PROJECT NAME	Project - A Novel Method for Handwritten Digit Recognition System
TEAM ID	PNT2022TMID32148
TEAM MEMBERS	Harshath.M, Priyanga.S, Suvetha.M, Ajeeth Kumar.S

1.INTRODUCTION

1.1 Project Overview

Our project is “A Novel Method for Handwritten Digit Recognition System.” This is a three- step process and it has user friendly interface. The three-step process are

1. Login
2. Upload
3. Result

Login - This is the first page; in this page you have to enter your email id and password. If you entered the correct credentials you will redirect to the next page.

Upload – This is the second page; in this page you can upload the image in your local system. In this page you could not upload the files except the jpeg, png and jpg files. It will also provide the facility to preview the image that you have uploaded.

Result – This is the third page; in this page the predicted value will be shown in the graph format. You can also download the page.

1.2 Purpose

The human handwritten digits are not perfect and it can be made with different sizes and shapes. To overcome this problem, it is needed some system that is faster than humans. The attractive solution for this problem is “Handwritten Digit Recognition System.” It is difficult to identify someone’s handwritten digits to recognize. It will make people stressed. They could not complete their work on time. To reduce these complications, it will be useful. Through this people can easily upload their handwritten digit image and they can get the predicted value. This handwritten digit recognition system can be useful in business perspective as well. Industries and organization can use this system as their part of work. Banks, Postal service can use this to recognize the digit code written by peoples. Our model is going to deploy in a web. So anyone on the internet can access the service provide by the system.

2.LITERATURE SURVEY

2.1 Existing problem

Handwritten recognition system has problems when it comes to accuracy. The issue is that there is a wide range of handwritings good and bad. This makes it tricky for programmers to provide enough examples of how every character might look. Sometimes, characters look very similar, making it hard for a computer to recognize accurately. If the system does not provide accurate prediction means it makes confusion to the users. It takes more time to predict the value it makes people anxious. These are all the problem in existing system.

2.2 References

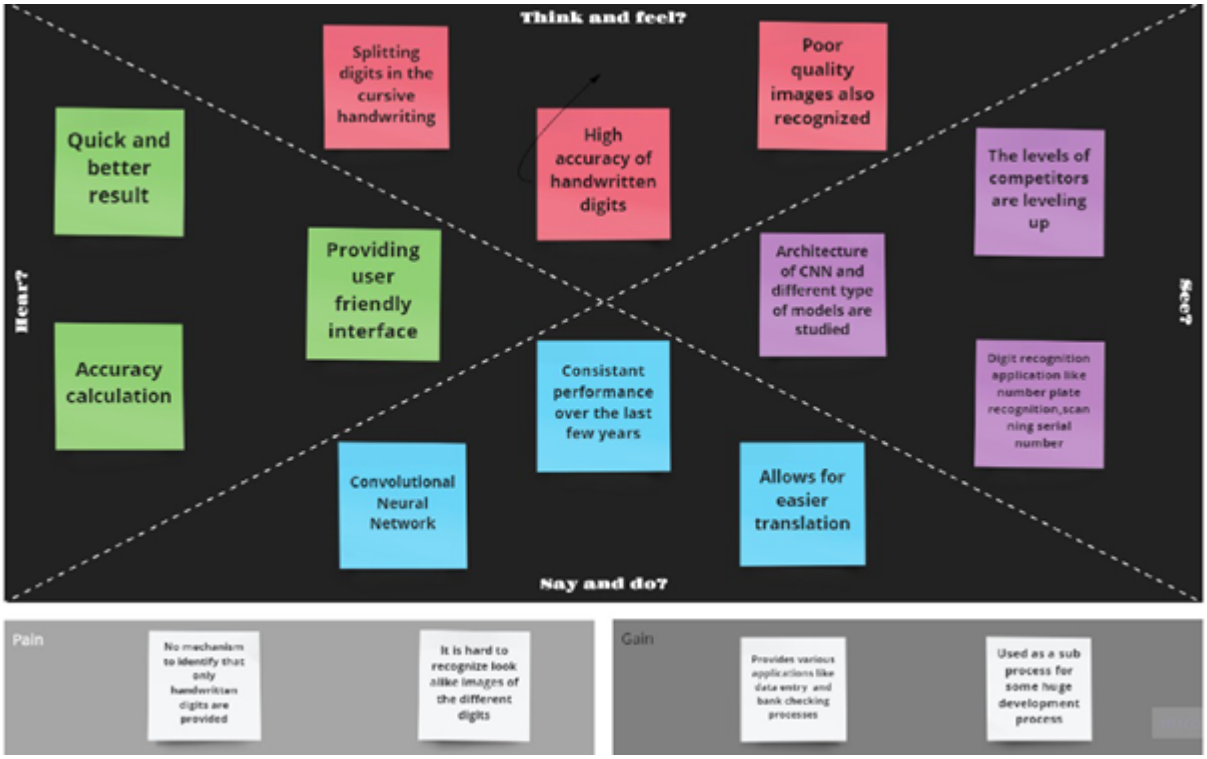
1. Dr.Kusumgupta2,"a comprehensive review on handwritten digit recognition using various neural network approaches", international journal of enhanced research in management & computer applications, vol. 5, no. 5, pp. 22-25, 2016.
2. Ishani Patel, ViragJagtap and OmpriyaKale."A Survey on Feature Extraction Methods for Handwritten Digits Recognition", International Journal of Computer Applications, vol. 107, no. 12, pp. 11-17, 2014.
3. Y LeCun,"COMPARISON OF LEARNING ALGORITHMS FOR HANDWRITTEN DIGIT RECOGNISATION". In:International conference on Artificial Neural networks, France, pp. 53–60. 1995.

2.3 Problem statement definition

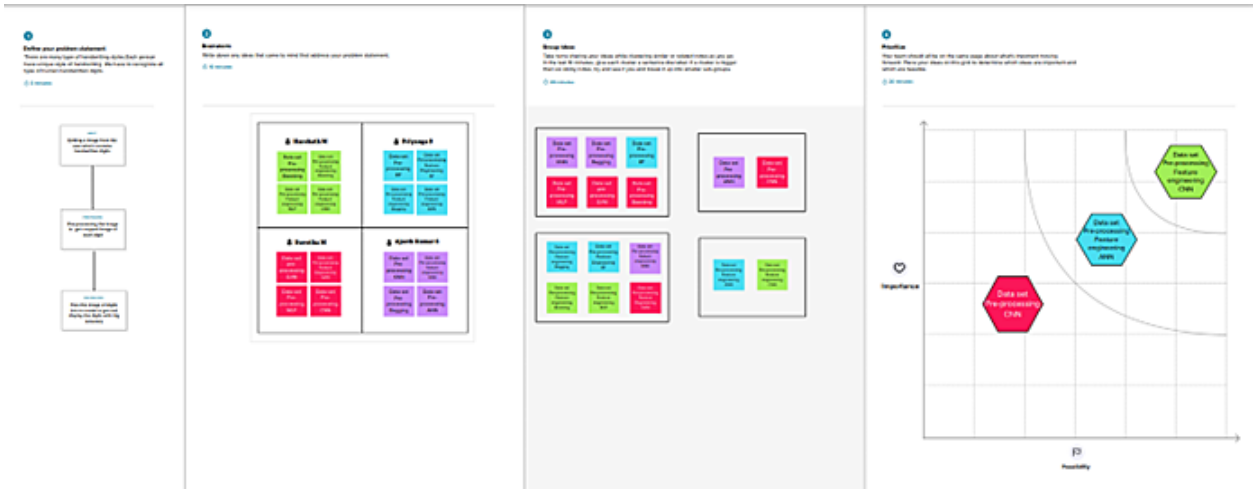
Everyone have different type of handwriting and it is also difficult to recognize the digit. The delay in recognition makes people anxious. It also makes delay in work completion. To reduce these types of problems we bring the solution that is digit recognizer.

