|  |  |
| --- | --- |
| Title | Project Report |
| Team ID | PNT2022TMID19491 |
| Project Name | A Novel Method for Handwritten Digit Recognition System |
| Date | 16/11/2022 |

**Project Report**

1. **INTRODUCTION** 
   1. Project Overview
   2. Purpose
2. **LITERATURE SURVEY**
   1. Existing problem
   2. References
   3. Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
   1. Empathy Map Canvas
   2. Ideation & Brainstorming
   3. Proposed Solution
   4. Problem Solution fit
4. **REQUIREMENT ANALYSIS**
   1. Functional requirement
   2. Non-Functional requirements
5. **PROJECT DESIGN**
   1. Data Flow Diagrams
   2. Solution & Technical Architecture
   3. User Stories
6. **PROJECT PLANNING & SCHEDULING**
   1. Sprint Planning & Estimation
   2. Sprint Delivery Schedule
   3. Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
   1. Feature 1
   2. Feature 2
   3. Database Schema (if Applicable)
8. **TESTING** 
   1. Test Cases
   2. User Acceptance Testing
9. **RESULTS**
   1. Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**

Source Code

GitHub & Project Demo Link

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

**1.INTRODUCTION :**

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

* 1. **PROJECT OVERVIEW :**
* Know fundamental concepts and techniques of the Artificial Neural Network and Convolution Neural Networks
* Gain a broad understanding of image data.
* Work with Sequential type of modeling
* Work with Keras capabilities
* Work with image processing techniques
* know how to build a web application using the Flask framework.
  1. **PURPOSE :**

Handwritten character recognition is one of the practically important issues in pattern recognition applications. The main purpose of this project is to build an automatic handwritten digit recognition method for the recognition of handwritten digit strings. To accomplish the recognition task, first, the digits will be segmented into individual digits. Then, a digit recognition module is employed to classify each segmented digit completing the handwritten digit string recognition task.

**2.LITERATURE SURVEY**

**2.1 EXISTING PROBLEM :**

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

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 (0-9). This has been a Content of bottomless- exploration in the field of deep literacy. Number recognition has numerous operations like number plate recognition, postal correspondence sorting, bank check processing, etc . (2). In Handwritten number recognition, we face numerous challenges . because of different styles of jotting of different peoples as it is not an Optic character recognition. This exploration provides a comprehensive comparison between different machine literacy and deep literacy algorithms for the purpose of handwritten number recognition. For this, we've used Support . Vector Machine, Multilayer Perceptron, and Convolutional . Neural Network. The comparison between these algorithms is carried out on the base of their delicacy, crimes, and .testing- training time corroborated by plots and maps that have been constructed using matplotlib for visualization.

**2.2 REFERENCES**

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

[2] K. Gaurav and Bhatia P. K., “Analytical Review of Preprocessing Techniques for Offline Handwritten Character Recognition”, 2nd International Conference on Emerging Trends in Engineering & Management, ICETEM, 2014.

[3] Salvador España-Boquera, Maria J. C. B., Jorge G. M. and Francisco Z. M., “Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 33, No. 4, April 2014

[4] Reena Bajaj, Lipika Dey, and S. Chaudhury, “Devnagari numeral recognition combining decision of multiple connectionist classifiers”, Sadhana, Vol.27, part. 1, pp.-59-72, 2011..

[5] U. Pal, T. Wakabayashi and F. Kimura, “Handwritten numeral recognition of six popular scripts,” Ninth International conference on Document Analysis and Recognition ICDAR 07, Vol.2, pp.749-753, 2010.

