TABLE OF CONTENTS

A Gesture-Based Tool For Sterile Browsing Of Radiology Images

1.INTRODUCTION

1.1 Project overview

Gestures as a basic form of non-verbal communication made with the hands. Humans are able to recognize body and sign language easily. This is possible due to the combination of vision and synaptic interactions that were formed along brain development . Inorder to replicate this skill in computers, some problems need to be solved: how to separate objects of interest in images and which image capture technology and classification technique are more appropriate, among others. In this project Gesture based Desktop automation, User interacts with the UI (User Interface) to upload the image as input. Depending on the different gesture inputs different operations are applied to the input image. Once model analyses the gesture, the prediction with operation applied on image is showcased on the UI. First the model is trained pre trained on the images of different hand gestures, such as a showing numbers with fingers as 1,2,3,4 by data collection and then image is processed by ImageDataGenerator library. The proposed project is done using open cv and IBM cloud. By CNN model is initialized and input,hidden,output layers are created and then is proposed for testing and training. This model uses the integrated webcam to capture the video frame. The image of the gesture captured in the video frame is compared with the Pre-trained model and the gesture is identified. If the gesture predicted is 1 then images is blurred;2, image is resized;3, image is rotated etc. Application layout is done by HTML for better outlook.

1.2 <u>Purpose</u>

Computer information technology is increasingly penetrating into the hospital domain. A major challenge involved in this process is to provide doctors with efficient, intuitive, accurate and safe means of interaction without affecting the quality of their work. Keyboards and pointing devices, such as a mouse, are today's principal method of human—computer interaction. However, the use of computer keyboards and mice by doctors and nurses in intensive care units (ICUs) is a common method for spreading infections.we suggest the use of hand gestures as an alternative to existing interface techniques, offering the major advantage of sterility In this work we refer to gestures as a basic form of non-verbal communication made with the hands. Psychological studies showed that young children use gestures to communicate before they learn to talk. Manipulation, as a form of gesticulation, is often used when people speak to each other about some object. Naturalness of expression, non-encumbered interaction, intuitiveness and high sterility are all good reasons to replace the current interface technology (e.g., keyboard, mouse, and joystick) with more natural interfaces.

2.<u>LITERATURE SURVEY</u>

2.1 <u>Existing problem</u>

The use of doctor-computer interaction devices in the operation room (OR) requires new modalities that support medical imaging manipulation while allowing doctors' hands to remain sterile, supporting their focus of attention, and providing fast response times

Keyboards and pointing devices, suchas a mouse, are today's principal method of human—computer interaction. However, the use of computer key-boards and mice by

doctors and nurses in intensive care units (ICUs) is a common method for spreading infections. In this paper, we suggest the use of hand gestures as an alternative to existing interface techniques, offering themajor advantage of sterility. Even though voice control alsoprovides sterility, the noise level in the operating room (OR)deems it problematic.

2.2 <u>References</u>

1.**Hand gesture recognition with depth images**
(*J. Suarez and R. R. Murphy 2012)*
The papers that use the Kinect and the OpenNI libraries for hand tracking tend to focus more on applications than on localization and classification methods, and show that the OpenNI hand tracking method is good enough for the applications tested. Kinect and other depth sensors for gesture recognition have yet to be tested in challenging applications and environments.

2. **Gesture-Based Affective Computing on Motion Capture Data**

(Kapur, A.,Kapur, A.,Virji-Babul,N.,Tzanetakis,G., Driessen,P.F. (2005).Gesture Based Affective Computing on Motion Capture Data.In: Tao, J.,Tan, T.,Picard, R.W.(eds) Affective Computing and Intelligent Interaction.ACII 2005.Lecture Notes in Computer Science, vol3784.)

Body skeletal movements captured using video-based sensor technology developed by Vicon Motion Systems, to train machine to identify different human emotions. automatic classification results into perspective a user study onthe human perception of the same data was conducted with average classification accuracy of 93%.Accuracy is not 100%

3. A gesture based interaction technique for a planning tool for construction and design

M.Rauterberg, M. Bichsel, M. Meier and M. Fjeld, "A gesture based interaction technique for a planning tool for construction and design," Proceedings 6th IEEEInternational Workshop on Robot and Human Communicatio n. RO-MAN'97 SENDAI, 1997, pp. 212-217, doi: 10.1109/ROM AN.1997.6469 84

The AR design strategy enables humans to behave in a nearly natural way. Natural interaction means human actions in the real world with other humans and/or with real world objects. Guided by the basic constraints of natural interaction, we derive a set of recommendations for the next generation of user interfaces: the natural user interface (NUI). Our approach to NUIs is discussed in the form of a general framework followed by a prototype.

4. Gesture-based interaction and communication: automated classification of hand gesture contours

L. Gupta and Suwei Ma, "Gesture- based interaction and communicatio n: automated classification of hand gesture contours," in IEEE Transactions on Systems, Man, and Cybernetics

The accurate classification of hand gestures is crucial in the development of novel hand gesture-based systems designed for human-computer interaction (HCI) and for human alternative and augmentative communication (HAAC). A complete vision-based system, consisting of hand gesture acquisition, segmentaion, filtering, representation and classification, is developed to robustly classify hand gestures.
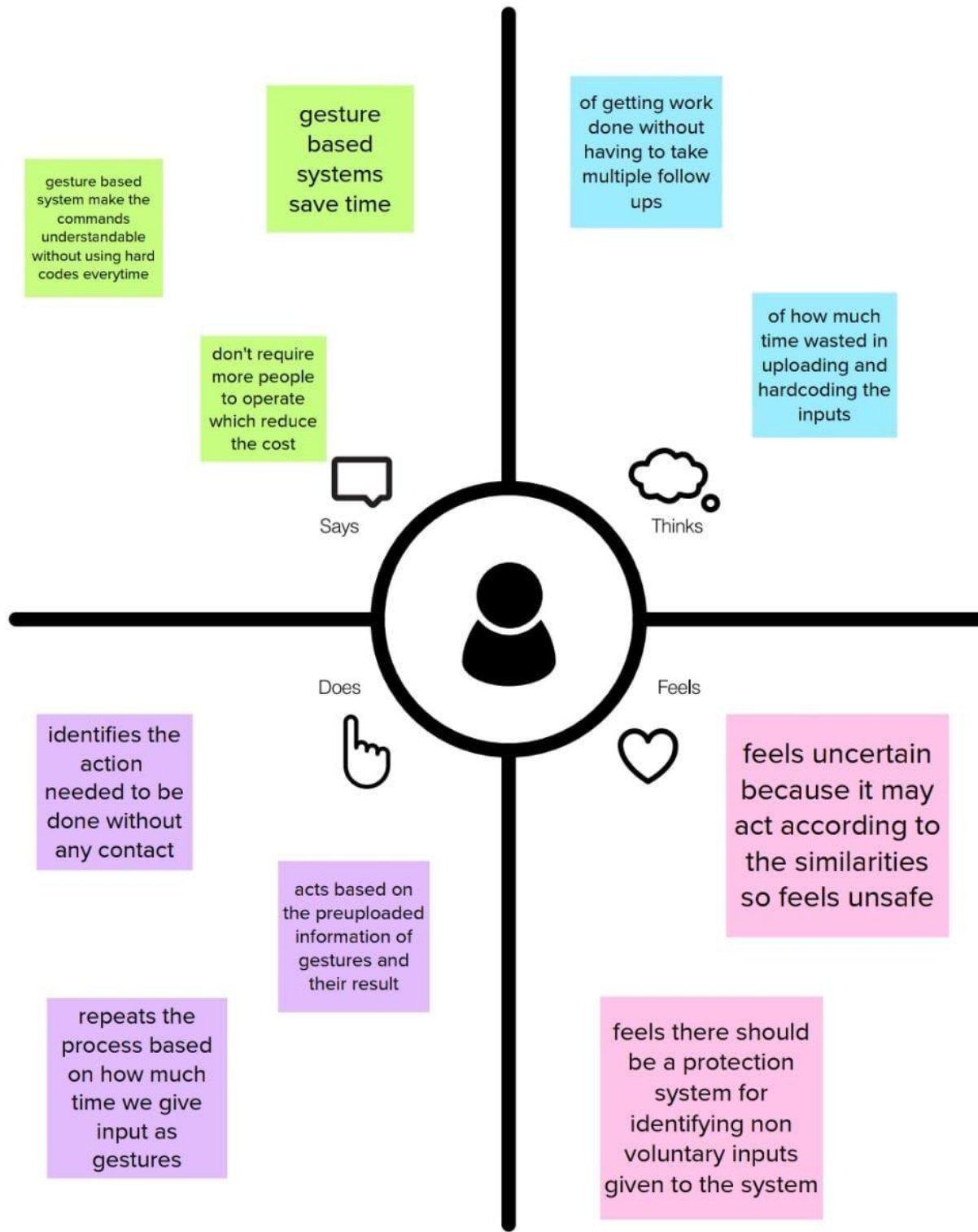
2.3 Problem Statement Definition

