A Gesture-based Tool for Sterile Browsing of Radiology Images

A PROJECT REPORT

Submitted by

A. AJAY KRISHNAN (961219205003)

M. LOKESWARAN (961219205021)

S. RAMESH (961219205501)

N.HARI PRASATH (961219205012)

M.SATHIS KUMAR (961219205028)

In partial fulfillment for award of the degree of

Bachelor of Technology

IN

INFORMATION TECHNOLOGY

LOYOLA INSTITUTE OF TECHNOLOGY AND SCIENCE

THOVALAI

ANNA UNIVERSITY: Chennai 600 025

JUNE 2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report" **A Gesture-based Tool for Sterile Browsing of Radiology Images**" is the Bonafede work of A. AJAY KRISHNAN, M. LOKESWARAN, S. RAMESH, N.HARI PRASATH, M.SATHIS KUMAR who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Mrs.Shakeela Joy	
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Professor	Professor
Information Technology	Computer Science and Engineering
Loyola Institute of Technology	Loyola Institute of Technology
and Science	and Science
Thovalai-629 302	Thovalai-629 302

ACKNOWLEDGEMENT

This project is the acknowledgement to the intensity, Passion, dedication and technical brilliance of many individuals who have guided us in the project to its completion.

First and foremost, we express our heartfelt gratitude to the **Almighty God** for giving us the opportunity to excel in our effort to complete the project in time. We are thankful for the blessing and support of chairman of our college **Dr. M. T. NICHOLAS, M.S., PhD**

We are deeply indebted to **Dr. J. DARVIN, M.E., PhD**, for the inspiration inculcated in us and for the apt guidance.

We would also like to express our sincere thanks to **Mrs.Shakeela Joy** for the kind support, valuable guidance that provided to be the real support for the completion ofthe project.

We also express our sincere thanks to for his motivation, inspiration and encouragement to undertake this work. We are extremely thankful to our parents and friends for their motivation and support.

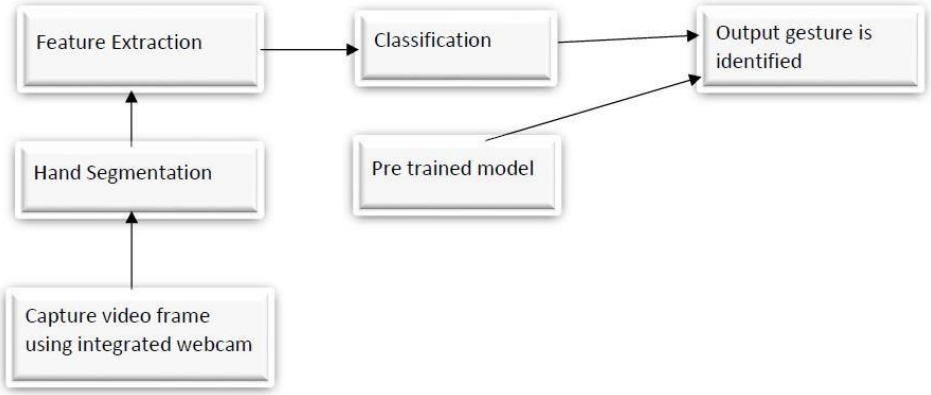
Introduction

1.1 Project Overview

Humans have the ability to recognize body and sign language but computers don't have this ability. Humans can recognize sign language because of the combination of vision and synaptic interactions with the brain. To make computers recognize sign language we need to replicate this skill to computers. Through the position and shape of the center of the palm and the fingers we can obtain certain information. The gesture can be both static and dynamic. Static hand gestures are obtained by analyzing the shape of the hand. Dynamic hand gestures are obtained by analyzing hand movements. The ability to spontaneously identify gestures without delay in hand motion is the problem. Through real-time hand gesture detection, we overcome these problems. Processing speed, image processing techniques and different recognition algorithms are used in this real-time hand gesture detection.

1.2 Purpose

In this project, the model is first pre-trained on the images of different hand gestures, such as showing numbers with fingers as 1, 2, 3, 4. This model uses the integrated webcam to capture the video frame. The image captured in the video frame is compared with the pretrained model and gesture is identified.



Literature Survey

2.1 Existing Problem

S. No	Paper	Work	Tools or Algorithms	Findings
1	E. Stergiopoulou, N. Papamarkos, "Hand gesture recognition using a neural network shape fitting technique"	Hand Gesture fitting procedure via Self-Growing and Self-Organized Neural Gas	Neural Network	The region of the hand is detected by applying a color segmentation technique based on a skin color filtering procedure in the YCbCr color space. Then, the SGONG network is applied on the hand area so as to approach its shape. Based on the output grid of neurons produced by the neural network, palm morphologic characteristics are extracted.
2	Gongfa Li, Heng Tang, "Hand gesture recognition based on convolution neural network"	The convolution neural network is applied to the recognition of gestures	Convolutional Neural Network	Geometric features based on the recognition method, the use of gestures of the edge characteristics and gestures of the regional structure characteristics as a recognition feature

3	G.R.S. Murthy, R.S. Jadon, “Hand gesture recognition using neural networks”	Supervised feed-forward neural net training and back propagation algorithm for classifying hand gestures	Image Acquisition Toolbox of MATLAB	Analyzing and classifying hand gestures for HCI include Glove based techniques and Vision based techniques
4	Hsien-I Lin, Ming-Hsiang Hsu, and Wei-Kai Chen “Hand gesture recognition using a convolutional neural networks”	The skin model and the calibration of hand position and orientation are applied to obtain the training and testing data for the CNN	Convolutional Neural Network, Gaussian Mixture Model	Since the light condition seriously affects the skin color. a Gaussian Mixture model (GMM) is used to train the skin model which is used to robustly filter out non-skin colors of an image
5	Pavlo Molchanov, Shalini Gupta, Kihwan Kim, Jan Kautz, “Hand Gesture Recognition With 3D Convolutional Neural Networks”	An algorithm for drivers' hand gesture recognition from challenging depth and intensity data using 3D convolutional neural networks	3D Convolutional Neural Network	Combines information from multiple spatial scales for the final prediction. Also, employs spatio-temporal data augmentation for more effective training and to reduce potential overfitting

2.1 References

[1] E. Stergiopoulou, N. Papamarkos, “Hand gesture recognition using a neural network shape fitting technique”, Engineering Applications of Artificial Intelligence 22(8):1141-1158, December 2009.

[2] Gongfa Li, Heng Tang, “Hand gesture recognition based on convolution neural network”, IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI), August 2019.

[3] G.R.S. Murthy, R.S. Jadon, “Hand gesture recognition using neural networks”, IEEE 2nd International Advance Computing Conference Conference (IACC), February 2010.

[4] Hsien-I Lin, Ming-Hsiang Hsu, and Wei-Kai Chen, “Hand gesture recognition using convolutional neural networks”, IEEE International Conference on Automation Science and Engineering (CASE), October 2014.

