 3 GB of memory which couldn’t fit the entire network. So the network was split across 2 GPUs, with half of the neurons(feature maps) on each GPU. So, a greater accuracy can be attained by allowing multi-GPU training by putting half of the model's neurons on one GPU and the other half on another GPU. |

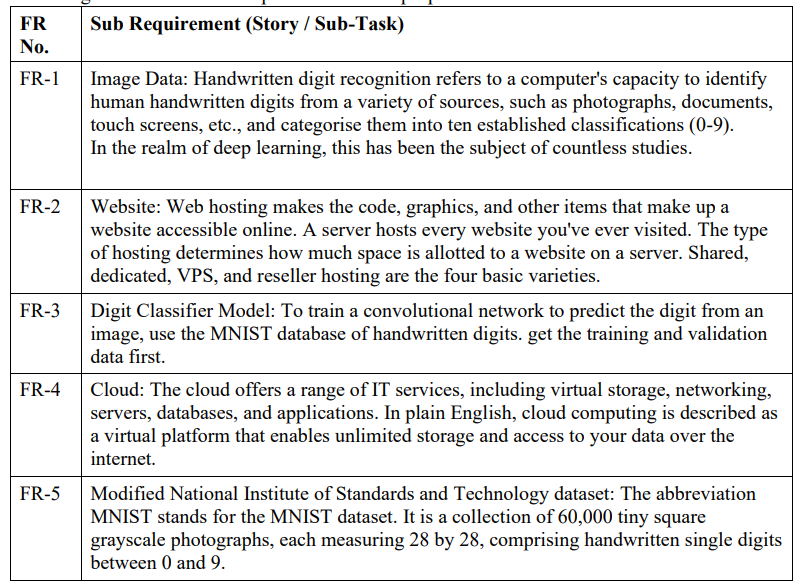
## 3.4 Problem Solution fit



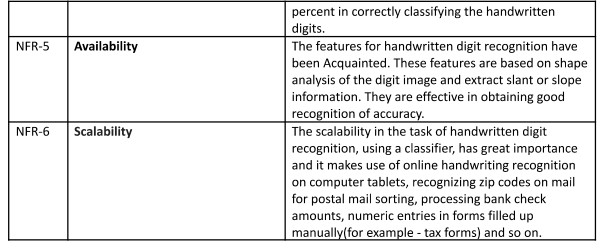
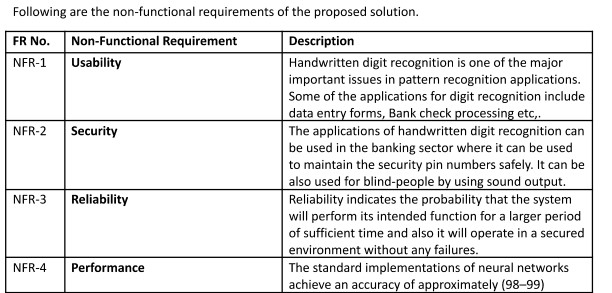


# 4. REQUIREMENT ANALYSIS

## 4.3 Functional requirement

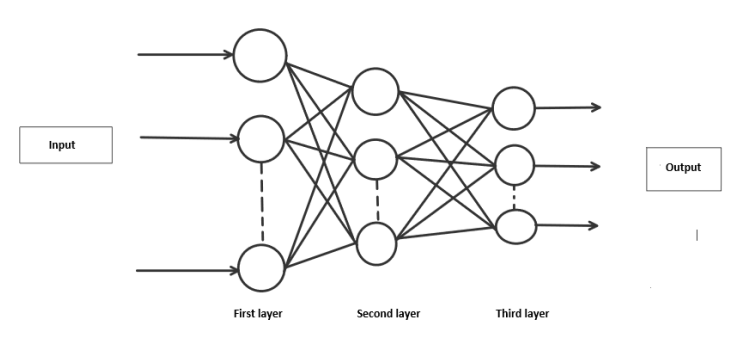


**4.4 Non-Functional requirements**



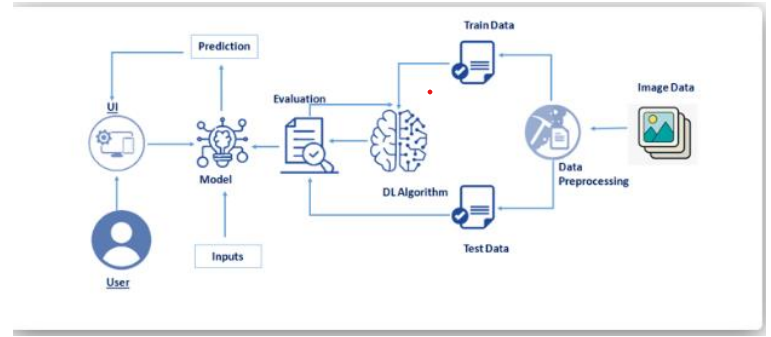
**5. PROJECT DESIGN**

## 5.1 Data Flow Diagrams

****

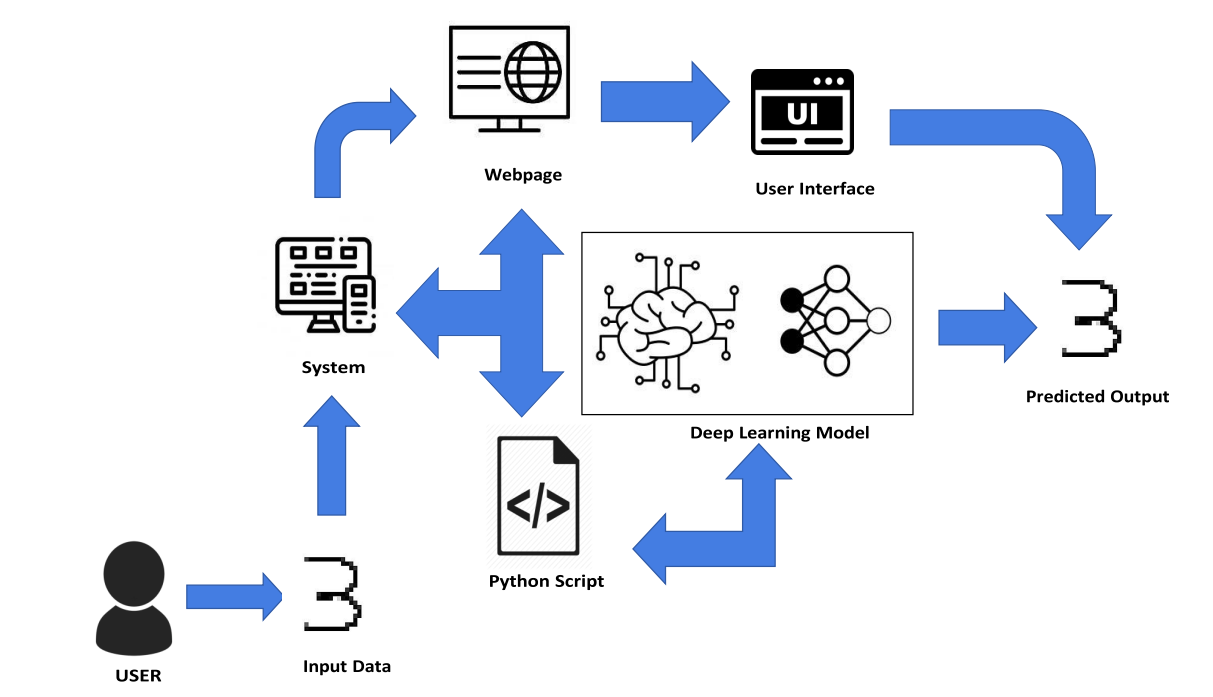
**5.2 Solution & Technical Architecture**

