PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION

TEAM ID: PNT2022TMID49570

TEAM MEMBERS

JEBA SALOMI. D

ANISHA SHAHINI. D

MALIGA FATHIMA ROWFINA. S

MARIA REXLINE. R

1. INTRODUCTION

1.1 Project Overview

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 digitized to reduce human effort. Hence, there comes a need for handwritten digit recognition in many real time applications. MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. We use artificial neural networks to train these images and build a deep learning model. Web application is created where the user can upload an image of a handwritten digit. this image is analyzed by the model and the detected result is returned on to UI.

1.2 Purpose

- Digit recognition plays an important role in the modern world.
- It can solve more complex problems and makes humans job easier. This type of system can be widely used in the world to recognize zip code or postal code for mail sorting
- In Banking Sector too where more handwritten numbers are involved like account number, figure of cash and checks.

2. LITERATURE SURVEY

2.1 Existing Problem

1] Microsoft Math solver:

Microsoft Math Solver app is used for solving variety of problems including arithmetic, algebra, trigonometry, calculus, statistics and other topics using an advanced AI.

Advantages:

- The image recognition is good in this app.
- It also recognizes mathematical solution.
- It is simple and very easy to use
- The handwriting detection of this app is good.
- This app scans and get the right numbers even in the bad lighting.

Disadvantages:

- This app is mostly used for solving mathematical equations but not for digit recognition.
- This app sometimes show wrong results.
- This app get slower when the target frame is resized.

2] Google Lens:

Google Lens app translates words, identifies plants, finds products and more using a camera. This app is used to scan and translate text, QR codes and bar codes.

Advantages:

- This app scans and translated text.
- This app is very responsive to photos.
- It translates the input photos accurately.
- It recognizes the handwritten text accurately.

Disadvantages:

- This app is not specially meant for digit recognition.
- This app has slow scanning process.

2.2 REFERENCES

1] Microsoft math solver

https://math.microsoft.com/

2] Google lens

https://lens.google/

2.3 PROBLEM STATEMENT DEFINITION

The user is a bank manager who needs a handwritten digit recognition system because the different handwriting are confusing. So, a novel method for handwritten digit recognition system must be developed.

Who does the problem affect?	Old people, bankers and customers
What are the boundaries of the problem?	Postal department, courier service and Banking sector
What is the issue?	Sometimes handwritten digits are confusing. so, the important details such as zip code, account number, figure of cash and checks may go wrong. By fixing this problem, handwritten digits are recognized correctly. If we did not solve this problem the transactions and mail sorting may gone wrong.
When does the issue occurs?	When the digits could not be recognized correctly. When the transactions are not successful. When the elder people unable to understand the smaller handwritten digits. When the courier service or postal department unable to recognize zip code or postal code for mail sorting.
Where is the issue occurring?	The issue occurs in banks and post office while transaction and mail sorting.

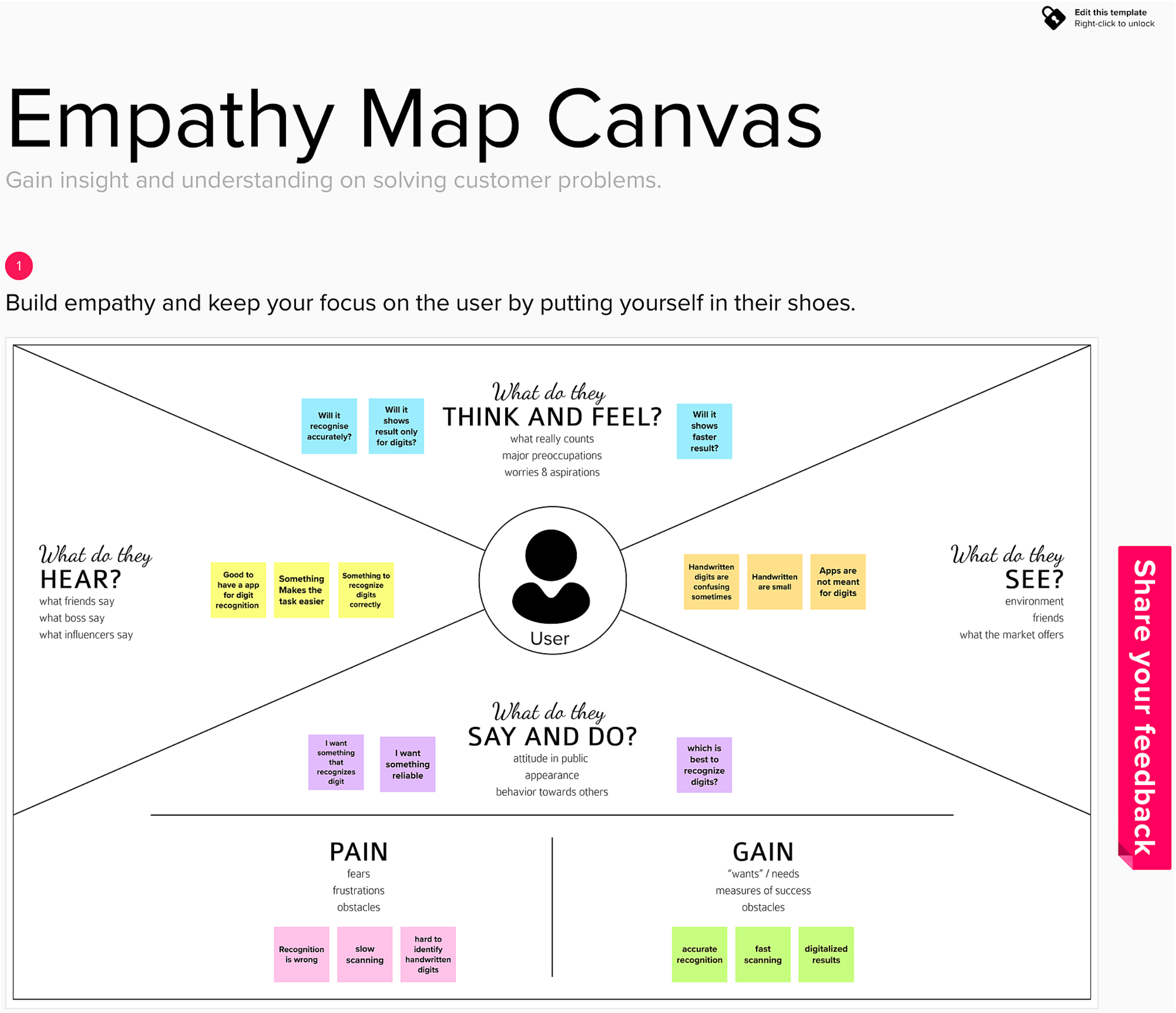
3. IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviors and attitudes.

It is a useful tool to help teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user’s perspective along with his or her goals and challenges.



Empathy Map Canvas

Gain insight and understanding on solving customer problems.

1

Build empathy and keep your focus on the user by putting yourself in their shoes.

What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations

- Will it recognise accurately?
- Will it shows result only for digits?
- Will it shows faster result?

What do they HEAR?
what friends say
what boss say
what influencers say

- Good to have a app for digit recognition
- Something Makes the task easier
- Something to recognize digits correctly

What do they SEE?
environment
friends
what the market offers

- Handwritten digits are confusing sometimes
- Handwritten are small
- Apps are not meant for digits

What do they SAY AND DO?
attitude in public
appearance
behavior towards others

- I want something that recognizes digit
- I want something reliable
- which is best to recognize digits?

PAIN
fears
frustrations
obstacles

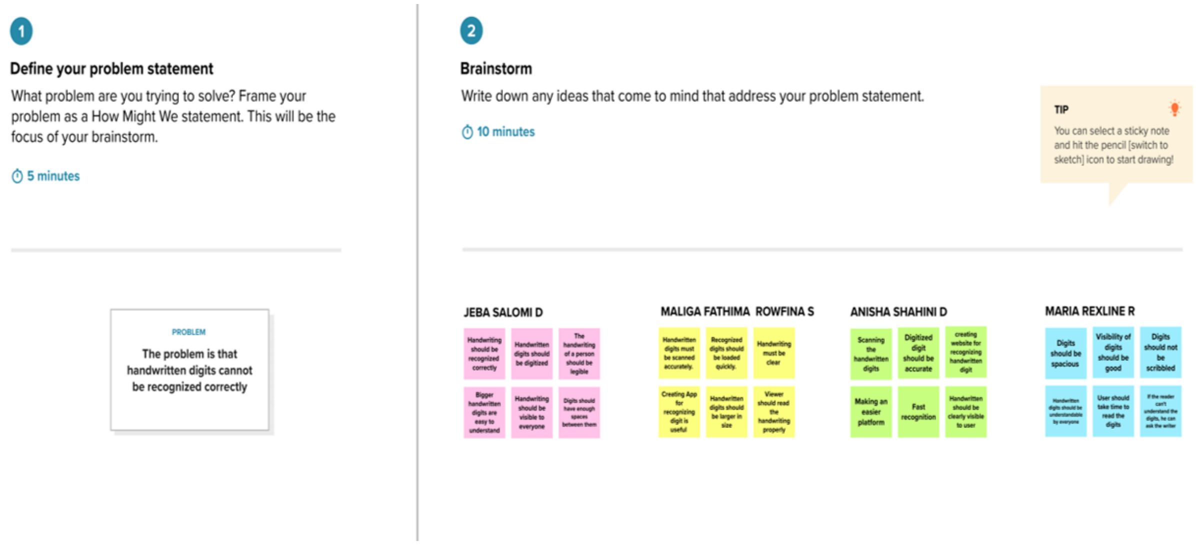
- Recognition is wrong
- slow scanning
- hard to identify handwritten digits

GAIN
“wants” / needs
measures of success
obstacles

- accurate recognition
- fast scanning
- digitalized results

3.2 Ideation and Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.



1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

The problem is that handwritten digits cannot be recognized correctly

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

JEBA SALOMI D

- Handwriting should be recognized correctly
- Handwritten digits should be digitized
- The handwriting of a person should be legible
- Bigger handwritten digits are easy to understand
- Handwriting should be visible to everyone
- Digits should have enough spaces between them