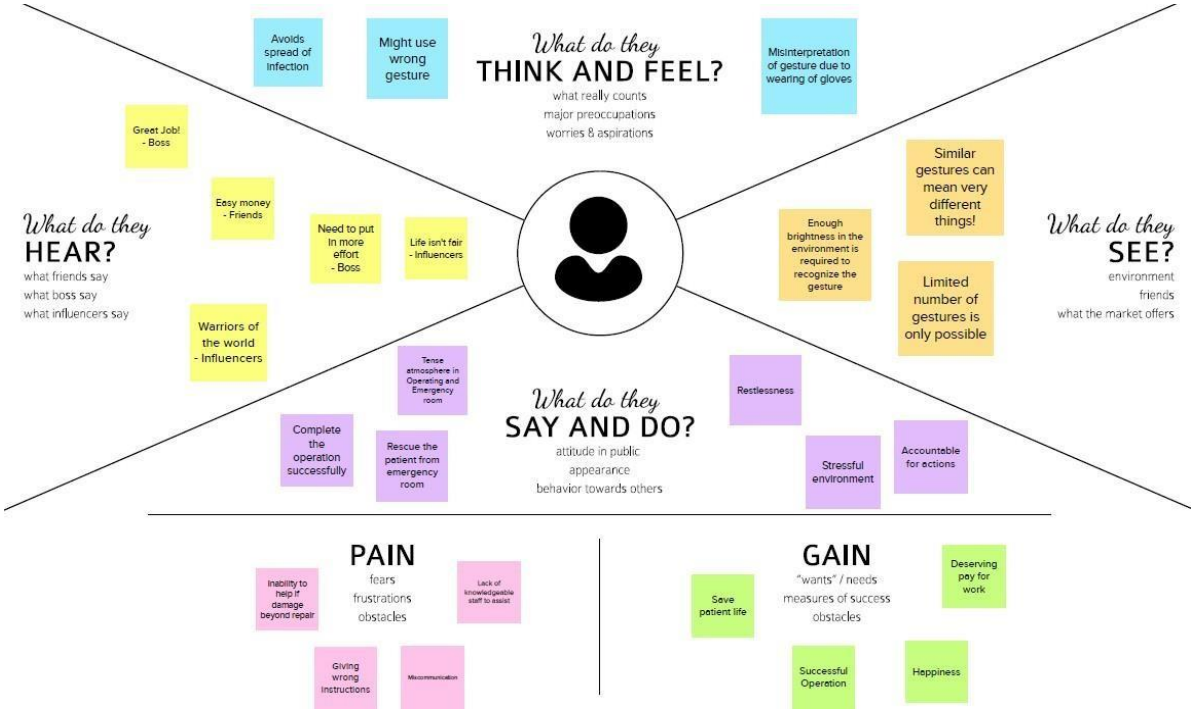
[5] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, Jan Kautz, “Hand Gesture Recognition With 3D Convolutional Neural Networks”, IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), June 2015.

2.2 Problem Statement Definition

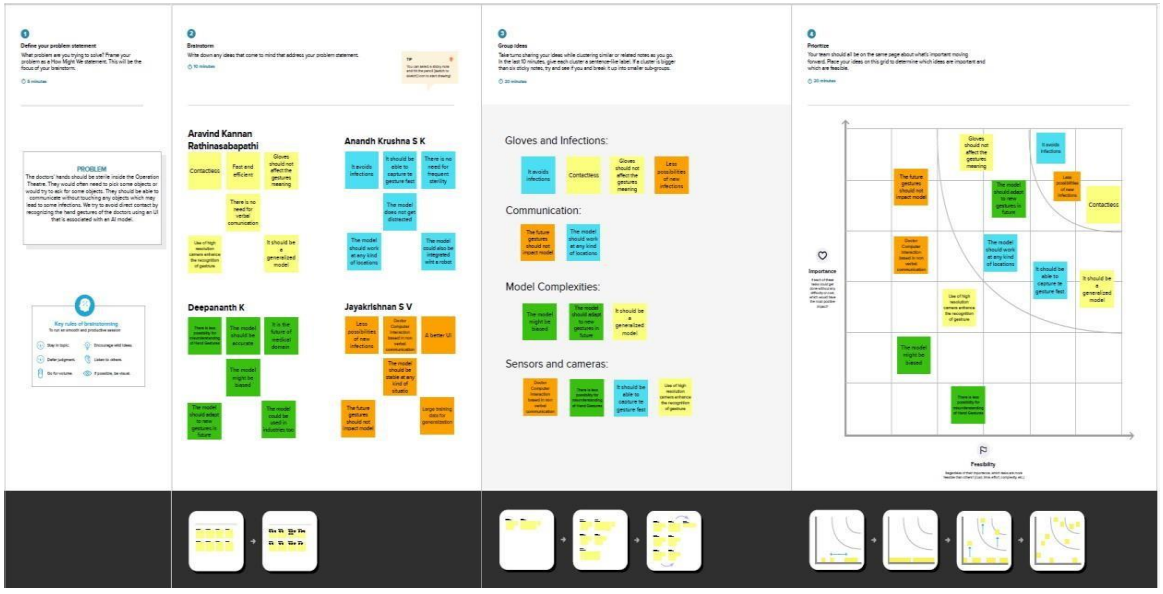
The aim of the project is to build an efficient hand gesture recognition model to recognize the gestures. In addition, the video input is taken and split into frames and the region of interest is found from the frames. The result is given as input to the hand gesture recognition model.