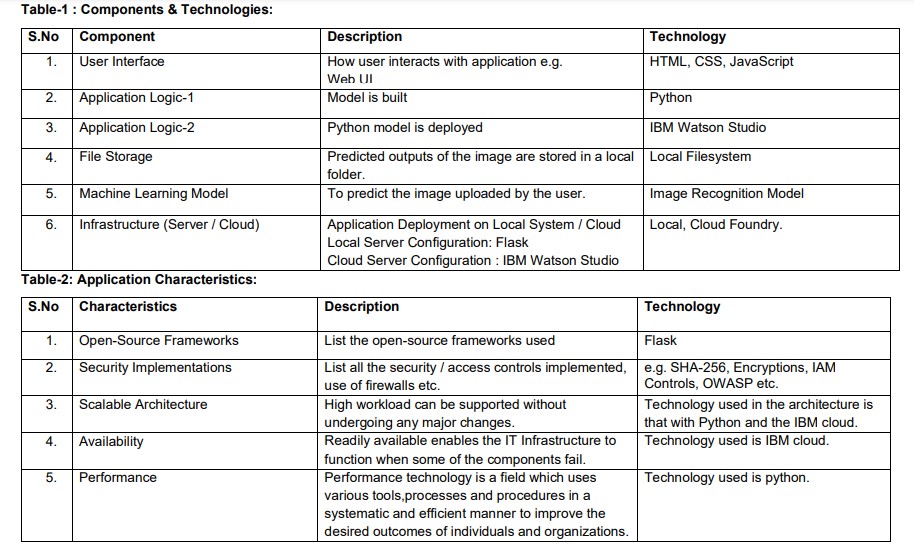
# Solution Architecture



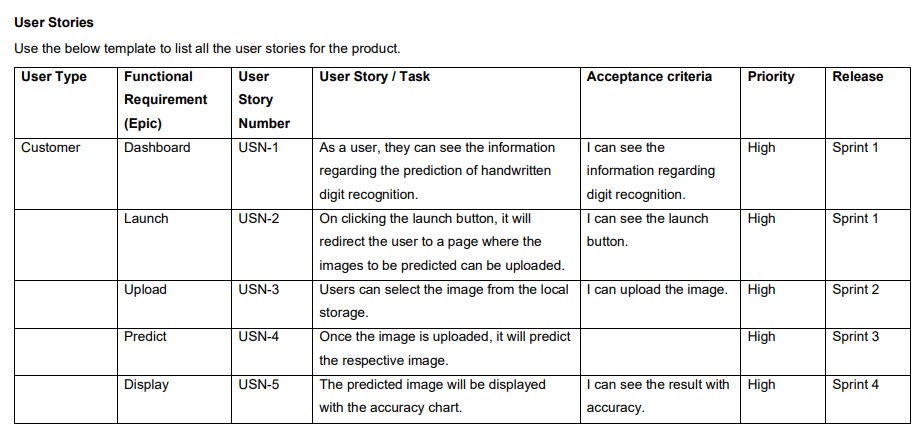
# 

# Technology Architecture



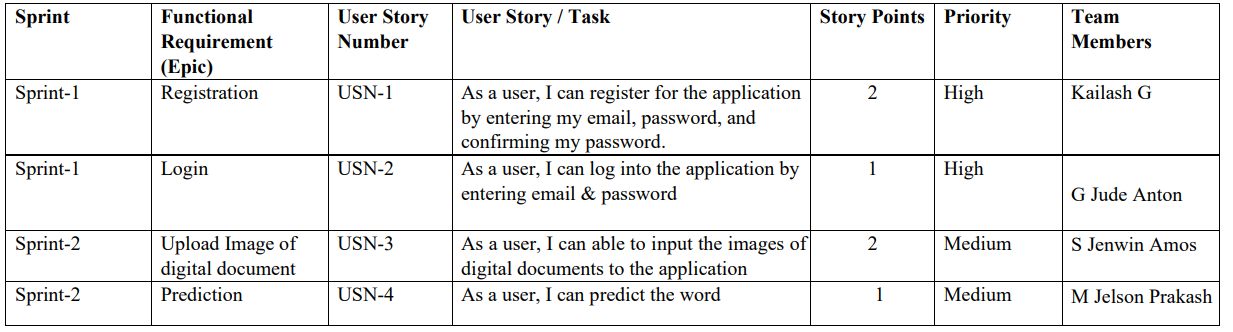


## 5.3 User Stories

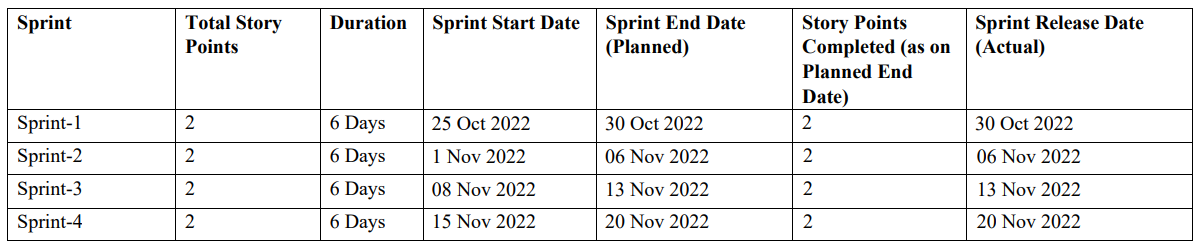


# 6. PROJECT PLANNING & SCHEDULING

## 6.1 Sprint Planning & Estimation

****

## 6.2 Sprint Delivery Schedule

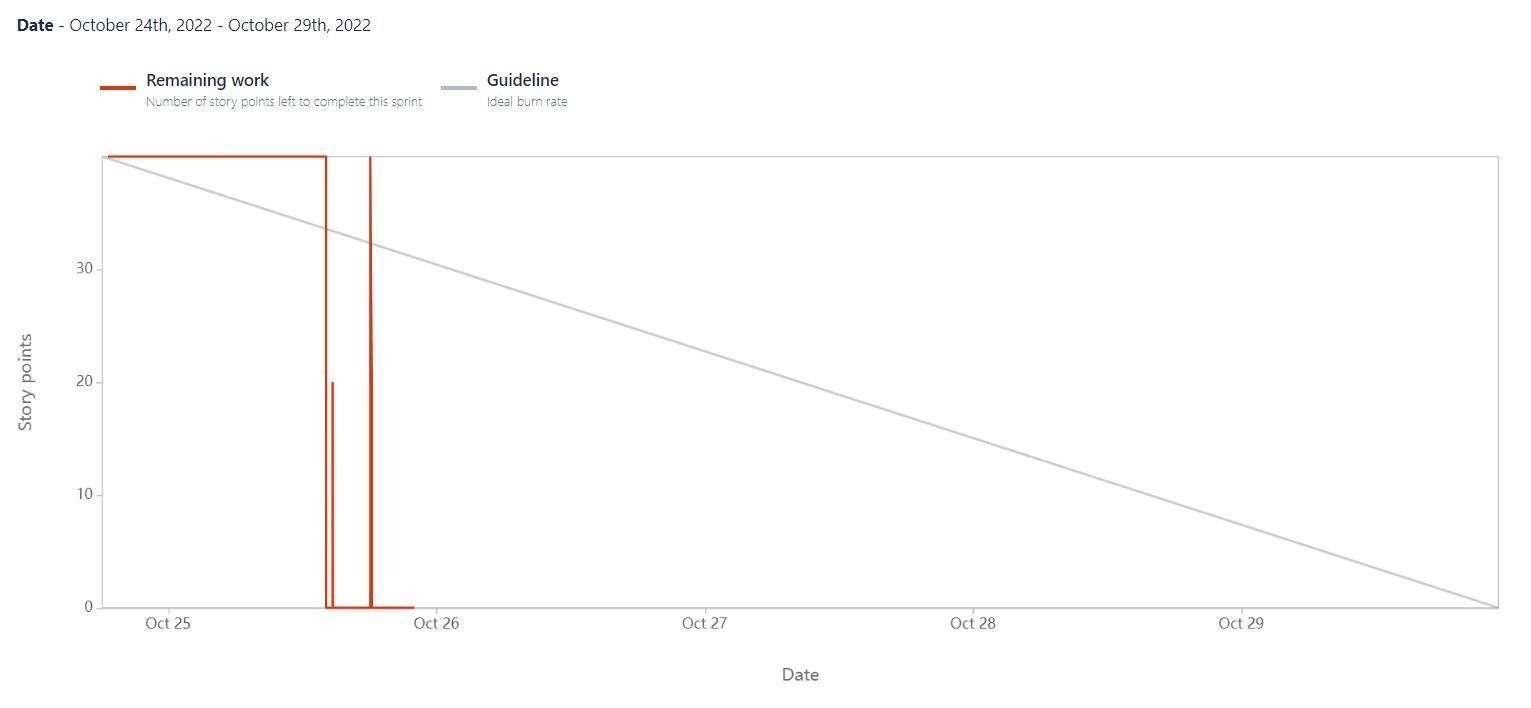
****

**6.3 Reports from JIRA**

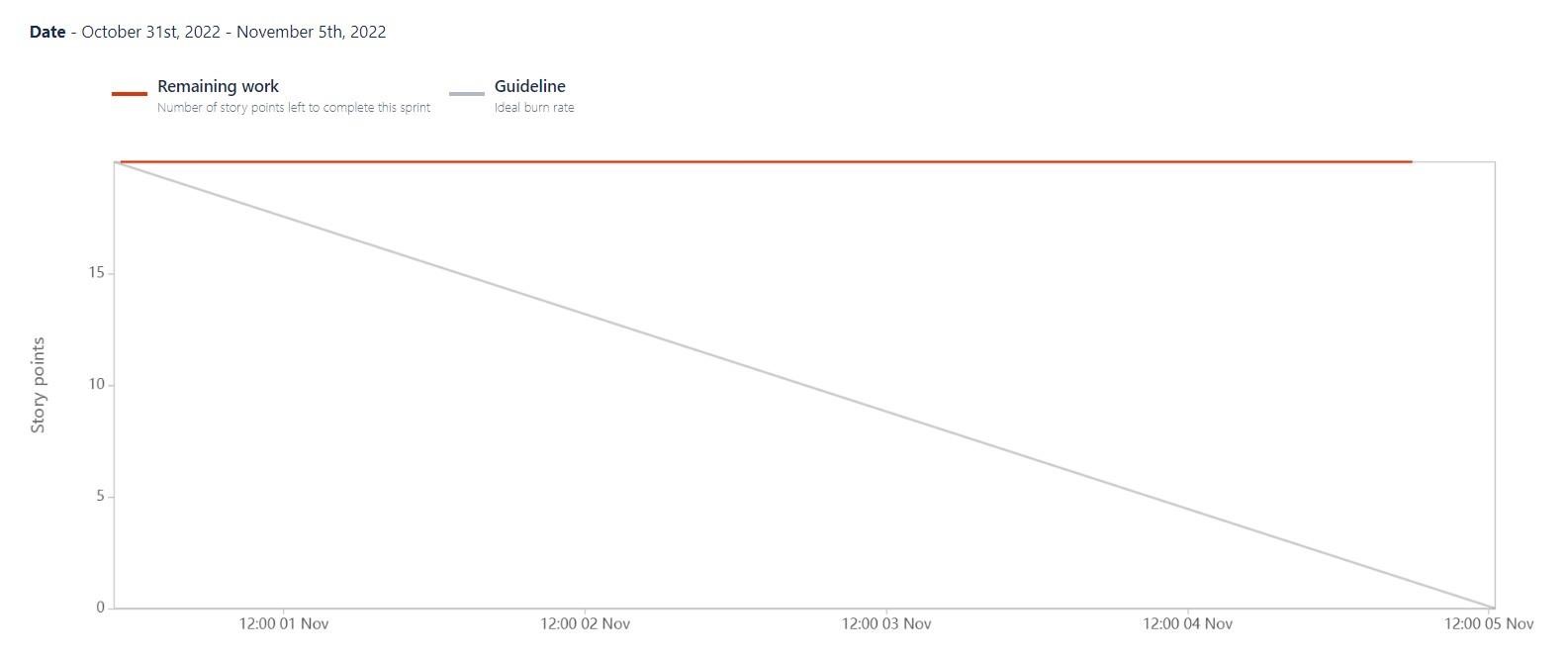
# Velocity Report



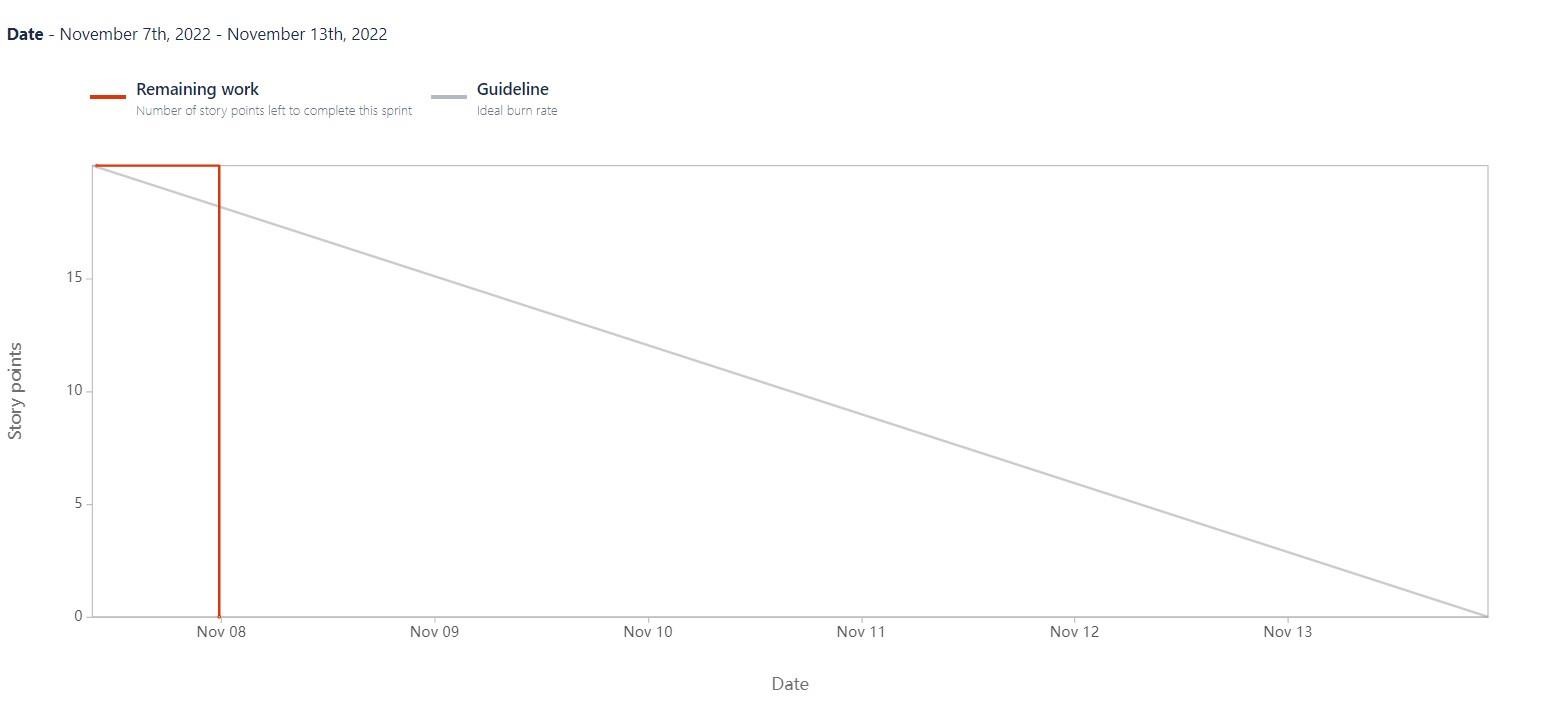
# ` Sprint 1

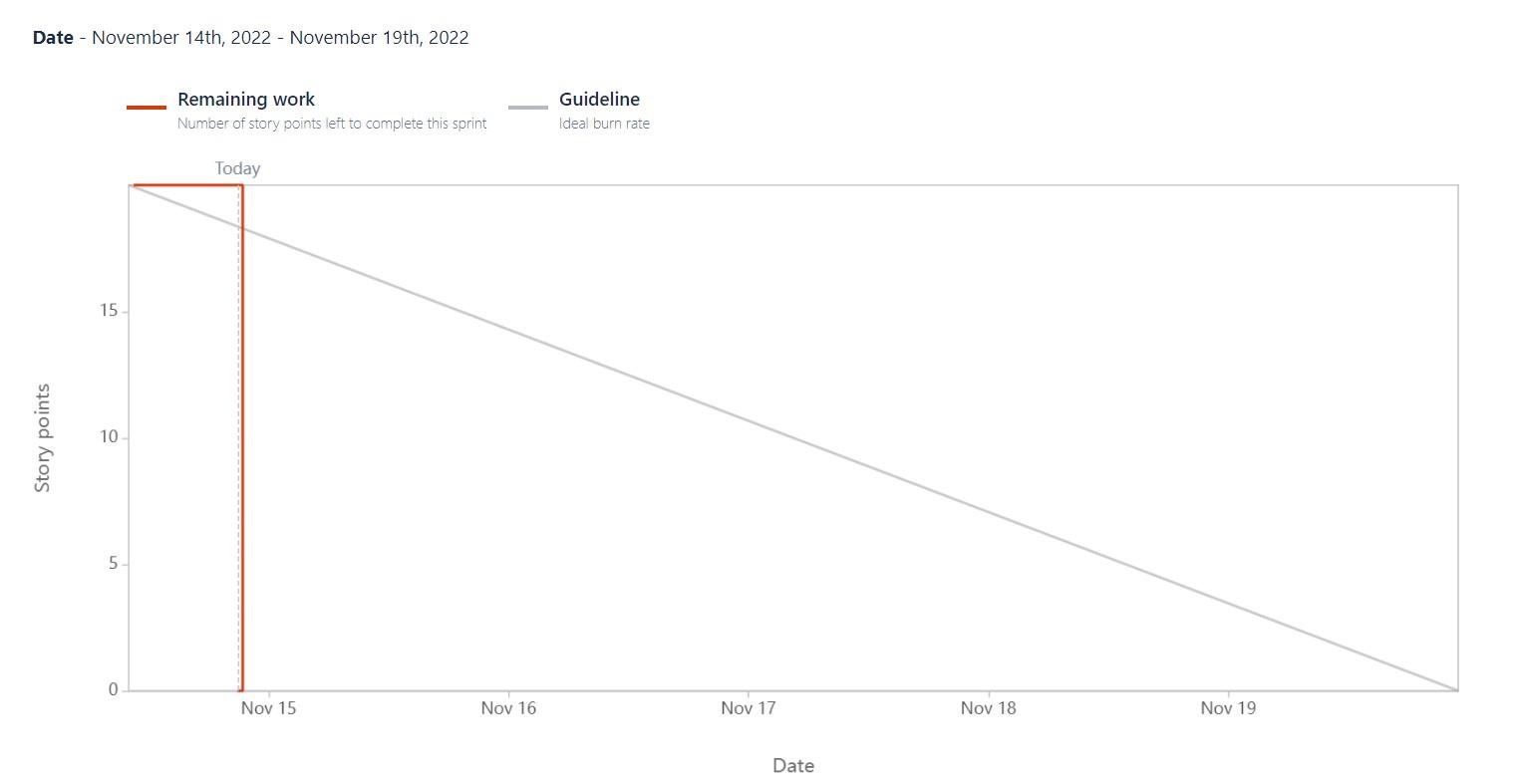


# Sprint 2



# Sprint 3



**Sprint 4** 

# 7. CODING & SOLUTIONING (Explain the features added in the project along with code)

|  |
| --- |
