A NOVEL METHOD FOR HADNWRITTEN DIGIT RECOGNITION SYSTEM

THE PROJECT REPORT

Submitted by

Team ID : PNT2022TMID21979

ADHIKESAV.M (113019104005)

DHANUSH.U(113019104024)

NIKESH.K(113019104030)

GOWTHARAJ.A(113019104057)

BACHELOR OF ENGINEEERING

IN

COMPUTER SCIENCE AND ENGINEERING

TABLE OF CONTENTS

ABSTRACT

One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Building an automatic handwritten digit recognition method is the major goal of this project, which will be used to recognise handwritten digit strings. The segmentation of the digits into separate digits is the first step in completing the recognition challenge. The handwritten digit string recognition challenge is then completed by using a digit recognition module to categorise each segmented digit. Applications for digit recognition include filling out forms, processing bank checks, and sorting mail. The capacity to create an effective algorithm that can distinguish handwritten digits and which is submitted by users via a scanner, tablet, and other digital devices is at the core of the issue.

List of Figure:

Abbreviations :

1. **CNN** – Convolutional Neural Network.
2. **MNIST-** Modified National Institute of Standard and Science Database.

Chapter 1

INTRODUCTION:

a. Project overview:

Artificial intelligence and computer technology both heavily rely on machine learning and deep learning. Human effort in identifying, learning, making predictions, and many other areas can be decreased with the application of deep learning and machine learning.

This report compares classifiers like KNN, PSVM, NN, and convolution neural network on the basis of performance, accuracy, time, sensitivity, positive productivity, and specificity while using different parameters with the classifiers. The handwritten digits (0 to 9) from the well-known MNIST dataset are recognised.

The developers are delving into machine learning and deep learning methods to make machines more intelligent. A human learns to accomplish a task by practising and repeating it repeatedly until the skill is learned by memory. then his brain's neurons.

b. Purpose :

The objective of this project is to develop a model that can identify and calculate the handwritten numbers from a picture using convolution neural network principles. Though the objective is to develop a model that can recognise digits, it may also be used to letters and a person's handwriting. Understanding Convolutional Neural Network and using it to improve the handwritten recognition system is the main objective of the presented system.

Chapter 2

LITERATURE SURVEY:

2.1 Existing Problem:

Digit recognition system is the working of a machine to train itself or recognizing the digits from different sources like images, etc. and in different real-world scenarios for online handwriting recognition on computer tablets or system, recognize number plates of , numeric entries in forms filled up by hand and so on.

2.2 References:

Paper 1: Novel Deep Neural Network Model for Handwritten Digit Classification and Recognition

Year: 2021 Authors: Ayush Kumar Agrawal and Vineet Kumar Awasthi

An artificial neural network has one hidden layer between the input and output layers, whereas a deep neural network has numerous hidden layers with input and output layers. Deep neural networks use several hidden layers to increase model performance and achieve higher accuracy compared to accuracy of machine learning models. Most researchers do their research in the area of pattern recognition. In the field of pattern recognition, there are many patterns that can be used, including handwritten numbers, characters, pictures, faces, sounds, and speech. This study focuses on the classification and recognition of handwritten digits.1000 were utilized as test samples and 1000 were training samples.10000 picture samples make up the USPS dataset, of which 7291 serve as training samples and 2007 serve as testing samples. We've used the proposed deep neural network technique in this paper to classify and identify data from the ARDIS and USPS datasets. The suggested model consists of six layers with SoftMax and relu activation functions. After model implementation, accuracy for ARDIS samples reached 98.70% testing and 99.76% training, which is greater

than accuracy from prior research. Additionally, using the USPS samples dataset, 98.22%

training accuracy and 93.01% testing accuracy were attained. When compared to earlier methodologies, the data show that deep neural networks perform incredibly well.

Paper 2: A Novel Handwritten Digit Classification System Based on Convolutional Neural Network Approach

Year: 2021 Authors: Ali Abdullah Yahya, Jieqing Tan, Min Hu

There have been a tonne of CNN classification algorithms put forth in the literature. However, these algorithms do not take into account the proper filter size selection, data preparation, dataset restrictions, or noise. As a result, few algorithms have been able to significantly increase classification accuracy. The paper makes the following contributions to solve these methods' drawbacks: First, the size of the effective receptive field (ERF) is determined after taking domain knowledge into account. They choose a typical filter size with the aid of the ERF calculation, improving the classification accuracy of our CNN. Second, excessive data produces inaccurate results, which has a detrimental impact on classification accuracy. Before carrying out the data classification task, data preparation is conducted to ensure that the dataset is devoid of any redundant or irrelevant variables to the goal variable. Thirdly, data augmentation has been suggested as a way to reduce training and validation errors and prevent dataset limitations. Fourthly, the paper suggests adding an additive white Gaussian noise with a threshold of 0.5 to the MNIST dataset in order to imitate the natural factors that can affect image quality in the real world. With a recognition accuracy of 99.98% and 99.40% with 50% noise, our CNN algorithm achieves state-of-the-art performance in handwritten digit recognition.

Paper 3: Handwritten Character Recognition using Neural Network and TensorFlow

Year : 2019 Authors : Megha Agarwal, Shalika, Vinam Tomar, Priyanka Gupta

The offline handwritten character recognition in this study will be carried out using TensorFlow and a convolutional neural network. a process known as using SoftMax

Regression, one may assign probabilities to one of the many characters in the handwritten text that offers the range of values from 0 to 1, summed to 1. The objective is to create software that is extremely accurate and that has a minimum level of spatial and temporal complexity. It was determined that strategies for feature extraction like diagonal and direction are significantly better at producing high accuracy. Outcomes in comparison to other conventional vertical and horizontal techniques moreover use the best Neural network tried layers provides the benefit of a higher accurate outcome by having a high noise tolerance. The feed forward model in neural networks is the back-propagation algorithm that was primarily used to classify the characters, recognise them, and receive training continually more. In addition to these, normalizing along with feature extraction, the results were better and more effective. Character recognition is the outcome of accuracy. The paper will describe the best approach to get more than 90% accuracy in the field of Handwritten Character Recognition (HCR).

Paper 4: Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN)

Year : 2020 Authors: Savita Ahlawat , Amit Choudhary , Anand Nayyar , Saurabh Singh and Byungun Yoon

Customized features and a vast quantity of past knowledge have been used in traditional handwriting recognition systems. It is difficult to train an optical character recognition (OCR) system based on these conditions. Deep learning approaches have enabled significant performance in the field of handwriting recognition research in recent years. Nonetheless, the increasing increase in the amount of handwritten data, along with the availability of vast computing capacity, necessitates improvements in recognition accuracy and warrants additional exploration. Convolutional neural networks (CNNs) are extremely excellent in perceiving the structure of handwritten characters/words in ways that aid in the automatic extraction of distinguishing features, making CNN the best solution for solving handwriting recognition challenges. The proposed work aims to investigate several design alternatives for CNN-based handwritten digit recognition, such as the number of layers, stride size, receptive field, kernel size, padding, and dilution. Furthermore, we intend to assess the effectiveness of several SGD optimization techniques in enhancing the performance of handwritten digit recognition. Using ensemble architecture improves the recognition accuracy of a network. In this case, we want to obtain equal accuracy by employing a pure CNN design without ensemble architecture, because ensemble structures increase computational overhead and testing complexity. As a result, a CNN design is developed in order to obtain higher accuracy than ensemble systems while reducing operational complexity and expense. Furthermore, we demonstrate an appropriate combination of learning parameters in the design of a CNN that leads us to a new absolute record in categorising MNIST handwritten digits. We conducted extensive trials and achieved 99.87% recognition accuracy for an MNIST dataset

Paper 5: Devnagari Numeral Recognition By Combining Decision Of Multiple Connectionist Classifiers

Year : 2002 Authors : Reena Bajaj, Lipika Dey, and S. Chaudhury

The classifiers used in this approach are based upon different representations of the input pattern. These representations, since they encode different types of property – style dependent and style-invariant, cannot be combined into a single monolithic feature vector.

Individual classifiers dealing with these representations output the class labels depending on the features used. These class labels have been combined using a meta-pi network because it can devise a combination scheme on the basis of consistency and accuracy of the individual classifiers. Each output unit of the meta-pi net modulates output of individual classifiers. The network is called the meta-pi network owing to the multiplicative function that its output units perform. This function serves to combine the outputs of sub-networks (or “modules”), independently trained to classify numerals based on different features. Hence, there are three output nodes in the present meta-pi net. The network has been trained with the samples used for training the individual nets along with newer examples. Through the training process the meta-pi net learns to choose any one of the valid classifier outputs or a combination of these valid outputs to produce the correct global output. The initial stage of the proposed architecture consists of connectionist modules for style based categorisation of the input. Since style groups are characteristic of each character, for each character there exists a style categorisation module which acquires knowledge about style categories of the corresponding character through unsupervised learning. Output of this stage would indicate similarity of an unknown input with style categories of the different characters (including the correct one). An interesting feature of this stage is that the classifiers are not forced to classify distinct looking samples of one character into one monolithic class. The classifiers can self-organise themselves to categorise them into distinct style categories. The novel feature of this work is the approach followed for identification and integration of style specific information in the recognition scheme. Use of multiple classifiers using the meta-pi network is another significant feature of this work. A complete hierarchical recognition architecture has been suggested in this work.

Paper 6: A novel method for Handwritten Digit Recognition with Neural Networks

Authors : MALOTHU NAGU, N VIJAY SHANKAR, K. ANNAPURNA

Character recognition plays an important role in the modern world. It can solve more complex problems and makes humans' job easier. An example is handwritten character recognition. Pattern recognition system consists of two-stage process(Feature Extraction and Classification). Feature extraction is the measurement on a population of entities that will be classified. This assists the classification stage by looking for features that allows fairly easy to distinguish between the different classes. There are Several Pattern Recognition Methods, they are: Bayesian decision theory, Nearest Neighbour rule, Linear Classification Discrimination The Bayesian decision theory is a system that minimizes the classification error. This method is based on priority basis, it classifies using priority information about something that we would like to classify. We can use Baye's formula, which states the following: $P(wj |x) = p(x|wj) P(wj)/p(x)$. The Nearest Neighbour (NN) rule is used to classify handwritten characters. The distance measured between the two-character images is needed in order to use this rule. The goal of Linear Classification is to assign observations into the classes. This can be used to establish a classifier rule so that it can assign a new observation into a class. In another words, the rule deals with assigning a new point in a vector space to a class separated by a boundary. Linear classification provides a mathematical formula to predict a binary result. This result is a true or false (positive or negative) result. The following is an equation that can be stated as the discriminator: $a1\ x1 + a2\ x2 + ... + an\ xn > x0$ Shape describes a spatial region. Most shapes are a 2-D space. Shape recognition works on the similarity measure so that it can determine that two shapes correspond to each other. The recognition needs to respect the properties of imperfect perception, for example: noise, rotation, shearing, etc. One of the techniques used in shape recognition is elastic matching distance. The difficult task is there are some handwritten digits that often run together or not fully connected. Numeral 5 is an example. But once these tasks have been carried out, the digits are available as individual items. But the digits are still in different sizes. Therefore, a normalization step has to be performed so we can have to have digits in equal sizes. After the digits are normalized, they are fed into the ANN. This is a feed-forward network with three hidden layers. The input is a 16 x 16 array that corresponds to the size of a normalized pixel image. The first hidden layer contains 12 groups of units with 64 units per group. Each unit in the group is connected to a 5 x 5 square in the input array and all 64 units in the group have the same 25 weight values. The second hidden layer consists of 12 groups of 16 units. This layer operates very similar to the first hidden layer, but now it seeks features in the first hidden layer. The third hidden layer consists of 30 units that are fully connected to the units in the previous layer. The output units are in turn fully connected to the third hidden layer.