Ideation & Proposed Solution

3.1 Empathy Map



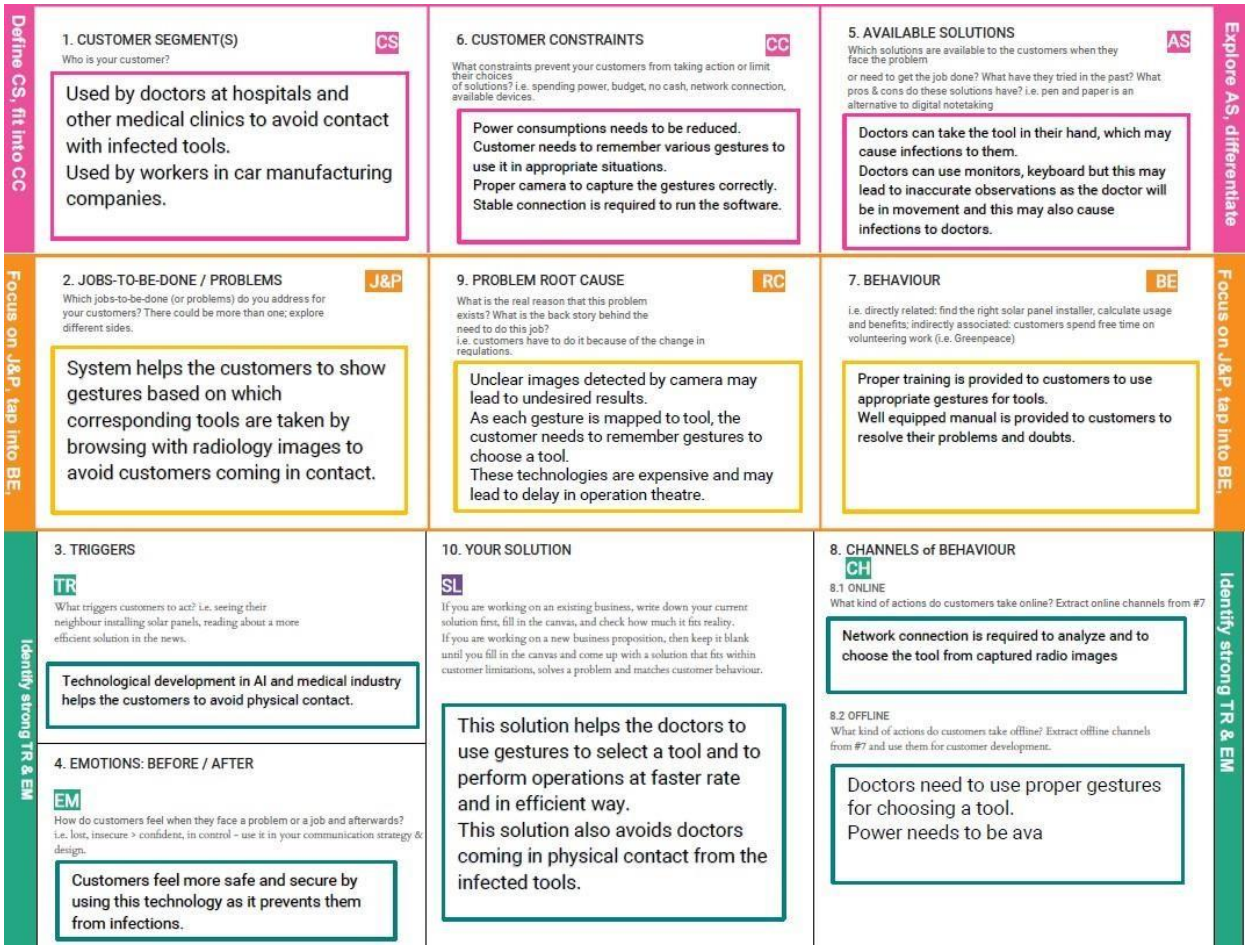
3.2 Ideation & Brainstorming



3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement	To design an ML model to identify and classify the hand gestures
2.	Idea / Solution description	To develop a CNN based classifier model, which would be trained on our training data
3.	Novelty / Uniqueness	We train a CNN based model to recognize the hand gesture. The training data include images that capture the hand gestures of 1,2,3,4,5 and 0. The image is resized without much loss of information and used for training a CNN based model. We use Python Flask to provide an interactive platform for our model.
4.	Social Impact / Customer Satisfaction	This project would help the doctors in operation theaters where physical contact between persons should be avoided in order to be sterilized and also prevent from any infections.
5.	Business Model (Revenue Model)	It can be sold as an open-source service to all the hospitals as a non-profitable work.
6.	Scalability of the Solution	The model could also be extended to other real world classifying problems like cancer detection from X-ray, COVID detection using X-ray images, mask detection, face detection etc..

3.4 Proposed Solution Fit



Requirement Analysis

4.1 Functional Requirements

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Identifying User Gestures	The user gestures are identified using the images of gestures captured by the camera
FR-2	Deployment in Cloud	The trained Deep Learning Model is deployed in cloud, which could be accessed anywhere around the world
FR-3	User Interface	The user interface, which helps in the Human Computer Interaction is designed
FR-4	Gestures related to the Application Domain	The model should be trained with the gestures related to the application domain.

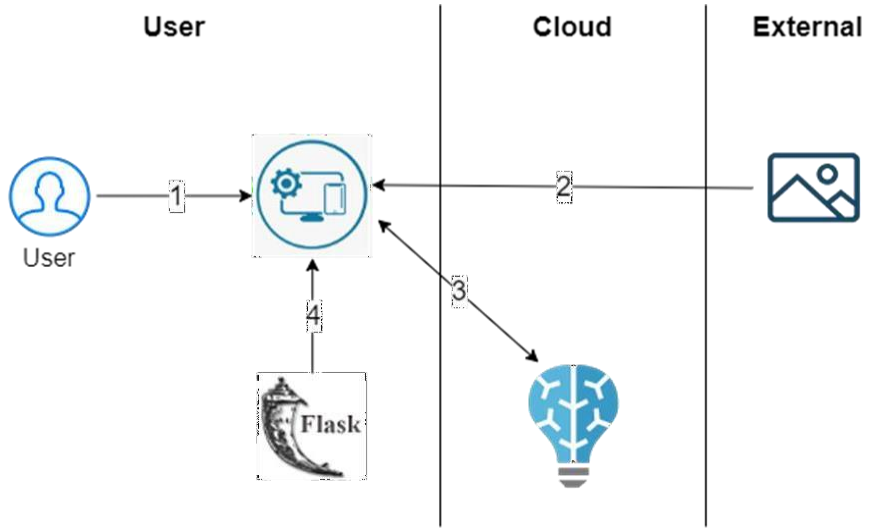
4.2 Non-functional Requirements

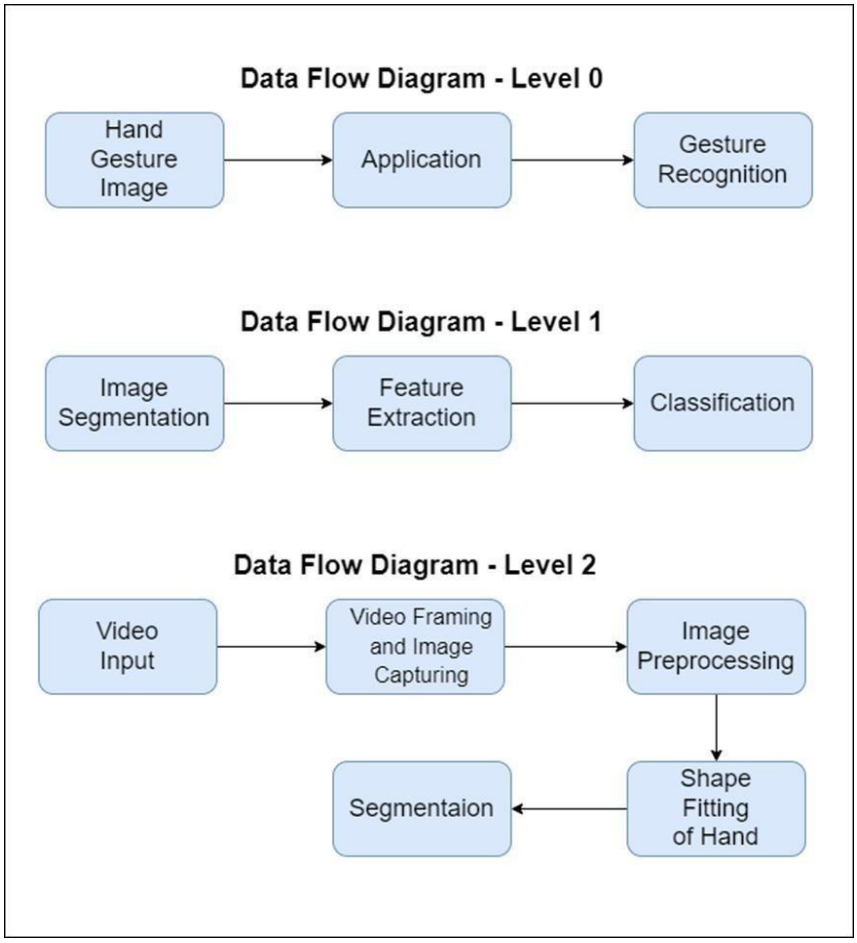
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The user interface which acts as an intermediate between the user and the DL Model which is deployed in the cloud
NFR-2	**Security**	The model deployed in the cloud should be accessible only by the approved users and it should be inaccessible by the attackers or the terrorists
NFR-3	**Reliability**	The tool or the system is 95% reliability for a year