3.IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



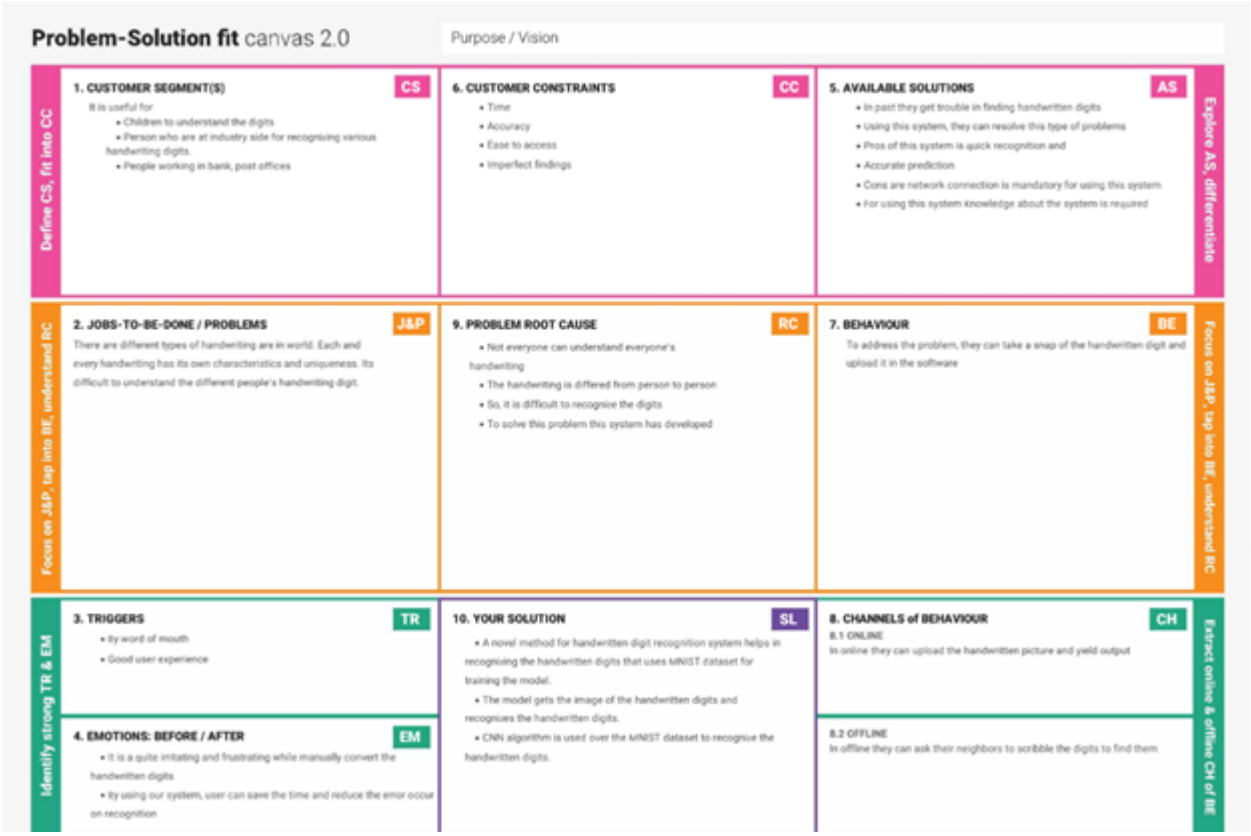
3.2 Ideation & Brainstorming



3.3 Proposed Solution

S.No.	Parameters	Description
1.	Problem Statement (problem to be solved)	● The human handwritten digits are not perfect and it can be made with different sizes and shapes. ● The handwritten digit recognition is the capability of computer applications to recognise the human handwritten digits.
2.	Idea / Solution Description	● Handwritten digit recognition using MNIST dataset. ● It is a major project made with the help of Convolutional Neural Network. ● It basically detects the scanned images of handwritten digits.
3.	Novelty / Uniqueness	Having a highly reliable and high accurate model with negotiable loss percentage.
4.	Social Impact / Customer satisfaction	● Easy to recognize the digits by a simple process. ● Everyone can easily upload the image containing handwritten digits, and get the output in a quicker way.
5.	Business Model (Revenue Model)	● This handwritten digit recognition system can be useful in business perspective as well. ● Industries and organization can use this system, as their part of work. ● Banks, Postal service can use this to recognize the digit code written by humans.
6.	Scalability of the solution	● Our model is going to deploy in a web. So, anyone on the internet can access the service provided by the system.

3.4 Problem Solution fit



4.REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Login	Login with verified email id and password using mongodb
FR-2	Image Upload	Upload a handwritten digit image in a supported format
FR-3	Web Browser	Mobile or Desktop browser is needed to make use of digit recognition.

4.2 Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Simple to understand the UI and easy to get the recognition of handwritten digits
NFR-2	**Security**	No security measures are taken for our application
NFR-3	**Reliability**	Withstand without any occurrence of error for a long period of time
NFR-4	**Performance**	Light-weight application makes the performance better.
NFR-5	**Availability**	New pages include will doesn’t affect the system
NFR-6	**Scalability**	Large number of users can recognize the digits at a time without any restriction.

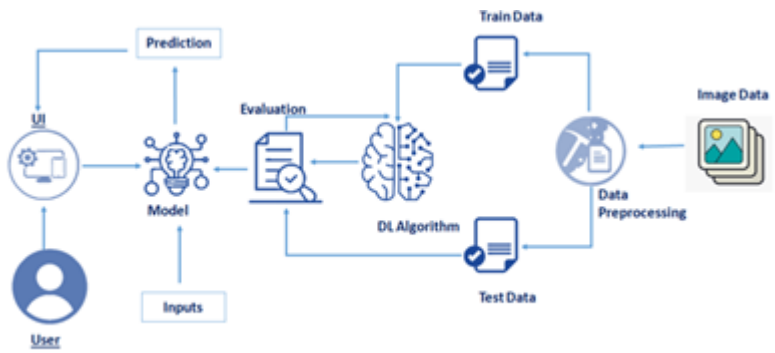
5.PROJECT DESIGN

5.1 Data Flow Diagrams

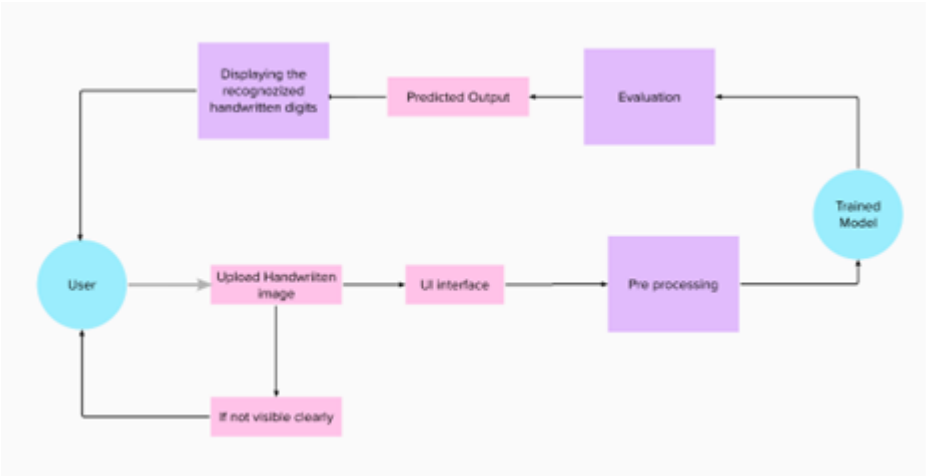
Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

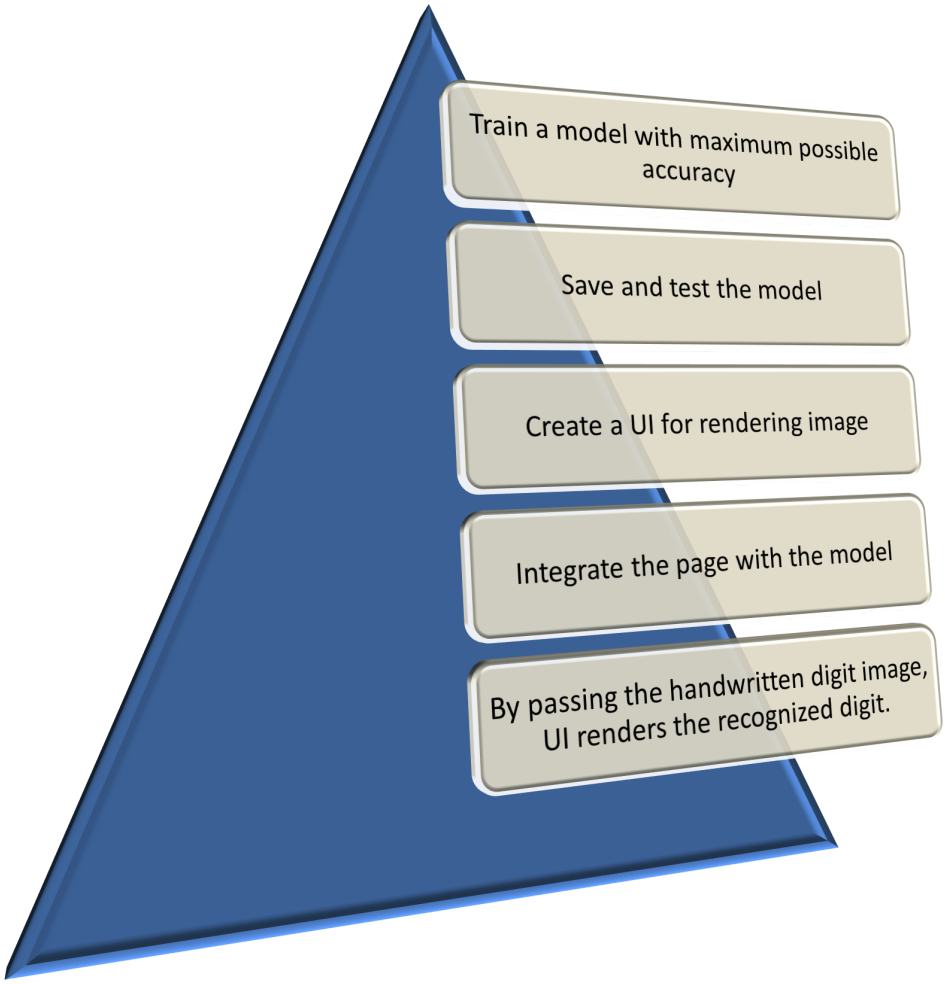
Flow Diagram:



DFD :



5.2 Solution & Technical Architecture



5.3 User stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Details	USN-1	As a user, I can know the details of the fundamental usage of the application.	I can access information of details page	Low	Sprint-2
	Login	USN-2	As a user, I will login with my email credentials and basic validation is verified.	I can access the next page if I am a verified user	High	Sprint-3
	Image upload	USN-3	As a user, I will upload the handwritten digit image to the application	I can upload the image from the local system	High	Sprint-2
	Recognized Result	USN-4	As a user, I can see the predicted / recognized digits in the application	I can see the output of the recognized digit	High	Sprint-3
Customer (Web user)	Details	USN-1	As a user, I can know the details of the fundamental usage of the application.	I can access information of details page	Low	Sprint-2
	Login	USN-2	As a user, I will login with my email credentials and basic validation is verified.	I can access the next page if I am a verified user	High	Sprint-3
	Image upload	USN-3	As a user, I will upload the handwritten digit image to the application	I can upload the image from the local system	High	Sprint-2
	Recognized Result	USN-4	As a user, I can see the predicted / recognized digits in the application	I can see the output of the recognized digit	High	Sprint-3

6.PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

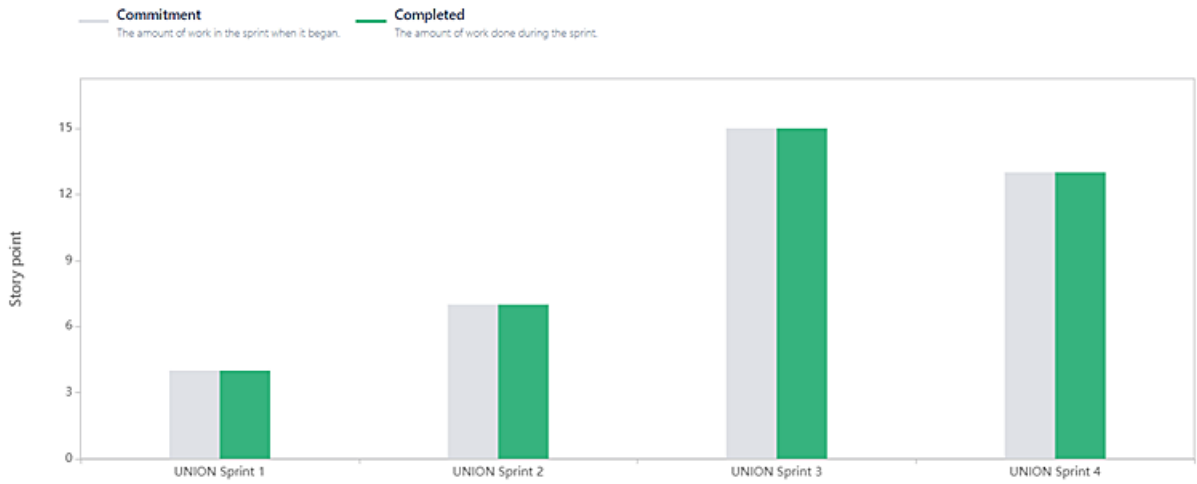
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Pre processing	USN-1	As a user, I can upload any kind of image with the pre-processing step is involved in it.	3	High	Priyanga, Suvetha
Sprint-1		USN-2	As a user, I can upload the image in any resolution.	1	Low	Ajeeth Kumar, Harshath
Sprint-2	Model	USN-3	As a user, I will get a application with ML model which provides high accuracy of recognized handwritten digit.	2	Medium	Suvetha, Ajeeth Kumar
Sprint-2		USN-4	As a user, I can pass the handwritten digit image for recognizing the digit.	2	Medium	Harshath, Suvetha
Sprint-2		USN-5	As a user, I can get the most suitable recognized digit.	3	High	Harshath, Priyanga
Sprint-3	User Interface	USN-6	As a user, I can login and I will upload the handwritten digit image to the application by clicking a upload button.	5	High	Ajeeth Kumar, Priyanga
Sprint-3		USN-7	As a user, I can know the details of the fundamental usage of the application.	2	Low	Ajeeth Kumar
Sprint-3		USN-8	As a user, I can see the predicted / recognized digits in the application	8	High	Harshath, Suvetha
Sprint-4	Cloud Deployment	USN-9	As a user, I can access the web application and make the use of the product from anywhere	13	High	Harshath, Priyanga

6.2 Sprint Delivery Schedule

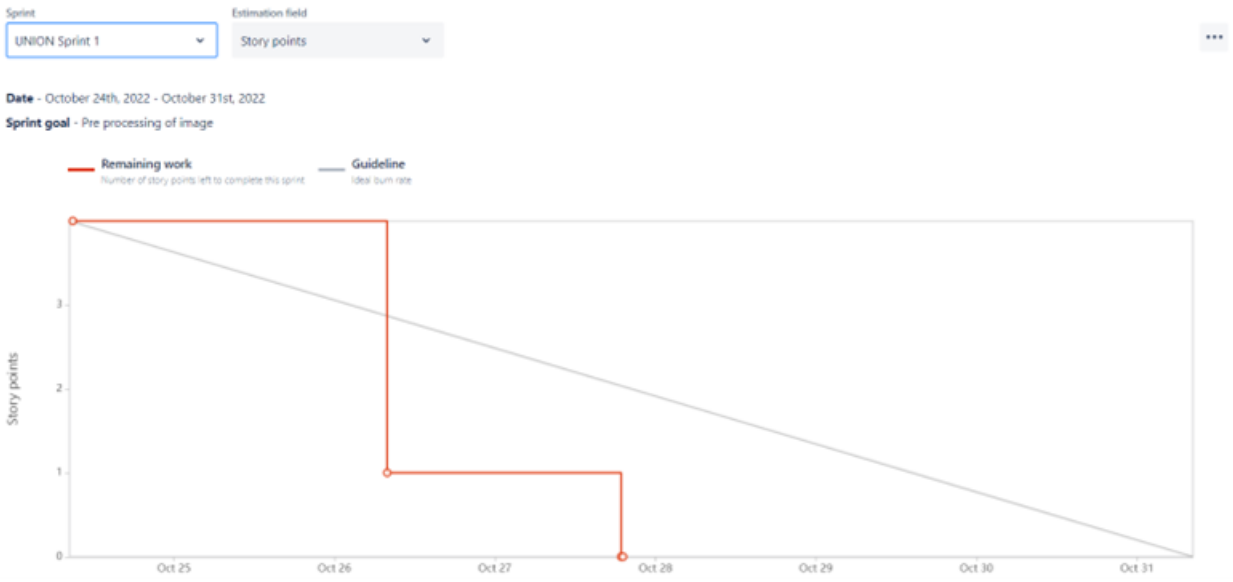
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	4	6 Days	24 Oct 2022	29 Oct 2022	4	29 Oct 2022
Sprint-2	7	6 Days	31 Oct 2022	05 Nov 2022	7	05 Nov 2022
Sprint-3	15	6 Days	07 Nov 2022	12 Nov 2022	15	12 Nov 2022
Sprint-4	13	6 Days	14 Nov 2022	19 Nov 2022	13	19 Nov 2022