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

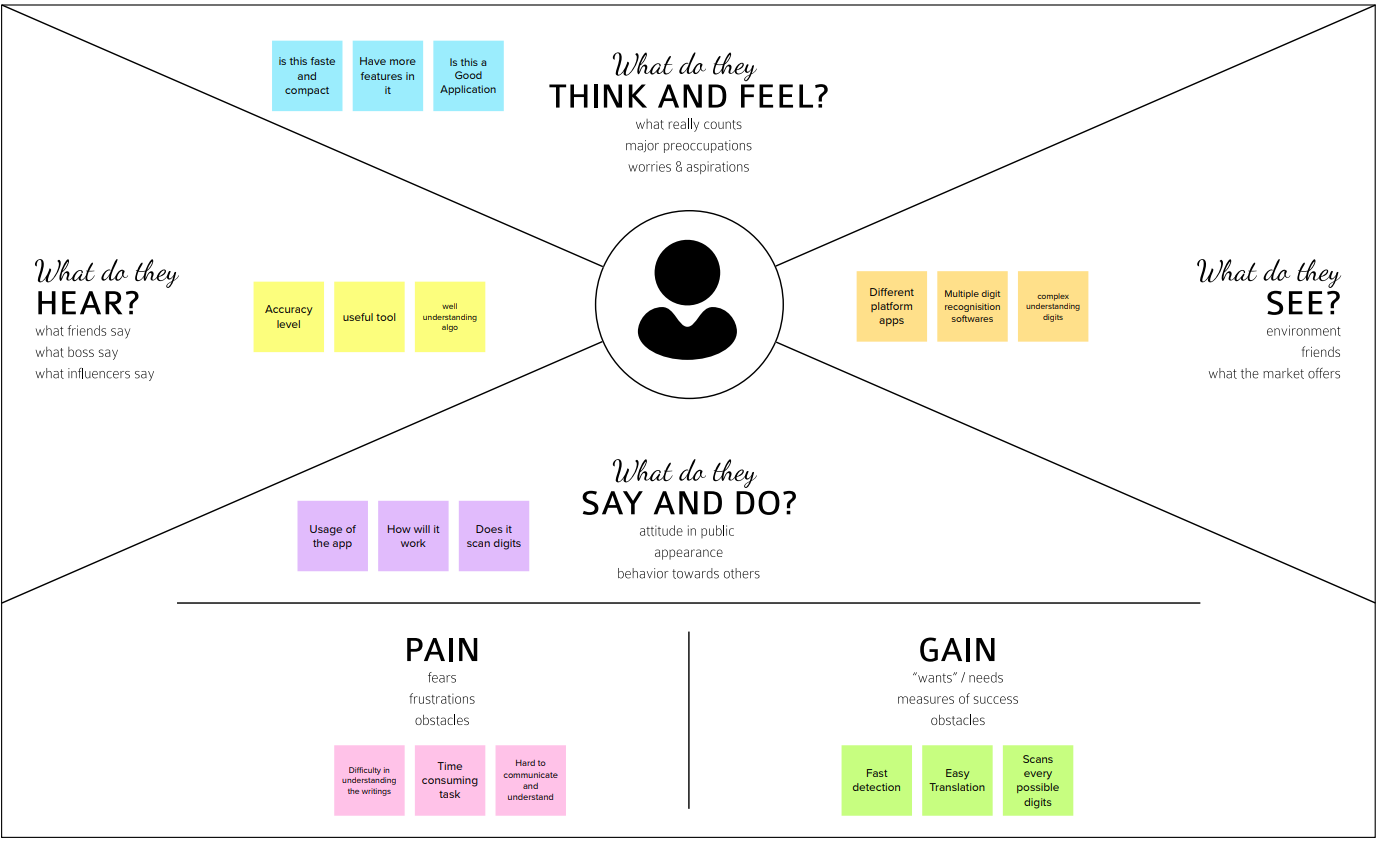
[7] Bharatratna P. Gaikwad, Ramesh R. Manza, Ganesh R. Manza, "Automatic Video Scene Segmentation to Separate Script and Recognition", Advances in Intelligent Systems and Computing Volume 328, pp 225-235, (2015).