NFR-4	**Performance**	The tool or the system should respond with the accurate response within 4-5 seconds
NFR-5	**Availability**	The model deployed in the cloud must be available to 99.8% of the people over a month during working hours
NFR-6	**Scalability**	The model deployed in the cloud must be accessible by over 10,00,000 people trying to access it using the user interface

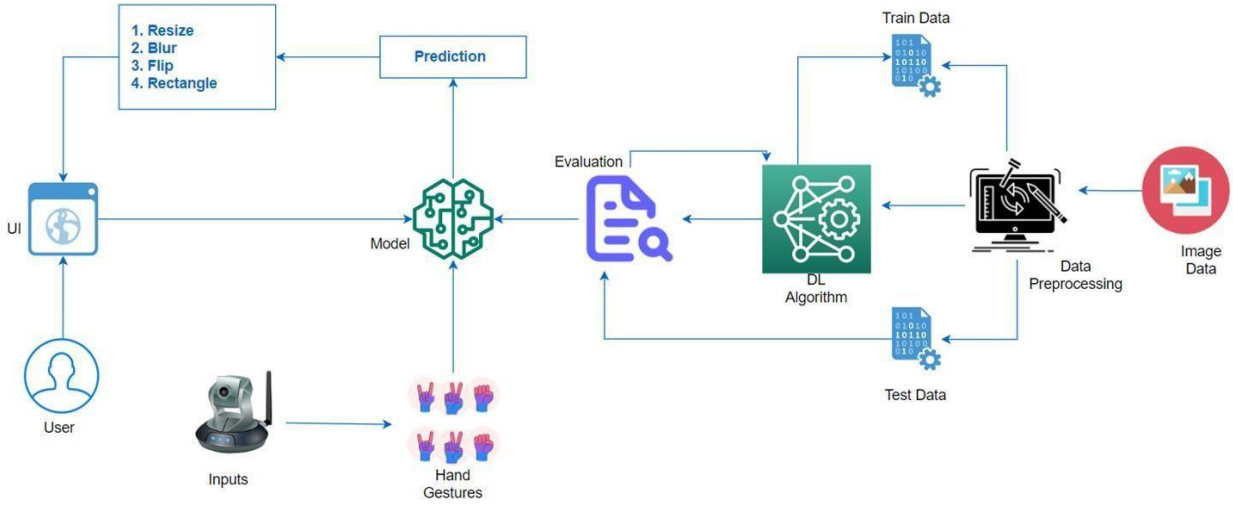
Project Design

5.1 Data 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 (Web user)	Launch Web App	USN-1	As a user, I can launch the web app where I can upload the images for browsing and show gestures	I can upload the images for classification	High	Sprint-4
	Prediction	USN-2	As a user, I can get the predicted results from the model deployed in the cloud	I can resize the radiology image, blur the image, flip based on the hand gesture	High	Sprint-4
	Deployment of web app	USN-3	As a user, I need the web app to be accessible all over the world	I can access the web app	Medium	Sprint-3

	Deployment of AI model in the cloud	USN-4	As a user, I need the AI model to be accessible all over the world	I can access the model deployed in the IBM cloud	Medium	Sprint-3
	Model Building	USN-5	As a user, I need an AI model which could classify or recognize the hand gestures	I can get the prediction from the AI model	Medium	Sprint-1
	User Interface Building	USN-6	As a user, I need a web app for human computer interaction	I get User Interface for interaction with the model	Medium	Sprint-2

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	Model Building	USN-5	As a user, I need an AI model which could classify or recognize the hand gestures	10	Medium	Ajaykrishnan A Lokeswaran M Hari Prasath M Ramesh S Sathish kumar M
Sprint-2	User Interface Building	USN-6	As a user, I need a web app for human computer interaction	10	Medium	Ajaykrishnan A Lokeswaran M Hari Prasath M Ramesh S Sathish kumar M
Sprint-3	Deployment of webapp in the cloud	USN-3	As a user, I need the webapp to be accessible all over the world	10	Medium	Ajaykrishnan A Lokeswaran M Hari Prasath M Ramesh S Sathish kumar M
	Deployment of AI model in the cloud	USN-4	As a user, I need the AI model to be accessible all over the world	10	Medium	Ajaykrishnan A Lokeswaran M Hari Prasath M Ramesh S Sathish kumar M

Sprint-4	Launch Web App deployed in cloud	USN-1	As a user, I can launch the webapp where I can upload the images for recognition	20	High	Ajaykrishnan A Lokeswaran M Hari Prasath M Ramesh S Sathish kumar M
Sprint-4	Prediction	USN-2	As a user, I can get the predicted results from the model deployed in the cloud	20	High	Ajaykrishnan A Lokeswaran M Hari Prasath M Ramesh S Sathish kumar M

6.2 Sprint Delivery Schedule

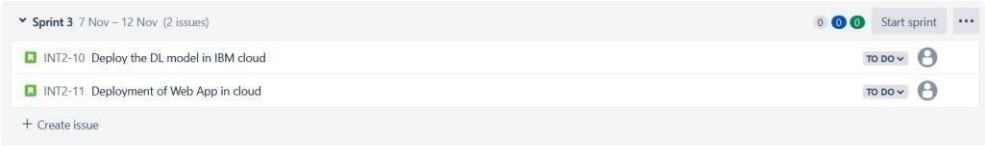
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022
Sprint-2	10	6 Days	31 Oct 2022	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022
Sprint-4	40	6 Days	14 Nov 2022	19 Nov 2022