The use of doctor-computer interaction devices in the operation room (OR) requires new modalities that support medical imaging manipulation while allowing doctors' hands to remain sterile, supporting their focus of attention, and providing fast response times

Keyboards and pointing devices, such as a mouse, are today's principal method of human—computer interaction. However, the use of computer key-boards and mice by doctors and nurses in intensive care units (ICUs) is a common method for spreading infections.we suggest the use of hand gestures as an alternative to existing interface techniques, offering the major advantage of sterility provides sterility, the noise level in the operating room (OR) deems it problematic.In this project Gesture based Desktop automation, User interacts with the UI (User Interface) to upload the image as input. Depending on the different gesture inputs different operations are applied to the input image. Once model analyses the gesture, the prediction with operation applied on image is showcased on the UI. First the model is trained pre trained on the images of different hand gestures, such as a showing numbers with fingers as 1,2,3,4 by data collection and then image is processed by ImageDataGenerator library . The proposed project is done using open cv and IBM cloud.By CNN model is initialized and input,hidden,output layers are created and then is proposed for testing and training. This model uses the integrated webcam to capture the video frame. The image of the gesture captured in the video frame is compared with the Pre-trained model and the gesture is identified. If the gesture predicted is 1 then images is blurred;2, image is resized;3,image is rotated etc. Application layout is done by HTML for better outlook.

3.IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map canvas

gesture based systems save time

gesture based system make the commands understandable without using hard codes everytime

of getting work done without having to take multiple follow ups

of how much time wasted in uploading and hardcoding the inputs

don't require more people to operate which reduce the cost

Says

Thinks

Does

Feels

identifies the action needed to be done without any contact

acts based on the preuploaded information of gestures and their result

repeats the process based on how much time we give input as gestures

feels uncertain because it may act according to the similarities so feels unsafe

feels there should be a protection system for identifying non voluntary inputs given to the system

3.2 Ideation and brainstorming

Gestures as a basic form of non-verbal communication made with the hands. Humans are able to recognize body and sign language easily. This is possible due to the combination of vision and synaptic interactions that were formed along brain development . Inorder to replicate this skill in computers, some problems need to be solved: how to separate objects of interest in images and which image capture technology and classification technique are more appropriate, among others. In this project Gesture based Desktop automation, User interacts with the UI (User Interface) to upload the image as input. Depending on the different gesture inputs different operations are applied to the input image. Once model analyses the gesture, the prediction with operation applied on image is showcased on the UI. First the model is trained pre trained on the images of different hand gestures, such as a showing numbers with fingers as 1,2,3,4 by data collection and then image is processed by ImageDataGenerator library. The proposed project is done using open cv and IBM cloud. By CNN model is initialized and input,hidden,output layers are created and then is proposed for testing and training. This model uses the integrated webcam to capture the video frame. The image of the gesture captured in the video frame is compared with the Pre-trained model and the gesture is identified. If the gesture predicted is 1 then images is blurred;2, image is resized;3, image is rotated etc. Application layout is done by HTML for better outlook.

3.3 Proposed Solution

Humans are able to recognize body and sign language easily. This is possible due to the combination of vision and synaptic interactions that were formed along brain development . In order to replicate this skill in computers, some problems need to be solved: how to separate objects of interest in images and which image capture technology and classification technique are more appropriate, among others.

The model is trained pre trained on the images of different hand gestures, such as a showing numbers with fingers as 1 ,2,3,4 .This model uses the integrated webcam to

capture the video frame. The image of the gesture captured in the video frame is compared with the Pre-trained model and the gesture is identified. If the gesture predicts is 1 then images is blurred;2, image is resized;3,image is rotated etc. Gesture based tool has been used in many sectors but for radiology for a medical use it is considered not efficient , this is the first time gesture based sterile browsing of radiology image is used for medical purposes that uses AI and Machine learning. This will create a impact in the health sector fast processing of medical images allows us to treat the diseases in more efficient way.

Many hospitals can use this as an outake for specially challenged people in emergency period. The proposed solution is highly effective in the areas actions takes a great place in diagnostic centers to multispeciality hospitals.

3.4 Problem solution fit

1. CUSTOMER SEGMENT(S)	6. CUSTOMER LIMITATIONS	5. AVAILABLE SOLUTION(S)
MultiSpecialityHospitals,Diagonostic Centers,CT scans,MRI.	Needed to wait for long time get the required results. Needed to wait for the specific personal to get the scan results no lay man can operate it.	Provides output processed image based on the input through the gestures provided by us.
2. PROBLEMS / PAINS Problems include extra time taken to feed the information about what to do for the patient in need of a surgery or an emergency	9. PROBLEM ROOT / CAUSE We use code everytime to process a image and we need to feed the information to what to do which takes more time	7. BEHAVIOR Slow process, only the technical personal can handle it because of the process involved is complex.
3.TRIGGERS TO ACT Providing faster results of scanned images Reducing the queue Providing it for cheaper costs 4. EMOTIONS Before - Risk of getting delayed for a surgery After -faster results gets rid of delayed results	10. SOLUTION Solutions include creating a AI with machine learning technology which detects our hand gestures , analyse it for giving accurate results. With this any anyone can operate this image processing technology with ease and the time required to get the results is drastically reduced.	8. CHANNELS OF BEHAVIOR Social media platforms like YouTube, Instagram ,facebook ,governmental and non governmental organizations