MALIGA FATHIMA ROWFINA S

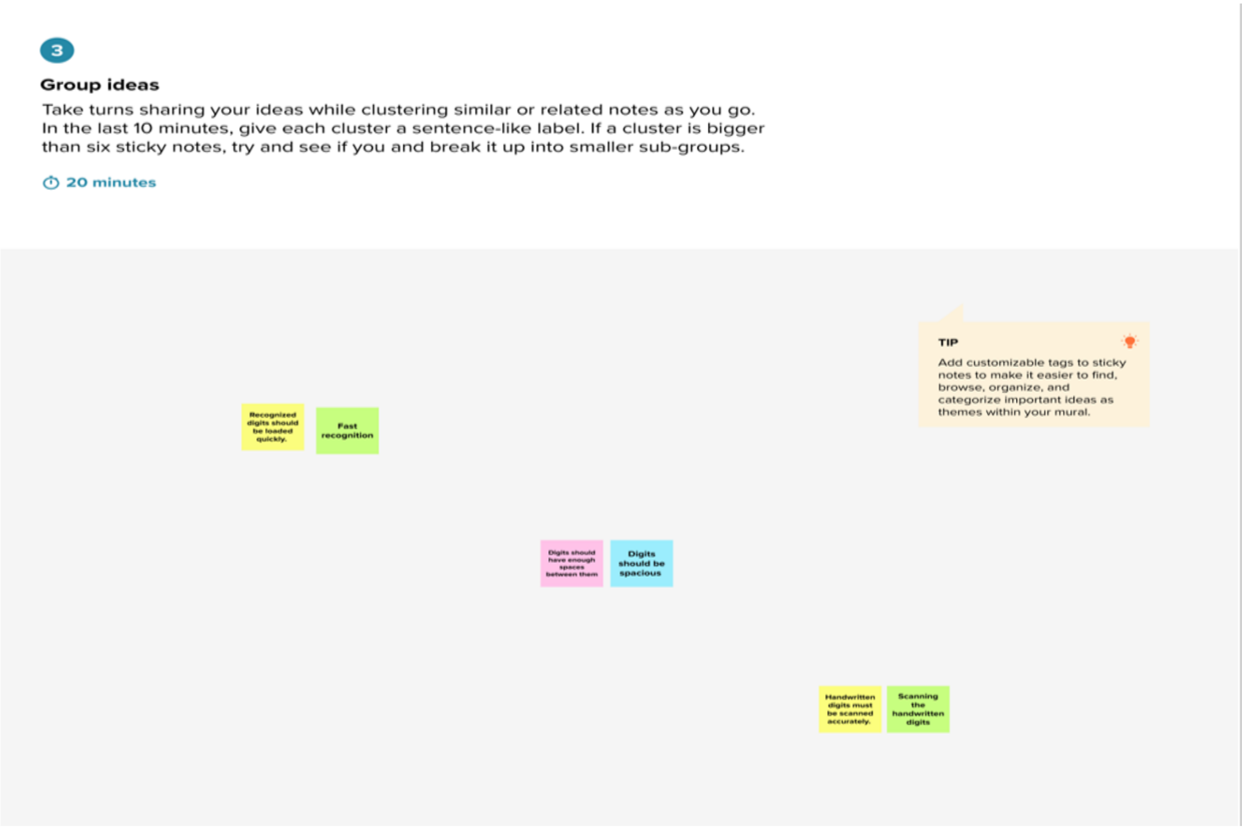
- Handwritten digits must be scanned accurately.
- Recognized digits should be loaded quickly.
- Handwriting must be clear
- Creating App for recognizing digit is useful
- Handwritten digits should be larger in size
- Viewer should read the handwriting properly

ANISHA SHAHINI D

- Scanning the handwritten digits
- Digitized digit should be accurate
- creating website for recognizing handwritten digit
- Making an easier platform
- Fast recognition
- Handwritten should be clearly visible to user

MARIA REXLINE R

- Digits should be spacious
- Visibility of digits should be good
- Digits should not be scribbled
- Handwritten digits should be understandable by everyone
- User should take time to read the digits
- If the reader can't understand the digits, he can ask the writer



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Recognized digits should be loaded quickly.

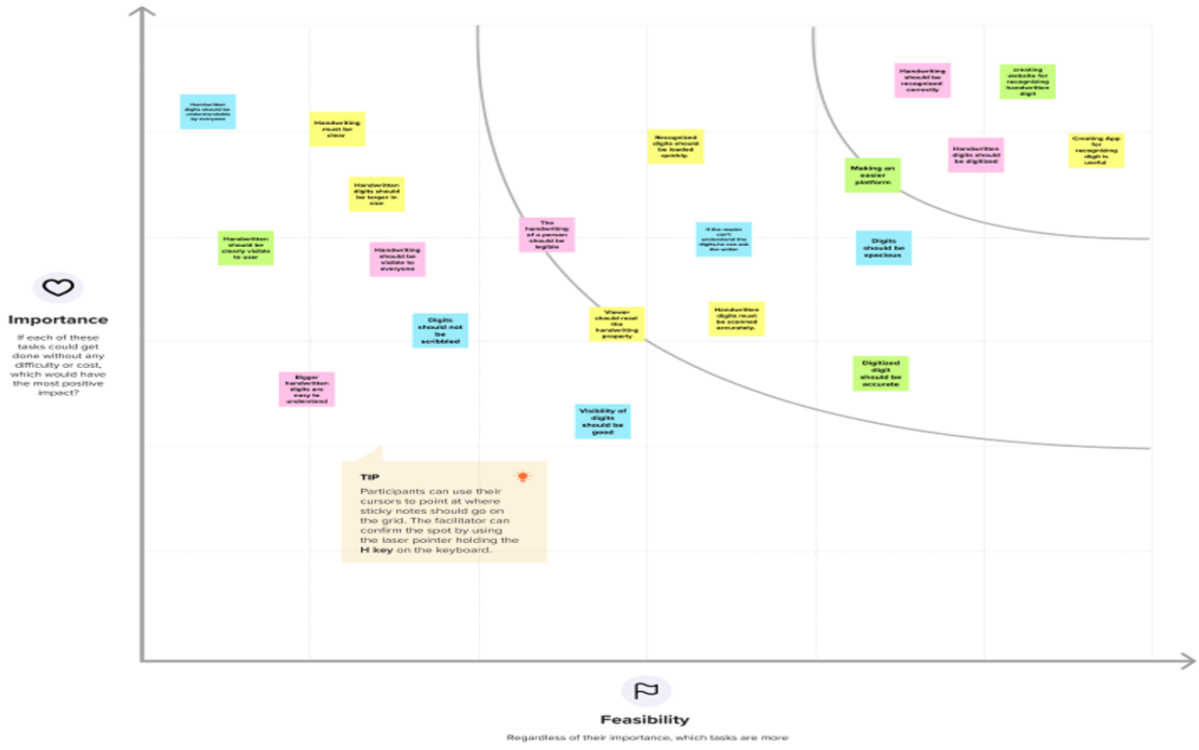
Fast recognition

Digits should have enough spaces between them

Digits should be spacious

Handwritten digits must be scanned accurately.

Scanning the handwritten digits


Handwritten digits should be understandable by everyone
Handwriting must be clear
Handwritten digits should be larger in size
Handwritten should be clearly visible to user
Handwriting should be visible to everyone
Digits should not be scribbled
Bigger handwritten digits are easy to understand
The handwriting of a person should be legible
Viewer should read the handwriting properly
Visibility of digits should be good
Recognized digits should be loaded quickly.
If the reader can't understand the digits,he can ask the writer
Handwritten digits must be scanned accurately.
Handwriting should be recognized correctly
creating website for recognizing handwritten digit
Making an easier platform
Handwritten digits should be digitized
Creating App for recognizing digit is useful
Digits should be spacious
Digitized digit should be accurate
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Feasibility
Regardless of their importance, which tasks are more

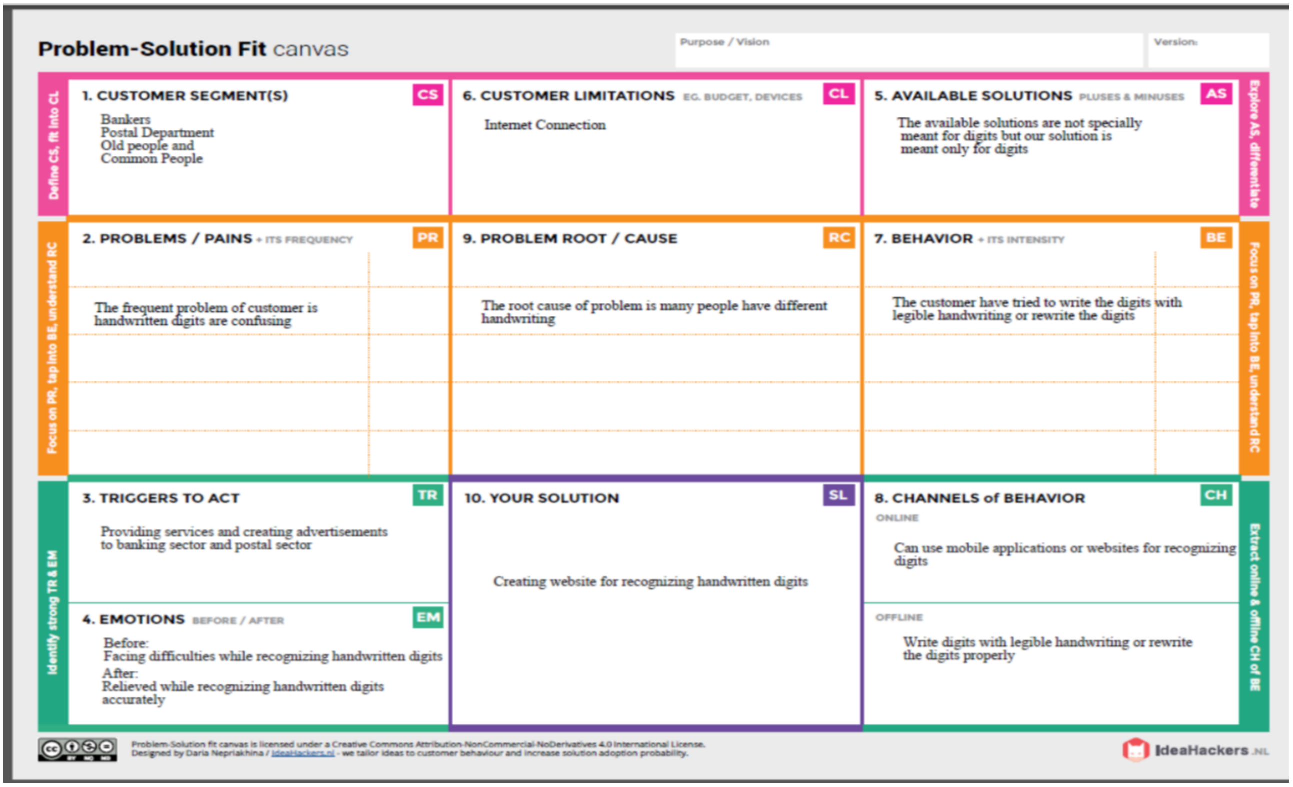
TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

3.3 Proposed Solution

S. NO	**PARAMETER**	**DESCRIPTION**
1.	Problem Statement (Problem to be solved)	Handwritten digits do not always have the same size, thickness and shape. So, the handwritten digits cannot be recognized correctly.
2.	Idea/Solution description	Handwritten digit recognition system is a way to tackle this problem which uses image of a digit to recognize the digit present in the image.
3.	Novelty/Uniqueness	This handwritten recognition system is meant for only digits, but other existed system is meant to recognize alphabets, expressions etc..
4.	Social Impact/Customer Satisfaction	○ Postal department and courier services can easily find the digits written. ○ Senior citizens who will have eye sight issues with handwritten digits can use this system to recognize the handwritten digits correctly
5.	Business Model	This system can be converted into a business model by providing services to the Banking sector and Postal sector.
6.	Scalability of the Solution	More number of handwritten digits can be recognized.