6.3 Reports from JIRA

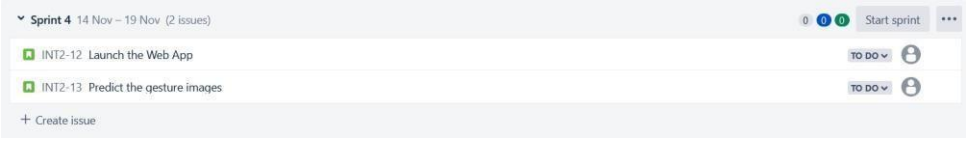
Sprint 1

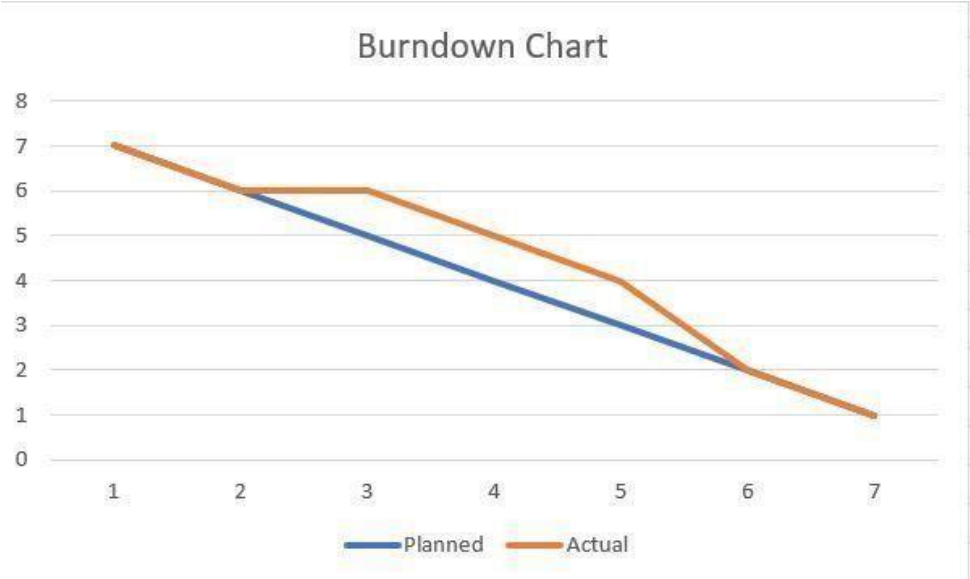
Sprint 2

Sprint 3



Sprint 4





Coding & Solutioning

7.1 Feature 1(Building Hand Gesture Recognizing Model)

Run the model on IBM

```
# Data Collection
from keras.preprocessing.image import ImageDataGenerator

## Configure ImageDataGenerator class
train_datagen =
ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,hor
izontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)

## Importing the Dataset
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def__iter__(self): return 0

# The following code accesses a file in your IBM Cloud Object
Storage. It includes your credentials.
# You might want to remove those credentials before you share the
notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id=IBM_API_KEY,
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-
storage.appdomain.cloud')

bucket =BUCKET_ID
object_key = 'Dataset-IBM.zip'

streaming_body_1 = cos_client.get_object(Bucket=bucket,
Key=object_key)['Body']
```

```
## Unzipping the Dataset
from io import BytesIO
import zipfile
unzip = zipfile.ZipFile(BytesIO(streaming_body_1.read()),'r')
file_paths = unzip.namelist()
for path in file_paths:
    unzip.extract(path)
pwd
filenames  =  os.listdir('/home/wsuser/work/Dataset/train')

## Apply ImageDataGenerator functionality to Trainset and Testset
x_train =
train_datagen.flow_from_directory('/home/wsuser/work/Dataset/train/'
, target_size=(64, 64), batch_size=5, color_mode='grayscale',
class_mode='categorical')
x_test =
test_datagen.flow_from_directory('/home/wsuser/work/Dataset/test/',
target_size=(64, 64), batch_size=5, color_mode='grayscale',
class_mode='categorical')

# Model building
## Importing the Model Building Libraries
import numpy as np
import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator

## Initializing the Model
classifier = Sequential()

### Adding CNN Layers
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 1)
,activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
classifier.add(Conv2D(32, (3, 3), activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
classifier.add(Flatten())
```

```
### Adding Dense Layers
classifier.add(Dense(units=128, activation='relu'))
classifier.add(Dense(units=6, activation='softmax'))
classifier.summary()

## Configure The Learning Process
classifier.compile(optimizer='adam',
loss='categorical_crossentropy', metrics=['accuracy'])
classifier.fit_generator(
    generator=x_train,  steps_per_epoch=len(x_train),
    epochs=20,  validation_data=x_test,  validation_steps=len(x_test)
)

## Save the Model
classifier.save('gesture.h5')
!tar -zcvf gesture.tgz gesture.h5
model_json = classifier.to_json()
with open("model-bw.json", "w") as f:
    f.write(model_json)

## Test the model
from tensorflow.keras.models import load_model
from keras.preprocessing import image
model = load_model("gesture.h5")
from tensorflow.keras.utils import load_img, img_to_array

img = load_img("/home/wsuser/work/Dataset/test/0/0.jpg",
grayscale=True, target_size=(64, 64))
x = img_to_array(img)
x = np.expand_dims(x,axis=0)
pred = model.predict(x)

pred
pred[0]

## Model on IBM
ls -1
!pip install watson-machine-learning-client --upgrade
from ibm_watson_machine_learning import APIClient
wml_credentials = {
```

```
    "url": URL,
    "apikey":  API_KEY
}
client = APIClient(wml_credentials)

def guid_from_space_name(client,space_name):
    space = client.spaces.get_details()
    return(next(item for item in space['resources']

if  item['entity']['name']==space_name)['metadata']['id'])

space_uid = guid_from_space_name(client, 'A Gesture Base Tool for
Sterile Browsing of Radiology Images - PNT2022TMID53035')
print("Space UID = "+space_uid)

client.set.default_space(space_uid)
client.software_specifications.list(limit  =  100)

software_spec_uid =
client.software_specifications.get_uid_by_name("tensorflow_rt22.1-
py3.9")

model_details =
client.repository.store_model(model="gesture.tgz",meta_props={
    client.repository.ModelMetaNames.NAME:"CNN",
    client.repository.ModelMetaNames.TYPE:"tensorflow_rt22.1",

client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid
})

model_id = client.repository.get_model_uid(model_details)
model_id
client.repository.download(model_id,'my_model.tar.gz')
```

Access the model on IBM

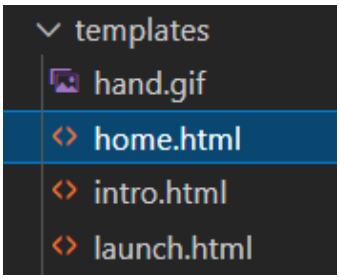
```
## Importing IBM Watson Machine Learning Library
from ibm_watson_machine_learning import APIClient
```

```
## Connecting to the model deployed in IBM Cloud with Credentials
wml_credentials = {
    "url": URL,
    "apikey":  API_KEY
}
client = APIClient(wml_credentials)

def guid_from_space_name(client,space_name):
    space = client.spaces.get_details()
    return(next(item for item in space['resources']

if  item['entity']['name']==space_name)['metadata']['id'])

space_uid = guid_from_space_name(client, 'A Gesture Base Tool for
Sterile Browsing of Radiology Images - PNT2022TMID53035')
print("Space UID = "+space_uid)

client.set.default_space(space_uid)

## Downloading model from IBM Cloud
client.repository.download(BUCKET_ID,'my_model.tar.gz')
```

7.2 Feature 2(Web Application Building)

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.