6.3 Reports from JIRA

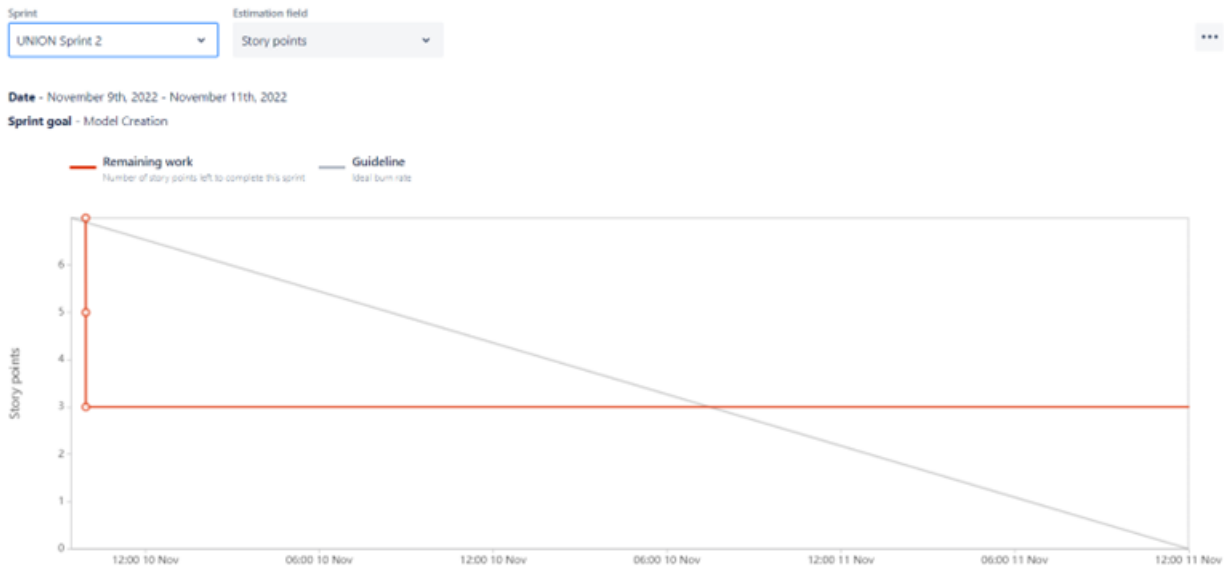
Velocity Report



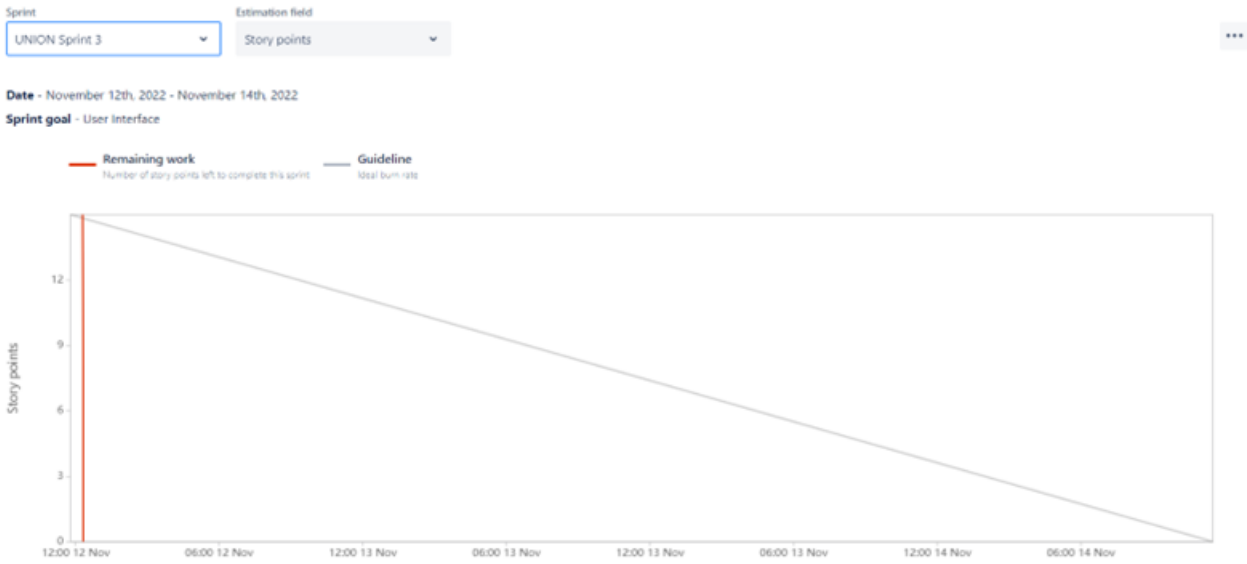
Sprint 1 – Burndown Chart



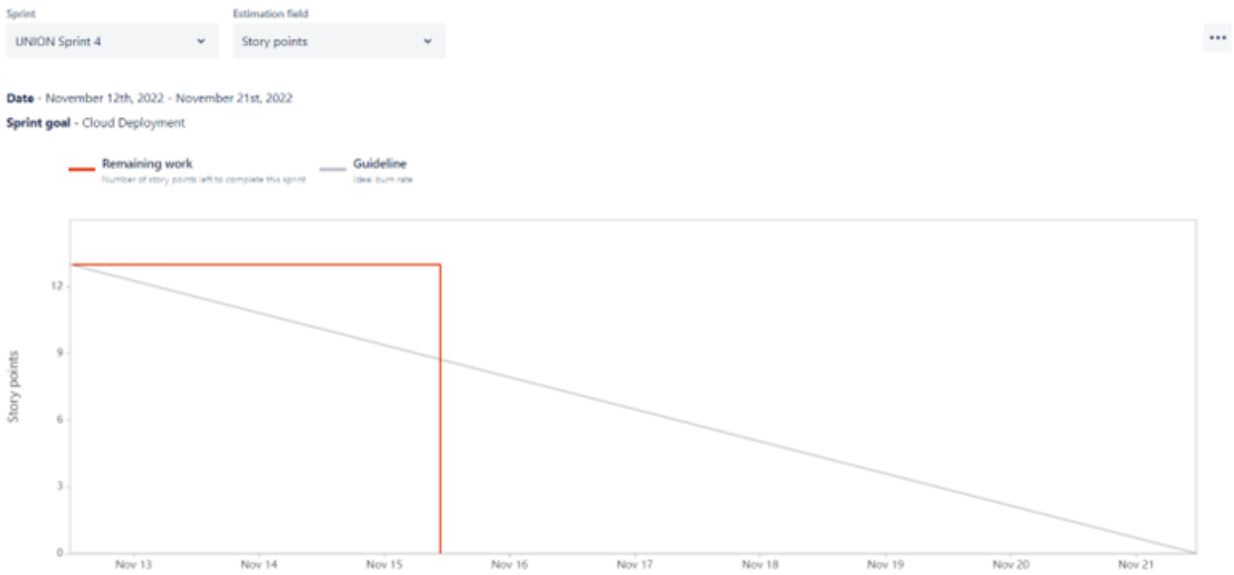
Sprint 2 – Burndown Chart



Sprint 3 – Burndown Chart



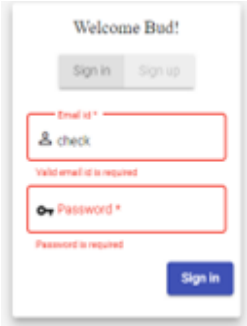
Sprint 4 – Burndown Chart



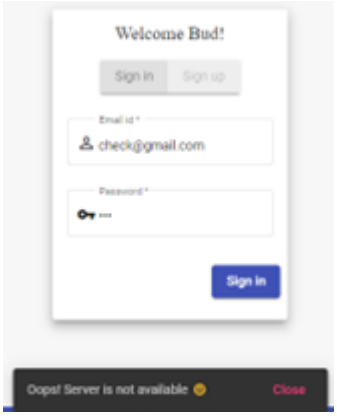
7.CODING & SOLUTIONING

7.1 Features

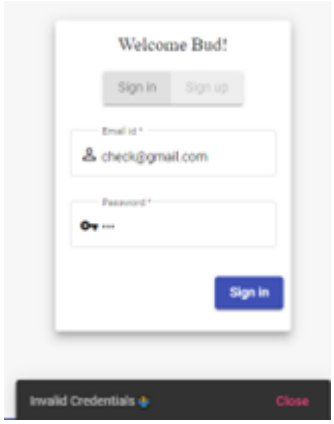
- Basic validations are verified, when user enters a credentials.



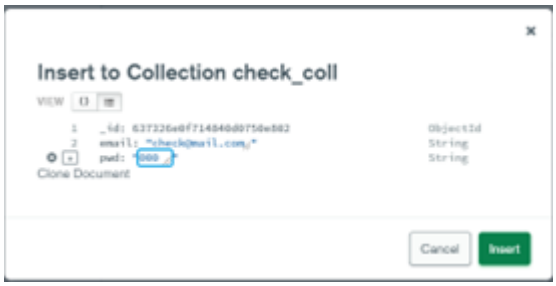
- If the server is not available, we should restrict the user without redirecting to the next page and let them wait in the same login page with an indication message “Oops server is not available”.



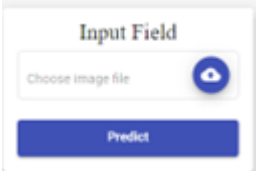
- When the user enters the invalid credential the snackbar will appears at the bottom and shows the corresponding message.



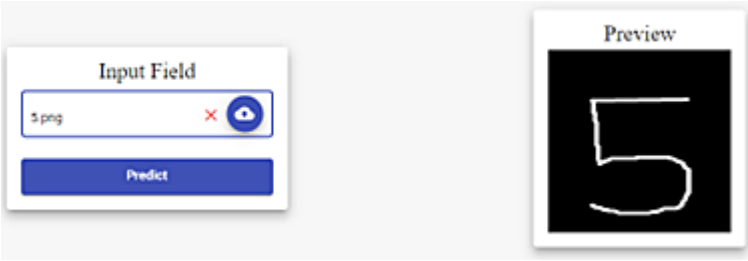
- If the user is available in mongodb collections, he/she is redirected to the next page. If the new user credentials are to be entered, it will done by creating a new document with necessary details in the mongodb.



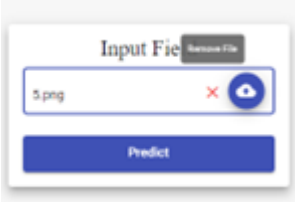
- When correct email is entered, user should redirect to the upload page by showing the indicating message that user is verified.



- Preview is useful in most of the places, here also we implement the preview card. When the user upload the pic, it stores in the database and show the preview immediately.



- Tiptool is useful to know about what we are doing. There is remove file button, to remove the file in formdata. Immediately after removing the file, the preview card is disappeared to maintain the flow.