Paper 7: A novel method for Handwritten Digit Recognition with Neural Networks

Authors: Cheng-Lin Liu, K. Nakashima, H. Sako, H. Fujisawa

This paper presents the latest results of handwritten digit recognition on well-known image databases using the state of-the-art feature extraction and classification techniques. The tested databases are CENPARMI, CEDAR, and MNIST. On the test dataset of each database, 56 recognition accuracies are given by combining 7 classifiers with 8 feature vectors. All the classifiers and feature vectors give high accuracies. Among the features, the chain-code feature and gradient feature show advantages, and the profile structure feature shows efficiency as a complementary feature. In comparison of classifiers, the support vector classifier with RBF kernel gives the highest accuracy but is extremely expensive in storage and computation. Among the non-SV classifiers, the polynomial classifier performs best, followed by a learning quadratic discriminant function classifier.

The results are competitive compared to previous ones and they provide a baseline for evaluation of future works.

Paper 8: Handwritten Digit Recognition Using Structural, Statistical Features and K-nearest Neighbour Classifier

Authors: U. Ravi Babu, Y. Venkateswarlu, Aneel Kumar Chintha

This paper presents a new approach to off-line handwritten digit recognition based on structural features which does not require thinning operation and size normalization technique. This paper uses four different types of structural features namely, number of holes, water reservoirs in four directions, maximum profile distances in four directions, and fill-hole density for the recognition of digits. The digit recognition system mainly depends on which kinds of features are used. The main objective of this paper is to provide efficient and reliable techniques for recognition of handwritten digits. A Euclidean minimum distance criterion is used to find minimum distances and k-nearest neighbour classifier is used to classify the digits. A MNIST database is used for both training and testing the system. 5000 images are used to test the proposed method a total 5000 numeral images are tested and get 96.94% recognition rate.

Paper 7: A Methodology for Feature Selection Using Multi-Objective Genetic Algorithms for Handwritten Digit String Recognition

Authors: L.S. Oliveira, R. Sabourin, F. Bortolozzi, C.Y

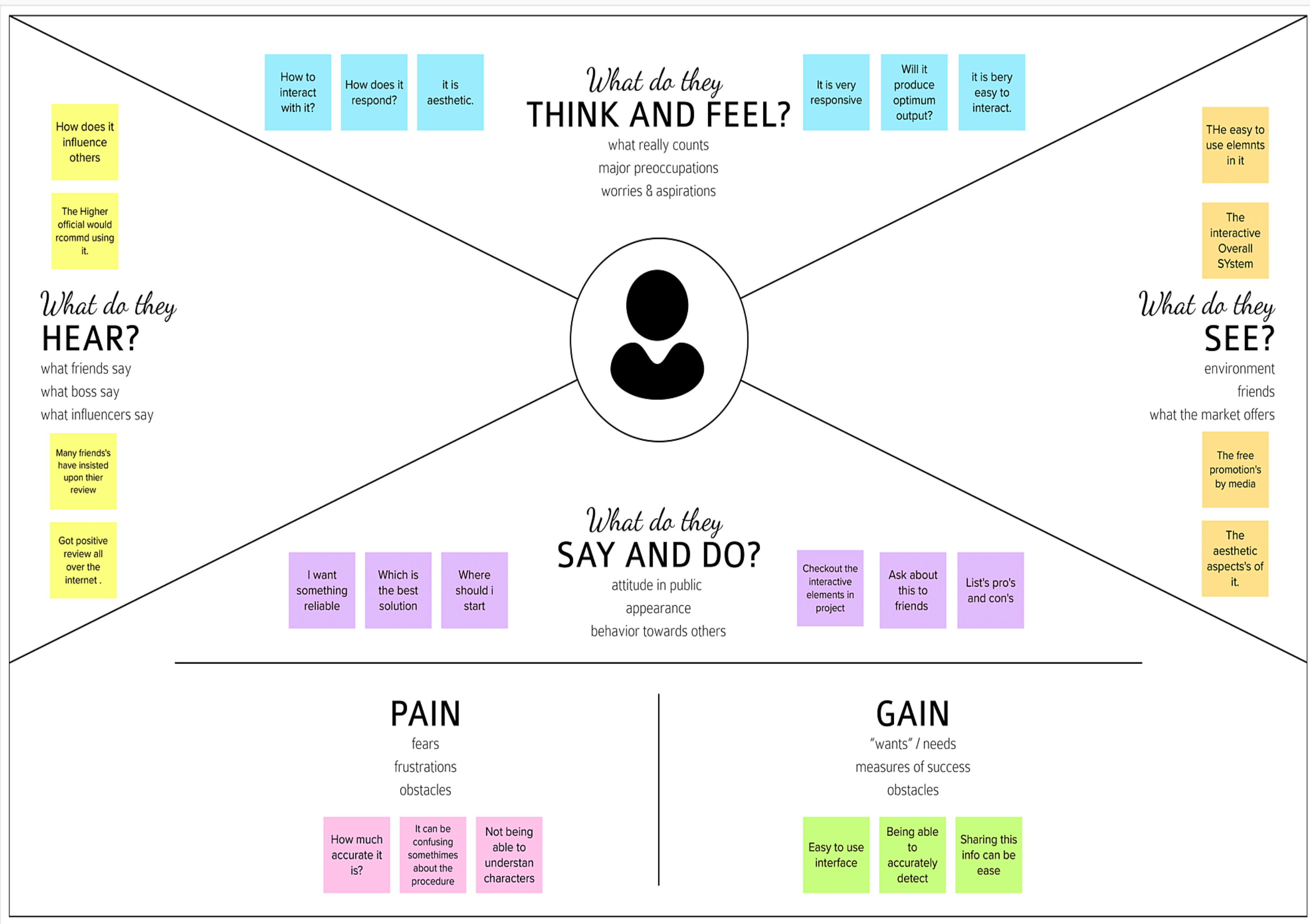
Suen discusses the use of genetic algorithms for feature selection for handwriting recognition. Its novelty lies in the use of multi-objective genetic algorithms where sensitivity analysis and neural networks are employed to allow the use of a representative database to evaluate fitness and the use of a validation database to identify the subsets of selected features that provide a good generalization. Comprehensive experiments on the NIST database confirm the effectiveness of the proposed strategy

2.3 Problem Statement Defintion:

In today's society, character recognition is becoming increasingly vital. It facilitates human work and aids with the resolution of more difficult issues. One illustration is handwritten character recognition, which is extensively used worldwide. This technique was created to recognise zip codes or postal codes for use in mail sorting. This can aid people in the difficult-to-read postal code mail sorting process. Researchers have been working on handwriting recognition for more than thirty years. The number of firms participating in handwriting recognition research has steadily expanded over the last several years. Handwriting processing has advanced due to a mix of factors such as improved recognition rates and the usage of complicated systems.

Chapter 3

IDEATION AND PROPOSED SOLUTION:



3.1 Empathy map canvas :

Fig3.1 : Empathy map canvas

3.2 Ideation And Brainstorming :

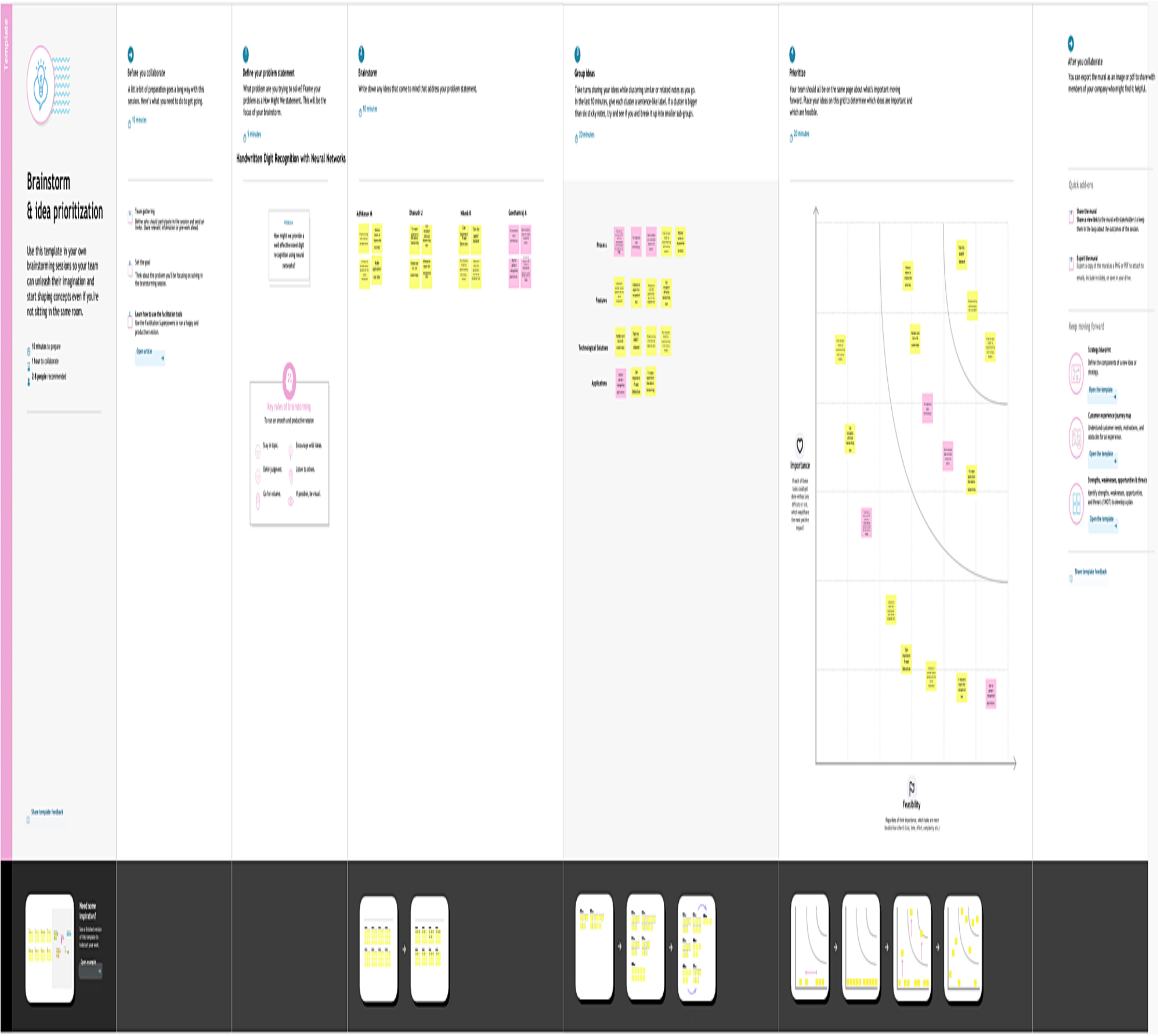


Fig .3.2: **Ideation And Brainstorming**

3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	In the modern world, digit recognition is crucial. It is capable of solving increasingly difficult problems and making humans' jobs easier. Handwritten digit recognition is one example. This is a worldwide system for recognizing zip codes or postal codes for mail sorting. Handwritten digit recognition can be accomplished using a variety of approaches. The machine has a difficult duty because handwritten digits are not flawless and can be generated with a variety of flavours. The solution to this issue is handwritten digit recognition, which uses an image of a digit and identifies the digit represented in the image.
2.	Idea / Solution description	Handwritten digit recognition is performed using the MNIST dataset which contains 60,000 training images of handwritten digits from zero to nine and 10,000 images for testing. So, the MNIST dataset has 10 different classes. In this project, we are going to implement a handwritten digit recognition application trained using the Convolutional Neural Networks model. In the end, a GUI is built where the user gives the handwritten digit as input where it is recognized and the result is displayed immediately.