4. <u>REQUIREMENT ANALYSIS</u>

4.1 Functional requirement

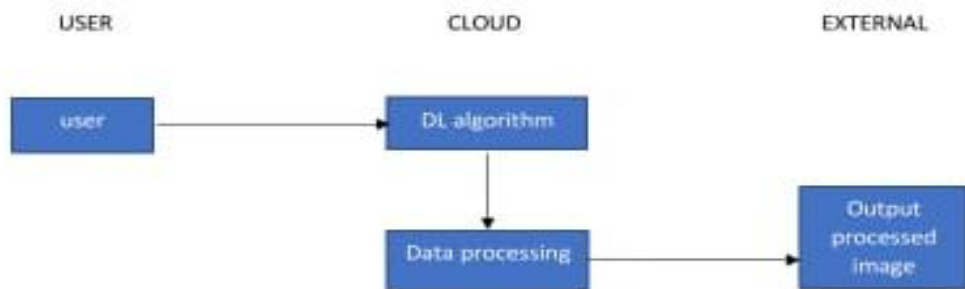| FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Hand detection	Filters out hand from video capturing device.
FR-2	Skin detection	Filter the objects of non skin-coloured.
FR-3	Filtered object detection	Read and recognize clusters skin coloured objects.
FR-4	Hand calibration	Perform adjustment according to user's dominant hand.
FR-5	Browsing gesture control mode	Hand gesture recognized for commands.

4.2 Non-Functional requirements

| FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Easy to use for all users with minimal instruction. It is understandable by non-technical users.
NFR-2	Security	Application will be permissible to used only in secure Networks and less feasibility of insecurity.
NFR-3	Reliability	It is operable in all lighting conditions. Regardless, of the brightness level in user's operating environment.
NFR-4	Performance	Minimize the number of calculation to perform hand gesture detection and to improve image quality resolution.
NFR-5	Availability	This application supports recognition of gesture only through internet connection.
NFR-6	Scalability	Scalable to support future developments and shall be at least 50% extensible to allow new gesture recognition features to be added to the systems.

5.PROJECT DESIGN

5.1 Data Flow Diagrams

Simplified form

Flow diagram
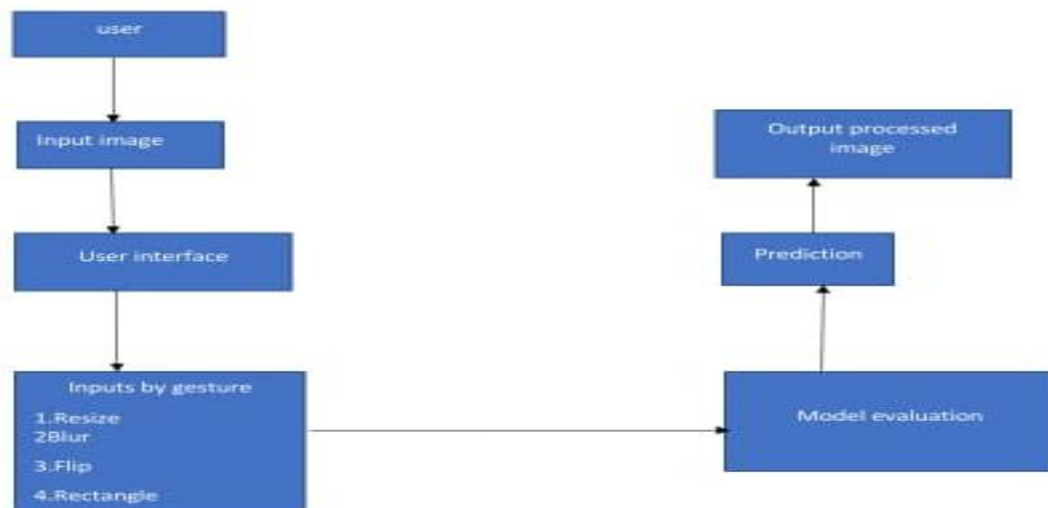
5.2 Solution & Technical Architecture
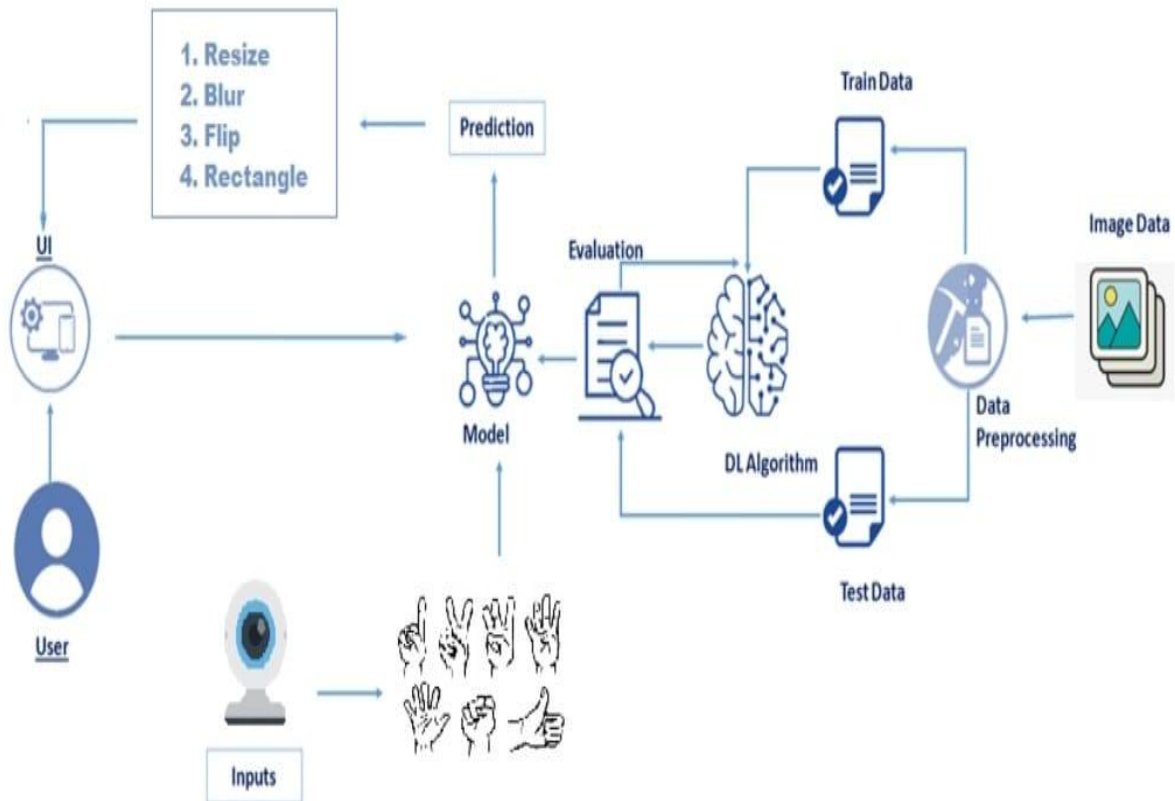
5.3 User stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Process images	USN-1	As a user, I can resize and blur the image	I can access my account for processing image	High	Sprint-1
		USN-2	As a user, I can reshape my image as rectangular.	I can use my account to reshape my image	High	Sprint-1
		USN-3	As a user, I can see the processing operations in the dashboard	I can visit all operations	Medium	Sprint-1