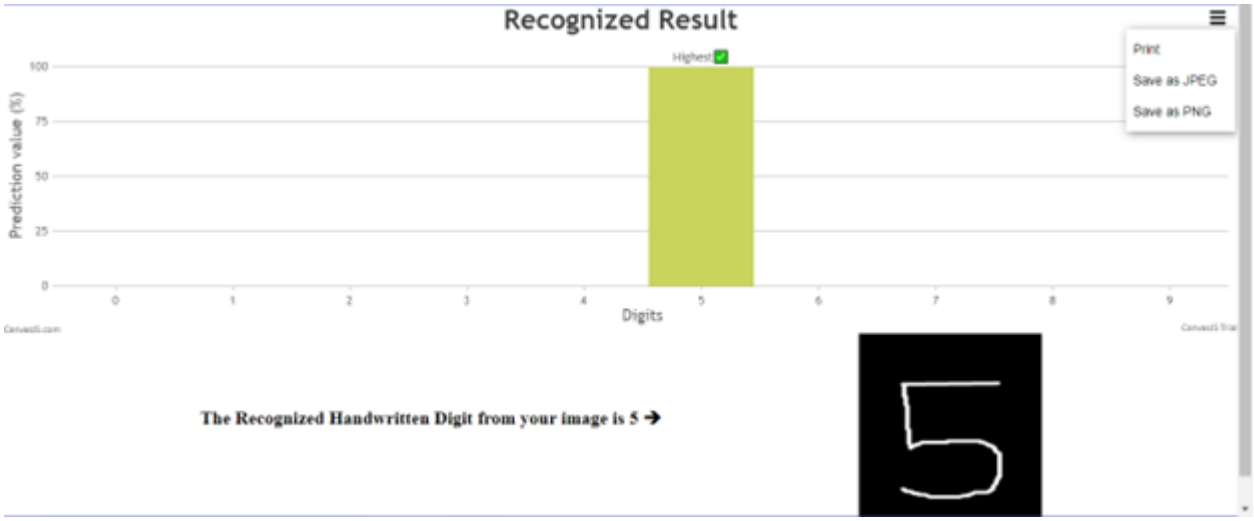


- Uploading the files is restricted only to image files. In special case, user enters a other format file, it will collapse the system. For overcoming that, new snackbar is created and shows the corresponding message if user enters the other format files.

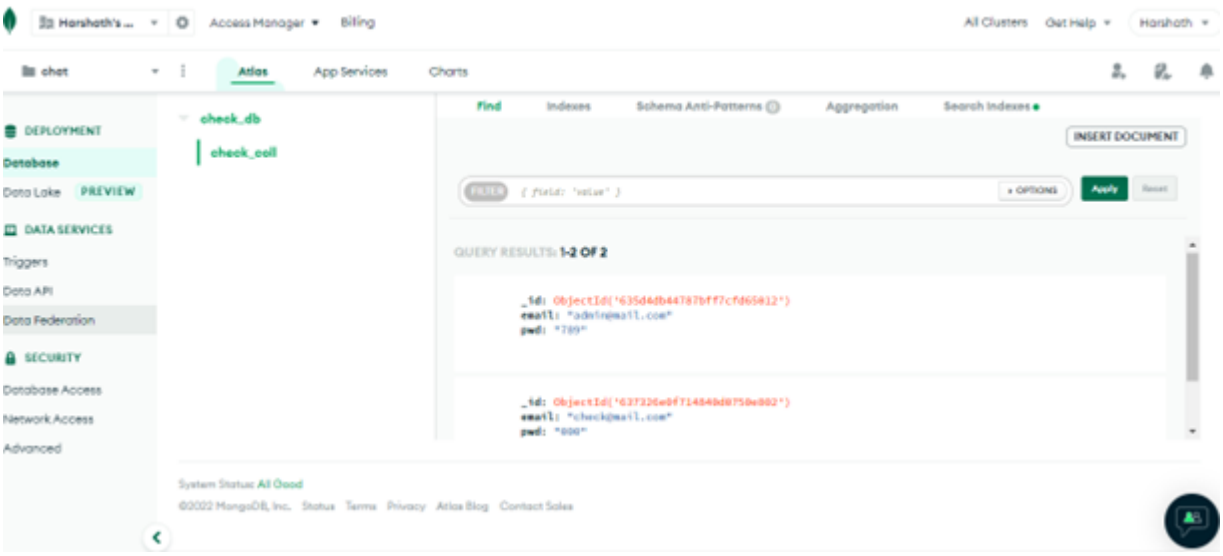
- Main feature of our project is showing the predicted results as a column chart, it makes the user to easily understand the results. Chart is created by canvas js library.

The Recognized Handwritten Digit from your image is 5 →

- Chart can be downloaded if it is needed.



7.2 Database Schema



8.TESTING

8.1 Test Cases

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_OO1	Functional	Login Page	Basic validations for email and password is to be done. Correspondings error should be displayed	—	1.Enter URL and click go 2.Click on email input and check by putting various types of inputs 3.Verify error is shown up	Mail id 1: check Password 1: - Mail id 2 : Check@gmail.com Password 2: 123	Validation and normal flow or error should be pop up	Working as expected	Pass	—	No	—	Ajeeth Kumar.S
LoginPage_TC_OO2	Functional	Login Page	Verify the user is authorized user or not	User Credentials	1.Enter URL and click go 2.Click on email input and check by putting various types of inputs 3.Verify error is shown up	Mail id : Check@gmail.com Password : 123	If user credentials are in the database, app will route to next upload page. If not, it should the message like invalid credentials.	Working as expected	Pass	—	No	—	Priyanga.S
LoginPage_TC_OO3	Functional	Login Page	Restrict the user in the login page when server is not availble.	—	1.Enter the user credentials. 2.Click sign in button 3.Verify server is available or not	—	Display the corresponfing message if server is not available	Working as expected	Pass	—	No	—	Suvetha.M
UploadPage_TC_OO1	Functional	Upload page	Uploading only image file. Restricting other format files	—	1.Click upload button and upload the files in other format.	File - Not a image format	Not taking the other format file and shows the error message to the user.	Working as expected	Pass	—	No	—	Priyanga.S
UploadPage_TC_OO2	Functional	Upload page	Showing the preview	—	1.Click upload button and upload the files in other format. 2.Verify the preview shown	image file	Preview showing in the page	Working as expected	Pass	—	No	—	Harshath.M
ResultPage_TC_OO1	Functional	Result Page	Column chart integration for the result page	—	1.After uploading the image, click predict button, 2.Check results are shown.	—	Predicted results are shown in text as well as chart format	Working as expected	Pass	—	No	—	Harshath.M

8.2. User Acceptance Testing

1.Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	17	3	4	3	27
Duplicate	1	3	2	1	7
External	2	5	3	1	11
Fixed	9	4	7	28	48
Not Reproduced	0	0	1	0	1
Skipped	1	0	1	1	3
Won't Fix	0	0	0	1	1
Totals	29	15	18	35	98

2.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	33	0	0	33
Security	10	0	0	10
Outsource Shipping	6	0	0	6
Exception Reporting	17	0	0	17
Final Report Output	8	0	0	8
Version Control	5	0	0	5

9.RESULTS

9.1 Performance Metrics

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Layers- Conv2d (Conv2D) Conv2d_1 (Conv2D) Flatten (Flatten) Dense (Dense)	Model: "sequential" Layer (type) Output Shape Param # conv2d (Conv2D) (None, 26, 26, 64) 640 conv2d_1 (Conv2D) (None, 24, 24, 32) 18464 flatten (Flatten) (None, 18432) 0 dense (Dense) (None, 10) 184330 Total params: 203,434 Trainable params: 203,434 Non-trainable params: 0
2.	Accuracy	Training Accuracy – 0.978 Validation Accuracy - 0.9786	loss & accuracy [0.2615804076194763, 0.978600025177002] val_loss: 0.2616 - val_accuracy: 0.9786

10.ADVANTAGES & DISADVANTAGES

ADVANTAGES

1. User friendly interface makes the user to navigate easily to other pages.
2. It provides high accuracy.
3. The quick prediction will save the time of the users.
4. It will show the preview of the image which the user uploaded. It will help the user to check whether he/she uploaded the correct image.
5. It provides results in graphical representation for easy understanding.
6. Handwritten Digit Recognizer is an angular js application and it is also deployed in github pages for easy access.
7. Login credentials are not static it will be fetched from mongodb atlas collections.
8. Users can download the prediction result as chart.

DISADVANTAGES

1. This is only for single digit recognition.
2. The persons who have the knowledge about this only can use this.
3. For now, the flask API is only run in the local host. So it is only used in offline.

11.CONCLUSION

An implementation of handwritten recognition using deep learning has been implemented. In this handwritten recognition system high accuracy is achieved. We have used the Machine Learning algorithm CNN for accuracy. Here mongodb is used to store the information like email id and password. During login, verification will be done by fetching the information which is stored in mongodb. It has many features and one of the best features is it will show the preview of the image after it is uploaded and showing the predicted results in graphical representation. Preview helps the user to check whether the correct image is uploaded. Column chart (Canvas js) makes the result page more attractive. The accuracy rate of this handwritten recognizer is 97.86%.

12.FUTURE SCOPE

Artificial Intelligence have more scope in these days. It plays a vital role in every places such as schools, colleges, offices, etc. Like that the Handwritten Recognition system will be more helpful in many fields. In post office it is used to recognize the digits of the postal codes. In medical coding it will be more useful to recognize the digits. 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 cheque amounts, numeric entries in forms filled up by hand and so on.