3.	Novelty / Uniqueness	This project introduces an operative strategy for dealing with novelty in the handwritten visual recognition domain. A perfect transcription agent would be able to distinguish known and unknown characters in a picture, as well as determine any aesthetic variations that may occur inside or between texts. The existence of novelty has shown to be a major stumbling block for even the most robust machine learning-based algorithms for these activities. Novelty in handwritten papers might include, among other things, a change in the writer, character properties, writing attributes, or overall document appearance. Instead of examining each element separately, we believe that an integrated agent capable of processing known characters and novelties concurrently is a superior technique. The handwritten digit recognition problem can be seen as a subtask of the optical character recognition (OCR) problem.

4.	Social Impact/ Customer Satisfaction	There are many benefits associated with the handwriting recognition system. In addition to reading postal addresses and bank check amounts, it is also useful for reading forms. Furthermore, it's used in fraud detection because it makes it easy to compare two texts and determine which one is a copy. As a result, this system fulfils customers' expectations, as it is a novel method for recognizing handwritten digits, ensuring high accuracy for the model and meeting all customer expectations. Users will save a lot of time and effort if the system provides various synonyms for the words recognized. Due to the fact that the users in rural areas will be using their own regional language, this proposed system should be able to detect those digits as well. As the system is being used in socially crowded places such as banks to check amounts, it should be fast and reliable. As it is designed to solve real-world problems, it should be highly reliable and trustworthy in every way, and users throughout the world should be able to use it effectively.
5.	Business Model (Revenue Model)	A revenue model means understanding how a start-up can make money. Our major revenue sources consist of sales, government funds, and public donations. The introduction of novel ideas increases revenue streams, such as introducing gesture or touch features , voice read out of recognised digits, etc..

6.	Scalability of the Solution	One of the approaches to make the handwritten digit recognition system scalable is to make use of cloud-native methods. For example, one of the cloud solutions for making AI scalable is IBM Cloud. IBM Cloud Build helps run and manage AI models, optimize decisions at scale across any cloud. The advantage of using cloud to make solutions scalable is that we can deploy our AI application on the specific cloud environment that best supports our business needs. We can take advantage of builtin security capabilities and AI model monitoring. We can Automate AI lifecycles with Model Ops pipelines, deploy and run models through one-click integration and also prepare and build models visually and programmatically. Looking at these advantages, we can drive better business outcomes by optimizing our decisions and also make our solution scalable using cloud

3.4 Problem Solution Fit:

S.No.	Parameter	Description
1.	CUSTOMER SEGMENT(S):	The Customers who deal with handwritten digits like Banking sectors , schools , colleges , railways , firms , etc.
2.	JOBS-TO-BEDONE/PROBLEMS:	Handwritten digits can be difficult to understand and interpret at times. It may cause errors when dealing with rough handwriting.
3.	TRIGGERS	To obtain the numbers accurately and quickly.
4.	EMOTIONS :BEFORE/AFTER	Feels frustrated and sad when numbers are not entered.

5.	AVAILABLE SOLUTIONS	There are no widely used software's to detect handwriting; instead, they check with other people to affirm what number it is.
6.	CUSTOMER CONSTRAINT(S):	They believe that the alternatives will result in errors and faults and will be inconvenient.
7.	BEHAVIOUR	Finding the best software for detecting accurate digits in a more efficient manner
8.	CHANNELS OF BEHAVIOUR	Using software that is available on the internet. Obtaining assistance from those nearby in order to recognise the digits written by their customers.
9.	PROBLEM ROOT CAUSE	We face numerous challenges in handwritten number recognition. because of different people's jotting styles and the lack of Optic character recognition This investigation offers an in-depth comparison of various machine literacy and deep literacy
10.	YOUR SOLUTION	A solution to this problem is the Handwritten digit recognition system, which uses a picture of a digit and recognises the digit present in the image. Convolutional Neural Network model built with PyTorch and applied to the MNIST dataset to recognise handwritten digits

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional Requirements:

Following are the functional requirements of the proposed solution.

FR-NO:	Sub Requirement (Story / Sub-Task)
FR-1	Image Data: Handwritten digit recognition refers to a computer's capacity to identify human handwritten digits from a variety of sources, such as photographs, documents, touch screens, etc., and categorise them into ten established classifications (0-9). In the realm of deep learning, this has been the subject of countless studies.
FR-2	Website: Web hosting makes the code, graphics, and other items that make up a website accessible online. A server hosts every website you've ever visited. The type of hosting determines how much space is allotted to a website on a server. Shared, dedicated, VPS, and reseller hosting are the four basic varieties.
FR-3	Digit Classifier Model: To train a convolutional network to predict the digit from an image, use the MNIST database of handwritten digits. get the training and validation data first.
FR-4	Cloud: The cloud offers a range of IT services, including virtual storage, networking, servers, databases, and applications. In plain English, cloud computing is described as a virtual platform that enables unlimited storage and access to your data over the internet.
FR-5	Modified National Institute of Standards and Technology dataset: The abbreviation MNIST stands for the MNIST dataset. It is a collection of 60,000 tiny square grayscale photographs, each measuring 28 by 28, comprising handwritten single digits between 0 and 9

4.2 Non-functional Requirements:

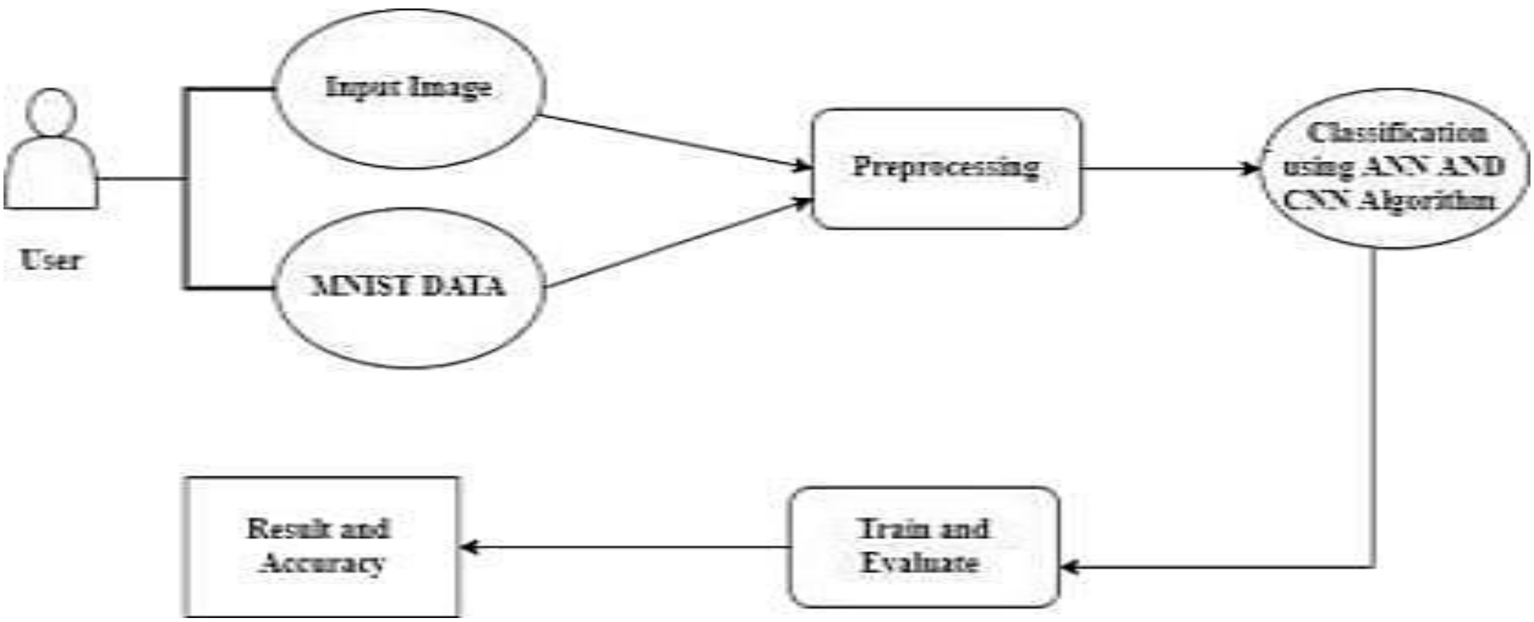
Following are the non-functional requirements of the proposed solution

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Applications for digit recognition include filling out forms, processing bank checks, and sorting mail.
NFR-2	Security	1) The system generates a thorough description of the instantiation parameters, which might reveal information like the writing style, in addition to a categorization of the digit. 2) The generative models are capable of segmentation driven by recognition. 3) The procedure uses a relatively.
NFR-3	Reliability	The samples are used by the neural network to automatically deduce rules for reading handwritten digits. Furthermore, the network may learn more about handwriting and hence enhance its accuracy by increasing the quantity of training instances. Numerous techniques and algorithms, such as Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests, etc., can be used to recognise handwritten numbers.
NFR-4	Accuracy	With typed text in high -quality photos, optical character recognition (OCR) technology offers accuracy rates of greater than 99%. However, variances in spacing, abnormalities in handwriting, and the variety of human writing styles result in less precise character identification.
NFR-5	Availability	It should be made available for everyone in the who can access the system.