3.4 Problem Solution fit



Problem-Solution Fit canvas

Purpose / Vision

Version:

Define CS, fit into CL

1. CUSTOMER SEGMENT(S) CS

Bankers
Postal Department
Old people and
Common People

6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES CL

Internet Connection

5. AVAILABLE SOLUTIONS PLUSES & MINUSES AS

The available solutions are not specially meant for digits but our solution is meant only for digits

Explore AS, differentiate

Focus on PR, tap into BE, understand RC

2. PROBLEMS / PAINS + ITS FREQUENCY PR

The frequent problem of customer is handwritten digits are confusing

9. PROBLEM ROOT / CAUSE RC

The root cause of problem is many people have different handwriting

7. BEHAVIOR + ITS INTENSITY BE

The customer have tried to write the digits with legible handwriting or rewrite the digits

Focus on PR, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TO ACT TR

Providing services and creating advertisements to banking sector and postal sector

4. EMOTIONS BEFORE / AFTER EM

Before:
Facing difficulties while recognizing handwritten digits
After:
Relieved while recognizing handwritten digits accurately

10. YOUR SOLUTION SL

Creating website for recognizing handwritten digits

8. CHANNELS of BEHAVIOR CH

ONLINE

Can use mobile applications or websites for recognizing digits

OFFLINE

Write digits with legible handwriting or rewrite the digits properly

Extract online & offline CH of BE

IdeaHackers.NL

4. REQUIREMENT ANALYSIS

4.1 Functional Requirements

Following are the functional requirements of the proposed solution.

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	User uploading image	Upload through Local system
FR-2	Image verification	Verification via message
FR-3	Getting the result	Get result via user Interface

4.2 Non-Functional Requirements

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

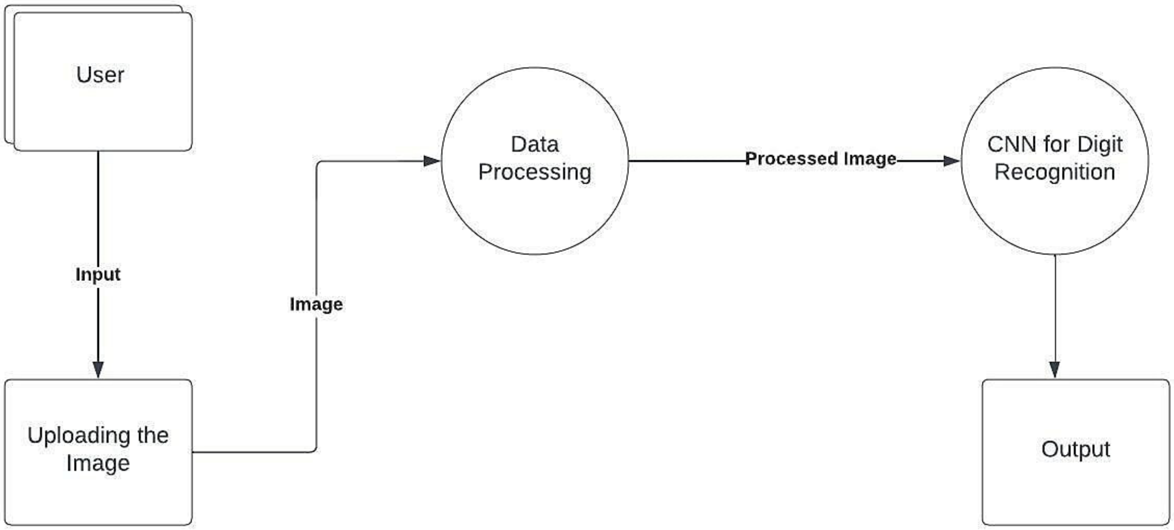
FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	People with no understanding to upload the image must be able to upload the image.
NFR-2	**Security**	Access permission for the local system must be given by the system's data administrator.
NFR-3	**Reliability**	The system will intimate the user to re upload the image if any failure occurs.
NFR-4	**Performance**	The front-page load time must be within a few seconds.
NFR-5	**Availability**	New module deployment must not impact front page and main page.
NFR-6	**Scalability**	The website traffic limit must be scalable.

5. PROJECT DESIGN

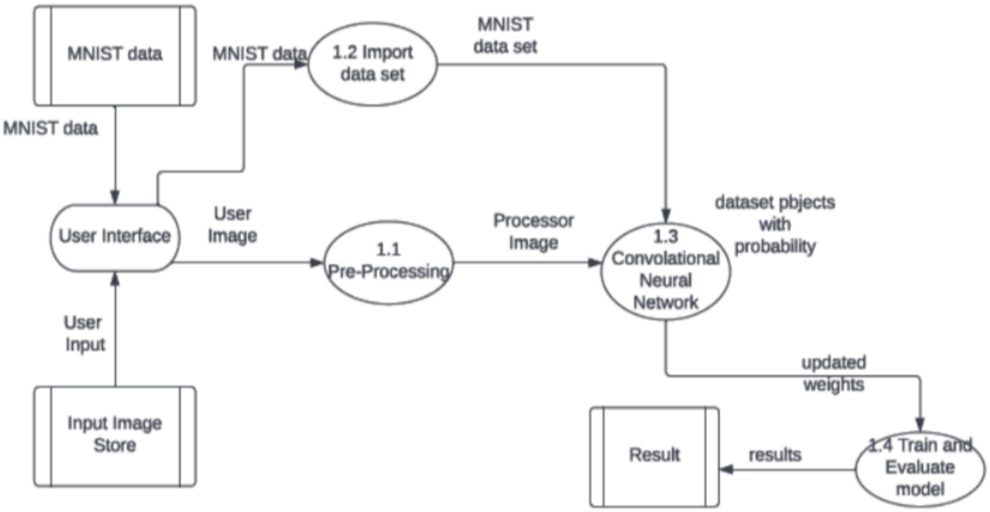
5.1 Data Flow Diagram

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.

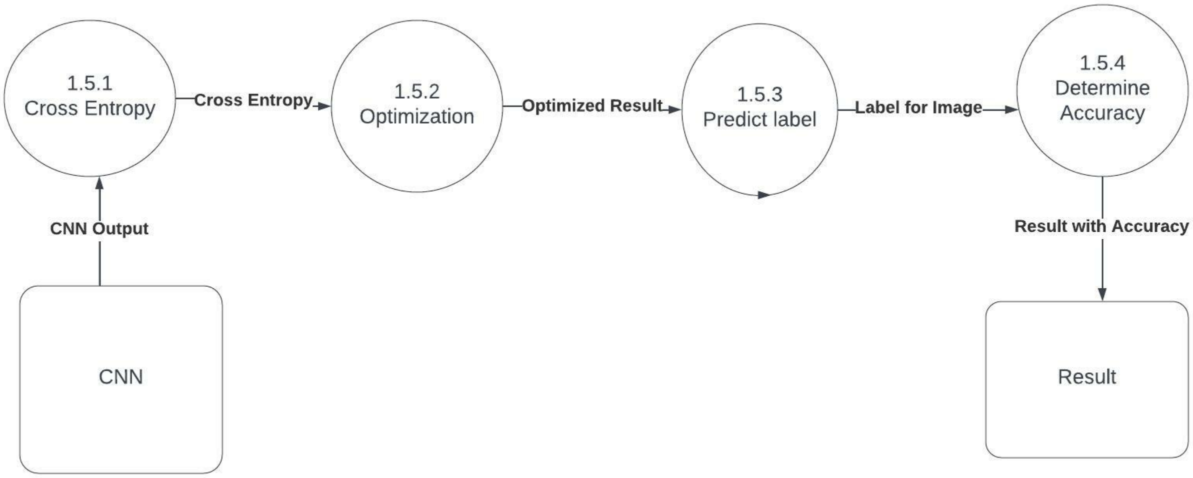
Level 0:



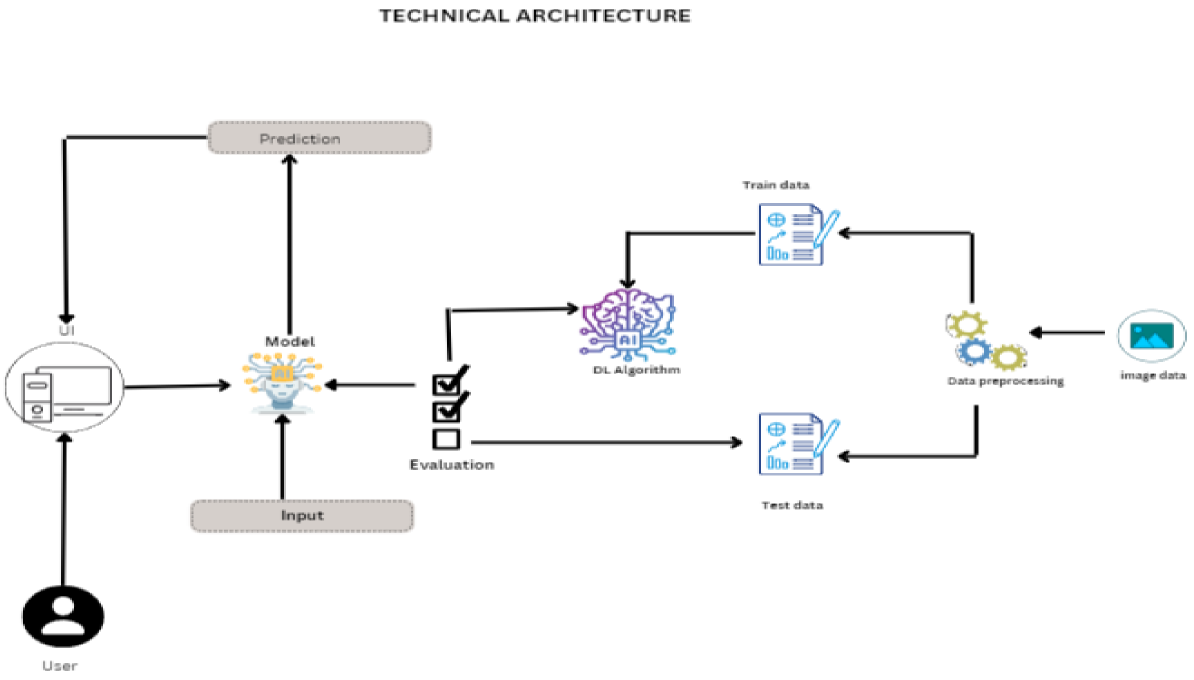
Level 1:



Level 2:



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 (Web user)	Front Page	USN-1	As a user, I can view the front page of the website where the description of the website is available	I can access my front page of the website	High	Sprint-1
	Choosing the Image	USN-2	As a user, I can choose the image from the local system	I can upload the image	High	Sprint-1
	Recognize the Image	USN-3	As a user, I can get message after validating the image	I can get a message	Low	Sprint-2
		USN-4	As a user, I can get the recognized digits	I can view digitized results	High	Sprint-3

6. PROJECT PLANNING AND SCHEDULING

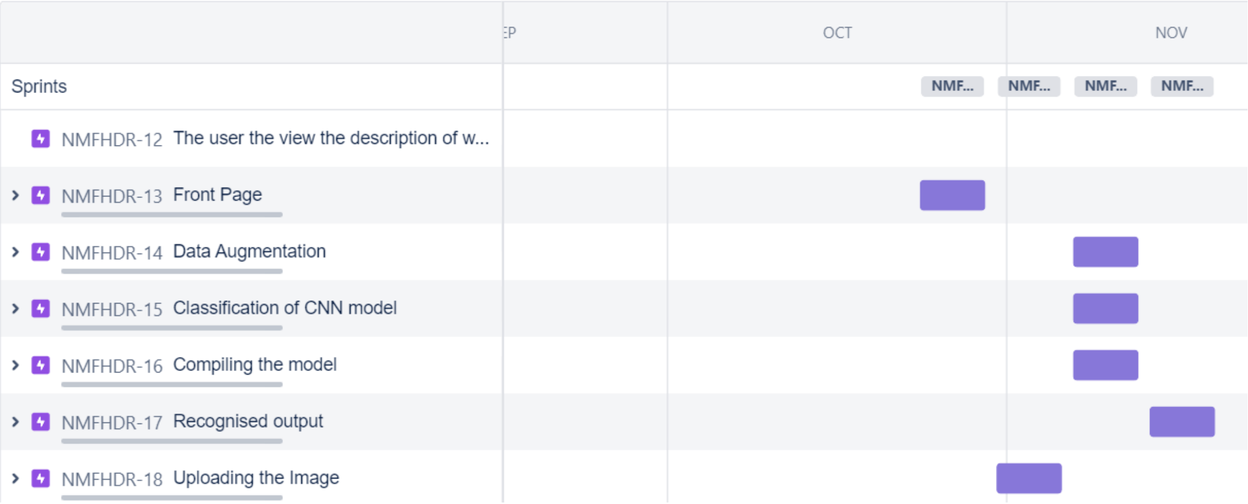
6.1 Sprint Planning and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Front Page	USN-1	As a user, I can view the front page of the website where the description of the website is available	2	High	Anisha Shahini D and Maliga Fathima Rowfina S
Sprint-2	Uploading the Image	USN-2	As a user, I can choose the image from the local system	2	Low	Maria Rexline R and Jeba Salomi D
Sprint-3	Data Augmentation	USN-3	As a developer, the image dataset must be augmented.	1	High	Anisha Shahini D and Maria Rexline R.
Sprint-3	Classification of CNN model	USN-4	As a developer, model must be classified	3	High	Jeba Salomi D, Maliga Fathima Rowfina S and Anisha Shahini D and Maria Rexline R.
Sprint-3	Compiling the model	USN-5	As a developer, a model must be compiled and fitted	3	High	Maliga Fathima Rowfina, S,and Jeba Salomi D
Sprint-4	Recognized output	USN-3	As a user, I can get the recognized digits.	2	High	Jeba Salomi D, Maliga Fathima Rowfina S, Maria Rexline R, Anisha Shahini D

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	20	6 Days	24 Oct 2022	29 Oct 2022	12	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	12	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	12	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	12	19 Nov 2022

6.3 Reports from JIRA



Sprints

NMFHDR-12 The user the view the description of w...

NMFHDR-13 Front Page

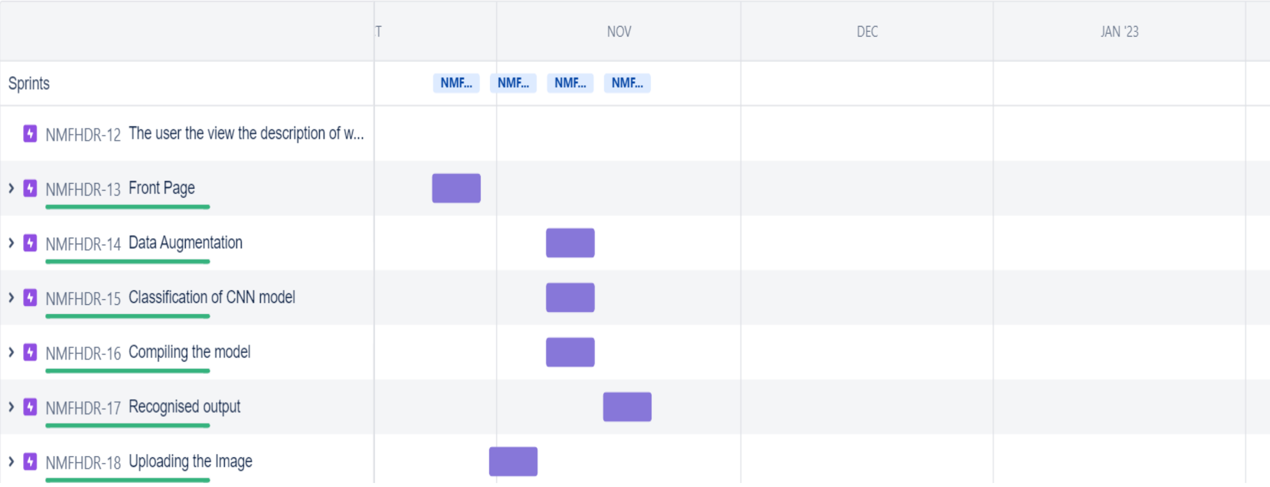
NMFHDR-14 Data Augmentation

NMFHDR-15 Classification of CNN model

NMFHDR-16 Compiling the model

NMFHDR-17 Recognised output

NMFHDR-18 Uploading the Image



Sprints

NMFHDR-12 The user the view the description of w...

NMFHDR-13 Front Page

NMFHDR-14 Data Augmentation

NMFHDR-15 Classification of CNN model

NMFHDR-16 Compiling the model

NMFHDR-17 Recognised output

NMFHDR-18 Uploading the Image

7. CODING AND SOLUTIONING

7.1 Feature 1

Uploading the image from local system :

Web application allows the user to upload the image from the local disk.

7.2 Feature 2

Recognise the digit:

Web application shows the result of predicted image

Source code:

recognise.html:

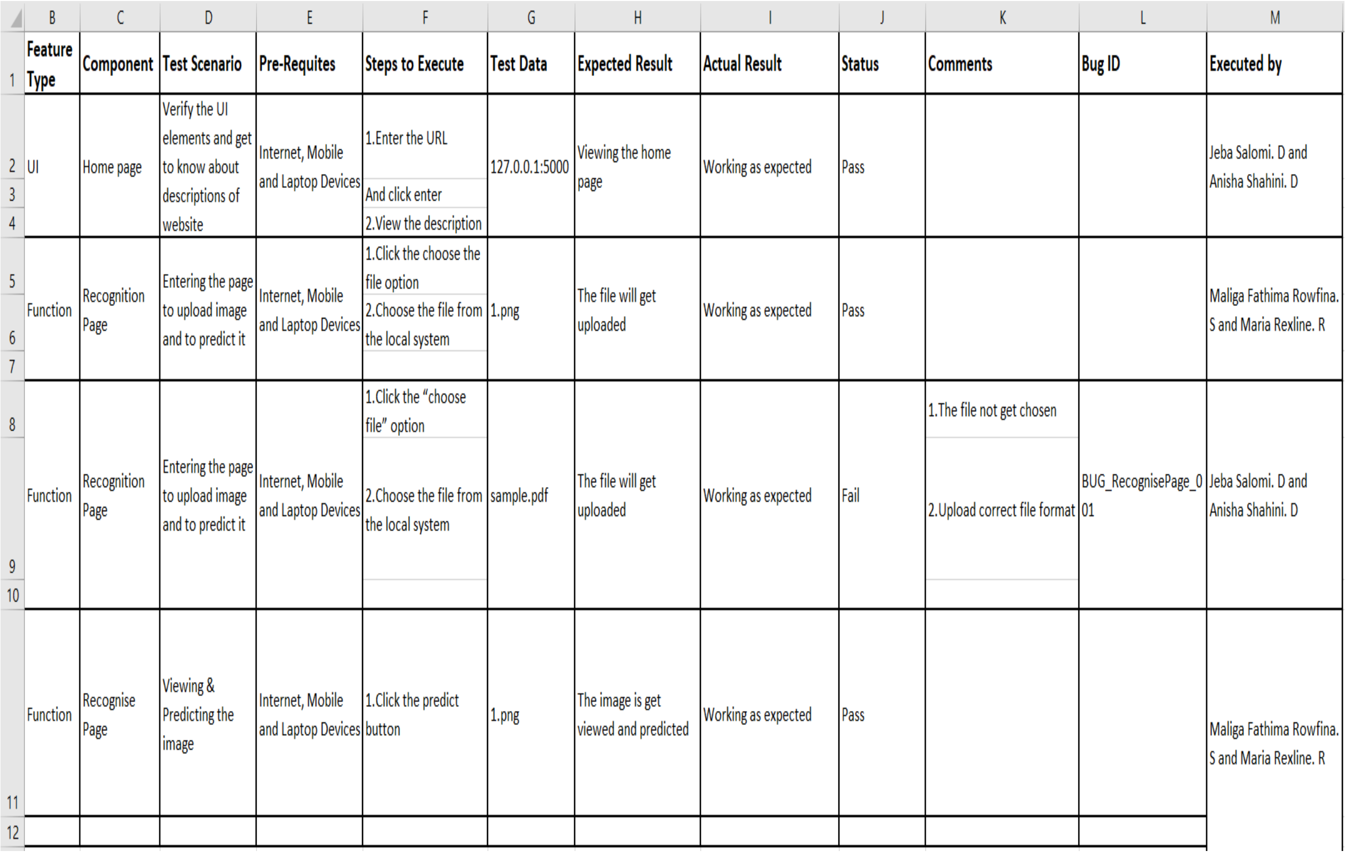
```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="{{ url_for('static', filename='recognize.css') }}">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js">
    document.getElementById("result").style.color="solid black";
    </script>
    <h1 style="color:rgb(25, 25, 112)"><center>DIGIT RECOGNITION</center></h1>
  </head>
  <body>
    <h2><center>Upload image here to recognize the digit</center></h2>
    <div class="bg">
    <p><center>
      {% with messages = get_flashed_messages() %}
        {% if messages %}

           {% for message in messages %}
            <font size ="+1"><b><center>{{ message }}</center></b></font>
           {% endfor %}



         {% endif %}
      {% endwith %}
     </center>
    </p>
    </div>
    <form method="post" action="/" enctype="multipart/form-data" >
      <dl>
        <p><center>

          <input class="upload-btn" type ="file" name="file" autocomplete="off" hidden="hidden"
required>

      </center>
      </p>
      </dl>
      {% if filename  %}
        <div style="padding:20px;"><center>
        <img src="{{ url_for('display_image', filename=filename) }}">
        </center>
        </div>
      {% endif %}
      <p><center>
        <input class="btn" type="submit" value="PREDICT">
      </center>
      </p>
    </form>


    <p><center><font size ="+2"><b>PREDICTED DIGIT IS: {{prediction}}
    </b></font></center></p>

  </body>

</html>
```