| 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   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))  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/modelCNN.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 |

**8. TESTING**

**8.1 Test Cases**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case ID** | **Feature Type** | **Component** | **Test Scenario** | **Expected Result** | **Actual Result** | **Status** |
| Homepage\_TC\_OO1 | Functional | Home Page | Verify user is able to see the  Homepage when clicked on the link | Home Page should be displayed. | Working as expected | Pass |
| Homepage\_TC\_OO2 | UI | Home Page | Verify the UI elements in  Homepage | Application should show below UI elements:  a.choose file button  b.predict button  c.clear button | Working as expected | Pass |
| Homepage\_TC\_OO3 | Functional | Home Page | Verify user is able to choose file from the local system  and click on predict | Choose file popup screen must be displayed and user should be able to click on predict button | Working as expected | Pass |
| Homepage\_TC\_OO4 | Functional | Home page | Verify user able to select  invalid file format | Application won't allow to attach formats other than ".png, .jiff,  .pjp, .jpeg,  .jpg, .pjpeg" | Working as expected | Pass |
| Predict\_TC\_OO5 | Functional | Predict page | Verify user is able to  navigate to the  predict to and view the  predicted result | User must be navigated to the predict page and must view the predicted result | Working as expected | Pass |

**8.2 User Acceptance Testing**

# Defect Analysis

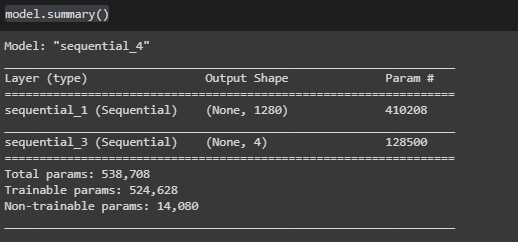