13. APPENDIX

Source Code

DigitAPI.py

```
from flask import Flask,Blueprint
from flask_cors import CORS
from flask_pymongo import PyMongo
from endpoints import api_endpoints

def create_app():
    webapp = Flask(__name__)
    CORS(webapp)

    api_blueprint = Blueprint('api_blueprint',__name__)
    api_blueprint = api_endpoints(api_blueprint)
    webapp.register_blueprint(api_blueprint, url_prefix= '/api')
    return webapp

app=create_app()
if('__main__'== __name__):
    app.run(host='0.0.0.0')
```

endpoints.py

```
from flask_pymongo import pymongo
from flask import request,send_file
from keras.models import load_model
from PIL import Image
import numpy as np

model = load_model("digit-recognition.h5")
uri = 'mongodb+srv://harsh:harsh@cluster0.rxvjk.mongodb.net/?retryWrites=true&w=majority'
client = pymongo.MongoClient(uri)
db = client.check_db
coll = db.check_coll
print('connection has made')

def api_endpoints(endpoints):
    @endpoints.route('/verify', methods=['POST'])
```

```python
def verify():
    try:
        email = request.form.get('email')
        pwd = request.form.get("pwd")
        flag = coll.find_one({"email":email, "pwd":pwd})
        status={
            'statuscode' : 200,
        }
        if(flag!=None):
            status['statusmessage'] = "true"
        else:
            status['statusmessage'] = "false"
    except Exception as e:
        status={
            'statuscode' : 400,
            'statusmessage' : str(e)
        }
    return status

@endpoints.route('/upload', methods=['POST'])
def upload():
    input = request.files.get("image")
    global format
    format = request.form.get("format")
    img= Image.open(input)
    img = img.resize((200,200))
    img.save("files/input."+format)
    return send_file(path_or_file = "files/input."+format)

@endpoints.route('/predict', methods=['GET'])
def predict():
    result = {};
    img=Image.open("files/input."+format).convert("L")
    img = img.resize((28,28))
    im2arr=np.array(img)
    im2arr = im2arr.reshape(1,28,28,1)
    y_pred = model.predict(im2arr)
    result["value"] = int(np.argmax(y_pred))
    print("Predicted value is",result)
    return result

@endpoints.route('/image', methods=['GET'])
def image():
```

```
        return send_file(path_or_file = "files/input."+format)
    return endpoints
```

Digit Recognizer (AngularJS files) :

Login Component

login.component.html

```
<div class="entire-login">
    <mat-card class="mat-elevation-z8">
        <mat-card-header class="flex-center">
            <mat-card-title >
                Welcome Bud!
            </mat-card-title>
        </mat-card-header>
        <mat-card-actions class="flex-center">
                  <mat-button-toggle-group style="margin: auto;" appearance="legacy">
                              <mat-button-toggle value="sign_in" checked="true">Sign in</mat-button-toggle>
                              <mat-button-toggle value="sign_up" disabled="true">Sign up</mat-button-toggle>
            </mat-button-toggle-group>
        </mat-card-actions>
        <mat-card-content class="card-content">
            <mat-form-field appearance="outline">
                <mat-label>Email id</mat-label>
                <mat-icon matPrefix>perm_identity</mat-icon>
                        <input id="email" name="email" matInput type='email' [formControl]="emailFC" placeholder="" #email/>
                <mat-error *ngIf="emailFC.hasError('required')">
                    Email id is required
                </mat-error>
                  <mat-error *ngIf="emailFC.hasError('email') && !emailFC.hasError('required')">
                    Valid email id is required
                </mat-error>
            </mat-form-field>
            <mat-form-field appearance="outline">
                <mat-label >Password</mat-label>
```

```
                <mat-icon matPrefix> vpn_key</mat-icon>
                   <input id="password" name="password" matInput type="password" [formControl]='passwordFC' #password>
                                                                <mat-error *ngIf="passwordFC.hasError('required')">
                     Password is required
                </mat-error>
            </mat-form-field>
            <mat-card-actions align="end">
                      <button  mat-raised-button  color="primary" (click)="val_credentials(email.value,password.value)">Sign in</button>
            </mat-card-actions>
        </mat-card-content>
    </mat-card>
</div>
```

login.component.css

```
.entire-login{
    display: flex;
    justify-content: center;
    align-items: center;
    height:85vh;
    background: #f7f7f7;
}

.card-content{
    display: flex;
    flex-direction: column;
}
.flex-center{
    display: flex;
    justify-content: center;
}
.mat-card{
    /* background-color: aliceblue; */
    box-shadow: 50px;
    font-family: 'Times New Roman', Times, serif;
}
```

login.component.spec.ts

```
// Done by Harshath.M

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { LoginComponent } from './login.component';

describe('LoginComponent', () => {
  let component: LoginComponent;
  let fixture: ComponentFixture<LoginComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ LoginComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(LoginComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

login.component.ts

```
// Done by Harshath.M

import { HttpClient } from '@angular/common/http';
import { Component, OnInit } from '@angular/core';
import { FormControl, Validators } from '@angular/forms';
import { MatSnackBar } from '@angular/material/snack-bar';
import { Router } from '@angular/router';

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.css']
})
```

```
14 export class LoginComponent implements OnInit {
15   email = "";
16   pwd = "";
17   invalid= true;
18   showbutton = true;
19
20     constructor(private  route:Router,  private  http:HttpClient,
   private snackbar:MatSnackBar) {
21
22   }
23
24   ngOnInit(): void {}
25                                emailFC              =              new
   FormControl('',[Validators.email,Validators.required]);
26   passwordFC = new FormControl('',[Validators.required]);
27
28   val_credentials(email:string,pwd:string){
29     let formdata = new FormData();
30     formdata.append("email",email);
31     formdata.append("pwd",pwd);
32     let api_url = "http://127.0.0.1:5000/api/";
33     this.http.post(api_url+"verify",formdata).subscribe({
34       next:((res:any)=>{
35         if(res.statusmessage=='true'){
36                       this.snackbar.open("Email  and  Password  is
   verified😎","Welcome", {duration:2000});
37           this.route.navigate(['/upload']);
38         }
39         else if(res.statusmessage=="false"){
40              this.snackbar.open("Invalid Credentials🙅♂️","Close",
   {duration:4000});
41         }
42         else{
43                        this.snackbar.open("Oops!  Something  went
   wrong😟","Close", {duration:4000});
44         }
45       }),
46       error:((()=>{
47         this.showbutton=true;
48              this.snackbar.open("Oops!  Server  is  not  available
   😟","Close", {duration:4000});
49       })
50     });
```

```
  }
}
```

Page not found Component

page-not-found.component.html

```
<div>
    <p>Page not found!!!</p>
    <p>Please enter the correct URL...</p>
</div>
```

page-not-found.component.css

```
div{
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    height: 80%;
}
```

page-not-found.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';
import { PageNotFoundComponent } from './page-not-found.component';
describe('PageNotFoundComponent', () => {
  let component: PageNotFoundComponent;
  let fixture: ComponentFixture<PageNotFoundComponent>;
  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ PageNotFoundComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(PageNotFoundComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
```

```
  });
});
```

page-not-found.component.ts

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-page-not-found',
  templateUrl: './page-not-found.component.html',
  styleUrls: ['./page-not-found.component.css']
})
export class PageNotFoundComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {
  }

}
```

Result Component

result.component.html

```
<div class="middle" *ngIf="!load_graph">
    <mat-progress-spinner mode="indeterminate"  diameter="60" strokeWidth="7" ></mat-progress-spinner>
</div>
<canvasjs-chart class="chart" *ngIf="load_graph" [options]="chartOptions" [styles]="{width: '100%', height:'360px'}"></canvasjs-chart>

<div *ngIf="load_graph" class="description">
  <h2>The Recognized Handwritten Digit from your image is {{pred_value}} ➔</h2>
  <img [src]="preview">
</div>
```

result.component.css

```
.middle{
    display: flex;
    justify-content: center;
```

```
    align-items: center;
    height:80%;
}
.chart{
    margin-top:20px;
}
.metrics-table{
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
    margin: 50px 0;
}
table{
    min-width: 350px;
    margin-bottom: 20px;
}
.description{
  display: flex;
  justify-content: space-evenly;
  flex-wrap: wrap;
  align-items: center;
}
.description h2{
  font-family: 'Times New Roman', Times, serif;
  font-weight: bold;
}
```

result.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ResultComponent } from './result.component';

describe('ResultComponent', () => {
  let component: ResultComponent;
  let fixture: ComponentFixture<ResultComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ ResultComponent ]
```

```
    })
    .compileComponents();

    fixture = TestBed.createComponent(ResultComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

result.component.ts

```
import { HttpClient } from '@angular/common/http';
import { Component, OnInit } from '@angular/core';
import { DomSanitizer } from '@angular/platform-browser';
import { saveAs } from 'file-saver';