app.py:

```
from flask import Flask, render_template, request, url_for, redirect, flash
from werkzeug.utils import secure_filename
import os
import urllib.request
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from PIL import Image
import numpy as np

model=load_model(r'models/mnistCNN.h5')

app = Flask(__name__)

UPLOAD_FOLDER = 'static/uploads/'
app.secret_key = "secret key"
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024

ALLOWED_EXTENSIONS = set(['png', 'jpg', 'jpeg'])

def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

@app.route('/')
def index():

    return render_template("index.html")

@app.route('/recognise', methods=['GET','POST'])
def recognise():
    if request.method == 'POST':
        return redirect(url_for('index'))
    return render_template('recognise.html')

@app.route('/', methods=['POST'])
def upload_image():
    if 'file' not in request.files:
        flash('No file part')
        return redirect(request.url)
    file = request.files['file']
    if file.filename == '':
        flash('No image selected for uploading')
        return redirect(request.url)
    if file and allowed_file(file.filename):
        filename = secure_filename(file.filename)
        file.save(os.path.join(app.config['UPLOAD_FOLDER'],filename))
        flash('Image uploaded successfully')
        img=Image.open(file.stream).convert("L")
        img=img.resize((28,28))
        im2arr = np.array(img)
        im2arr = im2arr.reshape(1,28,28,1)
        y_pred = model.predict(im2arr)
        print(np.argmax(y_pred))
        prediction=str(np.argmax(y_pred))

        return render_template('recognise.html',filename=filename,prediction=prediction)
    else:
        flash('Allowed image types are - png, jpg, jpeg')
        return redirect(request.url)

@app.route('/display/<filename>')
def display_image(filename):

    return redirect(url_for('static',filename='uploads/' +filename), code=301)


if __name__ == "__main__":
    app.run(debug = False)
```

8. TESTING

8.1 Test Cases



	B	C	D	E	F	G	H	I	J	K	L	M
1	Feature Type	Component	Test Scenario	Pre-Requites	Steps to Execute	Test Data	Expected Result	Actual Result	Status	Comments	Bug ID	Executed by
2 3 4	UI	Home page	Verify the UI elements and get to know about descriptions of website	Internet, Mobile and Laptop Devices	1.Enter the URL And click enter 2.View the description	127.0.0.1:5000	Viewing the home page	Working as expected	Pass			Jeba Salomi. D and Anisha Shahini. D
5 6 7	Function	Recognition Page	Entering the page to upload image and to predict it	Internet, Mobile and Laptop Devices	1.Click the choose the file option 2.Choose the file from the local system	1.png	The file will get uploaded	Working as expected	Pass			Maliga Fathima Rowfina. S and Maria Rexline. R
8 9 10	Function	Recognition Page	Entering the page to upload image and to predict it	Internet, Mobile and Laptop Devices	1.Click the “choose file” option 2.Choose the file from the local system	sample.pdf	The file will get uploaded	Working as expected	Fail	1.The file not get chosen 2.Upload correct file format	BUG_RecognisePage_001	Jeba Salomi. D and Anisha Shahini. D
11	Function	Recognise Page	Viewing & Predicting the image	Internet, Mobile and Laptop Devices	1.Click the predict button	1.png	The image is get viewed and predicted	Working as expected	Pass			Maliga Fathima Rowfina. S and Maria Rexline. R
12												

8.2 User Acceptance Testing

User Acceptance Testing (UAT) explains the test coverage and open issues of the handwriting digit recognition project at the time of the release to UAT

8.2.1 Defect Analysis

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won't Fix	1	0	1	0	2
Totals	6	3	1	4	14

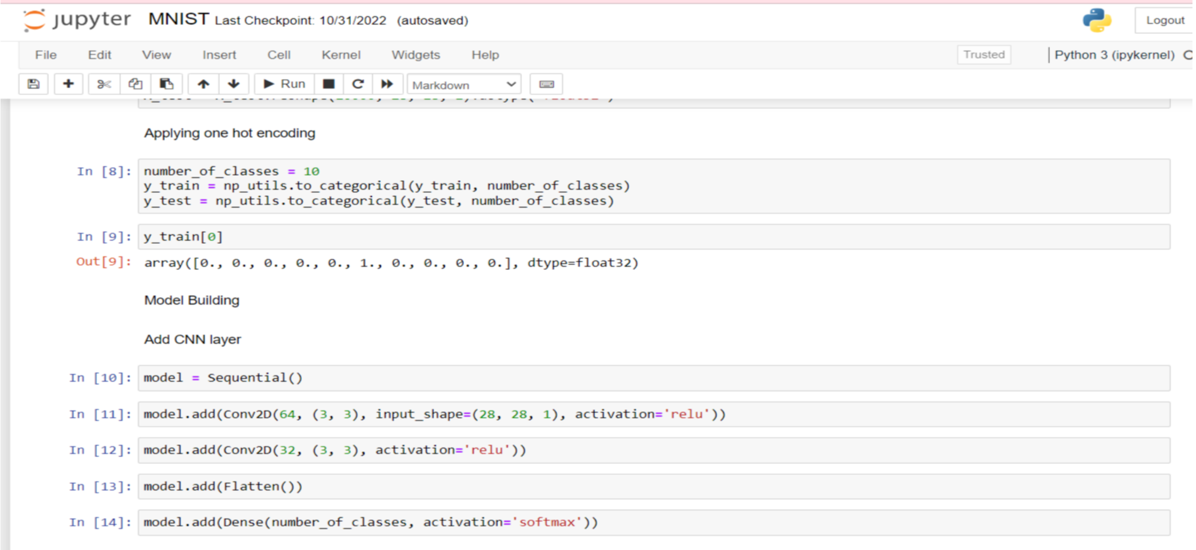
8.2.2 Test Case Analysis

Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	3	7
Security	2	0	1	1
Performance	3	0	1	2
Exception Reporting	2	0	0	2

9. RESULTS

9.1 Performance Metrics

We have used MNIST dataset for building Convolutional Neural Network model. The layers used in the model are Convolutional layer, flatten layer and dense layer. The model shows the test accuracy of 97% and loss accuracy 8%. The application shows accurate result for the images in the MNIST dataset.



jupyter MNIST Last Checkpoint: 10/31/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Markdown

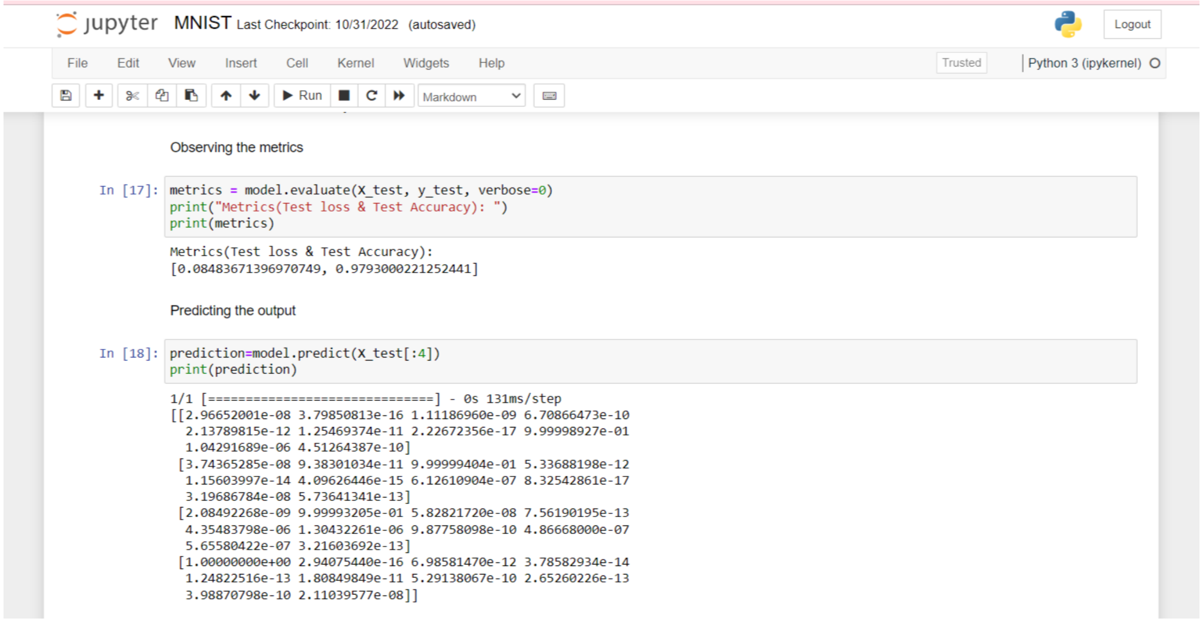
Applying one hot encoding

```
In [8]: number_of_classes = 10
        y_train = np_utils.to_categorical(y_train, number_of_classes)
        y_test = np_utils.to_categorical(y_test, number_of_classes)

In [9]: y_train[0]

Out[9]: array([0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)
```

Model Building

Add CNN layer

```
In [10]: model = Sequential()

In [11]: model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))

In [12]: model.add(Conv2D(32, (3, 3), activation='relu'))

In [13]: model.add(Flatten())

In [14]: model.add(Dense(number_of_classes, activation='softmax'))
```



jupyter MNIST Last Checkpoint: 10/31/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Markdown

Observing the metrics

```
In [17]: metrics = model.evaluate(X_test, y_test, verbose=0)
         print("Metrics(Test loss & Test Accuracy): ")
         print(metrics)

         Metrics(Test loss & Test Accuracy):
         [0.08483671396970749, 0.9793000221252441]
```

Predicting the output

```
In [18]: prediction=model.predict(X_test[:4])
         print(prediction)

1/1 [==============================] - 0s 131ms/step
[[2.96652001e-08 3.79850813e-16 1.11186960e-09 6.70866473e-10
  2.13789815e-12 1.25469374e-11 2.22672356e-17 9.99998927e-01
  1.04291689e-06 4.51264387e-10]
 [3.74365285e-08 9.38301034e-11 9.99999404e-01 5.33688198e-12
  1.15603997e-14 4.09626446e-15 6.12610904e-07 8.32542861e-17
  3.19686784e-08 5.73641341e-13]
 [2.08492268e-09 9.99993205e-01 5.82821720e-08 7.56190195e-13
  4.35483798e-06 1.30432261e-06 9.87758098e-10 4.86668000e-07
  5.65580422e-07 3.21603692e-13]
 [1.00000000e+00 2.94075440e-16 6.98581470e-12 3.78582934e-14
  1.24822516e-13 1.80849849e-11 5.29138067e-10 2.65260226e-13
  3.98870798e-10 2.11039577e-08]]
```

10. ADVANTAGES AND DISADVANTAGES

Advantages:

- This web application allows the user to upload the image
- This web application is developed only for recognizing the digits
- This web application predicts the image in the MNIST dataset accurately
- This application shows error message when other file formats are chosen
- This application simple User Interface
- This application is easy to use
- The web pages in this application are easy to navigate

Disadvantages:

- This application does not recognize some digits accurately
- This application does not allow the user to scan the image
- This application only allows png, jpg and jpeg image formats

11. CONCLUSION

The proposed web application is an Internet based system. In this system, the user can upload the image from the local system and recognize the digit in the uploaded image.