CHAPTER 5

PROJECT DESIGN

5.1 Data flow diagrams:



Simplified :

Fig 5.1: Data flow diagrams

Industry Standard:

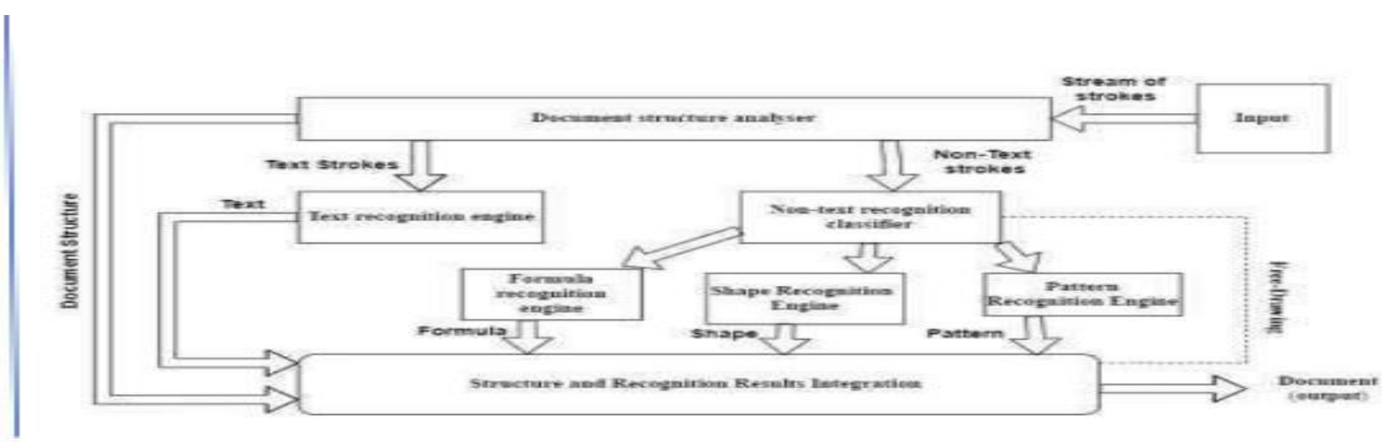


Fig 5.2 : Industry Standard

5.2 Solution And Technical Architecture :

Fig 5.3: Solution And Technical Architecture

Table-1 : Components & Technologies : References :-

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g.Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript /Angular Js /React Js etc.
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
7.	File Storage	File storage requirements	IBM Block Storage
10.	Machine LearningModel	Purpose of Machine Learning Model	Object Recognition Model
11.	Infrastructure (Server/ Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud ServerConfiguration	Local, Cloud Foundry

Table-2: Application Characteristics: Technology Stack

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Technology of Opensource framework
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASPetc.
3.	Scalable Architecture	Justify the scalability of architecture	3 – tier, Micro-services

4.	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 goodrecognition accuracies	Distributed servers, IBMcloud
5.	Performance	The standard implementations of neuralnetworks achieve an accuracy of ~ (98– 99) percent in correctly classifying the handwritten digits. .	Number of requests per sec, use of Cache,use ofCDN’s

5.3 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, Ican 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 onceI have registered for the application	I can receiveconfirmation email &click confirm	High	Sprint-1
		USN-3	As a user,I can register for the application through Facebook	I can register &access the dashboard with Facebook Login	Low	Sprint-2
	Home	USN-6	As a user, I can view the application’s home page whereI can read the instructions to use thisapplication	I can read instructions also and the home pageis user-friendly.	Low	Sprint-1

	Upload Image	USN-7	As a user, I can ableto input the images of digital documents to the application	As a user, I can able to inputthe images of digital documents to theapplicatio n	High	Sprint-3
	Predict	USN-8	As a user I can able to get the recognised digit as outputfrom the images of digital documents or images	I can access the recognized digits from digital document orimages	High	Sprint-3
		USN-9	As a user, I will trainand test the input to get themaximum accuracy of output.	I can able to train and testtheapplication until it gets maximum accuracy of theresult.	Mediu m	Sprint-4
Customer (Web user)	Accessibility	USN-10	As a user, I can usethe web application virtually anywhere.	I can use theapplication in any devicewith a browser	Mediu m	Sprint-4

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning &Estimation :

TITLE	DESCRIPTION	DATE
IDEATION PHASE	1. Literature survey 2. Empathy Map 3. Brainstorming 4. Problem Statement	29 August2022 - 17 September 2022
PROJECT DESIGN PHASE- I	1. Problem Solution Fit 2. Proposed Solution 3. Solution Architecture	19 September 2022 - 01 October 2022
PROJECT DESIGN PHASE-II	1. Requirement Analysis 2. Customer Journey 3. Data Flow Diagrams 4. Technical Architecture	03 October 2022 - 15 October 2022
PROJECT PLANNING PHASE	1. Sprint Delivery Plan 2. JIRA Files	17 October 2022 - 22 October 2022
PROJECT DEVELOPMENTPHASE	1. Sprint 1 2. Sprint 2 3. Sprint 3 4. Sprint 4	24 October 2022 - 19 November 2022

6.2 Sprint Delivery Schedule:

The following table represents user stories**:**

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	As a user, I can collectthe dataset from various resources with different handwritings.	8	High	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K
Sprint-1	Data Processing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test.	8	High	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K
Sprint-1	Model Building	USN-3	As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit.	5	High	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K
Sprint-2	Home page	USN-4	Description aboutHandwritten Digit Recognition System	8	Low	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K
Sprint-2			As a user, the model is saved & integrated withanandroid application or web applicationin order to predict.	13	Low	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K
Sprint-3	Upload Image	USN-8	As a user, I submit the required imagefor the prediction.	8	Medium	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K
Sprint-3	Result	USN-9	Predicted Numberwill be displayed.	13	High	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K

Sprint-4	Integrate the web app with the IBM model	USN-10	Use flask for the integration purpose.	13	High	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K
Sprint-4	Deploy the Model	USN-11	Deployment of model using IBM Watson Studio, objectstorage	8	Medium	Adhikesav.M Dhanush.U Gowthamraj.A NIkesk.K

Project Tracker:

Sprint	Total StoryPoints	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint ReleaseDate (Actual)
Sprint-1	21	6 Days	24 Oct 2022	29 Oct 2022	21	29 Oct 2022
Sprint-2	21	6 Days	31 Oct 2022	05 Nov 2022	21	05 Nov 2022
Sprint-3	21	6 Days	07 Nov 2022	12 Nov 2022	21	12 Nov 2022
Sprint-4	21	6 Days	14 Nov 2022	19 Nov 2022	21	19 Nov 2022

Velocity:

We have a 6-day sprint duration, and the velocity of the team is 21 (points per sprint). Let's calculate the team's average velocity .

$$AV = \underline{\text{sprint duration}} = \underline{21} = 3.5$$

velocity 6

6.3 JIRA Report:

Sprint-I:

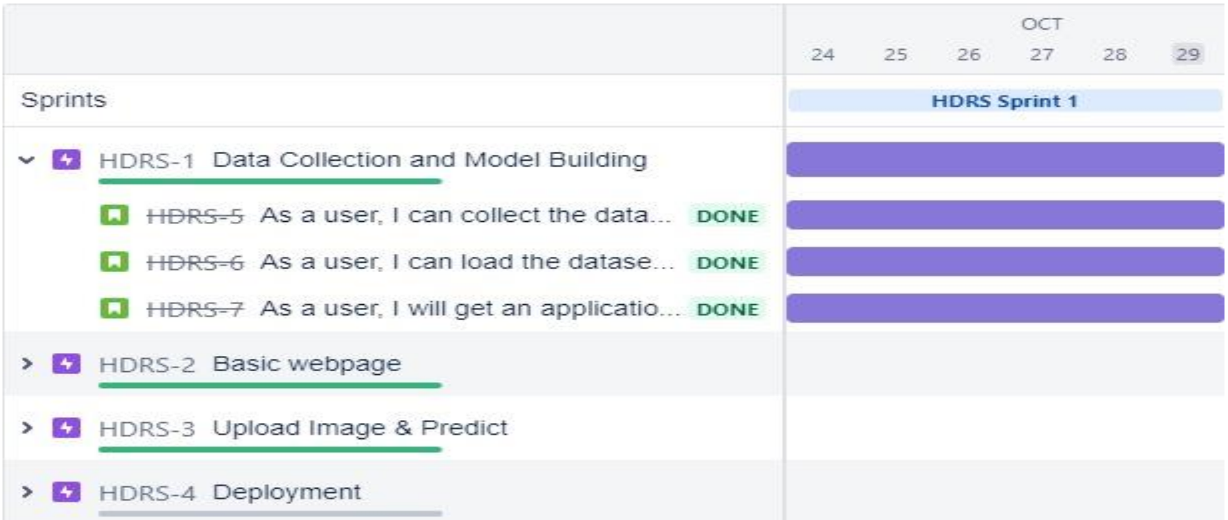


Fig 6.1 : Sprint-I

Sprint-II:

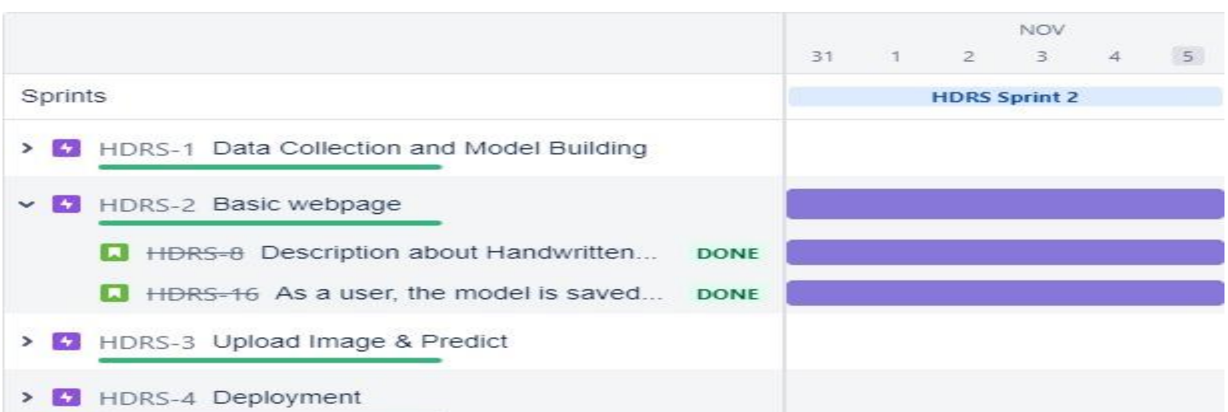


Fig 6.2: Sprint-II

Sprint-III:

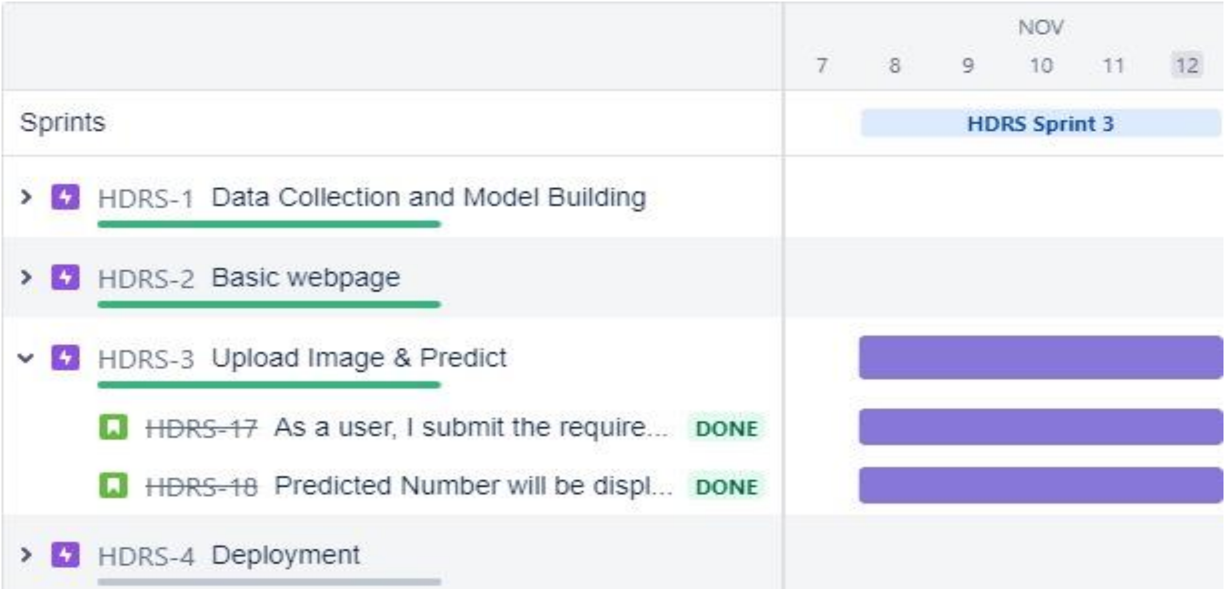


Fig 6.3: Sprint-III

Sprint-IV:

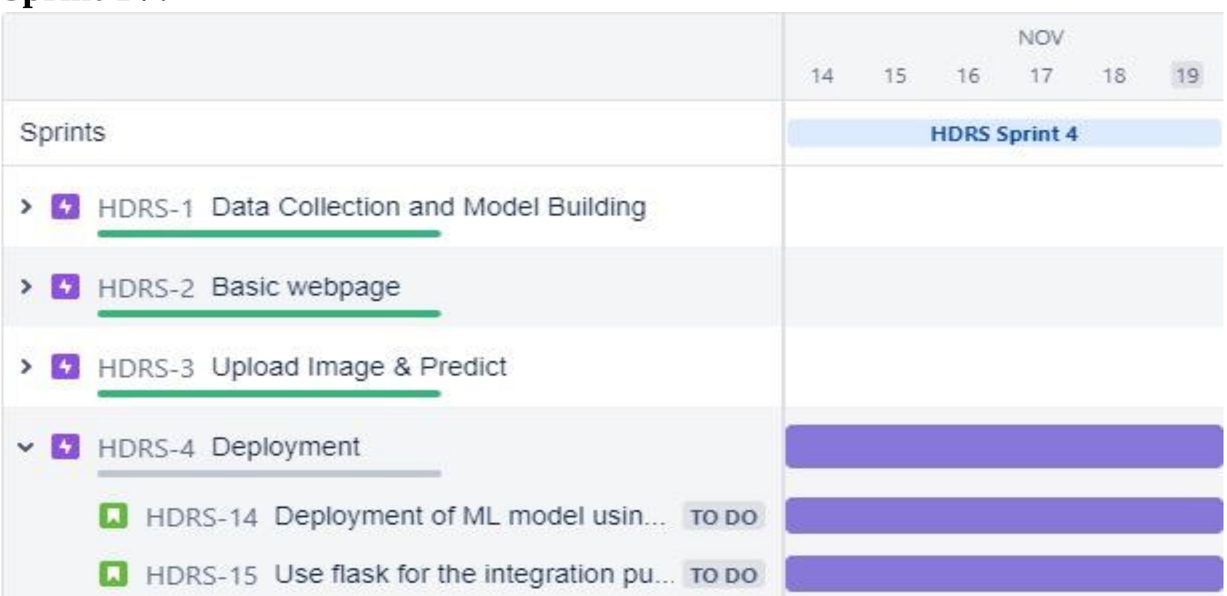


Fig 6.4: Sprint-IV

Overall :

The overall Report of the project is represented by the following figure :

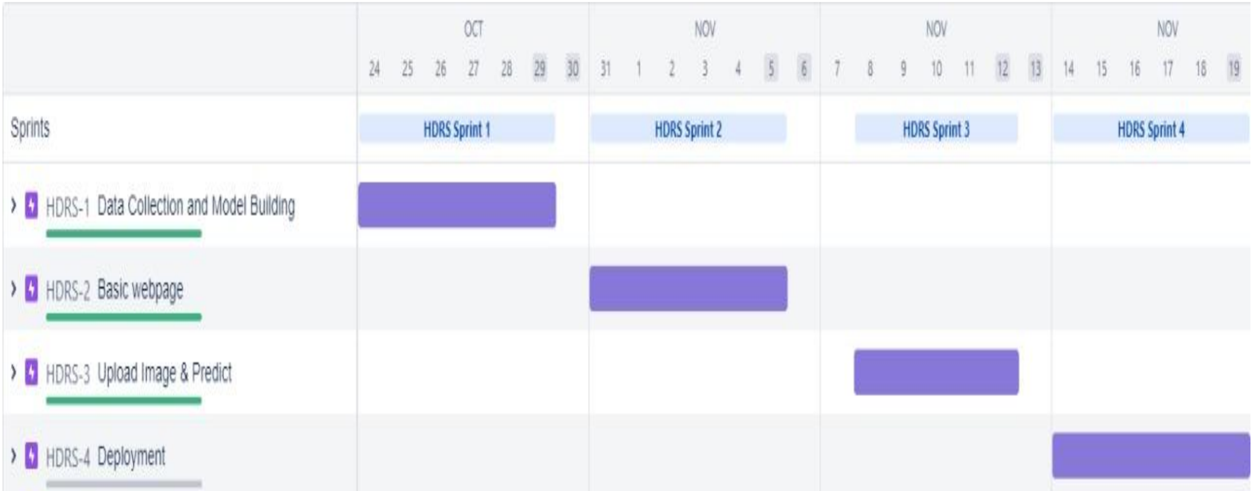
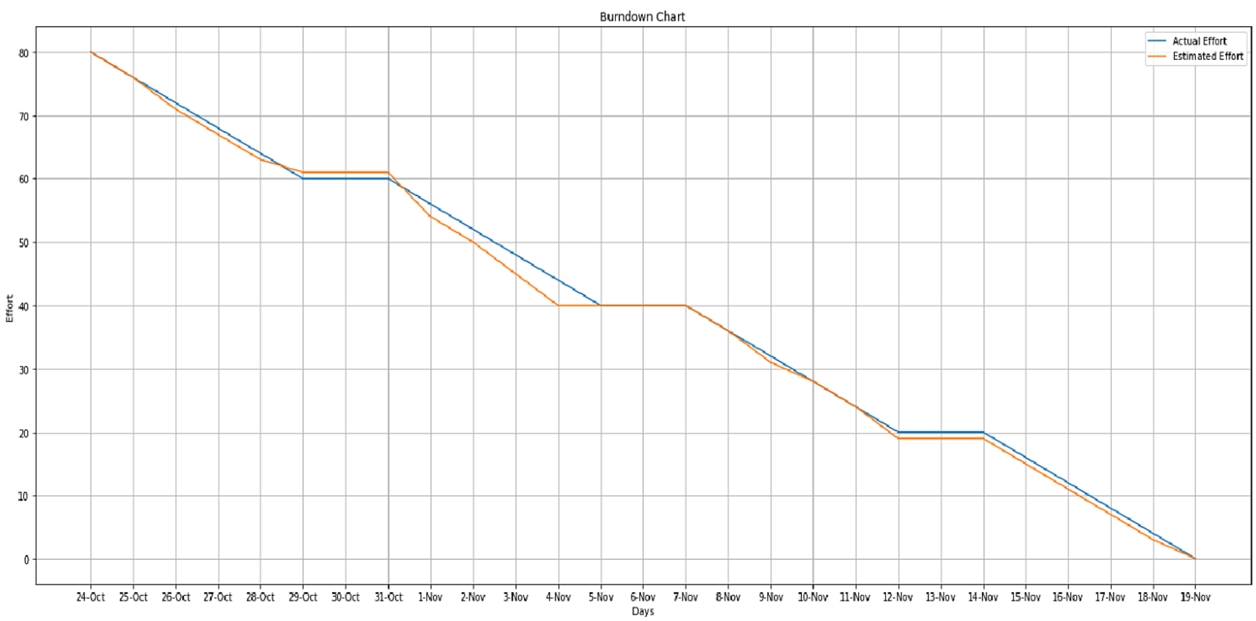


Fig 6.5: Overall

Burn Down Chart :

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies suchas Scrum. However, burn



downcharts can be applied to any project containing measurable progress over time.

Fig 6.6 : Burn Down Chart

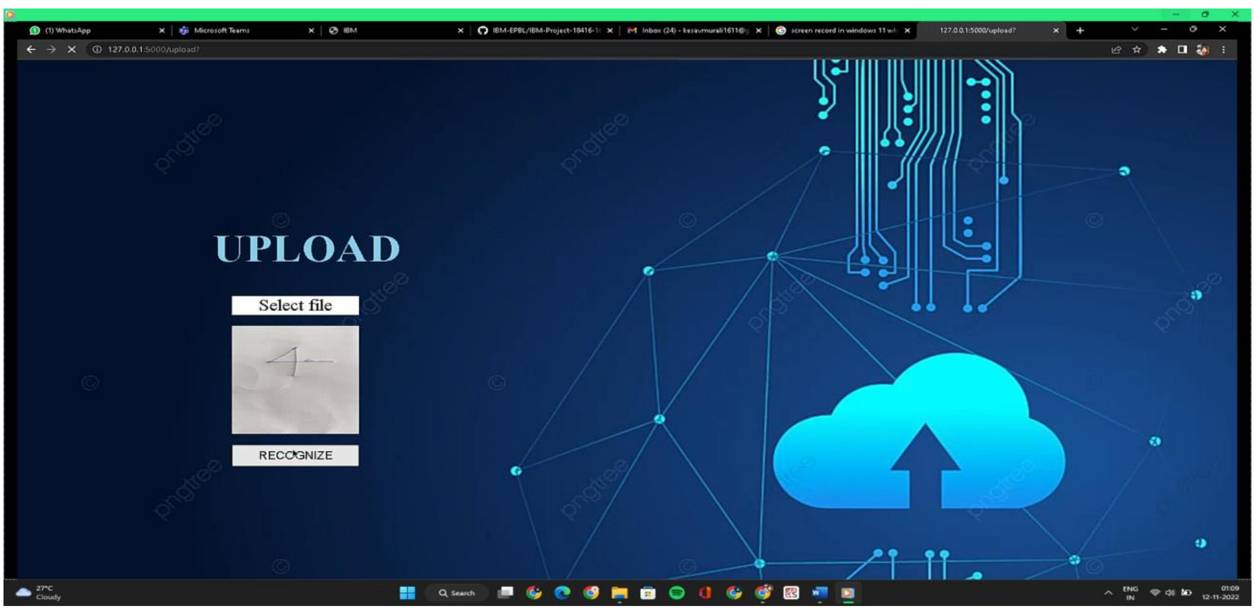
CHAPTER 7

CODING & SOLUTIONING

7.1 Feature 1 :

Preview of the image Being uploaded:

The first feature added to the code is a preview of the image that is being uploaded. This solves the problem of not knowing what file is being uploaded and also helps to decide whether the model is well trained. Using this preview, the validation of the answer with the given input can be performed and thus it can be ensured that the project is working properly. Preview also helps to clarify all the doubts regarding the working of the model which has been trained and tested in order to properly classify the image. It might interest you to learn the software is older than the OS inside today's Macs. Preview was part of the NextSTEP operating system that became the base of what we now call macOS. When part of NeXT it displayed and printed PostScript and TIFF files. Apple began weaving a



range of useful editing tools inside Preview when it launched Mac OS X Leopard in 2007.