Home.html

```
<!DOCTYPE html>
<html>
  <head>
        <script  src="https://cdn.tailwindcss.com"></script>
  </head>
  <body
    class="grid h-screen place-items-center text-center bg-gradient-
to-r from-lime-100 to-cyan-100">
    <div class="p-8 rounded-xl bg-white shadow-lg flex flex-col gap-
6 w-3/4 justify-center items-center">
      <header>
        <h1 class="bg-clip-text text-transparent bg-gradient-to-r
from-lime-600 to-cyan-600 text-5xl font-bold p-3">
```

```
          <a href="/home"> A Gesture Based Tool For Sterile Browsing
of Radiology Images </a>
        </h1>
        <div class="mt-4 flex gap-4 justify-center">
          <a href="/home">
            <button class="p-2 bg-blue-500 hover:bg-green-700 text-
white rounded-lg">
              Home
            </button>
          </a>
          <a href="/intro">
            <button class="p-2 bg-green-500 hover:bg-green-700 text-
white rounded-lg">
              Introduction
            </button>
          </a>
          <a href="/launch">
            <button class="p-2 bg-green-500 hover:bg-green-700 text-
white rounded-lg">
              Launch
            </button>
          </a>
        </div>
      </header>
      <img src="{{url_for('static', filename='gesture1.jpg')}}"
width='50%' height='50%' style="align-item:center"/>
      <h1 class="text-5xl bg-gradient-to-r from-red-600 to-orange-
600 bg-clip-text text-transparent font-bold">
        Welcome!
      </h1>
    </div>
  </body>
</html>
```

Intro.html

```
<!DOCTYPE html>
<html>
  <head>
    <script  src="https://cdn.tailwindcss.com"></script>
  </head>
  <body class="grid h-screen place-items-center text-center bg-
gradient-to-r from-lime-100 to-cyan-100">
    <div class="p-8 rounded-xl bg-white shadow-lg flex flex-col gap-
6 w-3/4 justify-center items-center">
      <header>
        <h1 class="bg-clip-text text-transparent bg-gradient-to-r
from-lime-600 to-cyan-600 text-5xl font-bold p-3">
          <a href="/home"> A Gesture Based Tool For Sterile Browsing
of Radiology Images </a>
        </h1>
        <div class="mt-4 flex gap-4 justify-center">
          <a href="/home">
            <button class="p-2 bg-green-500 hover:bg-green-700 text-
white rounded-lg">
              Home

            </button>
          </a>
          <a href="/intro">
            <button class="p-2 bg-blue-500 hover:bg-green-700 text-
white rounded-lg">
              Introduction
            </button>
          </a>
          <a href="/launch">
            <button class="p-2 bg-green-500 hover:bg-green-700 text-
white rounded-lg">
              Launch
            </button>
          </a>
    </div>
```

```
      </header>
      <img src="{{url_for('static', filename='slnarc.png')}}"
width='70%' height='70%' style="align-item:center"/>
      <p>
        Hand Gesture recognition system provides us an innovative,
natural, user friendly way of interaction with the computer which is
more familiar to the human beings.
        In our project, the hand region is extracted from the
background by using Region of interest.Then, we will be
predicting the labels based on the CNN trained model weights
of hand gestures using that predicted labelswe apply if
conditions to control some of the actions like reshaping,
blur,flip of the given image.
      </p>
    </div>
  </body>
</html>
```

Launch.html

```
<!DOCTYPE html>
<html>
  <head>
      <script  src="https://cdn.tailwindcss.com"></script>
  </head>
  <body class="grid h-screen place-items-center text-center bg-
gradient-to-r from-lime-100 to-cyan-100">
    <div class="p-8 rounded-xl bg-white shadow-lg flex flex-col gap-
6 w-3/4">
      <header>
        <h1 class="text-5xl bg-gradient-to-r from-lime-600 to-cyan-
600 bg-clip-text text-transparent font-bold p-3">
          <a href="/home"> A Gesture Based Tool For Sterile Browsing
of Radiology Images </a>
        </h1>
        <div class="mt-4 flex gap-4 justify-center">
          <a href="/home">
            <button class="p-2 bg-green-500 hover:bg-green-700 text-
white rounded-lg">
              Home
            </button>
          </a>
          <a href="/intro">
            <button class="p-2 bg-green-500 hover:bg-green-700 text-
white rounded-lg">
              Introduction
            </button>
          </a>
          <a href="/launch">
```

```
            <button class="p-2 bg-blue-500 hover:bg-green-700 text-
white rounded-lg">
              Launch
            </button>
          </a>
        </div>
      </header>

      <h3>Provide the image to perform the operations</h3>

      <form class="flex flex-col gap-2 bg-green-50 rounded-lg p-4
items-center"  action="predict" method="post"
        enctype="multipart/form-data">
        <h2>Upload Image Here</h2>
        <input class="bg-green-200 p-2 rounded-lg" type="file"
name="image" />
        <button class="w-1/3 bg-green-500 hover:bg-green-600 p-2
text-white rounded-lg" type="submit">
          Submit
        </button>
      </form>
    </div>
  </body>
</html>
```

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.

app.py

```
from flask import Flask,render_template,request
import numpy as np
import os
import operator
import cv2
from tensorflow.keras.models import load_model
from tensorflow.keras.utils import load_img, img_to_array
from werkzeug.utils import secure_filename

app = Flask(__name__,template_folder="templates")
model=load_model('../Cloud Model/gesture.h5')
print("Model is loaded from local system")
```

```
@app.route("/")
def root():
   return  render_template("home.html")

@app.route("/home")
def home():
   return  render_template("home.html")

@app.route("/intro")
def intro():
   return  render_template("intro.html")

@app.route("/launch")
def launch():
   return  render_template("launch.html")


@app.route('/index',methods=['GET','POST'])
def index():
    return  render_template("launch.html")

@app.route('/predict',methods=['GET','POST'])
def predict():
    #Getting input and storing it
    if request.method == 'POST':
        print('inside launch function')
        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('img saved successfully')
        print(file_path)

    cap=cv2.VideoCapture(0)
    image1=cv2.imread(file_path)
    cv2.imshow("Output",image1)
```

```
    prev='NULL'
    while True:
        _, frame=cap.read()
        frame=cv2.flip(frame,1)

        x1=int(0.5*frame.shape[1])
        y1=10
        x2=frame.shape[1]-10
        y2=int(0.5*frame.shape[1])

        cv2.rectangle(frame,(x1-1,y1-1),(x2+1,y2+1),(255,0,0)),1
        roi = frame[y1:y2,x1:x2]

        roi=cv2.resize(roi,(64,64))
        roi=cv2.cvtColor(roi,cv2.COLOR_BGR2GRAY)
        _,   test_image=cv2.threshold(roi,120,255,cv2.THRESH_BINARY)

        result = model.predict(test_image.reshape(1,64,64,1))
        print(result)
        prediction =
{'ZERO':result[0][0],'ONE':result[0][1],'TWO':result[0][2],'THREE':r
esult[0][3],'FOUR':result[0][4],'FIVE':result[0][5]}

prediction=sorted(prediction.items(),key=operator.itemgetter(1),reve
rse=True)

        cv2.putText(frame,prediction[0][0],(10,120),
cv2.FONT_HERSHEY_PLAIN,1,(0,255,255),1)
        cv2.imshow("frame",frame)


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

        if prev == prediction:
                continue

        prev = prediction
```

```
image1=cv2.imread(file_path)
if prediction[0][0]=='ONE':
    resized=cv2.resize(image1,(200,200))
    cv2.destroyWindow("Output")
    cv2.imshow("Output",resized)

elif prediction[0][0]=='ZERO':
    cv2.rectangle(image1,(480,170),(650,420),(0,0,255),2)
    cv2.destroyWindow("Output")
    cv2.imshow("Output",image1)
    #cv2.imshow("Rectangle  -  Zero",image1)
    #cv2.waitKey(0)

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.destroyWindow("Output")
    cv2.imshow("Output",rotated)

elif prediction[0][0]=='THREE':
    blurred=cv2.GaussianBlur(image1,(11,11),0)
    cv2.destroyWindow("Output")
    cv2.imshow("Output",blurred)

elif prediction[0][0]=='FOUR':
    zoomed=cv2.resize(image1,(400,400))
    cv2.destroyWindow("Output")
    cv2.imshow("Output",zoomed)

elif prediction[0][0]=='FIVE':
    neg=255-image1
    cv2.destroyWindow("Output")
    cv2.imshow("Output",neg)

else:
    continue
```

```
        cap.release()
        cv2.destroyAllWindows()

        return render_template("home.html")

    if_name__== "_main":

        app.run(debug=True)
```

Importing Libraries

```
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
```

- Creating our flask application and loading our model

```
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 the html Page

```
@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")
```

The above three route are used to render the home, introduction and the index html pages.