**2.3.PROBLEM STATEMENT DEFINITION:**

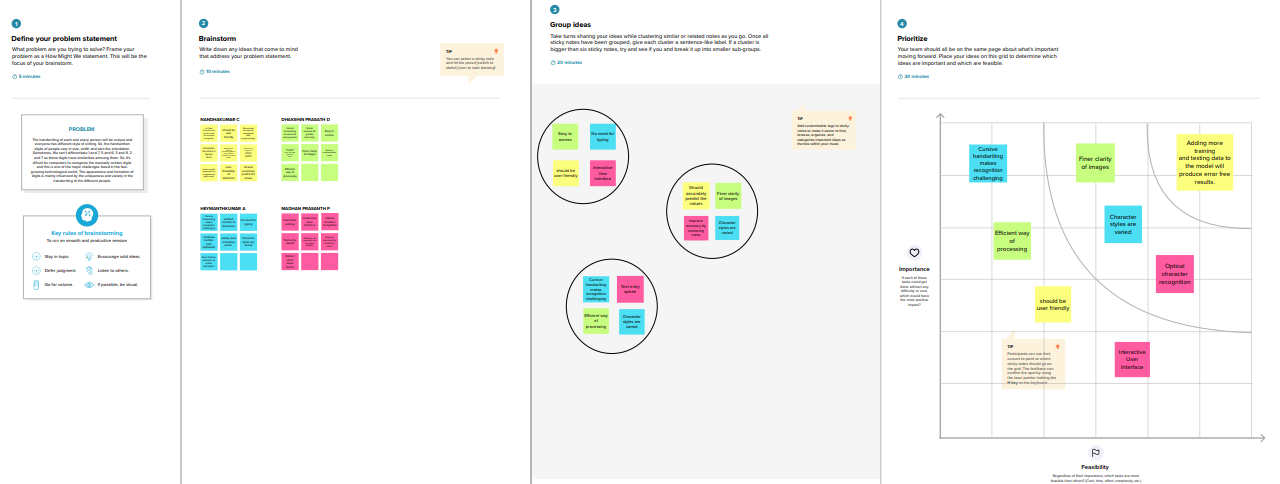
Handwritten Digit Recognition is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers, touch defenses, etc, and classify. them into 10 predefined classes (0-9). This has been a Content of bottomless- exploration in the field of deep literacy. Number recognition has numerous operations like number plate recognition, postal correspondence sorting, bank check processing, etc . (2). In Handwritten number recognition, we face numerous challenges . because of different styles of jotting of different peoples as it is not an Optic character recognition. This exploration provides a comprehensive comparison between different machine literacy and deep literacy algorithms for the purpose of handwritten number recognition. For this, we've used Support . Vector Machine, Multilayer Perceptron, and Convolutional . Neural Network. The comparison between these algorithms is carried out on the base of their delicacy, crimes, and .testing- training time corroborated by plots and maps that have been constructed using matplotlib for visualization