// Done by Harshath.M

@Component({
  selector: 'app-result',
  templateUrl: './result.component.html',
  styleUrls: ['./result.component.css']
})
export class ResultComponent implements OnInit {

  constructor(private http:HttpClient,
              private domsanitizer:DomSanitizer) { }
   chartOptions:any;
  pred_value = 0 ;
   load_graph = false;
  preview: any;
  api_url = "http://127.0.0.1:5000/api/";

  ngOnInit(): void {

this.http.get(this.api_url+'image',{responseType:'blob'}).subscr
ibe({
```

```
      next:((res:any)=>{
        let objecturl = URL.createObjectURL(res);
        this.preview = this.domsanitizer.bypassSecurityTrustUrl(objecturl);
      })
    });

this.http.get(this.api_url+"predict").subscribe((res:any)=>{
      this.pred_value = res.value;
            this.open_page();
            this.load_graph = true;
        });
  }
  getDataPoints() {
    let dataPoints =[];
    for (var i = 0; i <= 9 ; i++)
      dataPoints.push({
        x: i,
        y: 0
      });
    dataPoints[this.pred_value]= { x : this.pred_value ,y:100, indexLabel: "Highest\u2705"};
    console.log(dataPoints);
        return dataPoints;
   }

   open_page(){
        this.chartOptions = { //https://canvasjs.com/angular-charts/chart-index-data-label/
      animationEnabled: true,
            exportEnabled: true,
            theme: "light2",
            title: {
                  text: "Recognized Result"
            },
      axisX: {
        title: "Digits",
        interval: 1
      },
      axisY:{
        title: "Prediction value (%)",
        maximum: 110,
        interval:25
```

```
      },
              data: [{
                    type: "column",
            dataPoints: this.getDataPoints()
              }]
         }
   }
}
```

Upload Component

upload.component.html

```
<div class="entire">
    <mat-card class="card mat-elevation-z8">
        <mat-card-title style="text-align: center;">Input Field</mat-card-title>
        <mat-form-field appearance="outline">
            <input hidden type='file' accept="image/*" #fileclick (change)="select_file($event)" >
            <input readonly matInput value="{{this.fname}}" placeholder="Choose image file" >
            <button *ngIf="this.file" matSuffix (click)="deletefile()" matTooltip="Remove File" matTooltipPosition = "above" color="warn" mat-icon-button>
                <mat-icon>close</mat-icon>
            </button>
            <button matSuffix mat-mini-fab color="primary" (click)="fileclick.click()" matTooltip="Select a file" matTooltipPosition="right">
                <mat-icon>backup</mat-icon>
            </button>
        </mat-form-field>
        <button (click)="predict()" mat-raised-button color="primary" style="min-height: 40px;">
            <span>Predict</span>
        </button>
    </mat-card>

    <mat-card class="card mat-elevation-z8" *ngIf="enable_preview">
        <mat-card-title style="text-align: center;">Preview</mat-card-title>
```

```
        <img [src]="preview">
    </mat-card>
</div>
```

upload.component.css

```

.entire{
    display: flex;
    justify-content: space-evenly;
    flex-wrap: wrap;
    align-items: center;
    height:85vh;
    background: #f7f7f7;
}
.card{
    display: flex;
    flex-direction: column;
    justify-content: center;
}
.mat-card{
    box-shadow: 50px;
    font-family: 'Times New Roman', Times, serif;
}
```

upload.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { UploadComponent } from './upload.component';

describe('UploadComponent', () => {
  let component: UploadComponent;
  let fixture: ComponentFixture<UploadComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ UploadComponent ]
    })
    .compileComponentseateComponent(UploadComponent);
    component = fixture.componentInstance;
```

```
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
```

upload.component.ts

```
import { HttpClient } from '@angular/common/http';
import { Component, OnInit } from '@angular/core';
import { MatSnackBar } from '@angular/material/snack-bar';
import { Router } from '@angular/router';
import { DomSanitizer } from '@angular/platform-browser';

// Done by Harshath.M

@Component({
  selector: 'app-upload',
  templateUrl: './upload.component.html',
  styleUrls: ['./upload.component.css']
})
export class UploadComponent implements OnInit {

  all_formats=['png', 'jpg', 'jpeg']
  file :any;
  fname ='';
  fformat='';
  formdata:any;
  enable_preview =false;
  preview : any;

  constructor(private snackbar:MatSnackBar,
              private http:HttpClient,
              private route:Router,
              private domsanitizer:DomSanitizer) { }

  ngOnInit(): void {
  }

  select_file(event : any){
    try{
```

```
      this.file = event.target.files[0];
      if(this.file){
        this.fname = this.file.name;
        this.fformat = this.file.type.split('/')[1];
        if(this.all_formats.indexOf(this.fformat)!=-1){
          this.formdata= new FormData();
          this.formdata.append('image', this.file);
          this.formdata.append("format", this.fformat);

          let api_url = "http://127.0.0.1:5000/api/upload";

this.http.post(api_url,this.formdata,{responseType:'blob'}).subscribe({
            next:((res:any)=>{
              let objecturl = URL.createObjectURL(res);
              this.preview = this.domsanitizer.bypassSecurityTrustUrl(objecturl);
            }),
            error:(()=>{
              this.snackbar.open("Oops! Server is not available 😟","Close", {duration:4000});
            }),
            complete:(()=> this.enable_preview=true)
          });
        }
        else{
          this.snackbar.open("Please select a jpg/jpeg/png file","Got it" ,{duration :3000});
          this.fname='';
          this.fformat='';
          this.file=null;
        }
      }
    }
    catch(err){
      console.log(err);
    }
  }

  deletefile(){
    this.fname='';
    this.fformat='';
    this.file=null;
```

```
    this.formdata.delete("image");
    this.formdata.delete("format")
    this.enable_preview=false;
  }

// .subscribe(next?: ((value: string) => void) | null | undefined,
// error?: ((error: any) => void) | null | undefined,
//  complete?: (() => void) | null | undefined): Subscription (+2 overloads)

  predict(){
      if(this.file){
        this.route.navigate(['result']);
      }
      else{
        this.snackbar.open("Please select a file ⊠","Okay",{duration:3000});
      }
  }
}
```

about-dialog.html

```
<mat-card-title>About Handwritten Digit Recognizer</mat-card-title>
<mat-dialog-content>
  <br>
  <h3>Abstract</h3>
  <p>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.</p>

  <h3>Procedure</h3>
  <ol>
    <li><b>Login -- </b>This is first page when you entered into the webapp. If you entered the user credentials(i.e.,Email id, Password) correctly, you are redirected to the next page</li>
    <li><b>Upload -- </b>In this page, you can upload the handwritten digit image from your local system. Immediately after pick the image, preview of the image is shown to you for extra verification.</li>
    <li><b>Result -- </b>The predicted value of the image that you upload is shown in this page. Column chart is also provided to see the result in graphical representation.</li>
  </ol>

  <h3>Upload a image which is similar to the image shown below.</h3>
  <img src="https://drive.google.com/uc?export=view&id=1yBFVSuzMFnIFxXh7PmhG3eR5nYLPZMDO" alt="example image" style="display: block; margin:auto;">

  <h3>Developed by:</h3>
  <ul>
    <li>Harshath.M</li>
    <li>Priyanga.S</li>
    <li>Suvetha.M</li>
    <li>Ajeeth Kumar.S</li>
  </ul>

</mat-dialog-content>
<mat-dialog-actions align="end">
  <button mat-button mat-dialog-close>Cancel</button>
</mat-dialog-actions>
```

app-routing.module.ts

```
import { NgModule } from '@angular/core';
```

```
import { RouterModule, Routes } from '@angular/router';
import { LoginComponent } from './login/login.component';
import { UploadComponent } from './upload/upload.component';
import { ResultComponent } from './result/result.component';
import { PageNotFoundComponent } from './page-not-found/page-not-found.component';

const routes: Routes = [
  {path:'', redirectTo:'login', pathMatch:'full'},
  {path:'login', component:LoginComponent},
  {path:'upload', component:UploadComponent},
  {path:'result', component:ResultComponent},
  {path:"**", component:PageNotFoundComponent}
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

app.component.css

```
.abt-btn{
    background:#fff;
    color: #3f51b5;
}
.toolbar{
    display: flex;
    justify-content: space-around;
    flex-wrap: wrap;

}
.footer {
  display: flex;
  justify-content: space-around;
  flex-wrap: wrap;
  height: auto;
}
.toolbar span{
  display: flex;

```

```
}
```

app.component.html

```
<mat-toolbar color="primary" class="toolbar">
    <span>
      <img src="assets\white icon.svg" alt="" height="30px">
       Handwritten Digit Recognizer
    </span>
    <button mat-raised-button (click)="openDialog()" class="abt-btn">About</button>
</mat-toolbar>
<router-outlet></router-outlet>
<mat-toolbar color="primary" class="footer">
  Developed by:
    <li>Harshath.M</li>
    <li>Priyanga.S</li>
    <li>Suvetha.M</li>
    <li>Ajeeth Kumar.S</li>
</mat-toolbar>
```

app.component.spec.ts