```
@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.

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.

```
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.

```
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.

```
# 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)
```

Predicting our results

After placing the ROI and getting the frames from the web cam now its time to predict the gesture result using the model which we trained and stored it into a variable for the further operations.

```
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.

```
#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")
```

```
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")
```

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
HomePage_TC_OO1	UI	Home Page	Verify user is able to see the home page for our gesture based tool for sterile browsing of radiology images	-	1.Enter URL and click go 2. Check whether home page can be seen	http://127.0.0.1:5000	The homepage should be displayed	Working as expected	Pass		N		Anandh Krushna S K
HomePage_TC_OO2	UI	Home Page	Verify whether the navigation bar is displayed	-	1.Enter URL and click go 2. Check whether the navigation bar is displayed in the homepage	http://127.0.0.1:5000	The navigation bar should be displayed in the homepage	Working as expected	Pass		N		Anandh Krushna S K
IntroPage_TC_001	UI	Intoduction Page	Verify user is able to see the introduction page for our gesture based tool for sterile browsing of radiology images	-	1.Enter URL and click go 2. Check whether intoduction page can be seen	http://127.0.0.1:5000/intro	The Introduction should be displayed	Working as expected	Pass		N		Anandh Krushna S K
LaunchPage_TC_001	UI	Launch Page	Verify whether ther user is able to access the launch page	-	1.Enter URL and click go 2. Check whether launch page can be seen	http://127.0.0.1:5000/launch	The launch page should be displayed	Working as expected	Pass		N		Jayakrishnan S V
LaunchPage_TC_002	UI	Launch Page	Verify if the user could see the utility for uploading the rediology image	-	1.Enter URL and click go 2. Check whether the launch page upload image utility can be seen	http://127.0.0.1:5000/launch	The utility to upload image should be displayed	Working as expected	Pass		N		Jayakrishnan S V
LaunchPage_TC_003	Functional	Launch Page	Verify if the user can upload the image from their local system	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated	http://127.0.0.1:5000/launch Upload the image to be manipulated	The name of the image uploaded should be displayed	Working as expected	Pass		N		Jayakrishnan S V
LaunchPage_TC_004	Functional	Launch Page	Verify if the user can submit	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter	The web-camera starts	Working as expected	Pass		N		Deepananth K
LaunchPage_TC_005	Functional	Launch Page	Verify if the image uploaded is accessible in the backend	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter	The image is stored in a folder in the backend	Working as expected	Pass		N		Deepananth K
LaunchPage_TC_006	Functional	Launch Page	Verify if the web cam starts	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter	The webcam starts	Working as expected	Pass		N		Deepananth K
LaunchPage_TC_007	Functional	Launch Page	Verify if the region of interest can be found for the gesture	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit 5. Show gesture in the webcam	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter Show Gesture 0 in the webcam	The gesture shown in camera should be captured ad the region of interest should be found	Working as expected	Pass		Y		Aravind Kannan Rathinasabapathi
LaunchPage_TC_008	Functional	Launch Page	Verify if the gesture 0 shown in camera is recogized	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit 5. Show gesture in the webcam	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter Show Gesture 0 in the webcam	The uploaded image should be resized and should be shown	Working as expected	Pass		Y		Aravind Kannan Rathinasabapathi
LaunchPage_TC_008	Functional	Launch Page	Verify if the gesture 0 shown in camera is recogized	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit 5. Show gesture in the webcam	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter Show Gesture 0 in the webcam	The uploaded image should be resized and should be shown	Working as expected	Pass		Y		Aravind Kannan Rathinasabapathi
LaunchPage_TC_009	Functional	Launch Page	Verify if the gesture 1 shown in camera is recogized	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit 5. Show gesture in the webcam	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter Show Gesture 1 in the webcam	The uploaded image should be resized as rectangle and should be shown	Working as expected	Pass		Y		Anandh Krushna S K
LaunchPage_TC_010	Functional	Launch Page	Verify if the gesture 2 shown in camera is recogized	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit 5. Show gesture in the webcam	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter Show Gesture 2 in the webcam	The uploaded image should be rotated 45 degrees and should be shown	Working as expected	Pass		Y		Aravind Kannan Rathinasabapathi

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnents	TC for Automation(Y/N)	BUG ID	Executed By
LaunchPage_TC_011	Functional	Launch Page	Verify if the gesture 3 shown in camera is recogized	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit 5. Show gesture in the webcam	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter Show Gesture 3 in the webcam	The uploaded image should be blurred and should be shown	Working as expected	Pass		Y		Jayakrishnan S V
LaunchPage_TC_012	Functional	Launch Page	Verify if the gesture 4 shown in camera is recogized	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit 5. Show gesture in the webcam	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter Show Gesture 4 in the webcam	The uploaded image should be zoomed and should be shown	Working as expected	Pass		Y		Aravind Kannan Rathinasabapathi
LaunchPage_TC_013	Functional	Launch Page	Verify if the gesture 5 shown in camera is recogized	-	1.Enter URL 2.Click on upload image utility 3. Choose the image thet needs to be manipulated 4. Click Submit 5. Show gesture in the webcam	http://127.0.0.1:5000/launch Upload the image to be manipulated Click Enter Show Gesture 5 in the webcam	The uploaded image should be negated and should be shown	Working as expected	Pass		Y		Deepananth K

8.1 User Acceptance Testing

Defect Analysis

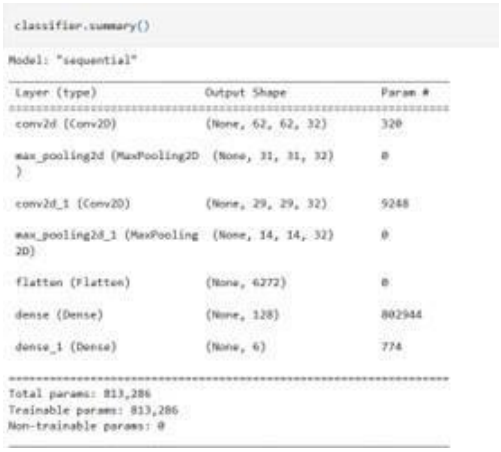
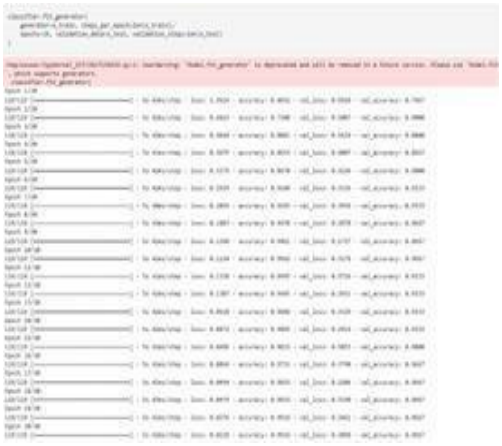
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	3	2	1	0	6
Duplicate	1	0	0	0	1
External	2	3	0	2	7
Fixed	3	2	1	0	6
Not Reproduced	0	0	1	1	2
Skipped	0	0	0	1	1
Won't Fix	0	2	1	0	3
Totals	9	9	4	4	26

Test Case Analysis

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	6	0	0	6
Client Application	16	0	0	16
Outsource Shipping	3	0	0	3
Exception Reporting	2	0	0	2
Final Report Output	7	0	0	7
Version Control	2	0	0	2

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

Run The Application

At last, we will run our flask application