**3.IDEATION & PROPOSED SOLUTION**

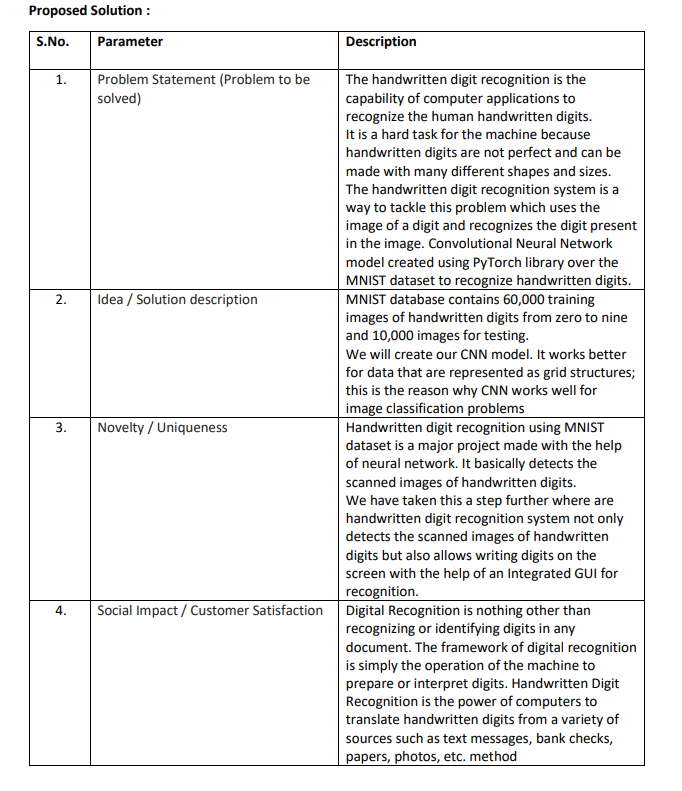
**3.1.Empathy Map Canvas**

****

* 1. **Ideation & Brainstorming :**

****

* 1. **Proposed Solution :**

****

* 1. **Problem Solution fit :**

****

1. **REQUIREMENT ANALYSIS**

**Functional Requirements:**

Following are the functional requirements of the proposed solution.

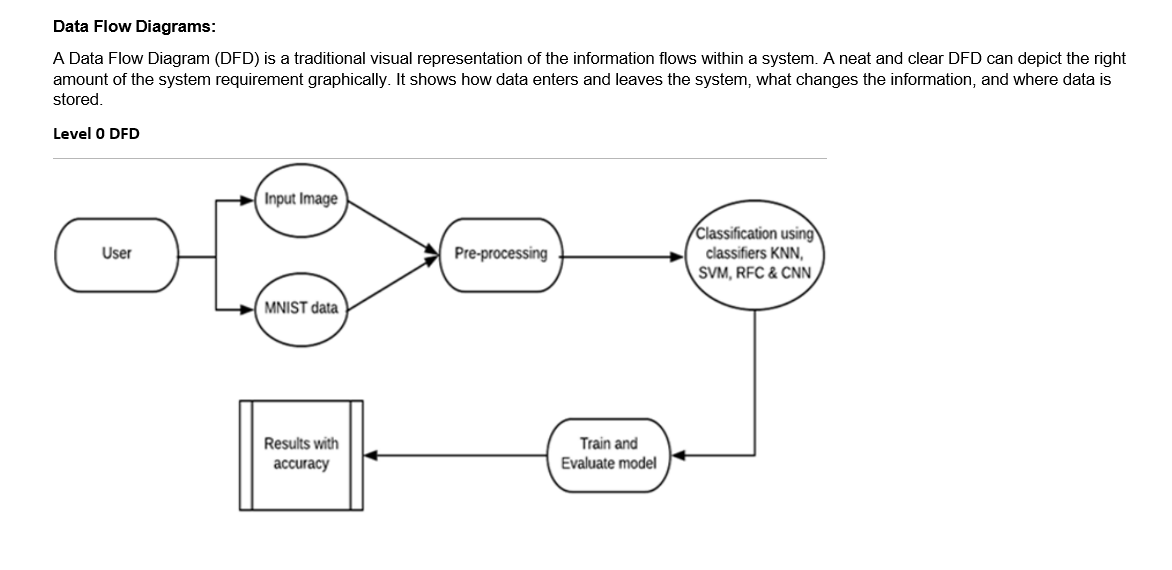
|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through Form Registration through Gmail |
| FR-2 | User Confirmation | Confirmation via Email Confirmation via OTP |
| FR-3 | User Login | Login page gets displayed and User is Logged in |
| FR-4 | Upload Handwritten Files | Upload files to Recognize the digits |
| FR-5 | Displaying Output | Appropriate digits gets displayed to the user |