Fig 7.1 : Preview

7.2 Feature 2 :

Upload again and again:

The Uploading is the transmission of a file from one computer system to another, usually larger computer system. The upload again button helps to upload a different file again as input and get different answer as the output . thus the upload again efficiently save time . it saves time by not forcing us to restart the whole app just to upload single image file . this efficiently improves the speed of accessibility and also save a lot of time discarding the need for restart of the whole application upload again is one of few features which might make a web application much more interactive and efficient in nature.

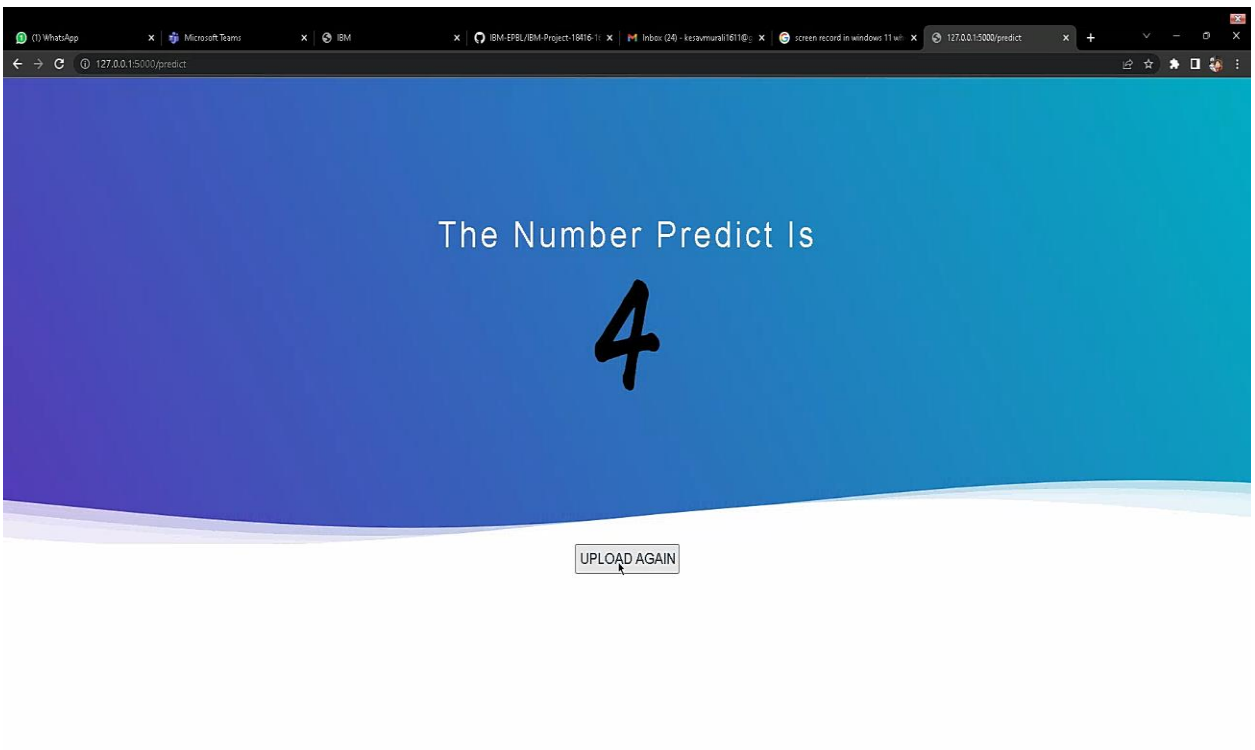


Fig 7.2: Upload again

7.3 Coding and solution :

Create An HTML File

- We use HTML to create the front end part of the web page.
- Here, we created 2 html pages- index.html, web.html.
- index.html displays home page.
- web.html accepts the values from the input and displays the prediction.
- For more information regarding HTML refer the link below

Let's see how our index.html file looks like

This is the main page which describes about the project and summarizes it.

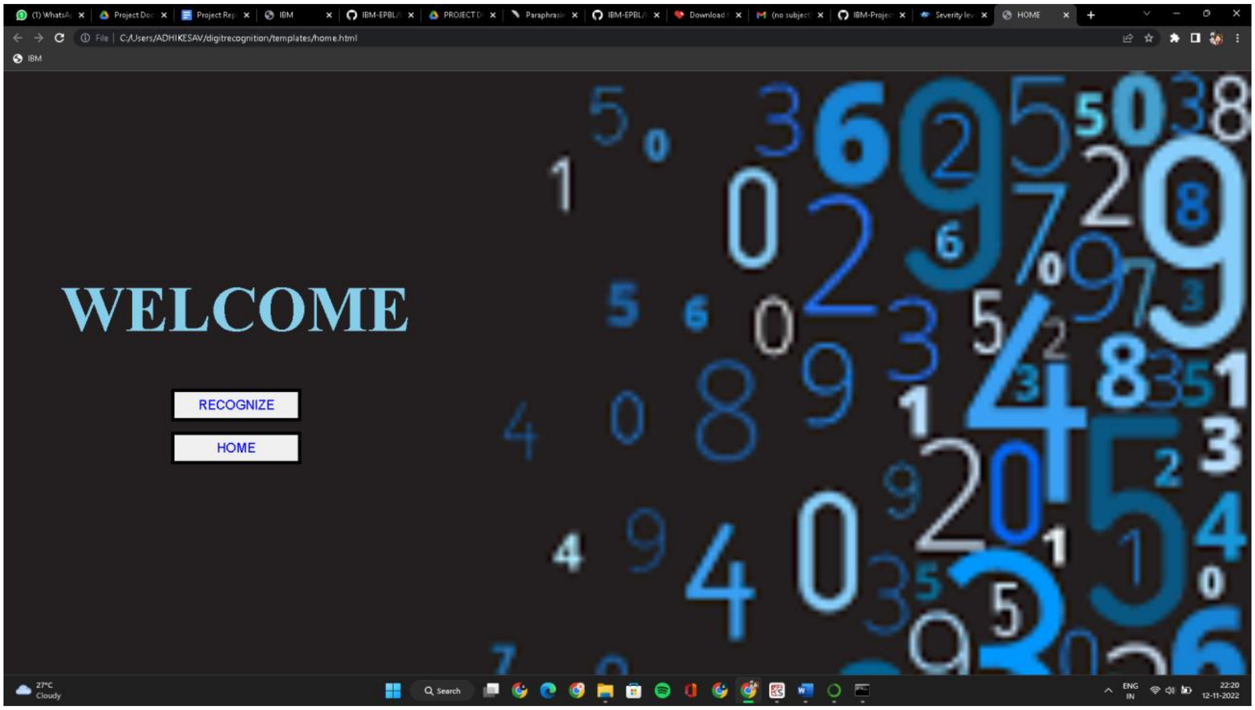


Fig 7.3 : Home Page

Let's see how our web.html page looks likeThis is the prediction page where we get to choose the image from our local system and predict the output.

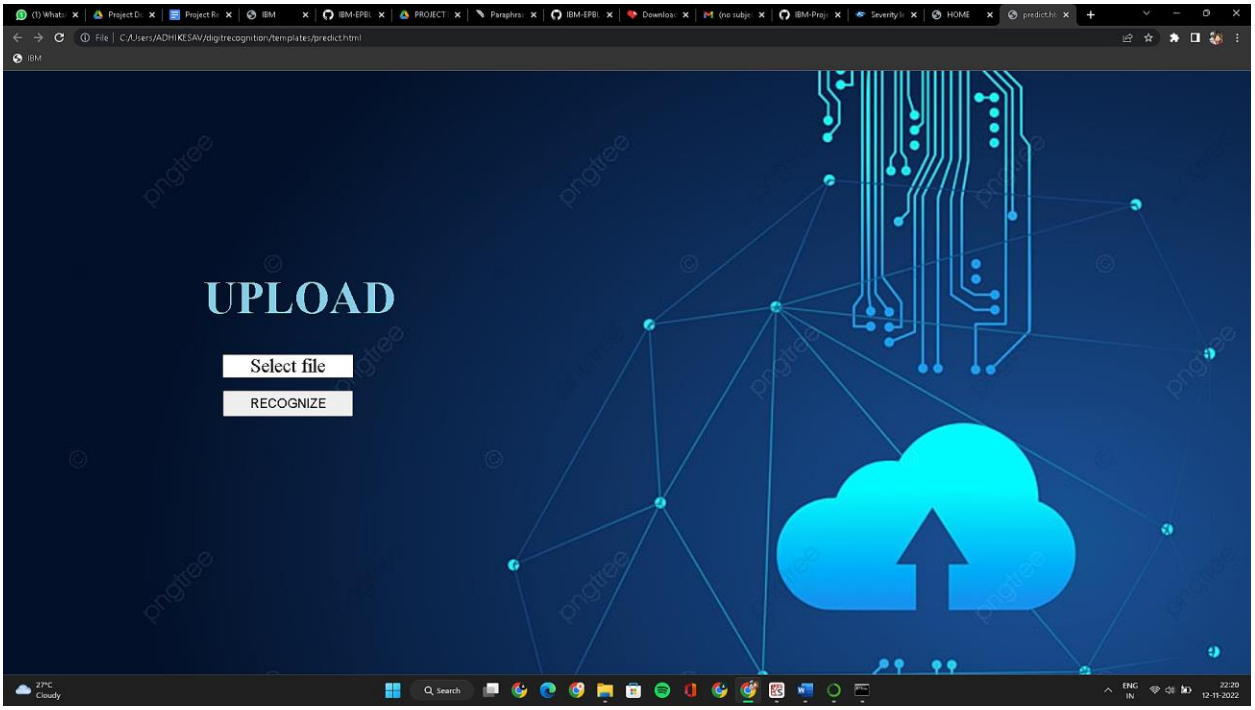


Fig 7.4: Predict Page

Build Python Code (Part 1)

Let us build the flask file 'app.py' which is a web framework written in python for server-side scripting. Let's see step by step procedure for building the backend application.

- App starts running when the "__name__" constructor is called in main.
- render template is used to return HTML file.
- "GET" method is used to take input from the user.
- "POST" method is used to display the output to the user.
- Libraries required for the app to run are to be imported.
- We are routing the app to the HTML templates which we want to render.
- Firstly we are rendering the main.html template and from there we are navigating to our prediction page that is predict.html.