```
import { TestBed } from '@angular/core/testing';
import { RouterTestingModule } from '@angular/router/testing';
import { AppComponent } from './app.component';

describe('AppComponent', () => {
  beforeEach(async () => {
    await TestBed.configureTestingModule({
      imports: [
        RouterTestingModule
      ],
      declarations: [
        AppComponent
      ],
    }).compileComponents();
  });

  it('should create the app', () => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.componentInstance;
```

```
    expect(app).toBeTruthy();
  });

  it(`should have as title 'Digit_Recognizer'`, () => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.componentInstance;
    expect(app.title).toEqual('Digit_Recognizer');
  });

  it('should render title', () => {
    const fixture = TestBed.createComponent(AppComponent);
    fixture.detectChanges();
    const compiled = fixture.nativeElement as HTMLElement;
    expect(compiled.querySelector('.content span')?.textContent).toContain('Digit_Recognizer app is running!');
  });
});
```

app.component.ts

```
import { Component } from '@angular/core';
import { MatDialog } from '@angular/material/dialog';
import { MAT_DIALOG_DATA } from '@angular/material/dialog';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})

export class AppComponent {
  constructor(private dialog:MatDialog){}

  openDialog(){
    this.dialog.open(AboutDialog);
  }

  title = 'Digit_Recognizer';
}

@Component({
```

```
  selector: 'about-dialog',
  templateUrl:'./about-dialog.html'
})
export class AboutDialog{}
```

app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import {MatCardModule} from '@angular/material/card';
import {MatFormFieldModule} from '@angular/material/form-field';
import {MatInputModule} from '@angular/material/input';
import { ReactiveFormsModule } from '@angular/forms';
import {MatButtonModule} from '@angular/material/button';
import {MatButtonToggleModule} from '@angular/material/button-toggle';
import { HttpClientModule } from '@angular/common/http';
import {MatSelectModule} from '@angular/material/select';
import {MatTableModule} from '@angular/material/table';
import {MatToolbarModule} from '@angular/material/toolbar';
import {MatIconModule} from '@angular/material/icon';
import {MatTooltipModule} from '@angular/material/tooltip';
import {MatSnackBarModule} from '@angular/material/snack-bar';
import {MatProgressSpinnerModule} from '@angular/material/progress-spinner';
import {MatDialogModule} from '@angular/material/dialog';

import * as CanvasJSAngularChart from '../assets/canvasjs.angular.component';
var CanvasJSChart = CanvasJSAngularChart.CanvasJSChart;

import { AboutDialog } from './app.component';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { LoginComponent } from './login/login.component';
import { UploadComponent } from './upload/upload.component';
import { ResultComponent } from './result/result.component';
import { PageNotFoundComponent } from './page-not-found/page-not-found.component';
```

```
31
32 @NgModule({
33   declarations: [
34     AppComponent,
35     LoginComponent,
36     UploadComponent,
37     ResultComponent,
38     PageNotFoundComponent,
39     CanvasJSChart,
40     AboutDialog
41   ],
42   imports: [
43     BrowserModule,
44     AppRoutingModule,
45     MatCardModule,
46     MatFormFieldModule,
47     MatInputModule,
48     ReactiveFormsModule,
49     MatButtonModule,
50     MatButtonToggleModule,
51     HttpClientModule,
52     MatSelectModule,
53     MatTableModule,
54     MatToolbarModule,
55     BrowserAnimationsModule,
56     MatIconModule,
57     MatTooltipModule,
58     MatSnackBarModule,
59     MatProgressSpinnerModule,
60     MatDialogModule
61   ],
62   providers: [],
63   bootstrap: [AppComponent]
64 })
65 export class AppModule { }
66
```

angular.json

```
1  {
2    "$schema":
   "./node_modules/@angular/cli/lib/config/schema.json",
```

```
  "version": 1,
  "newProjectRoot": "projects",
  "projects": {
    "Digit_Recognizer": {
      "projectType": "application",
      "schematics": {},
      "root": "",
      "sourceRoot": "src",
      "prefix": "app",
      "architect": {
        "build": {
          "builder": "@angular-devkit/build-angular:browser",
          "options": {
            "outputPath": "dist/digit-recognizer",
            "index": "src/index.html",
            "main": "src/main.ts",
            "polyfills": "src/polyfills.ts",
            "tsConfig": "tsconfig.app.json",
            "assets": [
              "src/favicon.ico",
              "src/assets"
            ],
            "styles": [
              "./node_modules/@angular/material/prebuilt-themes/indigo-pink.css",
              "src/styles.css"
            ],
            "scripts": []
          },
          "configurations": {
            "production": {
              "budgets": [
                {
                  "type": "initial",
                  "maximumWarning": "1mb",
                  "maximumError": "2mb"
                },
                {
                  "type": "anyComponentStyle",
                  "maximumWarning": "2kb",
                  "maximumError": "4kb"
                }
              ],
```

```
45              "fileReplacements": [
46                {
47                  "replace": "src/environments/environment.ts",
48                  "with": "src/environments/environment.prod.ts"
49                }
50              ],
51              "outputHashing": "all"
52            },
53            "development": {
54              "buildOptimizer": false,
55              "optimization": false,
56              "vendorChunk": true,
57              "extractLicenses": false,
58              "sourceMap": true,
59              "namedChunks": true
60            }
61          },
62          "defaultConfiguration": "production"
63        },
64        "serve": {
65          "builder": "@angular-devkit/build-angular:dev-server",
66          "configurations": {
67            "production": {
68              "browserTarget":
   "Digit_Recognizer:build:production"
69            },
70            "development": {
71              "browserTarget":
   "Digit_Recognizer:build:development"
72            }
73          },
74          "defaultConfiguration": "development"
75        },
76        "extract-i18n": {
77          "builder": "@angular-devkit/build-angular:extract-
   i18n",
78          "options": {
79            "browserTarget": "Digit_Recognizer:build"
80          }
81        },
82        "test": {
83          "builder": "@angular-devkit/build-angular:karma",
84          "options": {
```

```
            "main": "src/test.ts",
            "polyfills": "src/polyfills.ts",
            "tsConfig": "tsconfig.spec.json",
            "karmaConfig": "karma.conf.js",
            "assets": [
              "src/favicon.ico",
              "src/assets"
            ],
            "styles": [
              "./node_modules/@angular/material/prebuilt-themes/indigo-pink.css",
              "src/styles.css"
            ],
            "scripts": []
          }
        },
          "deploy": {
            "builder": "angular-cli-ghpages:deploy"
          }
        }
      }
    }
  }
```

package.json

```
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "version": 1,
  "newProjectRoot": "projects",
  "projects": {
    "Digit_Recognizer": {
      "projectType": "application",
      "schematics": {},
      "root": "",
      "sourceRoot": "src",
      "prefix": "app",
      "architect": {
        "build": {
          "builder": "@angular-devkit/build-angular:browser",
          "options": {
```

```
            "outputPath": "dist/digit-recognizer",
            "index": "src/index.html",
            "main": "src/main.ts",
            "polyfills": "src/polyfills.ts",
            "tsConfig": "tsconfig.app.json",
            "assets": [
              "src/favicon.ico",
              "src/assets"
            ],
            "styles": [
              "./node_modules/@angular/material/prebuilt-themes/indigo-pink.css",
              "src/styles.css"
            ],
            "scripts": []
          },
          "configurations": {
            "production": {
              "budgets": [
                {
                  "type": "initial",
                  "maximumWarning": "1mb",
                  "maximumError": "2mb"
                },
                {
                  "type": "anyComponentStyle",
                  "maximumWarning": "2kb",
                  "maximumError": "4kb"
                }
              ],
              "fileReplacements": [
                {
                  "replace": "src/environments/environment.ts",
                  "with": "src/environments/environment.prod.ts"
                }
              ],
              "outputHashing": "all"
            },
            "development": {
              "buildOptimizer": false,
              "optimization": false,
              "vendorChunk": true,
              "extractLicenses": false,
```

```
58              "sourceMap": true,
59              "namedChunks": true
60            }
61          },
62          "defaultConfiguration": "production"
63        },
64        "serve": {
65          "builder": "@angular-devkit/build-angular:dev-server",
66          "configurations": {
67            "production": {
68              "browserTarget":
   "Digit_Recognizer:build:production"
69            },
70            "development": {
71              "browserTarget":
   "Digit_Recognizer:build:development"
72            }
73          },
74          "defaultConfiguration": "development"
75        },
76        "extract-i18n": {
77          "builder": "@angular-devkit/build-angular:extract-
   i18n",
78          "options": {
79            "browserTarget": "Digit_Recognizer:build"
80          }
81        },
82        "test": {
83          "builder": "@angular-devkit/build-angular:karma",
84          "options": {
85            "main": "src/test.ts",
86            "polyfills": "src/polyfills.ts",
87            "tsConfig": "tsconfig.spec.json",
88            "karmaConfig": "karma.conf.js",
89            "assets": [
90              "src/favicon.ico",
91              "src/assets"
92            ],
93            "styles": [
94              "./node_modules/@angular/material/prebuilt-
   themes/indigo-pink.css",
95              "src/styles.css"
96            ],
```

```
            "scripts": []
          }
        },
          "deploy": {
            "builder": "angular-cli-ghpages:deploy"
          }
        }
      }
    }
  }
```

Links

[Github Link]

[Demo Link]

[Deployed angular UI link]