**Non-functional Requirements:**

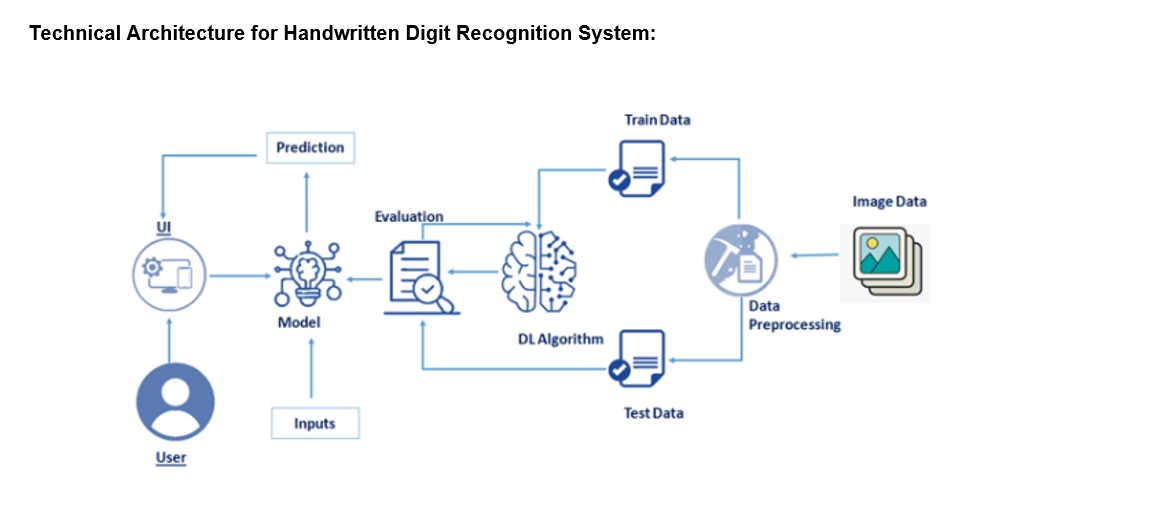
Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Handwritten character recognition is one of the practically important issues in pattern recognition applications. The applications of digit recognition include in postal mail sorting,  bank check processing, form data entry, etc |
| NFR-2 | **Security** | The application where this handwritten digit recognition can be used are banking sector where it can be used to maintain the security pin numbers, it can be also used for blind-people by using sound output. |
| NFR-3 | **Reliability** | The probability that the system will perform its intended function adequately for a large period of time and will operate in a secured environment without failure. |
| NFR-4 | **Performance** | The standard implementations of neural networks achieve an accuracy of ~ (98–99) percent in correctly classifying the handwritten digits. |
| NFR-5 | **Availability** | Abstract and Figures. The features for handwritten digit recognition have been introduced. These features are based on shape analysis of the digit image and extract slant or slope information. They are effective in obtaining good recognition accuracies. |

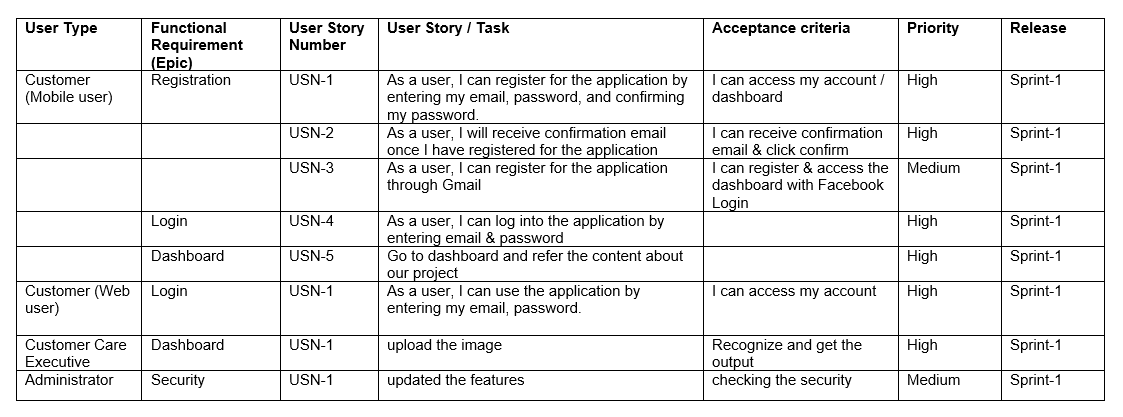
1. **PROJECT DESIGN**
   1. Data Flow Diagrams