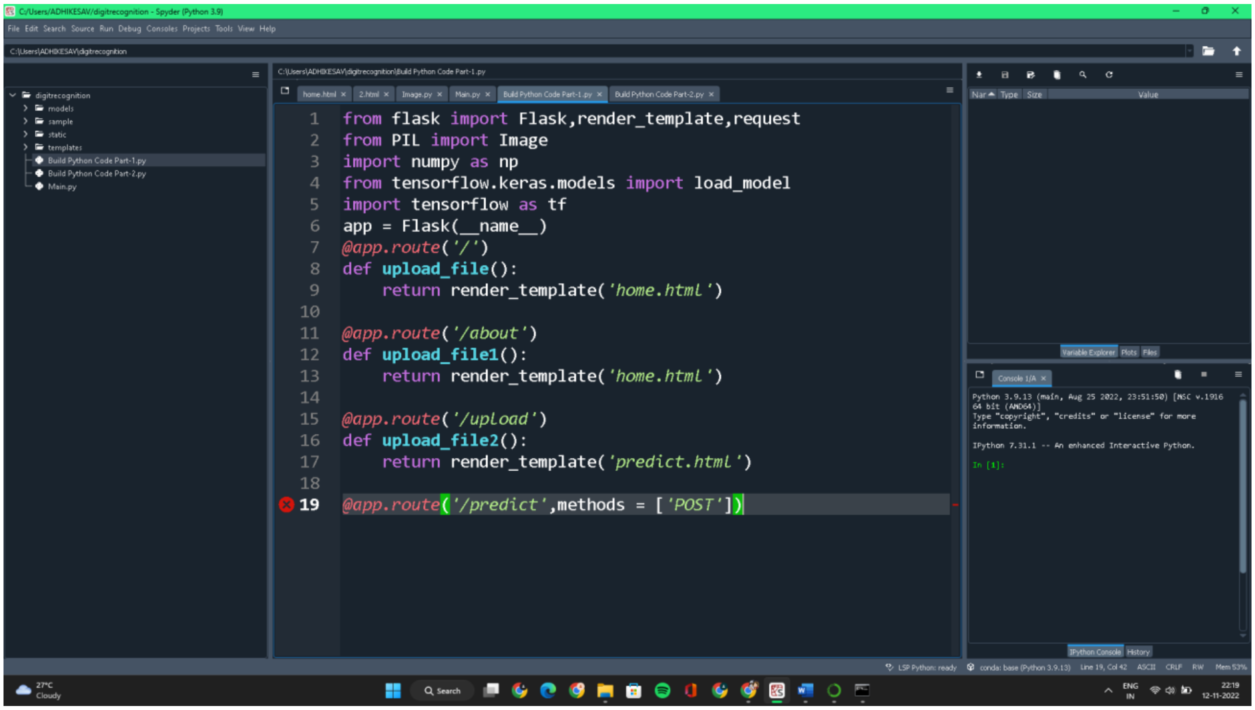
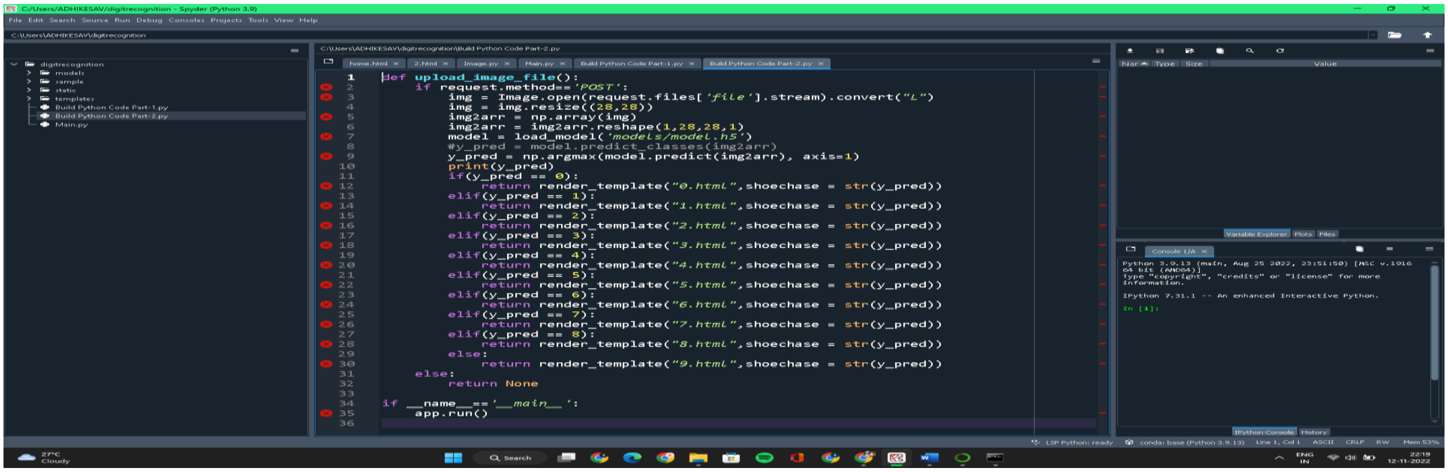


Fig 7.5 : Build Python Code Part-I

Build Python Code (Part 2)

Here the route for prediction is given and necessary steps are performed in order to get the predicted output. Necessary conditions are given according to the input classes and the app will be returning the templates according to that. Main Function: This function runs



your app in a web browser Lastly, we run our app on the localhost.

Fig 7.6: Build Code Part-II

Run The Application

- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.

- Now type “python app.py” command

Fig 7.7: Run The Application

Navigate to the localhost where you can view your web page
Upload an image and see the predicted output on UI your page and output looks like:

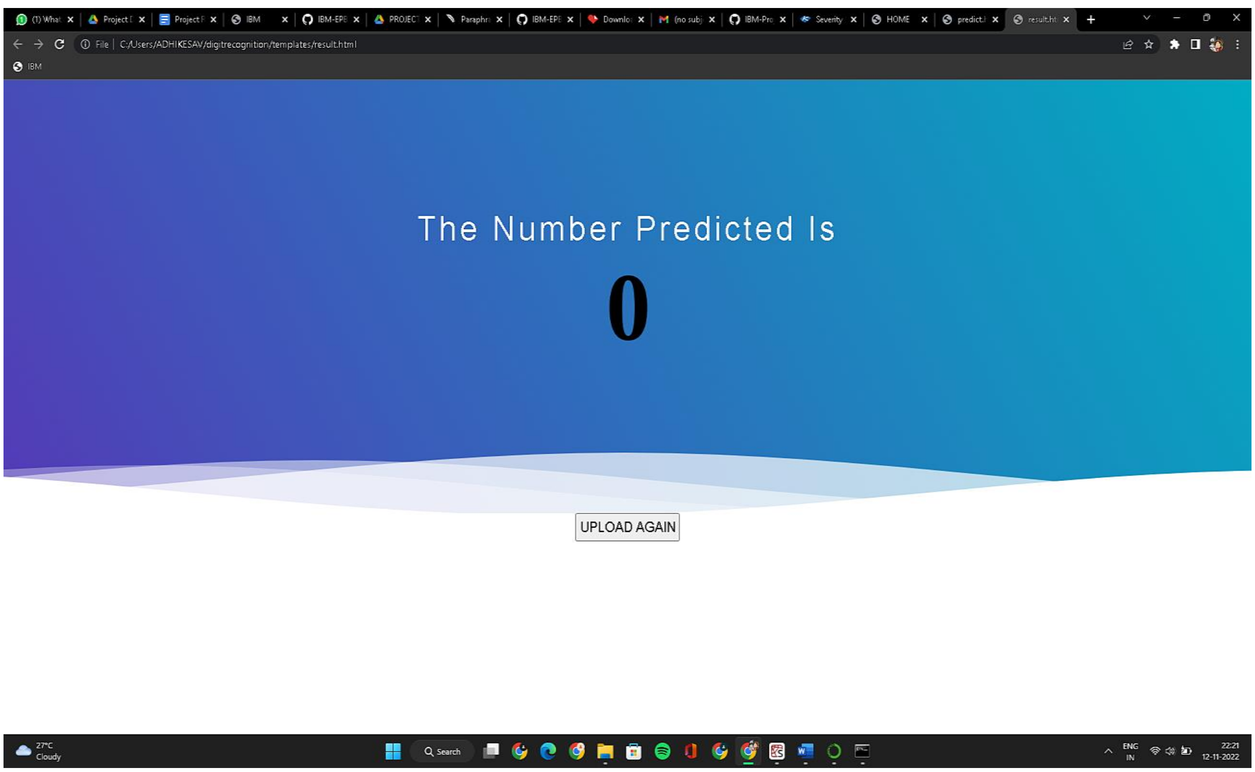


Fig 7.8 : Result page

Exception Handling :

The purpose of handling errors is to make the program robust. The word ‘robust’ means ‘strong’. A robust program does not terminate in the middle. Also, when there is an error in the program, it will display an appropriate message to the user and continue execution. Designing such programs is needed in any software development. For this purpose, the programmer should handle the errors. When the errors can be handled, they are called exceptions.

The goal of this book is help you make your code better

when we say "code", we literally mean the lines of code you are staring at in your editor. We’re not talking about the overall architecture of your project, or your choice of design patterns. Those are certainly important, but in our experience most of our day-to-day lives as programmers are spent on the “basic” stuff, like naming variables, writing loops, and attacking problems down at the function level. And a big part of this is reading and editing the code that’s already there.

CHAPTER 8

TESTING

Test Cases:

TEST ID	Feature Type	Component	Test Scenario	Test Data	Steps To Execute	Expected Result	Actual Result	Status
HOME page 1	Functional	Home page	Verify whether user is able use the recognize button	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2. check whether the recognize button is working	user is able use the recognize button	Working as expected	pass
HOME page 2	Functional	Home page	Verify whether the user is able to use the home button	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2. check whether the recognize button is working	ser is able to use the home button	Working as expected	pass
Predict page 1	Functional	Predict page	Verify whether the predict page is opening	https://github.com/IBMEPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2. check whether the button is working 3. verify whether the predict page is opening up	predict page is opening	Working as expected	pass
Predict page 2	Functional	Predict page	Verify if we are able to upload the file to be recognized	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2. check whether the button is working 3. verify whether the predict page is opening up	we are able to upload the file to be recognized	Working as expected	pass
Predict page 3	UI	Predict page	Verify whether user is able to view the preview the image which is being uploaded.	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2.. Check whether we are able to view a preview of the file which is being uploaded	user is able to view the preview the image which is being uploaded.	Working as expected	pass

Predict page 4	Functional	Predict page	Verify whether user is able use the recognize button	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2.. Check whether we are able to view a preview of the file which is being uploaded. 3. verify whether the recognize button is working or not.	user is able use the recognize button	Working as expected	pass
Predict page 5	Functional	Predict page	Verify whether it validates the file	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2.Check whether we are able to view a preview of the file which is being uploaded. 3. verify whether the recognize button is working or not.	validates the file	Working as expected	pass
Main python program	Functional	Main python program	verify whether it accepts the file uploaded and store it in a variable	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2.verify whether python program accepts the file loaded	it accepts the file uploaded and store it in a variable	Working as expected	pass
Main python program	Functional	Main python program	verify whether it loads the dataset and predict the digit	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2.verify whether python program accepts the file loaded. 3. verify whether it loads the dataset and predict the digit	it loads the dataset and predict the digit	Working as expected	pass
Result page1	Functional	Result page	Verify whether it loads the dataset and predict the digit	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2.verify whether it loads the dataset and predict the digit 3. Verify whether it shows the result page	it loads the dataset and predict the digit	Working as expected	pass
Result page	Functional	Result page	Verify whether the result is displayed	https://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2.verify whether it loads the dataset and predict the digit 3. Verify whether it shows the result page	the result is displayed	Working as expected	pass

Result page	UI	Result page	verify whether to upload again page maps to the predict page again	ttps://github.com/IBM-EPBL/IBM-Project-18416-1659684839/tree/main/Final%20Deliverables	1. run the application in Spyder application using flask 2.verify whether it loads the dataset and predict the digit 3. Verify whether it shows the result page	verify whether to upload again page maps to the predict page again		

Thus the testcases and the result were tested and recorded as above.