The concepts of Neural Networks, machine learning and data mining are being implemented in most problems faced by technologists and programmers around the world. The idea is to train a computer to think and make decisions like a human being. The concepts used in this application helps us to understand the essential requirements to build a Convolutional Neural network. The Convolution Neural Network model is build on MNIST dataset.

The purpose of this web application is to recognise the handwritten digit.

12. FUTURE SCOPE

As part of our future enhancements, we aim to tune our model to find the most accurate solution for classification. Furthermore, it would be worthwhile to run this system for multi -digit recognition and character recognition. Likely, this would aid in complete handling of occlusion and would lead to improved detection and classification results. Data storage should be as efficient as possible, in spite of having a many training samples.

13. APPENDIX

Source code:

index.html:

```
<!DOCTYPE html>
<head>
   <link rel="stylesheet" href="{{ url_for('static', filename='index.css') }}">

<meta charset="UTF-8">
   <title>index page</title>

</head>
<body>
   <h1 class="heading">HANDWRITTEN DIGIT RECOGNITION SYSTEM</h1>
   <div class="icon">
      <p>Handwritten Text Recognition is a technology that is much needed  in this as of
today. This digit Recognition system
         is used to recognize the digits from different source like emails,bank
cheque,papers,images.ex. Before proper
         implementation of this technology we have relied on writing texts with our own
hands which can result in errors. It's
         difficult to store and access physical data with efficiency.The project presents
recognizing the handwriting digits (0 to 9)
         from the famous MNIST dataset. Here we will be using artificial neural
networks/convolutional network.
</p>
```

```
    <div class="btn">
    <a  href="{{ url_for('recognise') }}" target="_self">continue</a>
    </div>
  </div>
</body>
</html>
```

recognise.html:

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="{{ url_for('static', filename='recognize.css') }}">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js">
    document.getElementById("result").style.color="solid black";
    </script>
    <h1 style="color:rgb(25, 25, 112)"><center>DIGIT RECOGNITION</center></h1>
  </head>
  <body>
    <h2><center>Upload image here to recognize the digit</center></h2>
    <div class="bg">
    <p><center>
      {% with messages = get_flashed_messages() %}
        {% if messages %}

          {% for message in messages %}
           <font size ="+1"><b><center>{{ message }}</center></b></font>
          {% endfor %}
```

```
                {% endif %}
            {% endwith %}
         </center>
        </p>
        </div>

        <form method="post" action="/" enctype="multipart/form-data" >
            <dl>
                <p><center>

                    <input class="upload-btn" type ="file" name="file" autocomplete="off"
hidden="hidden" required>

            </center>
            </p>
            </dl>
            {% if filename  %}
                <div style="padding:20px;"><center>
                <img src="{{ url_for('display_image', filename=filename) }}">
                </center>
                </div>
            {% endif %}
            <p><center>
                <input class="btn" type="submit" value="PREDICT">
            </center>
            </p>
        </form>
        <p><center><font size ="+2"><b>PREDICTED DIGIT IS: {{prediction}}
        </b></font></center></p>

    </body>

</html>
```

index.css:

```
*{
    margin: 0;
    padding: 0;
}
body{
    background-image: url('bg.jpg');
    background-size: cover;
    background-repeat: no-repeat;
    overflow: hidden;
}
.heading{
    color: #eadf7e;
    padding-right: 50px;
    right: 0;
    margin-top: 150px;
    margin-left: 100px;
    text-align: left;
}
```

```
.icon{
    width: 100%;




}

p{
    color: #fff;
    font-size: 1.5em;
    text-align: center;
    font-family:cursive;
    font-weight: 100;
    right: -20px;
    width: 595px;
    height: 270px;
    padding-left: 156px;
    padding-top: 174px;
    margin-top: -141px;

}

.btn{
    width: 130px;
    height: 40px;
    background: #eadf7e;
    border: none;
    margin-bottom: 74px;
    margin-left: 350px;
    margin-top: 200px;
    font-size: 32px;
    border-radius: 60px;
    cursor: pointer;
    transition: .4s ease;
    text-align: center;
}
.btn a{
    text-decoration: none;
    color: #000;
    transition: .3s ease;
}
.btn:hover{
    color: rgb(255, 250, 250);
}
```

recognise.css:

```
body {
    background-color: rgb(209, 209, 227);

}
div{

    text-align: right;
}
h1 {
    color: azure;
    text-align: left;
```

```
}

p {
    color:rgb(188, 195, 216);
    font-style: italic;
    text-align: right;
    text-indent: 30px;
    font-size: large;
 }
.pre {
 color: black;
}
.btn {
  border: 2px solid gray;
  color: white;
  background-color:rgb(25, 25, 112);
  padding: 8px 20px;
  border-radius: 8px;
  font-size: 20px;
  font-weight: bold;
}
.upload-btn{
    border: 3px solid rgb(25, 25, 112);
    cursor: pointer;
    color: rgb(25, 25, 112);
    display: inline-block;
    font-size: 20px;
    font-weight: bold;
    border-radius: 8px;
    height: 39px;
    line-height: 36px;

}
.upload-btn::-webkit-file-upload-button{
    background: rgb(25, 25, 112);
    color: white;
    padding: 8px 16px;
    border: none;
    cursor: pointer;
}

.msg {
    color: solid red;
    font-size: large;

}
/*.upload-btn input[type=file] {
 display: none;
}*/

/*#custom-button {
    padding: 10px;
    color: white;
    background-color: blue;
    border: 1px solid #000;
    border-radius: 5px;
    cursor: pointer;
}
```

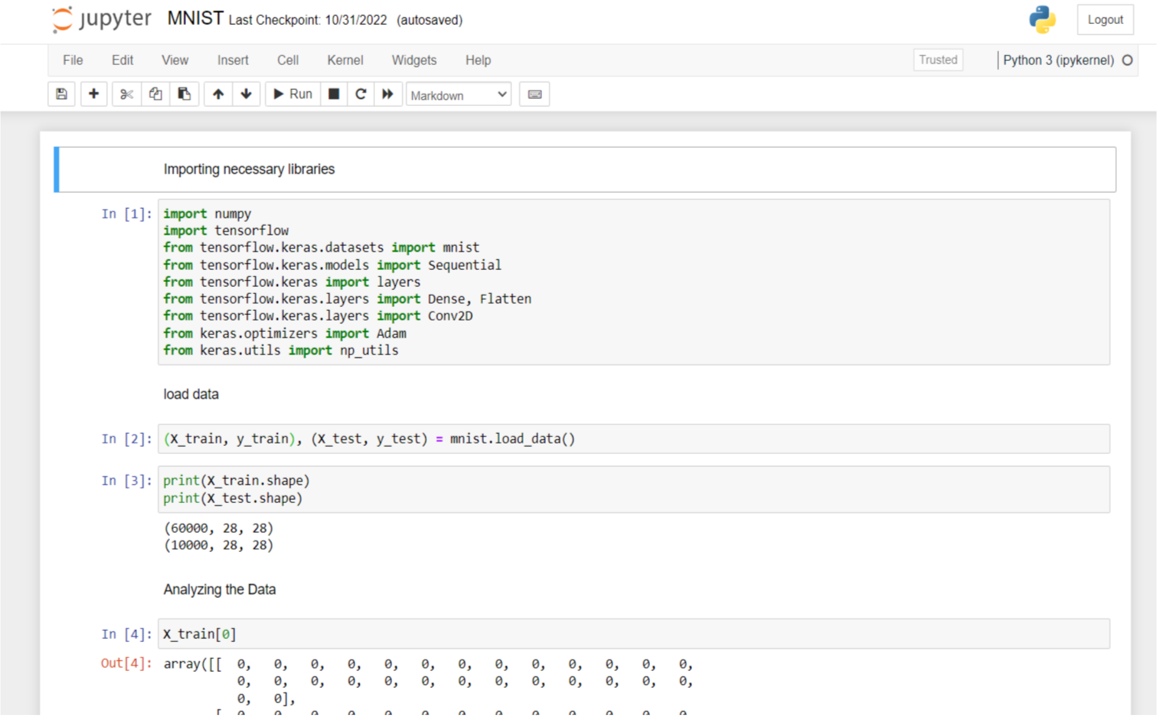
```
#custom-button:hover{
   background-color: #00b28f;
}
#custom-text{
   margin-left: 10px;
   font-family: sans-serif;
   color: #aaa;
}*/
#display_image{
   width: 375px;
   height: 211px;
   border: 1px solid black;
   background-position: center;
   background-size: cover;
   }
```

app.py:

```
from flask import Flask, render_template, request, url_for, redirect, flash
from werkzeug.utils import secure_filename
import os
import urllib.request
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from PIL import Image
import numpy as np

model=load_model(r'models/mnistCNN.h5')
```

```
app = Flask(__name__)

UPLOAD_FOLDER = 'static/uploads/'
app.secret_key = "secret key"
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024

ALLOWED_EXTENSIONS = set(['png', 'jpg', 'jpeg'])

def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

@app.route('/')
def index():

    return render_template("index.html")

@app.route('/recognise', methods=['GET','POST'])
def recognise():
    if request.method == 'POST':
        return redirect(url_for('index'))
    return render_template('recognise.html')

@app.route('/', methods=['POST'])
def upload_image():
    if 'file' not in request.files:
        flash('No file part')
        return redirect(request.url)
    file = request.files['file']
    if file.filename == '':
        flash('No image selected for uploading')
        return redirect(request.url)
    if file and allowed_file(file.filename):
        filename = secure_filename(file.filename)
        file.save(os.path.join(app.config['UPLOAD_FOLDER'],filename))
        flash('Image uploaded successfully')
        img=Image.open(file.stream).convert("L")
        img=img.resize((28,28))
        im2arr = np.array(img)
        im2arr = im2arr.reshape(1,28,28,1)
        y_pred = model.predict(im2arr)
        print(np.argmax(y_pred))
        prediction=str(np.argmax(y_pred))

        return render_template('recognise.html',filename=filename,prediction=prediction)
    else:
        flash('Allowed image types are - png, jpg, jpeg')
        return redirect(request.url)

@app.route('/display/<filename>')
def display_image(filename):

    return redirect(url_for('static',filename='uploads/' +filename), code=301)



if __name__ == "__main__":
    app.run(debug = False)
```

MNIST.ipynb:



Importing necessary libraries