* 1. Solution & Technical Architecture

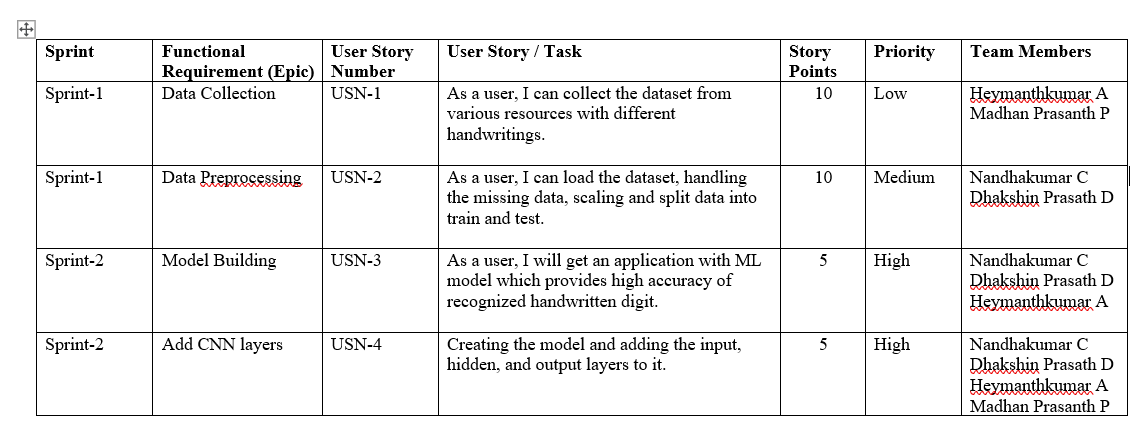


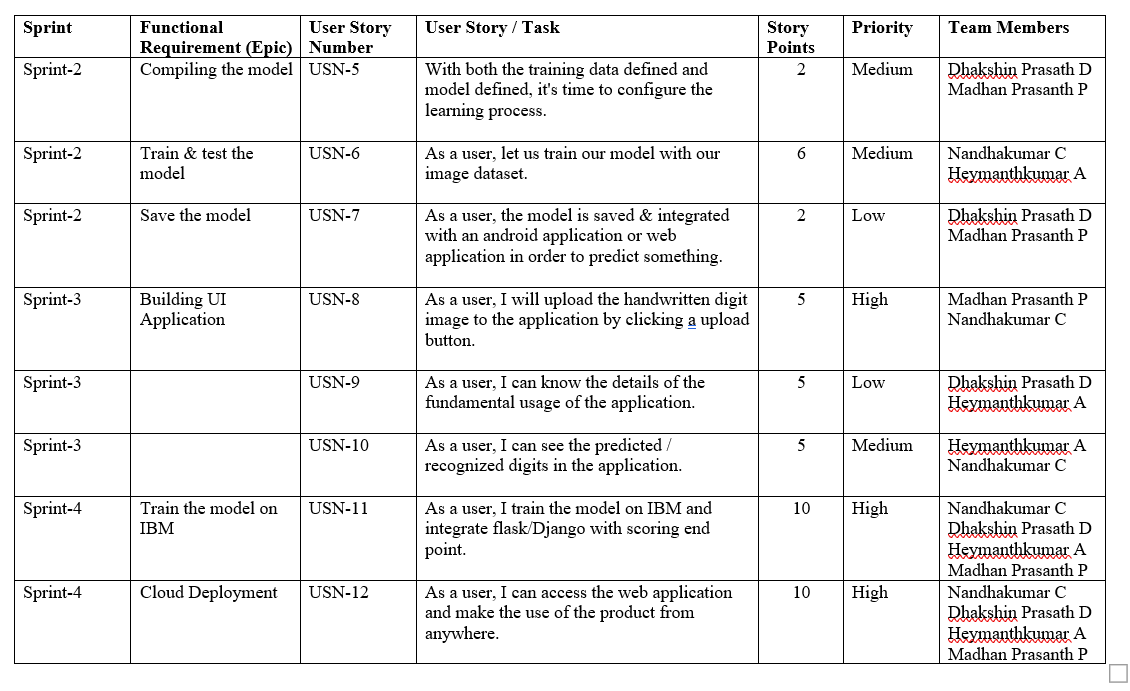
* 1. User Stories



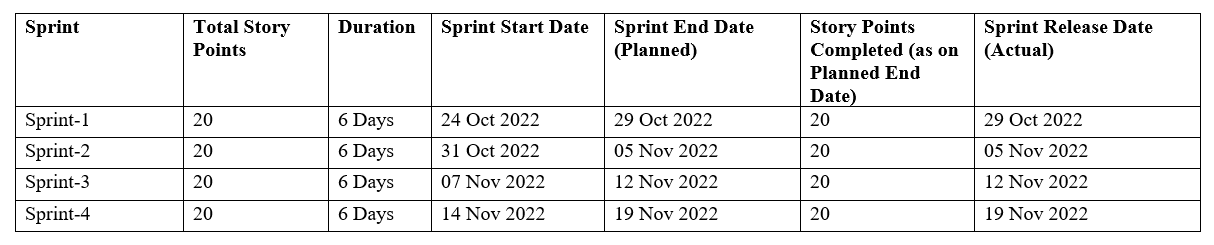
1. **PROJECT PLANNING & SCHEDULING**

6.1.Sprint Planning & Estimation

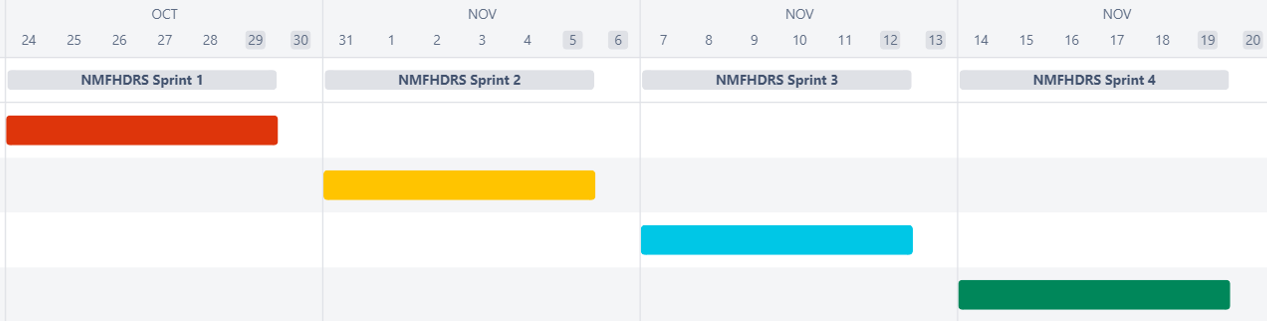




6.2.Sprint Delivery Schedule



6.3.Reports from JIRA



1. **CODING & SOLUTIONING :**

**IMPORTING THE REQUIRED LIBRARIES :**

import numpy as np

import tensorflow #open source used for both ML and DL for computation

from tensorflow.keras.datasets import mnist #mnist dataset

from tensorflow.keras.models import Sequential #it is a plain stack of layers

from tensorflow.keras import layers #A Layer consists of a tensor- in tensor-out computat ion funct ion

from tensorflow.keras.layers import Dense, Flatten #Dense-Dense Layer is the regular deeply connected r

#faltten -used fot flattening the input or change the dimension

from tensorflow.keras.layers import Conv2D #onvoLutiona l Layer

from keras.optimizers import Adam #opt imizer

from keras. utils import np\_utils #used for one-hot encoding

import matplotlib.pyplot as plt   #used for data visualization

**LOAD DATA**

(x\_train, y\_train), (x\_test, y\_test)=mnist.load\_data () #splitting the mnist data into train and test