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Resolution** | **Severity 1** | **Severity 2** | **Severity 3** | **Severity 4** | **Subtotal** |
| By Design | 0 | 0 | 0 | 0 | 0 |
| Duplicate | 0 | 0 | 0 | 0 | 0 |
| External | 0 | 0 | 0 | 0 | 0 |
| Fixed | 0 | 0 | 0 | 0 | 0 |
| Not Reproduced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

# Test Case Analysis

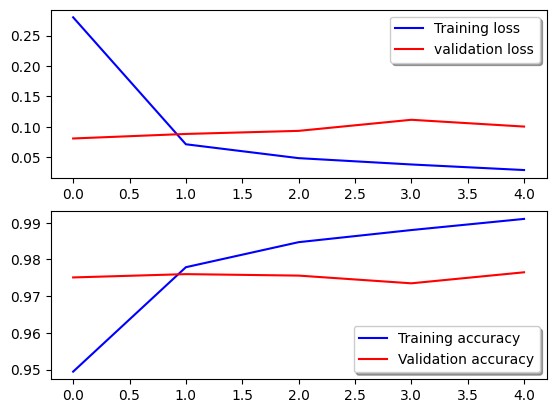
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Section** | **Total Cases** | **Not Tested** | **Fail** | **Pass** |
| Client Application | 5 | 0 | 0 | 5 |
| Security | 5 | 0 | 0 | 5 |
| Final Report Output | 5 | 0 | 0 | 5 |
| Performance | 5 | 0 | 0 | 5 |

# 9. RESULTS

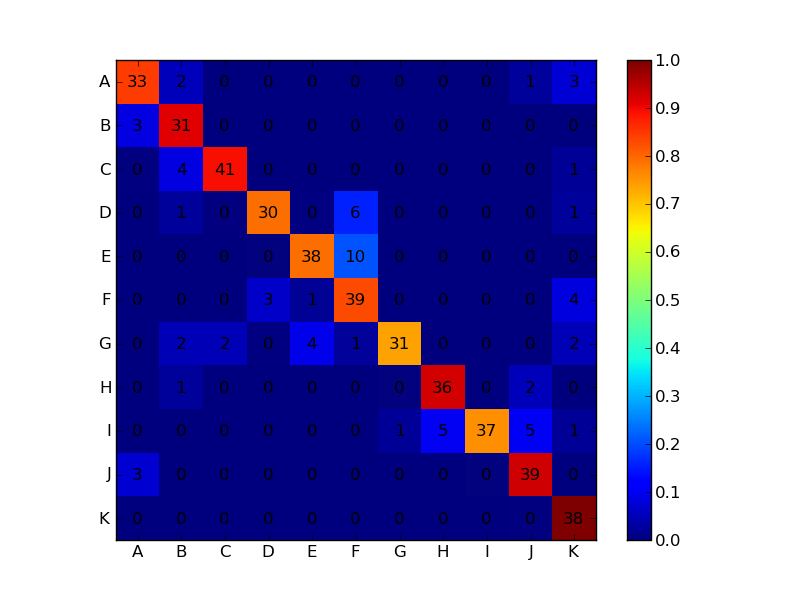
**9.1 Performance Metrics Model Summary**:



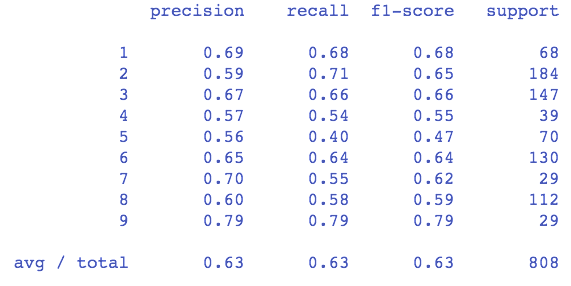
**Accuracy:**



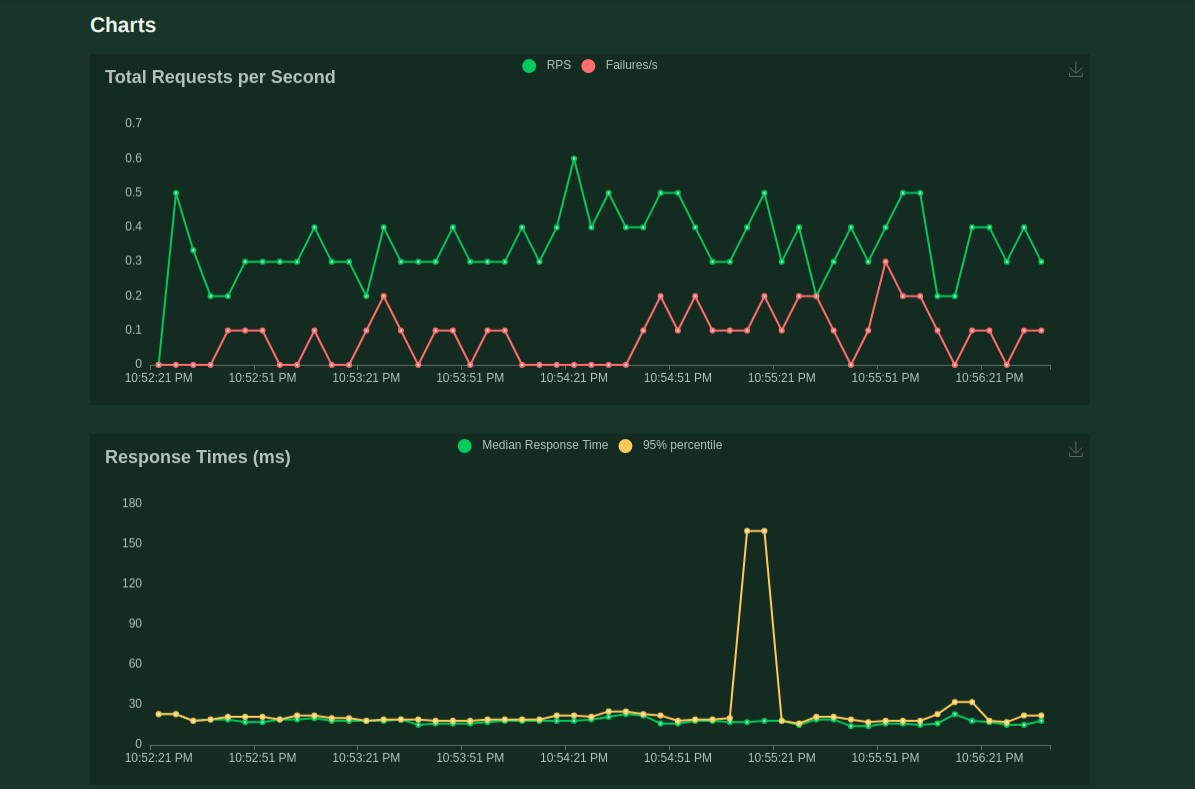
**Confusion Matrix:**

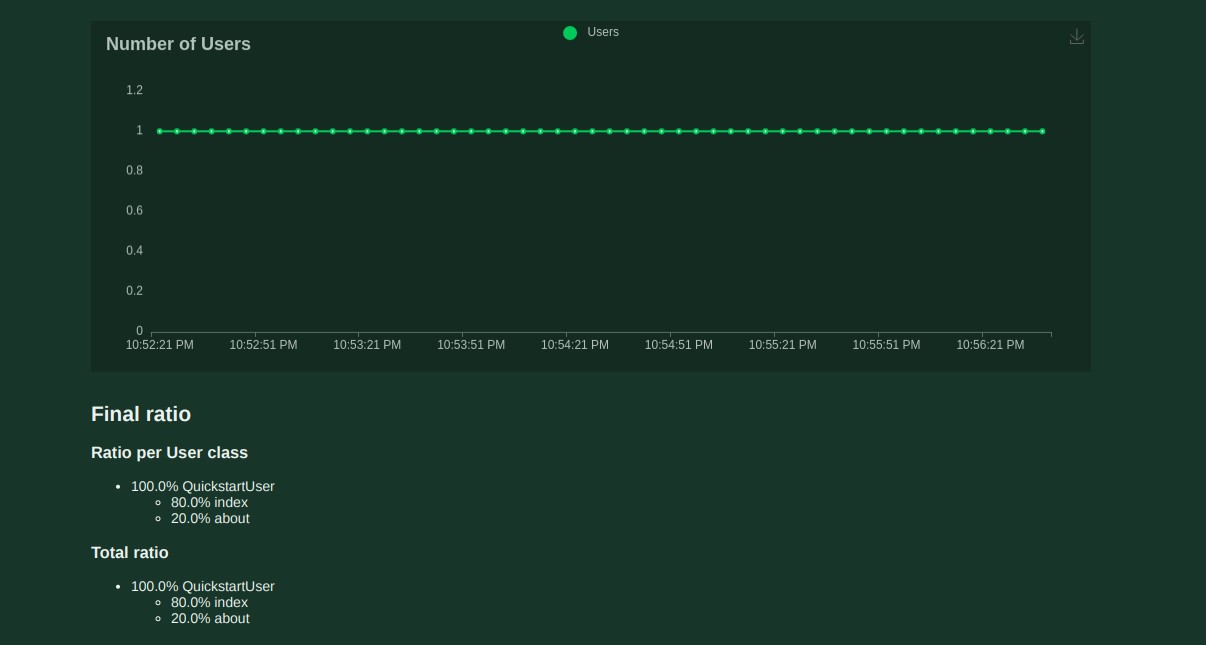


**Classification Report:**

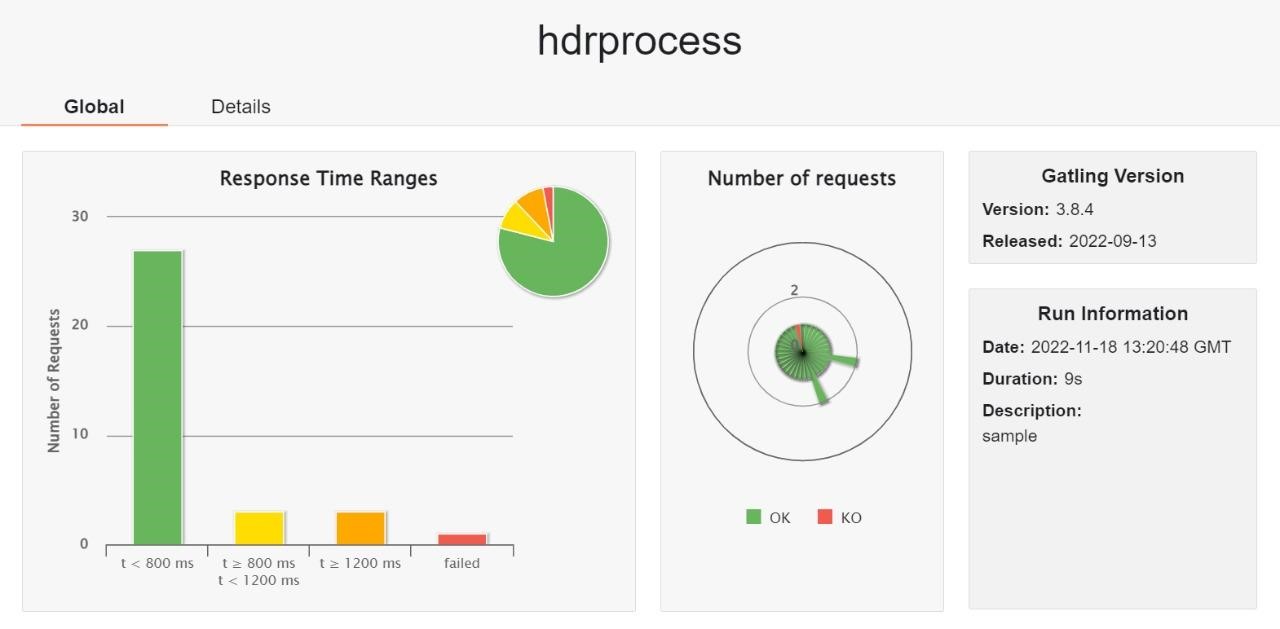


**Performance Metrics Result:**





# Gatling report



**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 high performance server for faster predictions**.**
* Prone to occasional errors.

# 11. CONCLUSION

This project demonstrated a web application that uses machine learning to recognie 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 realworld 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.

# 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.

# 13. APPENDIX Source Code

**HTML AND CSS: home.html:**

|  |
| --- |
| <html>  <head>  <meta name="viewport" content="width=device-width, initial-scale=1.0" />  <title>Handwritten Digit Recognition</title>  <h3>TEAM ID : PNT2022TMID28098</h3>  <link rel="icon" type="image/svg" sizes="32x32" href="{{url\_for('static',filename='images/icon.svg')}}" />  <link rel="stylesheet" href="{{url\_for('static',filename='css/main.css')}}" />  <script src="https://unpkg.com/feather-icons"></script>  <script defer src="{{url\_for('static',filename='js/script.js')}}"></script> </head>  <body>  <div class="container">  <div class="heading">  <h1 class="heading\_\_main" data-shadow='Handwritten Digit Recognizer'>Handwritten Digit Recognizer</h1>  <h2 class="heading\_\_sub">Easily detect handwritten digits and Verfiy it</h2>  <h2 class="heading\_\_sub">TEAM ID: PNT2022TMID28098</h2>  </div>  <div class="upload-container">  <div class="form-wrapper">  <form class="upload" action="/predict" method="post" enctype="multipart/form-data">  <label id="label" for="upload-image"><i data-feather="file-plus"></i>Upload File</label>  <input type="file" name="photo" id="upload-image" hidden />  <button type="submit" id="up\_btn"></button>  </form>  <img id="loading" src="{{url\_for('static',filename='images/loading.gif')}}">  </div>  </div>  </div> </body>  </html> |