```
In [1]: import numpy
        import tensorflow
        from tensorflow.keras.datasets import mnist
        from tensorflow.keras.models import Sequential
        from tensorflow.keras import layers
        from tensorflow.keras.layers import Dense, Flatten
        from tensorflow.keras.layers import Conv2D
        from keras.optimizers import Adam
        from keras.utils import np_utils
```

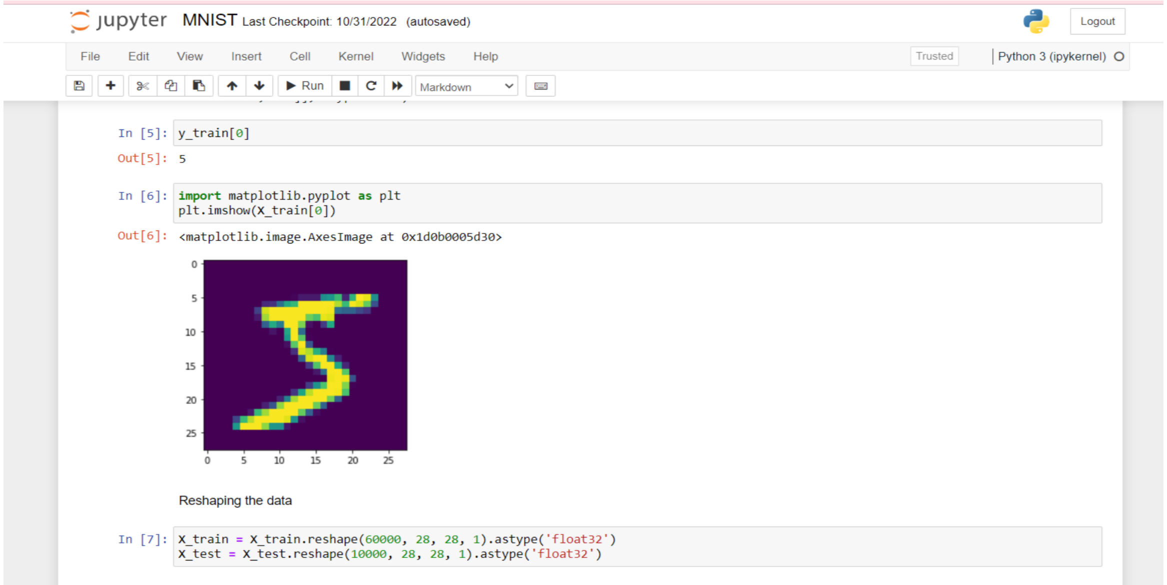
load data

```
In [2]: (X_train, y_train), (X_test, y_test) = mnist.load_data()
```

```
In [3]: print(X_train.shape)
        print(X_test.shape)

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

Analyzing the Data

```
In [4]: X_train[0]

Out[4]: array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
                  0,   0],
```

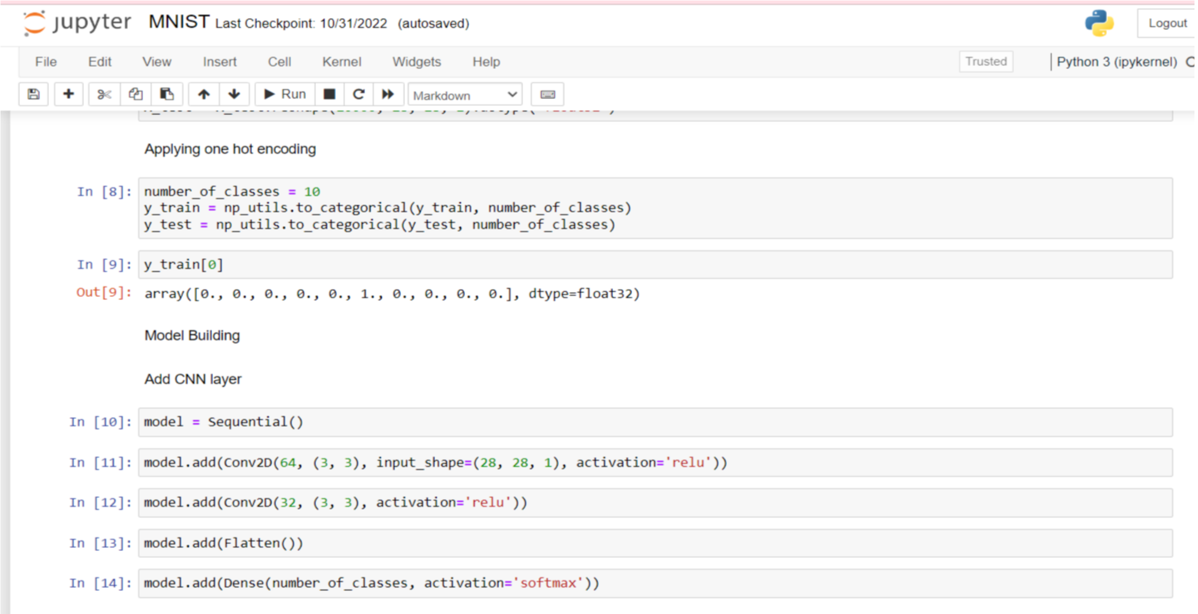


```
In [5]: y_train[0]

Out[5]: 5
```

```
In [6]: import matplotlib.pyplot as plt
        plt.imshow(X_train[0])

Out[6]: <matplotlib.image.AxesImage at 0x1d0b0005d30>
```

Reshaping the data

```
In [7]: X_train = X_train.reshape(60000, 28, 28, 1).astype('float32')
        X_test = X_test.reshape(10000, 28, 28, 1).astype('float32')
```


jupyter MNIST Last Checkpoint: 10/31/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Markdown

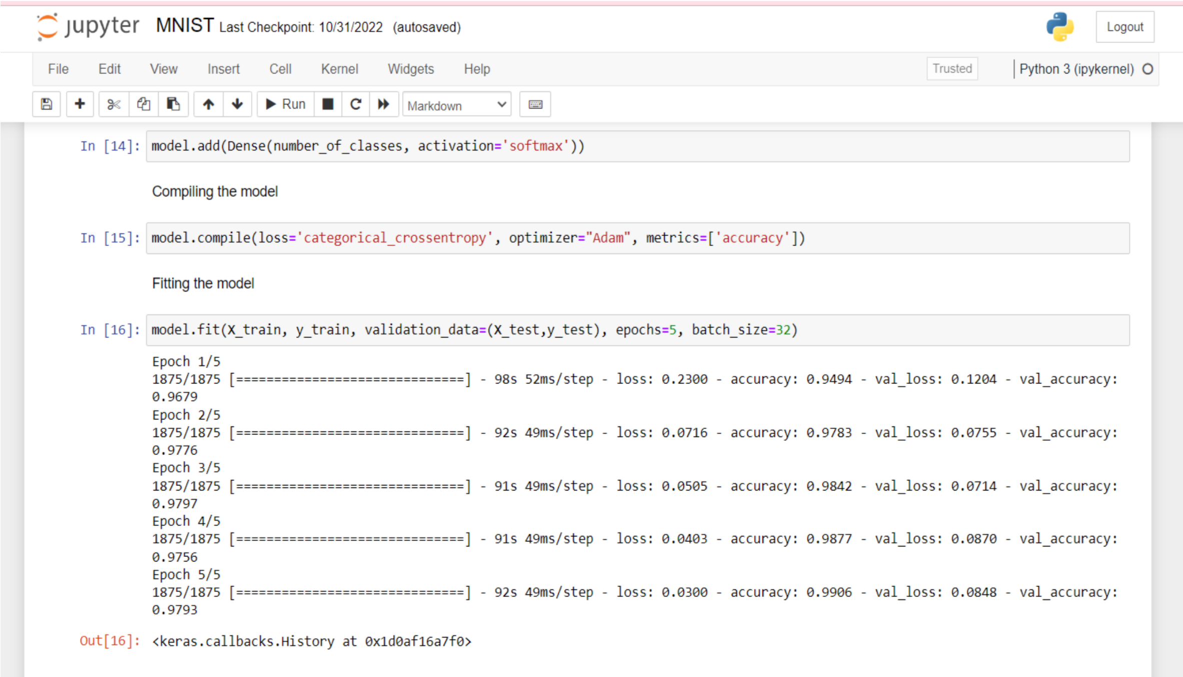
Applying one hot encoding

```
In [8]: number_of_classes = 10
        y_train = np_utils.to_categorical(y_train, number_of_classes)
        y_test = np_utils.to_categorical(y_test, number_of_classes)

In [9]: y_train[0]

Out[9]: array([0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)
```

Model Building

Add CNN layer

```
In [10]: model = Sequential()

In [11]: model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))

In [12]: model.add(Conv2D(32, (3, 3), activation='relu'))

In [13]: model.add(Flatten())

In [14]: model.add(Dense(number_of_classes, activation='softmax'))
```


jupyter MNIST Last Checkpoint: 10/31/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Markdown

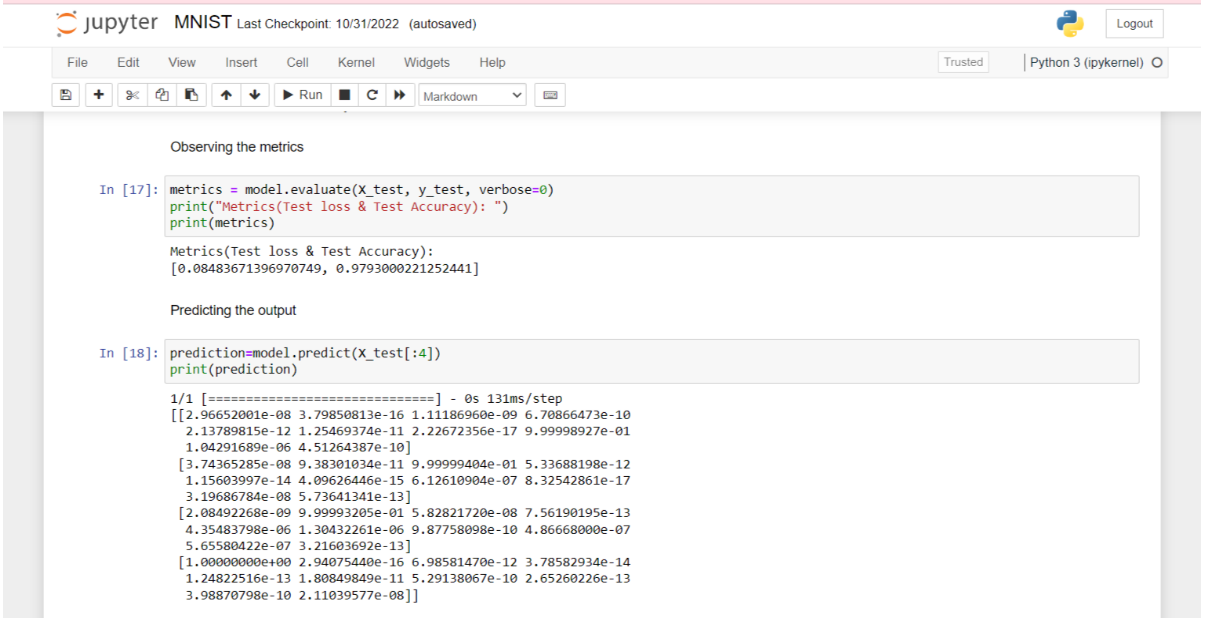
```
In [14]: model.add(Dense(number_of_classes, activation='softmax'))
```