**RESHAPING DATASET**

#Reshaping to format which CNN expects (batch, height, width, channels)

x\_train=x\_train.reshape (60000, 28, 28, 1).astype('float32')

x\_test=x\_test.reshape (10000, 28, 28, 1).astype ('float32')

**APPLYING ONE HOT ENCODING**

number\_of\_classes = 10  #storing the no of classes in a variable

y\_train = np\_utils.to\_categorical (y\_train, number\_of\_classes) #converts the output in binary format

y\_test = np\_utils.to\_categorical (y\_test, number\_of\_classes)

**ADD CNN LAYERS**

#create model

model=Sequential ()

#adding modeL Layer

model.add(Conv2D(64, (3, 3), input\_shape=(28, 28, 1), activation='relu'))

model.add(Conv2D(32, (3, 3), activation = 'relu'))

#flatten the dimension of the image

model.add(Flatten())

#output layer with 10 neurons

model.add(Dense(number\_of\_classes,activation = 'softmax'))

1. **TESTING** 
   1. Test Cases

TRAINING THE MODEL

#fit the model

model.fit(x\_train, y\_train, validation\_data=(x\_test, y\_test), epochs=5, batch\_size=32)

OBSERVING THE METRICS

# Final evaluation of the model

metrics = model.evaluate(x\_test, y\_test, verbose=0)

print("Metrics (Test loss &Test Accuracy) : ")

print(metrics)

prediction=model.predict(x\_test[6000:6001])

print(prediction)

plt.imshow(x\_test[6000])

* 1. User Acceptance Testing

TEST MODEL

from tensorflow.keras.models import load\_model

from keras.preprocessing import image

from PIL import Image

import numpy as np

model = load\_model("trainedModel.h5")

import os, types

import pandas as pd

from botocore.client import Config

import ibm\_boto3

def \_\_iter\_\_(self): return 0

# @hidden\_cell

# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.

# You might want to remove those credentials before you share the notebook.

cos\_client = ibm\_boto3.client(service\_name='s3',

    ibm\_api\_key\_id='xglVQkzEf5qe6v6VWbdq9KCLfz7-8qScQQe9t9umoy8W',

    ibm\_auth\_endpoint="https://iam.cloud.ibm.com/oidc/token",

    config=Config(signature\_version='oauth'),

    endpoint\_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'anovelmethodforhandwrittendigitre-donotdelete-pr-waapzywq80yk7w'

object\_key = 'mnist-dataset-1024x424 (5).png'

streaming\_body\_2 = cos\_client.get\_object(Bucket=bucket, Key=object\_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.

# Please read the documentation of ibm\_boto3 and pandas to learn more about the possibilities to load the data.

# ibm\_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/

# pandas documentation: http://pandas.pydata.org/

1. **RESULTS**
   1. Performance Metrics

img = Image.open(streaming\_body\_2).convert("L") # convert image to monochrome

img = img.resize( (28,28) ) # resizing of input image

im2arr = np.array(img) #converting to image

im2arr = im2arr.reshape(1, 28, 28, 1) #reshaping according to our requirement

pred = model.predict(im2arr)

print(pred)

1. **CONCLUSION**

Our project HANDWRITTEN DIGIT RECOGNITION deals with identifying the digits. The main purpose of this project is to build an automatic handwritten digit recognition method for the recognition of handwritten digit strings.

1. **FUTURE SCOPE**

The proposed system takes 28x28 pixel sized images as input. The same system with further modifications and improvements in the dataset and the model can be used to build Handwritten Character Recognition System which recognizes human handwritten characters and predicts the output

1. **APPENDIX**

Source Code:

<https://github.com/IBM-EPBL/IBM-Project-31596-1660203292/tree/main/Final%20Deliverables/Final%20Code>

GitHub & Project Demo Link

<https://github.com/IBM-EPBL/IBM-Project-31596-1660203292>

https://github.com/IBM-EPBL/IBM-Project-31596-1660203292/blob/main/Final%20Deliverables/Demo%20Video/Demo%20Video.mkv