6.PROJECT PLANNING & SCHEDULING

6.1Sprint Planning and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	High	Naveen, Praveen
		USN-2	As a user, I will receive confirmation email once I have registered for the application	High	Nitiyasri, Nithiyasree
		USN-3	As a user, I can register for the application through Facebook	Low	Naveen, Nitiyasri
		USN-4	As a user, I can register for the application through Gmail	Medium	Praveen,Nithiyasree
Sprint 2	Login	USN-5	As a user, I can log into the application by entering email & password	High	Naveen,Praveen
	Dashboard	USN-6	As a user, I can view my profile and update my details	Medium	Naveen, Nithiyasree
		USN-7	As a user, I can view all images uploaded	High	Praveen,Nitiyasri
		USN-8	As a user, I can change my password	High	Nitiyasri, Nithiyasree
Sprint-3	Image Capturing	USN-9	As a user, I can capture images of hand gestures made by me	High	Naveen, Nitiyasri

	Image Processing	USN-10	In the application, the captured images are processed to identify the hand gesture	High	Praveen, Nithiyasree
	Data Storage	USN-11	In the application radiology images uploaded by the user are stored using a database	High	Naveen ,Praveen
Sprint-4	Sterile Browsing	USN-12	Depending on the different gesture inputs different operations are performed on the input image	High	Nitiyasri, Nithiyasree
	Displaying the operations performed	USN-13	Once the model analyses the gesture, the prediction with operation applied on the image is showcased on	High	Naveen, Nitiyasri

			the user interface		

6.2 Sprint Delivery schedule

Sl. NO	MILESTONE	ACTIVITIES	DATE
1	Preparation Phase	Pre-requisites	25 Aug 2022
		Prior knowledge	26 Aug 2022
		Project Structure	24 Aug 2022
		Project Flow	24 Aug 2022
		Project Objectives	23 Aug 2022
		Registrations	27 Aug 2022
		Environment Set-up	28 Aug 2022
2	Ideation Phase	Literature Survey	30 Aug 2022 – 03 Sept 2022
		Empathy Map	6 Sept 2022 – 7 Sept 2022
		Problem Statement	9 Sept 2022 –10 Sept 2022
		Ideation	13 Sept 2022 - 16 Sept 2022
3	Project Design Phase -I	Proposed Solution	20 Sept 2022 –23 Sept 2022
		Problem SolutionFit	25Sept 2022 – 26 Sept 2022
		Solution Architecture	28 Sept 2022 – 30 Sept 2022
4	Project Design Phase -II	Customer Journey	4 Oct 2022 – 8 Oct 2022
		Requirement Analysis	10 Oct 2022 – 11 Oct 2022
		Data Flow Diagram	12 Oct 2022 – 14 Oct 2022

		Technology Architecture	16 Oct 2022 – 17 Oct 2022
5	Project Planning Phase	Milestones & Tasks	18 Oct 2022 – 19 Oct 2022
		Sprint Schedules	20 Oct 2022 - 23 Oct 2022
6	Project Development Phase	Sprint -1	25 Oct 2022 – 29 Oct 2022
		Sprint -2	31 Oct 2022 – 05 Nov 2022
		Sprint -3	08 Nov 2022 –13 Nov 2022
		Sprint -4	14 Nov 2022 – 20 Nov 2022

7.<u>CODING & SOLUTIONING</u>

<u>Project structure</u>

● Dataset folder contains the training and testing images for training our model.

● We are building a Flask Application which needs HTML pages stored in the templates folder and a python script app.py for server side scripting

● we need the model which is saved and the saved model in this content is gesture.h5

● The static folder will contain js and cssfiles

● Whenever we upload a image to predict, that images is saved in uploads folder.

<u>Data collection</u>

ML depends heavily on data, without data, it is impossible for a machine to learn. It is the most crucial aspect that makes algorithm training possible. In

Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions.

Image processing

In this step we improve the image data that suppresses unwilling distortions or enhances some image features important for further processing, although perform some geometric transformations of images like rotation, scaling, translation etc.

```python
from keras.preprocessing.image import ImageDataGenerator
```

Image Data Agumentation

```python
#setting parameter for Image Data agumentation to the traing data
train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
#Image Data agumentation to the testing data
test_datagen=ImageDataGenerator(rescale=1./255)
```

Loading our data and performing data agumentation

```python
#performing data agumentation to train data
x_train = train_datagen.flow_from_directory('data/train',target_size=(64, 64),batch_size=5,
                                            color_mode='grayscale',class_mode='categorical')
#performing data agumentation to test data
x_test = test_datagen.flow_from_directory('data/test',target_size=(64, 64),batch_size=5,
                                            color_mode='grayscale',class_mode='categorical')
```

```
Found 600 images belonging to 6 classes.
Found 30 images belonging to 6 classes.
```

```python
#performing data agumentation to train data
x_train = train_datagen.flow_from_directory(r'C:\\Users\\Anura\\OneDrive\\Desktop\\Gesture-Based-Number-Recognition-main\\New folder\\Data\\train',
                                            target_size=(64, 64),
                                            batch_size=3,
                                            color_mode='grayscale',
                                            class_mode='categorical')
#performing data agumentation to test data
x_test = test_datagen.flow_from_directory(r'C:\\Users\\Anura\\OneDrive\\Desktop\\Gesture-Based-Number-Recognition-main\\New folder\\Data\\test',
                                            target_size=(64, 64),
                                            batch_size=3,
                                            color_mode='grayscale',
                                            class_mode='categorical')
```

Model building

Importing the libraries

```
Importing Neccessary Libraries

import numpy as np #used for numerical analysis
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A layer consists of a tensor-in tensor-out computation function
#Dense layer is the regular deeply connected neural network layer
from tensorflow.keras.layers import Dense,Flatten
#Faltten-used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D,MaxPooling2D #Convolutional layer
#MaxPooling2D-for downsampling the image
from keras.preprocessing.image import ImageDataGenerator
```

```
model=Sequential()
```

Adding cnn layers

```
# First convolution layer and pooling
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 1), activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
# Second convolution layer and pooling
classifier.add(Conv2D(32, (3, 3), activation='relu'))
# input_shape is going to be the pooled feature maps from the previous convolution layer
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Flattening the layers
classifier.add(Flatten())
```

Adding dense layers

Dense layer is deeply connected neural network layer. It is most common and frequently used layer.

```
# Adding a fully connected layer, i.e. Hidden Layer
model.add(Dense(units=512 , activation='relu'))



# softmax for categorical analysis, Output Layer
model.add(Dense(units=6, activation='softmax'))
```

Understanding the model is very important phase to properly use it for training and prediction purposes. Keras provides a simple method, summary to get the full information about the model and its layers.

```
classifier.summary() #summary of our model

Model: "sequential_4"
_____
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_6 (Conv2D)            (None, 62, 62, 32)        320
_____
max_pooling2d_6 (MaxPooling2 (None, 31, 31, 32)        0
_____
conv2d_7 (Conv2D)            (None, 29, 29, 32)        9248
_____
max_pooling2d_7 (MaxPooling2 (None, 14, 14, 32)        0
_____
flatten_3 (Flatten)          (None, 6272)              0
_____
dense_6 (Dense)              (None, 128)               802944
_____
dense_7 (Dense)              (None, 6)                 774
=================================================================
Total params: 813,286
Trainable params: 813,286
Non-trainable params: 0
```

Configure the learning process

● The compilation is the final step in creating a model. Once the compilation is done, we can move on to training phase. Loss function is used to find error or deviation in the learning process. Keras requires loss function during model compilation process.