Compiling the model

```
In [15]: model.compile(loss='categorical_crossentropy', optimizer="Adam", metrics=['accuracy'])
```

Fitting the model

```
In [16]: model.fit(X_train, y_train, validation_data=(X_test,y_test), epochs=5, batch_size=32)

         Epoch 1/5
         1875/1875 [==============================] - 98s 52ms/step - loss: 0.2300 - accuracy: 0.9494 - val_loss: 0.1204 - val_accuracy:
         0.9679
         Epoch 2/5
         1875/1875 [==============================] - 92s 49ms/step - loss: 0.0716 - accuracy: 0.9783 - val_loss: 0.0755 - val_accuracy:
         0.9776
         Epoch 3/5
         1875/1875 [==============================] - 91s 49ms/step - loss: 0.0505 - accuracy: 0.9842 - val_loss: 0.0714 - val_accuracy:
         0.9797
         Epoch 4/5
         1875/1875 [==============================] - 91s 49ms/step - loss: 0.0403 - accuracy: 0.9877 - val_loss: 0.0870 - val_accuracy:
         0.9756
         Epoch 5/5
         1875/1875 [==============================] - 92s 49ms/step - loss: 0.0300 - accuracy: 0.9906 - val_loss: 0.0848 - val_accuracy:
         0.9793

Out[16]: <keras.callbacks.History at 0x1d0af16a7f0>
```



Observing the metrics

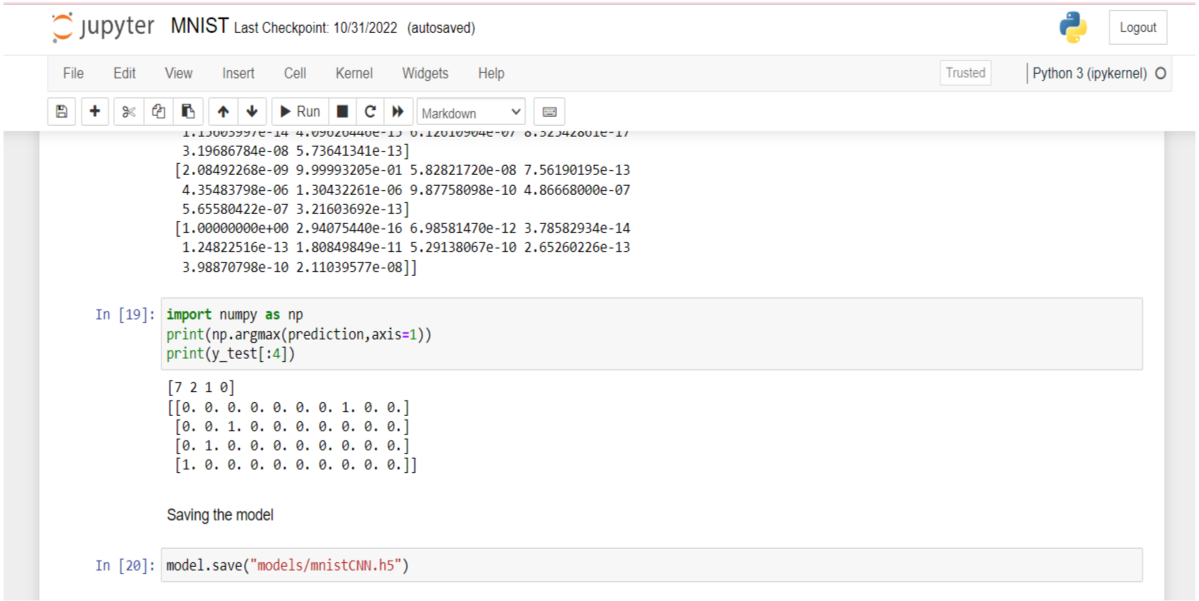
```
In [17]: metrics = model.evaluate(X_test, y_test, verbose=0)
         print("Metrics(Test loss & Test Accuracy): ")
         print(metrics)
```

```
Metrics(Test loss & Test Accuracy): 
[0.08483671396970749, 0.9793000221252441]
```

Predicting the output

```
In [18]: prediction=model.predict(X_test[:4])
         print(prediction)
```

```
1/1 [==============================] - 0s 131ms/step
[[2.96652001e-08 3.79850813e-16 1.11186960e-09 6.70866473e-10
  2.13789815e-12 1.25469374e-11 2.22672356e-17 9.99998927e-01
  1.04291689e-06 4.51264387e-10]
 [3.74365285e-08 9.38301034e-11 9.99999404e-01 5.33688198e-12
  1.15603997e-14 4.09626446e-15 6.12610904e-07 8.32542861e-17
  3.19686784e-08 5.73641341e-13]
 [2.08492268e-09 9.99993205e-01 5.82821720e-08 7.56190195e-13
  4.35483798e-06 1.30432261e-06 9.87758098e-10 4.86668000e-07
  5.65580422e-07 3.21603692e-13]
 [1.00000000e+00 2.94075440e-16 6.98581470e-12 3.78582934e-14
  1.24822516e-13 1.80849849e-11 5.29138067e-10 2.65260226e-13
  3.98870798e-10 2.11039577e-08]]
```



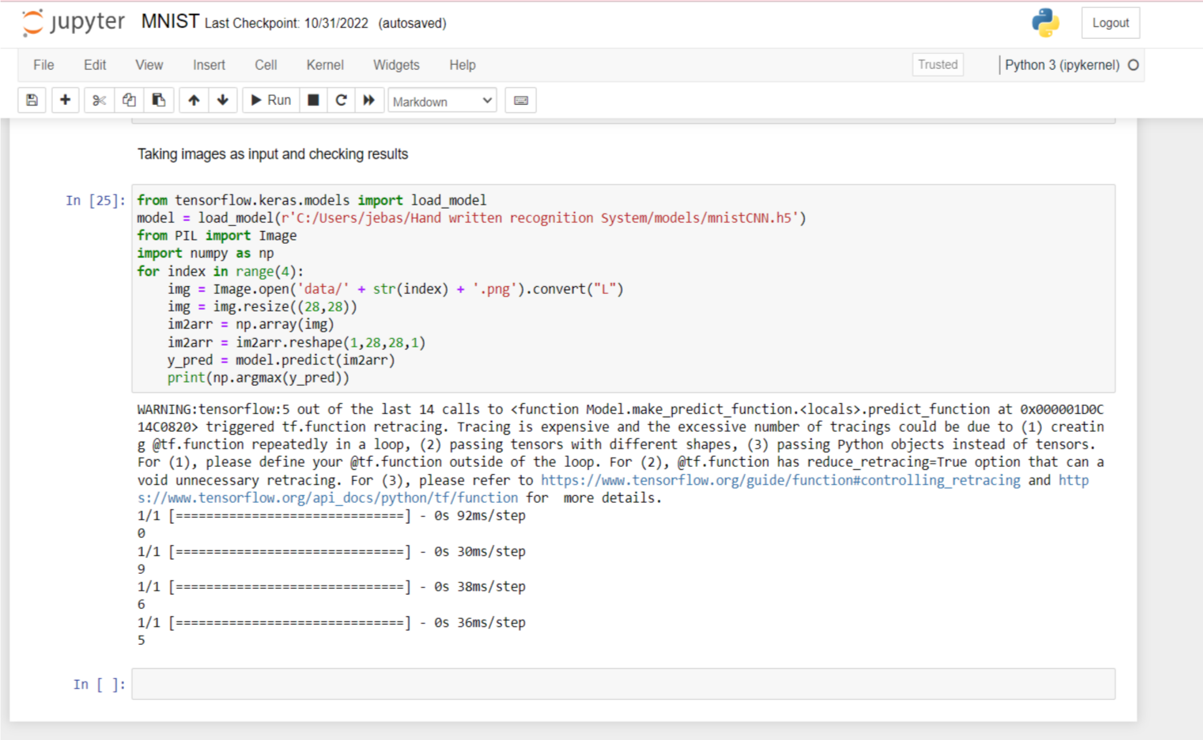
```
  1.15603997e-14 4.09626446e-15 6.12610904e-07 8.32542861e-17
  3.19686784e-08 5.73641341e-13]
 [2.08492268e-09 9.99993205e-01 5.82821720e-08 7.56190195e-13
  4.35483798e-06 1.30432261e-06 9.87758098e-10 4.86668000e-07
  5.65580422e-07 3.21603692e-13]
 [1.00000000e+00 2.94075440e-16 6.98581470e-12 3.78582934e-14
  1.24822516e-13 1.80849849e-11 5.29138067e-10 2.65260226e-13
  3.98870798e-10 2.11039577e-08]]
```

```
In [19]: import numpy as np
         print(np.argmax(prediction,axis=1))
         print(y_test[:4])
```

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

Saving the model

```
In [20]: model.save("models/mnistCNN.h5")
```



Taking images as input and checking results

In [25]:

```
from tensorflow.keras.models import load_model
model = load_model(r'C:/Users/jebas/Hand written recognition System/models/mnistCNN.h5')
from PIL import Image
import numpy as np
for index in range(4):
    img = Image.open('data/' + str(index) + '.png').convert("L")
    img = img.resize((28,28))
    im2arr = np.array(img)
    im2arr = im2arr.reshape(1,28,28,1)
    y_pred = model.predict(im2arr)
    print(np.argmax(y_pred))
```

```
WARNING:tensorflow:5 out of the last 14 calls to <function Model.make_predict_function.<locals>.predict_function at 0x000001D0C
14C0820> triggered tf.function retracing. Tracing is expensive and the excessive number of tracings could be due to (1) creatin
g @tf.function repeatedly in a loop, (2) passing tensors with different shapes, (3) passing Python objects instead of tensors.
For (1), please define your @tf.function outside of the loop. For (2), @tf.function has reduce_retracing=True option that can a
void unnecessary retracing. For (3), please refer to https://www.tensorflow.org/guide/function#controlling_retracing and http
s://www.tensorflow.org/api_docs/python/tf/function for  more details.
1/1 [==============================] - 0s 92ms/step
0
1/1 [==============================] - 0s 30ms/step
9
1/1 [==============================] - 0s 38ms/step
6
1/1 [==============================] - 0s 36ms/step
5
```

In [ ]:

Github link:

https://github.com/IBM-EPBL/IBM-Project-23458-1659883628.git

Demo link:

https://drive.google.com/file/d/1OtjZhQiieDGB9mf1ASchXYOcEr0L-Viu/view?usp=share_link