**Predict.html:**

</head> <style> body{ background-image: url('static/images/bc1.jpg'); <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>PREDICTIONS</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 PREDICITONS</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>

**Main.css**

@import url('https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;500;600&family=Righteous&family=Sono:wght@400;500;600;700;800&display=swap');  
  
\* {  
 padding: 0;  
 margin: 0;  
}  
  
body {  
 color: #f1faee;  
 font-family: 'Brush Script MT', cursive;  
}  
  
  
  
@keyframes gradient {  
 0% {  
 background-position: 0% 50%;  
 }  
  
 50% {  
 background-position: 100% 50%;  
 }  
  
 100% {  
 background-position: 0% 50%;  
 }  
}  
  
  
  
.container {  
 width: 100%;  
 height: 100%;  
 display: flex;  
 flex-direction: column;  
 justify-content: center;  
 align-items: center;  
 background-image: url("paper.gif");  
 background-attachment:fixed;  
 background-size: 100% 100%;  
 animation: gradient 5s ease infinite;  
  
}  
  
.heading {  
 margin-top: -2rem;  
 padding-bottom: 2rem;  
 width: fit-content;  
 text-align: center;  
 font-family: 'Brush Script MT', cursive;  
}  
  
.heading .heading\_\_main {  
 font-size: 3rem;  
 color: #ff0000;  
 font-weight: 550;  
}  
  
.heading .heading\_\_sub {  
 font-size: 1rem;  
 color: #ff0000;  
 font-family: 'Brush Script MT', cursive;  
  
}  
  
.upload-container {  
 box-shadow: 0 0 20px rgb(33, 195, 183);  
 width: 40rem;  
 height: 25rem;  
 padding: 1.5rem;  
}  
  
.form-wrapper {  
 background-color: rgba(190, 190, 190, 0.5);  
 width: 100%;  
 height: 100%;  
 display: flex;  
 border: 2px dashed black;  
 justify-content: center;  
 align-items: center;  
}  
  
.form-wrapper #loading {  
 display: none;  
 position: absolute;  
}  
  
.form-wrapper .upload {  
 display: flex;  
 justify-content: center;  
 align-items: center;  
 width: 9rem;  
 height: -webkit-fit-content;  
 height: -moz-fit-content;  
 height: fit-content;  
 font-family: 'Brush Script MT', cursive;  
 border-radius: 6px;  
 color: white;  
 background-color: rgb(33, 195, 183);  
 box-shadow: 0 5px 10px rgb(33, 195, 183);  
}  
  
.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;  
}  
  
.buttons {  
 margin: 20%;  
 text-align: center;  
}  
  
  
  