 ● Optimization is an important process which optimize the input weights by comparing the prediction and the loss function. Here we are using Adam optimizer Metrics is used to evaluate the performance of your model. It is similar to loss function, but not used in training process.

Compiling the model

```
# Compiling the CNN
# categorical_crossentropy for more than 2
classifier.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Train the model

Train the model with our image dataset. fit_generator functions used to train a deep learning neural network Arguments:

● steps_per_epoch : it specifies the total number of steps taken from the generator as soon as one epoch is finished and next epoch has started. We can calculate the value of steps_per_epoch as the total number of samples in your dataset divided by the batch size.

● Epochs : an integer and number of epochs we want to train our model for.

● validation_data can be either: 1. an inputs and targets list 2. a generator 3. an inputs, targets, and sample_weights list which can be used to evaluate the loss and metrics for any model after any epoch has ended.

● validation_steps :only if the validation_data is a generator then only this argument can be used. It specifies the total number of steps taken from the generator before it is stopped at every epoch and its value is calculated as the total number of validation data points in your dataset divided by the validation batch size.

```
# It will generate packets of train and test data for training
model.fit_generator(x_train,
                    steps_per_epoch = 594/3 ,
                    epochs = 25,
                    validation_data = x_test,
                    validation_steps = 30/3 )
```

```
Epoch 1/25
198/198 [==============================] - 7s 34ms/step - loss: 1.3144 - accuracy: 0.4798 - val_loss: 0.7614 - val_accuracy: 0.7000
Epoch 2/25
198/198 [==============================] - 7s 34ms/step - loss: 0.6828 - accuracy: 0.7155 - val_loss: 0.5644 - val_accuracy: 0.8000
Epoch 3/25
198/198 [==============================] - 7s 33ms/step - loss: 0.4049 - accuracy: 0.8552 - val_loss: 0.7858 - val_accuracy: 0.7667
Epoch 4/25
198/198 [==============================] - 7s 34ms/step - loss: 0.3155 - accuracy: 0.8721 - val_loss: 0.2433 - val_accuracy: 0.9667
Epoch 5/25
198/198 [==============================] - 7s 34ms/step - loss: 0.1929 - accuracy: 0.9327 - val_loss: 0.3210 - val_accuracy: 0.9667
Epoch 6/25
198/198 [==============================] - 7s 34ms/step - loss: 0.1761 - accuracy: 0.9495 - val_loss: 0.5928 - val_accuracy: 0.9000
Epoch 7/25
198/198 [==============================] - 7s 34ms/step - loss: 0.1257 - accuracy: 0.9613 - val_loss: 0.3547 - val_accuracy: 0.9333
Epoch 8/25
198/198 [==============================] - 7s 34ms/step - loss: 0.0959 - accuracy: 0.9630 - val_loss: 0.4215 - val_accuracy: 0.9667
Epoch 9/25
198/198 [==============================] - 7s 35ms/step - loss: 0.1353 - accuracy: 0.9444 - val_loss: 0.3127 - val_accuracy: 0.9667
Epoch 10/25
198/198 [==============================] - 7s 34ms/step - loss: 0.0985 - accuracy: 0.9697 - val_loss: 0.3157 - val_accuracy: 0.9667
Epoch 11/25
198/198 [==============================] - 7s 34ms/step - loss: 0.0824 - accuracy: 0.9747 - val_loss: 0.3259 - val_accuracy: 0.9667
Epoch 12/25
198/198 [==============================] - 7s 34ms/step - loss: 0.0474 - accuracy: 0.9865 - val_loss: 0.4769 - val_accuracy: 0.9333
Epoch 13/25

198/198 [==============================] - 7s 34ms/step - loss: 0.0319 - accuracy: 0.9865 - val_loss: 0.3459 - val_accuracy: 0.9667
Epoch 24/25
198/198 [==============================] - 7s 35ms/step - loss: 0.0385 - accuracy: 0.9815 - val_loss: 0.3301 - val_accuracy: 0.9667
Epoch 25/25
198/198 [==============================] - ETA: 0s - loss: 0.0138 - accuracy: 0.99 - 7s 34ms/step - loss: 0.0138 - accuracy: 0.9966 - val_loss: 0.2801 - val_accuracy: 0.9667
<tensorflow.python.keras.callbacks.History at 0x1b89d20da60>
```

```python
# Save the model
model.save('gesture.h5')



model_json = model.to_json()
with open("model-bw.json", "w") as json_file:
    json_file.write(model_json)
```

Testing the model

Evaluation is a process during development of the model to check whether the model is best fit for the given problem and corresponding data. Load the saved model using load_model

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model = load_model("gesture.h5") #loading the model for testing
path = "C:\\Users\\Anura\\OneDrive\\Desktop\\Gesture-Based-Number-Recognition-main\\im6.jpg"
```

Plotting images:

```
%pylab inline
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
imgs = mpimg.imread(path)
imgplot = plt.imshow(imgs)
plt.show()
```
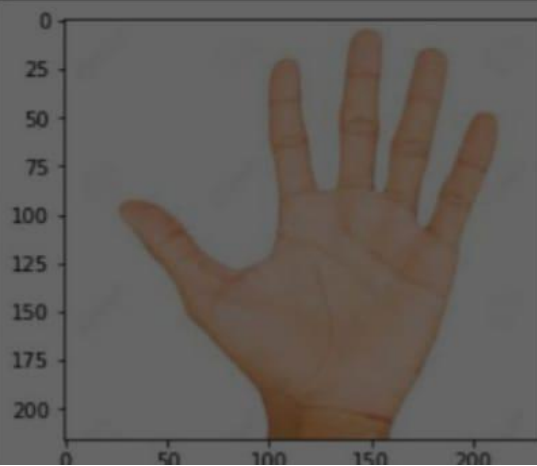

Taking an image as input and checking the results

```
index=['0','1','2','3','4','5']
result=str(index[pred[0]])
result

'1'
```

```
import numpy as np
p = []

for i in range(0,6):
    for j in range(0,5):
        path = "C:\\Users\\Anura\\OneDrive\\Desktop\\Gesture-Based-Number-Recognition-main\\New folder\\Data\\test\\"+str(i)+"\\"+str(j)+".jpg"
        img = image.load_img(path,color_mode = "grayscale",target_size= (64,64))
        x = image.img_to_array(img)
        x = np.expand_dims(x,axis = 0)
        pred = np.argmax(model.predict(x), axis=-1)
        p.append(pred)

print(p)
```
Pytho

```
[array([0], dtype=int64), array([0], dtype=int64), array([0], dtype=int64), array([0], dtype=int64), array([0], dtype=int64), array([1],
dtype=int64), array([1], dtype=int64), array([1], dtype=int64), array([1], dtype=int64), array([1], dtype=int64), array([2], dtype=int64),
array([2], dtype=int64), array([1], dtype=int64), array([2], dtype=int64), array([2], dtype=int64), array([3], dtype=int64), array([3],
dtype=int64), array([3], dtype=int64), array([3], dtype=int64), array([3], dtype=int64), array([4], dtype=int64), array([4], dtype=int64),
array([4], dtype=int64), array([4], dtype=int64), array([4], dtype=int64), array([5], dtype=int64), array([5], dtype=int64), array([5],
dtype=int64), array([5], dtype=int64), array([5], dtype=int64)]
```

<u>Application building</u>

After the model is trained in this particular step, we will be building our flask application which will be running in our local browser with a user interface.