```
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
```

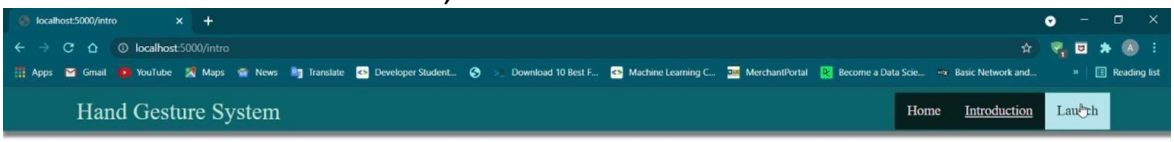
Then it will run on localhost:5000

```
* 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.

Let’s see how our home.html page looks like:

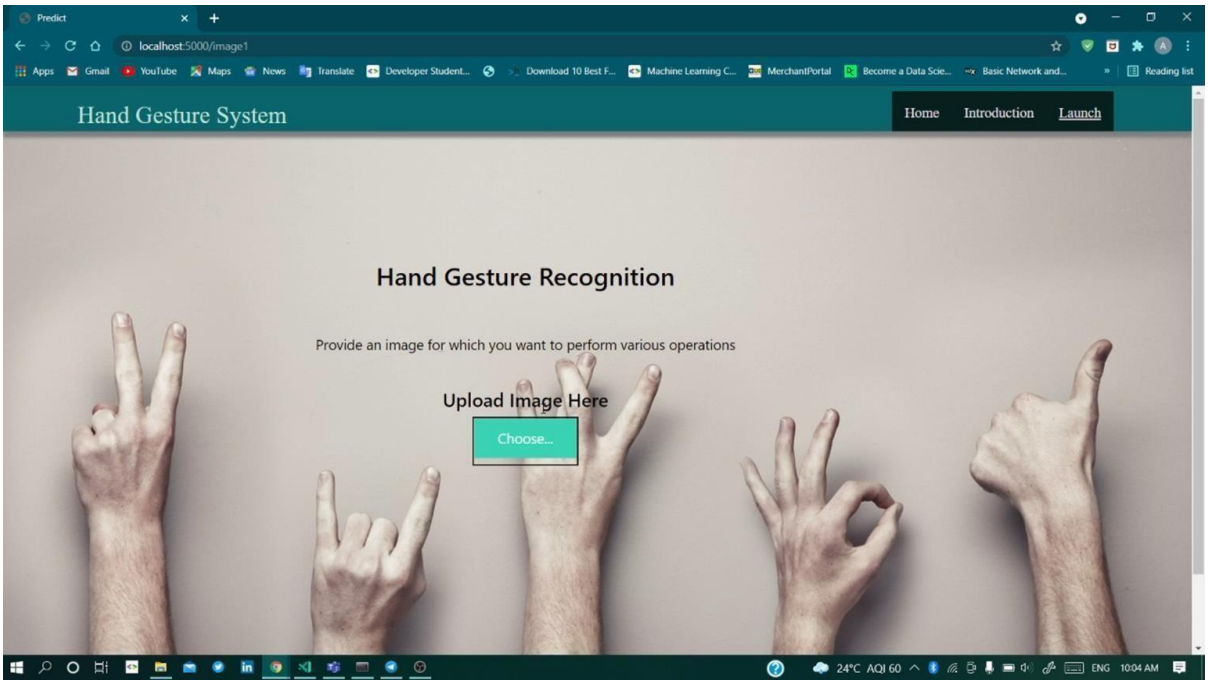
When “Info” button is clicked, localhost redirects to “intro.html”



INTRODUCTION

Hand Gesture recognition system provides us an innovative, natural, user friendly way of interaction with the computer which is more familiar to the human beings. In our project, the hand region is extracted from the background by using Region of intrest. Then, we will be predicting the labels based on the CNN trained model weights of hand gestures using that predicted labels we apply if conditions to control some of the actions like reshaping , blur, flip of the given image.

Upload the image and click on Predict button to view the result



Advantages & Disadvantage

Advantages

1) Gesture based tool for sterile browsing of radiology images helps to recognize the hand gestures even with gloves.
2) As there is no contact with computers, it prevents infections from contact sources like the keyboard and/or other input devices.
3) Doctors can use this system in the operation theater with ease while engaged in other surgical procedures.
4) Easy-to-use, intuitive user interface making it easy to utilize the tool at once even for a layman
5) Real time gesture recognition and image manipulation makes the system smooth to use

Disadvantages

1) Requires computer infrastructure to be present in the operating room
2) All the radiological images must be present locally in the computer to be manipulated
3) Prerequisite of gesture to action mapping must be known to utilize the tool acts as an overhead
4) Should have clear background to distinguish the region of interest to identify the gesture
5) Limited number of hand gestures, so not able provide a lot of options to the target user

Conclusion

Gesture based tool for sterile browsing of radiology images is a revolutionary tool, beneficial to the medical community to multi-task and perform surgical procedures without any hassle. The unique selling point of project is that it is a real time tool, so that it avoids any unnecessary delays opposed to the current system in place. Integration of the tool in the operating theater is the target use case of the project, benefitting both the doctor and patient. It is useful to the doctor as it helps him perform the surgical procedure flawlessly and the patient as it saves him from his illness. Say goodbye to mistakes, with the help of this project.

Future Scope

- Move the camera access facility from the local desktop to the web browser to make the application deployable in a cloud environment.
- This project can be deployed as a web application to the cloud and sold as a multi-tenant Software-as-a-Service product.
- The model can be improved to handle different backgrounds and detect multiple hands and choose the hand of interest to manipulate the image.
- Model can be re-trained using the images from the video feed, if stored, to improve the efficiency and accuracy of the application.
- Retraining the image can happen at regular intervals by creating a cronjob to do the same, resulting in an iteratively improved model.

Appendix

GitHub & Project Demo Link

GitHub Link:
source code
https://github.com/IBM-EPBL/IBM-Project-18952-1659691551

Demonstration link:

https://drive.google.com/file/d/1wD4toogEmfyvOaQsfK5GpjuoDRYHHCA_/view?usp=share_link