  
@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;  
 }  
}

**Predict.css**

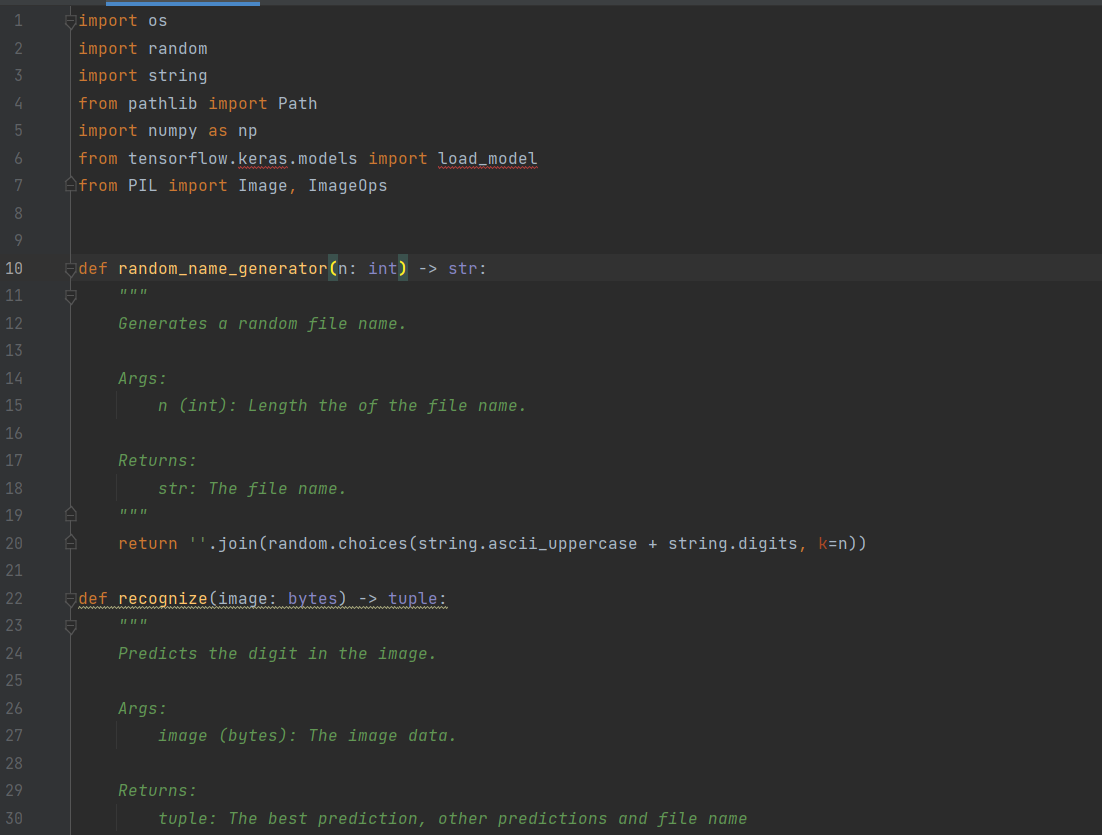
@import url('https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;500;600&family=Righteous&family=Sono:wght@400;500;600;700;800&display=swap');  
  
body {  
 background-image: url("pred.gif");  
   
 background-size: 100% 100%;  
 animation: gradient 15s ease infinite;  
 font-family: 'Brush Script MT', cursive;  
}  
  
@keyframes gradient {  
 0% {  
 background-position: 0% 50%;  
 }  
  
 50% {  
 background-position: 100% 50%;  
 }  
  
 100% {  
 background-position: 0% 50%;  
 }  
}  
  
h1 {  
 padding-top: 2rem;  
 color: #c1d8ff;  
}  
  
.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);  
 box-shadow: 0 0 20px rgb(17, 211, 255);  
}  
  
.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;  
 background-color: #e6e6e6;  
 box-shadow: 0 0 7px rgb(23, 236, 255);  
}  
  
.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;  
 }  
}

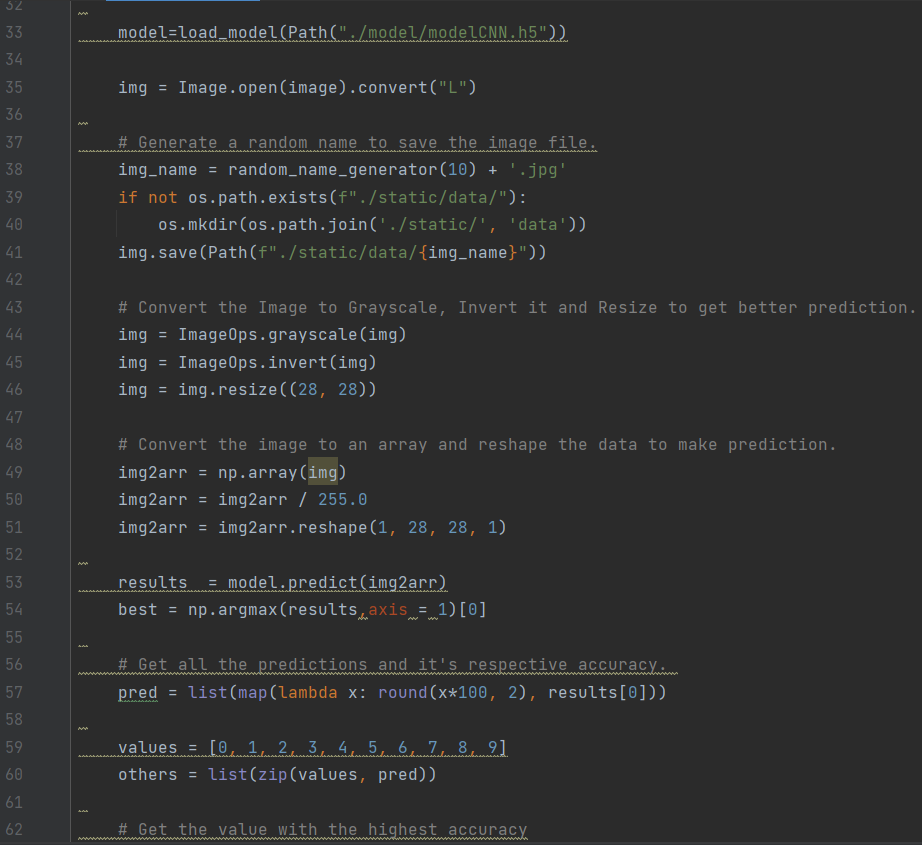
**FLASK**

**App.py**

****

**Recognizer.py**

****

****

**MODEL CREATION:**

# GitHub & Project Demo Link

**GitHub Link** https://github.com/IBM-EPBL/IBM-Project-28021-1660105663

**Demo Video** https://drive.google.com/drive/folders/1KDoHk8enpSK4BW8Y14tZj8WuxB7Nndfn?usp=sharing