Create HTML Pages

● We use HTML to create the front end part of the web page.

● Here, we created 3 html pages- home.html, intro.html and index6.html

● home.html displays home page.

● Intro.html displays introduction about the hand gesture recognition

● index6.html accepts input from the user and predicts the values.

● We also use JavaScript-main.js and CSS-main.css to enhance our functionality and view of HTML pages.
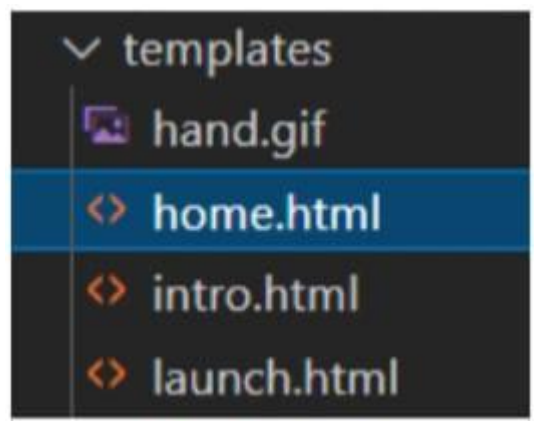
Build Python Code

● Build flask file 'app.py' which is a web framework written in python for server-side scripting.

● App starts running when " name " constructor is called in main.

● render_template is used to return html file.

● "GET" method is used to take input from the user.

● "POST" method is used to display the output to the user.

● Importing Libraries

```python
from flask import Flask,render_template,request
# Flask-It is our framework which we are going to use to run/serve our application.
#request-for accessing file which was uploaded by the user on our application.
import operator
import cv2 # opencv library
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np

from tensorflow.keras.models import load_model#to load our trained model
import os
from werkzeug.utils import secure_filename
```

```python
app = Flask(__name__,template_folder="templates") # initializing a flask app
# Loading the model
model=load_model('gesture.h5')
print("Loaded model from disk")
```

Routing to html page

```python
@app.route('/')# route to display the home page
def home():
    return render_template('home.html')#rendering the home page


@app.route('/intro') # routes to the intro page
def intro():
    return render_template('intro.html')#rendering the intro page

@app.route('/image1',methods=['GET','POST'])# routes to the index html
def image1():
    return render_template("index6.html")
```

```python
@app.route('/predict',methods=['GET', 'POST'])# route to show the predictions in a web UI
def launch():
```

And the predict route is used for prediction and it contains all the codes which are used for predicting our results.

And the predict route is used for prediction and it contains all the codes which are used for predicting our results. Firstly, inside launch function we are having the following things:

● Getting our input and storing it

 ● Grab the frames from the web cam.

● Creating ROI

● Predicting our results

● Showcase the results with the help of opencv

● Finally run the application Getting our input and storing it Once the predict route is called, we will check whether the method is POST or not if is POST then we will request the image files and with the help of os function we will be storing the image in the uploads folder in our local system.

```python
if request.method == 'POST':
    print("inside image")
    f = request.files['image']

    basepath = os.path.dirname(__file__)
    file_path = os.path.join(basepath, 'uploads', secure_filename(f.filename))
    f.save(file_path)
    print(file_path)
```

Grab the frames from the web cam

when we run the code a web cam will be opening to take the gesture input so we will be capturing the frames of the gesture for predicting our results.

```python
cap = cv2.VideoCapture(0)
while True:
    _, frame = cap.read() #capturing the video frame values
    # Simulating mirror image
    frame = cv2.flip(frame, 1)
```

Creating ROI

A region of interest (ROI) is a portion of an image that you want to filter or operate on in some way. The toolbox supports a set of ROI objects that you can use to create ROIs of many shapes, such circles, ellipses, polygons, rectangles,

and hand-drawn shapes. A common use of an ROI is to create a binary mask image.

```python
# Got this from collect-data.py
# Coordinates of the ROI
x1 = int(0.5*frame.shape[1])
y1 = 10
x2 = frame.shape[1]-10
y2 = int(0.5*frame.shape[1])
# Drawing the ROI
# The increment/decrement by 1 is to compensate for the bounding box
cv2.rectangle(frame, (x1-1, y1-1), (x2+1, y2+1), (255,0,0) ,1)
# Extracting the ROI
roi = frame[y1:y2, x1:x2]

# Resizing the ROI so it can be fed to the model for prediction
roi = cv2.resize(roi, (64, 64))
roi = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
_, test_image = cv2.threshold(roi, 120, 255, cv2.THRESH_BINARY)
cv2.imshow("test", test_image)
```

```python
result = model.predict(test_image.reshape(1, 64, 64, 1))
prediction = {'ZERO': result[0][0],
              'ONE': result[0][1],
              'TWO': result[0][2],
              'THREE': result[0][3],
              'FOUR': result[0][4],
              'FIVE': result[0][5]}
# Sorting based on top prediction
prediction = sorted(prediction.items(), key=operator.itemgetter(1), reverse=True)

# Displaying the predictions
cv2.putText(frame, prediction[0][0], (10, 120), cv2.FONT_HERSHEY_PLAIN, 1, (0,255,255), 1)
```