8.2 User Acceptance Testing :

Purpose of Document

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

Defect Analysis

This report showsthe 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	0	4	2	3	9
Duplicate	0	0	3	0	3
External	0	0	0	1	1
Fixed	0	4	5	4	13
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	1	1
Won't Fix	0	0	0	1	1
Totals	0	8	11	10	26

Test Case Analysis:

This reportshows the numberof test cases that have passed, failed,and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	25	0	0	25
Security	1	0	0	1
Outsource Shipping	2	0	0	2
Exception Reporting	9	0	0	9
Final ReportOutput	4	0	0	4
Version Control	2	0	0	2

CHAPTER 9

RESULTS

Performance Metrics: [LINK]

Model Performance Testing:

Project team shall fill the following information in model performance testing template. **Fig 9.1**

<table>
<tr><th>S.No.</th><th>Parameter</th><th>Values</th><th>Screenshot</th></tr>
<tr><td>1.</td><td>Model Summary</td><td>MNIST is a collection of handwritten digits ranging from the number 0 to 9. t has a

training set of 60,000 images, and 10,000 test images that are classified into corresponding

categories or labels. To use the MNIST dataset in Keras, an APl is provided to download and

extract images and labels automatically.</td><td>Saving the Model
In [ ]: from tensorflow.keras.models import load_model

model.save('CNN.h5')
print('Model Saved!')

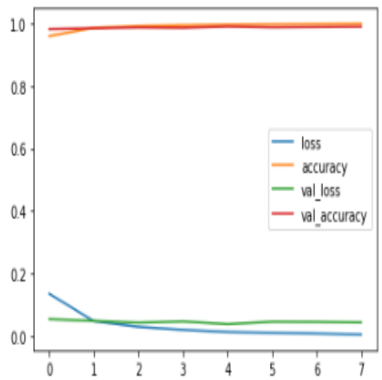
savedModel=load_model('CNN.h5')
savedModel.summary()

Model Saved!
Model: "sequential"
Layer (type) | Output Shape | Param #
conv2d (Conv2D) | (None, 25, 25, 32) | 544
max_pooling2d (MaxPooling2D) | (None, 12, 12, 32) | 0
flatten (Flatten) | (None, 4608) | 0
dense (Dense) | (None, 128) | 589952
dense_1 (Dense) | (None, 10) | 1290
Total params: 591,786
Trainable params: 591,786
Non-trainable params: 0</td></tr>
</table>

In [ ]:

```
metrics.plot()
```

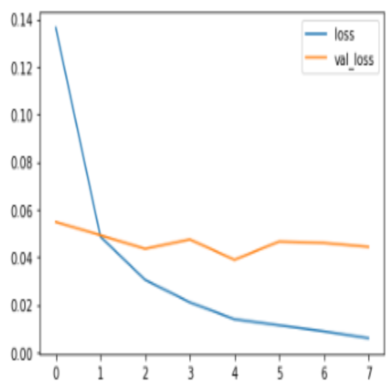
Out[ ]:



In [ ]:

```
metrics[['loss','val_loss']].plot()
```

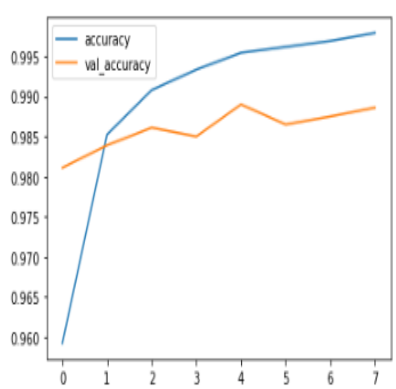
Out[ ]:



In [ ]:

```
metrics[['accuracy','val_accuracy']].plot()
```

Out[ ]:



CHAPTER 10
ADVANTAGES & DISADVANTAGES

10.1Advantage's:

1) the system not only produces a classification of the digit but also a rich description of the instantiation parameters which can yield information such as the writing style;
2) the generative models can perform recognition driven segmentation;
3) the method involves a relatively small number of parameters and hence training is relatively easy and fast; and
4) unlike many other recognition schemes, it does not rely on some form of pre-normalization of input images, but can handle arbitrary scaling, translations and a limited degree of image rotation. We have demonstrated that our method of fitting models to images does not get trapped in poor local minima.

10.2 Disadvantage's /limitations:

One of the most difficult aspects of developing a handwritten digit recognition system is that there are numerous different handwriting styles, which is a highly personal behaviour. Numbers can have different stress parts, be written in different angles, and have different lengths of specific segments. Although machine learning developers face these challenges, several steps have already been taken, such as fine tuning already defined models and developing cutting-edge classification methods for predicting handwritten digits effectively by reducing computational cost, time, and also improving accuracy. Extensive research is also being conducted in this field to ensure its long-term viability. If this model is implemented on a large scale, several issues may arise. Recognizing handwritten digits, when used maliciously.

CHAPTER 11

CONCLUSION

We successfully built a Python artificial intelligence project for handwritten digit recognition. We built and trained a Convolutional Neural Network (CNN) that excels at image classification. Deep Learning Methods for Handwritten Digit Recognition have been developed. The most commonly used Machine learning algorithms CNN were trained and tested on the same data to provide a comparison of classifiers. These deep learning approaches can achieve a high level of accuracy. Unlike other research methodologies, this one is concerned with determining which classifier performs best by increasing accuracy. A CNN model that uses Keras as the backend as well as TensorFlow as the software can achieve 98% accuracy.

Chapter 12

FUTURE SCOPE

We can work on a denser or hybrid algorithm in the future to solve many problems with more diverse data than the current set of algorithms. As a result of the above differentiation and future development, we can achieve high-level functioning applications that can be used in classified or government agencies as well as for the general public. These algorithms can be used in hospitals for detailed medical diagnosis, treatment, and patient monitoring. It can be used in surveillance systems to track suspicious activity within the system. It can be used in fingerprint scanners and retinal scanners. Applications for database filtering. Checking equipment for national forces. Using these algorithms for both day to day and high-level application can be helpful in achieving safety and solace.

CHAPTER 13

APENDIX

13.1 Source Code :

HOME.html:

```
<html>
<head>
<title>HOME</title>
<style>
body{
background-image:url("C:/Users/ADHIKESAV/digitrecognition/static/image/a.png");
background-repeat:no-repeat;
background-size:cover;
backgroud-attachment:fixed;
}
#a{
color:skyblue;
font-size:99px;
padding-top:300px;
padding-left:80px;
}
#b{
margin-left:250px;
border:5px black solid;
height:50px;
width:200px;
font-size:20px;
```

```
color:blue;
}
#c{
margin-left:250px;
border:5px black solid;
height:50px;
width:200px;
font-size:20px;
color:blue;
}
</style>
<body>
<h1 id="a">WELCOME</h1>
<form action="upload">
<input id="b" type="submit" value="RECOGNIZE">
</form>
<form action ="about">
<input id="c" type="submit" value="HOME">
</form>
</body></head></html>
```

Predict.html:

```
<html>
<head>
<link rel="shortcut icon" href="data:image/x-icon;," type="image/x-icon">
<style>
body{
```

```
background-image:url("C:/Users/ADHIKESAV/digitrecognition/static/image/c.png");
background-repeat:no-repeat;
background-size:cover;
backgroud-attachment:fixed;
height:700px;
width:1000px;
}
#a{
color:skyblue;
font-size:70px;
padding-top:300px;
padding-left:300px;
}
#b{
margin-left:250px;
border:5px black solid;
height:50px;
width:200px;
font-size:50px;
display:none;
}
#c{
margin-left:330px;

height:40px;
width:200px;
```

```
font-size:20px;
}
.label{
  background:white;
  margin-left:330px;

  height:35px;
  width:200px;
  font-size:30px;
  display: block;
  text-align:center;
}
.preview{
display:flex;
flex-direction:row;
align-items:center;
justify-content:center;
}
#previewimg{
width:200px;
height:200px;
```



```
margin-bottom:20px;
margin-right:140px;
display:none;
```

```
}
</style>
<script>
  function showPreview(event){
  if(event.target.files.length > 0){
    var src = URL.createObjectURL(event.target.files[0]);
    var preview = document.getElementById("previewimg");
    preview.src = src;
    preview.style.display = "block";
  }
  }
function validateImage(event){
  event.preventDefault();
  element = document.querySelector("#b");
  if(element.files.length==0){
     alert("Please Upload an image");
  }
  else{
        const form = document.querySelector('form');
        form.submit();
  }
}
</script>
    <body>
    <h1 id ="a" >UPLOAD</h1>
         <form action ="predict" method = "POST" enctype = "multipart/form-data">
```

```
        <label class='label' for='b'>Select file</label>
        <input id="b" type = "file" name = "file"  onchange='showPreview(event);'/><br>
        <div class='preview'><img id='previewimg' /></div>
        <input id="c" type = "submit"  value="RECOGNIZE" 
onclick='validateImage(event);'/>
        </form>
   </body>
   </head>
</html>
```

Main.py:

```
from flask import Flask,render_template,request
from PIL import Image
import numpy as np
from tensorflow.keras.models import load_model
import tensorflow as tf
app = Flask(__name__)
@app.route('/')
def upload_file():
    return render_template('home.html')
@app.route('/about')
def upload_file1():
    return render_template('home.html')

@app.route('/upload')
def upload_file2():
```

```
    return render_template('predict.html')


@app.route('/predict',methods = ['POST'])
def upload_image_file():
    if request.method=='POST':
        img = Image.open(request.files['file'].stream).convert("L")
        img = img.resize((28,28))
        img2arr = np.array(img)
        img2arr = img2arr.reshape(1,28,28,1)
        model = load_model('models/model.h5')
        #y_pred = model.predict_classes(img2arr)
        y_pred = np.argmax(model.predict(img2arr), axis=1)
        print(y_pred)
        if(y_pred == 0):
            return render_template("0.html",shoechase = str(y_pred))
        elif(y_pred == 1):
            return render_template("1.html",shoechase = str(y_pred))
        elif(y_pred == 2):
            return render_template("2.html",shoechase = str(y_pred))
        elif(y_pred == 3):
            return render_template("3.html",shoechase = str(y_pred))
 elif(y_pred == 4):
            return render_template("4.html",shoechase = str(y_pred))
        elif(y_pred == 5):
            return render_template("5.html",shoechase = str(y_pred))
        elif(y_pred == 6):
```

```
            return render_template("6.html",shoechase = str(y_pred))
        elif(y_pred == 7):
            return render_template("7.html",shoechase = str(y_pred))
        elif(y_pred == 8):
            return render_template("8.html",shoechase = str(y_pred))
        else:
            return render_template("9.html",shoechase = str(y_pred))
    else:
        return None


if __name__=='__main__':
    app.run()
```

13.2 Github And Demo Link:

GITHUB LINK:https://github.com/IBM-EPBL/IBM-Project-18416-1659684839

DEMO LINK: https://www.youtube.com/embed/AkICV18HOfY