Finally according to the result predicted with our model we will be performing certain operations like resize, blur , rotate etc.

```python
#loading an image
image1=cv2.imread(file_path)
if prediction[0][0]=='ONE':


    resized = cv2.resize(image1, (200, 200))
    cv2.imshow("Fixed Resizing", resized)
    key=cv2.waitKey(3000)


    if (key & 0xFF) == ord("1"):
        cv2.destroyWindow("Fixed Resizing")


elif prediction[0][0]=='ZERO':


    cv2.rectangle(image1, (480, 170), (650, 420), (0, 0, 255), 2)
    cv2.imshow("Rectangle", image1)
    cv2.waitKey(0)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("0"):
        cv2.destroyWindow("Rectangle")


elif prediction[0][0]=='TWO':
    (h, w, d) = image1.shape
    center = (w // 2, h // 2)
    M = cv2.getRotationMatrix2D(center, -45, 1.0)
    rotated = cv2.warpAffine(image1, M, (w, h))
    cv2.imshow("OpenCV Rotation", rotated)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("2"):
        cv2.destroyWindow("OpenCV Rotation")
```

```python
elif prediction[0][0]=='THREE':
    blurred = cv2.GaussianBlur(image1, (21, 21), 0)
    cv2.imshow("Blurred", blurred)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("3"):
        cv2.destroyWindow("Blurred")

elif prediction[0][0]=='FOUR':

    resized = cv2.resize(image1, (400, 400))
    cv2.imshow("Fixed Resizing", resized)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("4"):
        cv2.destroyWindow("Fixed Resizing")

elif prediction[0][0]=='FIVE':
    '''(h, w, d) = image1.shape
    center = (w // 2, h // 2)
    M = cv2.getRotationMatrix2D(center, 45, 1.0)
    rotated = cv2.warpAffine(image1, M, (w, h))'''
    gray = cv2.cvtColor(image1, cv2.COLOR_RGB2GRAY)
    cv2.imshow("OpenCV Gray Scale", gray)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("5"):
        cv2.destroyWindow("OpenCV Gray Scale")

else:
    continue

        interrupt = cv2.waitKey(10)
        if interrupt & 0xFF == 27: # esc key
            break

    cap.release()
    cv2.destroyAllWindows()
return render_template("home.html")
```

Run the application

```python
if __name__ == "__main__":
    # running the app
    app.run(debug=False)
```

Run The app in local browser

● Open anaconda prompt from the start menu

● Navigate to the folder where your python script is.

● Now type "python app.py" command Navigate to the localhost where you can view your web page

```
(base) E:\>cd E:\PROJECTS\number-sign-recognition\Flask

(base) E:\PROJECTS\number-sign-recognition\Flask>python app.py
```

Then it will run on localhost:5000

```
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Navigate to the localhost (http://127.0.0.1:5000/)where you can view your web page.

8.TESTING

In this the input is given as 3,the image is blurred.
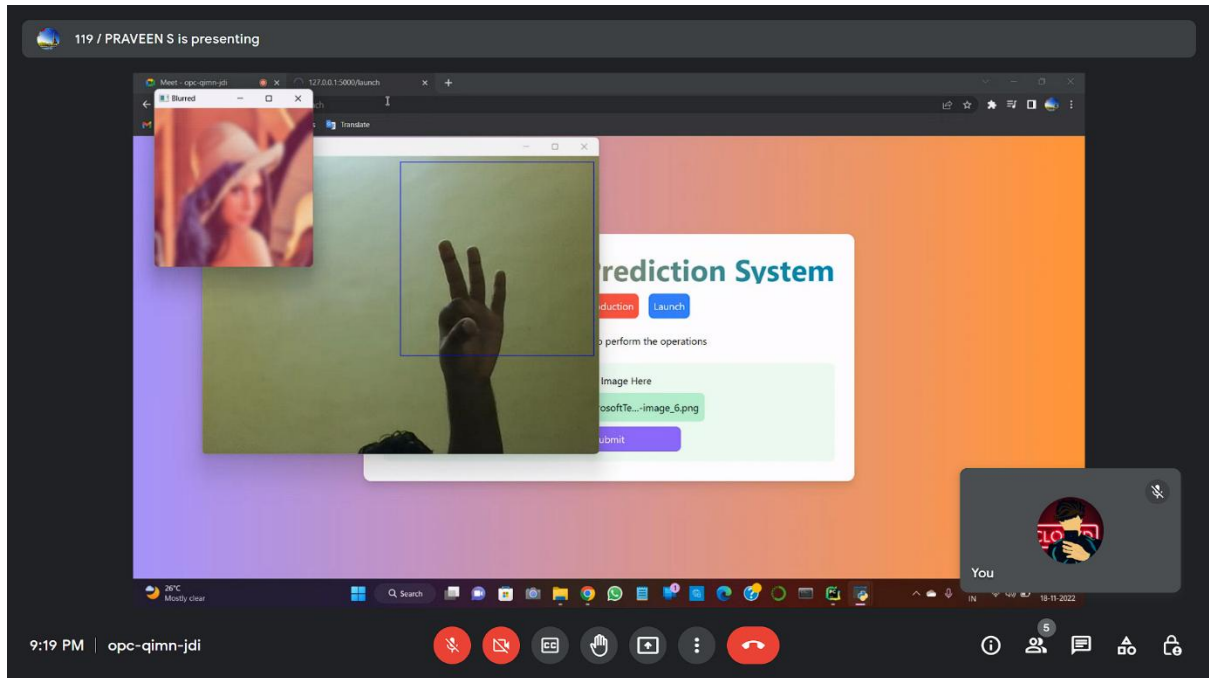
In this the input is given as 2,the image is resized.
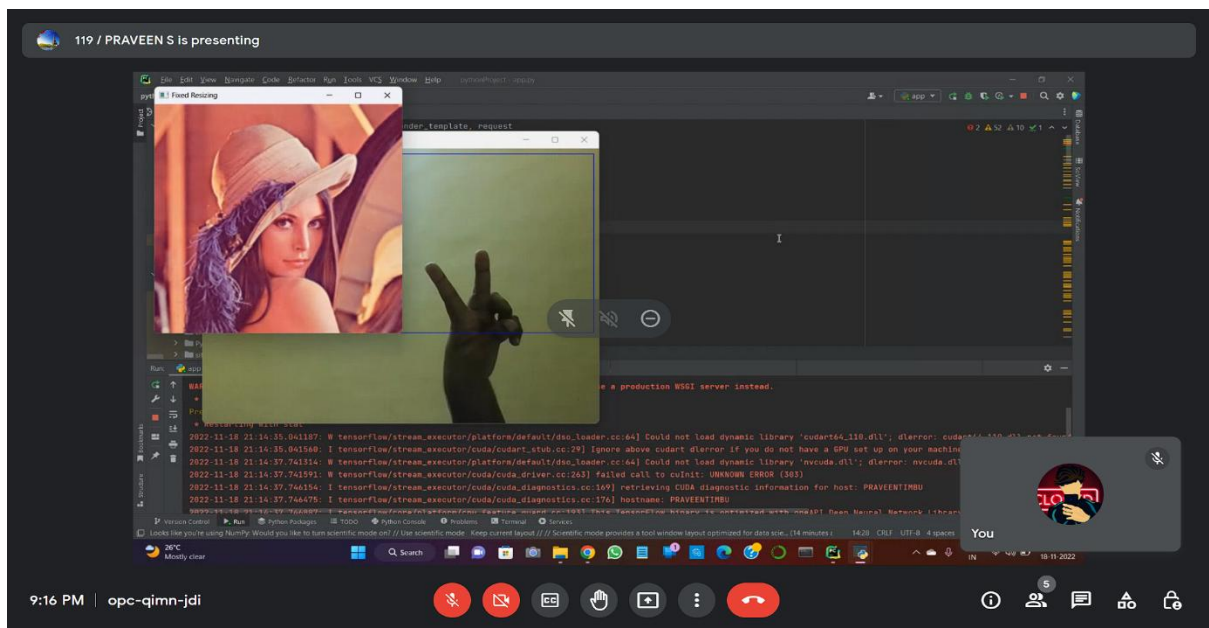
9.RESULTS

9.1 PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot
1.	Model Summary	conv2d (Conv2D) - 320 max_pooling2d (MaxPooling2D) - 0 conv2d_1 (Conv2D) - 9248 max_pooling2d_1 (MaxPooling2D) - 0 flatten (Flatten) - 0 dense (Dense) - 802944 dense_1 (Dense) - 774 ================================ Total params: 813,286 Trainable params: 813,286 Non-trainable params: 0	
2.	Accuracy	Training Accuracy - 99.16% Validation Accuracy – 96.67%	
3.	Confidence Score (Only Yolo Projects)	Class Detected - Confidence Score -	NA

Screenshots:

1. Model Summary:

```
classifier.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 62, 62, 32)	320
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
conv2d_1 (Conv2D)	(None, 29, 29, 32)	9248
max_pooling2d_1 (MaxPooling 2D)	(None, 14, 14, 32)	0
flatten (Flatten)	(None, 6272)	0
dense (Dense)	(None, 128)	802944
dense_1 (Dense)	(None, 6)	774

```
Total params: 813,286
Trainable params: 813,286
Non-trainable params: 0
```

2. Accuracy:

```
classifier.fit_generator(
    generator=x_train, steps_per_epoch=len(x_train),
    epochs=20, validation_data=x_test, validation_steps=len(x_test)
)
```

/tmp/wsuser/ipykernel_217/2617134232.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  classifier.fit_generator(

```
Epoch 1/20
119/119 [==============================] - 6s 43ms/step - loss: 1.3614 - accuracy: 0.4832 - val_loss: 0.6568 - val_accuracy: 0.7667
Epoch 2/20
119/119 [==============================] - 5s 42ms/step - loss: 0.6663 - accuracy: 0.7340 - val_loss: 0.5007 - val_accuracy: 0.9000
Epoch 3/20
119/119 [==============================] - 5s 42ms/step - loss: 0.4844 - accuracy: 0.8081 - val_loss: 0.5624 - val_accuracy: 0.8000
Epoch 4/20
119/119 [==============================] - 5s 41ms/step - loss: 0.3675 - accuracy: 0.8653 - val_loss: 0.4007 - val_accuracy: 0.8667
Epoch 5/20
119/119 [==============================] - 5s 42ms/step - loss: 0.3375 - accuracy: 0.8670 - val_loss: 0.3236 - val_accuracy: 0.9000
Epoch 6/20
119/119 [==============================] - 5s 42ms/step - loss: 0.2559 - accuracy: 0.9108 - val_loss: 0.3335 - val_accuracy: 0.9333
Epoch 7/20
119/119 [==============================] - 5s 40ms/step - loss: 0.2045 - accuracy: 0.9293 - val_loss: 0.3956 - val_accuracy: 0.9333
Epoch 8/20
119/119 [==============================] - 5s 42ms/step - loss: 0.1807 - accuracy: 0.9478 - val_loss: 0.2878 - val_accuracy: 0.9667
Epoch 9/20
119/119 [==============================] - 5s 41ms/step - loss: 0.1360 - accuracy: 0.9461 - val_loss: 0.2737 - val_accuracy: 0.8667
Epoch 10/20
119/119 [==============================] - 5s 41ms/step - loss: 0.1136 - accuracy: 0.9562 - val_loss: 0.3276 - val_accuracy: 0.9667
Epoch 11/20
119/119 [==============================] - 5s 41ms/step - loss: 0.1338 - accuracy: 0.9495 - val_loss: 0.5726 - val_accuracy: 0.9333
Epoch 12/20
119/119 [==============================] - 5s 42ms/step - loss: 0.1307 - accuracy: 0.9495 - val_loss: 0.2451 - val_accuracy: 0.9333
Epoch 13/20
119/119 [==============================] - 5s 42ms/step - loss: 0.0528 - accuracy: 0.9848 - val_loss: 0.3329 - val_accuracy: 0.9333
Epoch 14/20
119/119 [==============================] - 5s 42ms/step - loss: 0.0472 - accuracy: 0.9865 - val_loss: 0.2916 - val_accuracy: 0.9333
Epoch 15/20
119/119 [==============================] - 5s 41ms/step - loss: 0.0496 - accuracy: 0.9815 - val_loss: 0.5053 - val_accuracy: 0.9000
Epoch 16/20
119/119 [==============================] - 5s 41ms/step - loss: 0.0866 - accuracy: 0.9731 - val_loss: 0.3790 - val_accuracy: 0.9667
Epoch 17/20
119/119 [==============================] - 5s 41ms/step - loss: 0.0454 - accuracy: 0.9815 - val_loss: 0.2206 - val_accuracy: 0.9667
Epoch 18/20
119/119 [==============================] - 5s 43ms/step - loss: 0.0479 - accuracy: 0.9815 - val_loss: 0.3190 - val_accuracy: 0.9667
Epoch 19/20
119/119 [==============================] - 5s 41ms/step - loss: 0.0276 - accuracy: 0.9916 - val_loss: 0.3461 - val_accuracy: 0.9667
Epoch 20/20
119/119 [==============================] - 5s 42ms/step - loss: 0.0228 - accuracy: 0.9916 - val_loss: 0.3098 - val_accuracy: 0.9667
```

10.ADVANTAGES & DISADVANTAGES

Advantages

(i) ease of use—the system allows the surgeon to use his/her hands, their natural work tool;

(ii) rapid reaction—nonverbal instructions by hand gesture commands are intuitive and fast

(iii) an unencumbered interface—the proposed system does not require the surgeon to attach a microphone, use head-mounted (body-contact) sensing devices or to use foot pedals

(iv) distance control—the hand gestures can be performed up to 5 meters from the camera and still be recognized accurately.

Disadvantages

Such systems are difficult to devolope because of the complexity and the cost of implementation .

As each gesture is assigned a specific control command this system is not a platform independent since certain commands vary as the operating system varies.

11.CONCLUSION

In this project we developed a tool which recognises hand gestures and enables doctors to browse through radiology images using these gestures. This enables doctors and surgeons to maintain the sterility as they would not have to touch any mouse or keyboard to go through the images. This tool is also easy to use and is quicker than the regular method of using mouse/keyboard. It can be used regardless of the users location since they don't have to be in contact with any device. It also does not require the user to have any device on them to use it.

Further this technology can be extended to other industries like it can be used by presenters, by teachers for show images in the classroom, etc.

12.FUTURE SCOPE

• The tool can be made quicker by increasing the recognition speed.

• More number of gestures can be added thereby increasing this tool's functionality and useability for different purposes.

• Tracking of both hands can be added to increase the set of commands. Voice commands can also be added to further increase the functionality.

13.APPENDIX

13.1 SOURCE CODE

https://github.com/IBM-EPBL/IBM-Project-24225-1659940279/blob/main/Project%20Development%20Phase/Sprint-4/FFinal_code.ipynb

13.2 GIT HUB AND PROJECT DEMO LINK

Git repo link

https://github.com/IBM-EPBL/IBM-Project-24225-1659940279

project demolink

https://drive.google.com/drive/folders/1Zzk9XM9Dwx-yR_CtEyshcs4Hoyt-mZ2P?usp=sharing