Industrial Training Report

On

Machine Learning With Python Programming
(Proctored Examination System)

A training report submitted in partial fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY
in
ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

Name	Sreeparna Ray
University. Roll No.	10800319054
Registration No.	038430

Department of
ELECTRONICS AND COMMUNICATION ENGINEERING
ASANSOL ENGINEERING COLLEGE
Affiliated To
MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, WEST BENGAL

OCTOBER 2022

DEPARTMENT OF
ELECTRONICS AND COMMUNICATION ENGINEERING
ASANSOL ENGINEERING COLLEGE
Vivekananda Sarani, Kanyapur, Asansol, West Bengal – 713305

Certificate of Recommendation

It is hereby recommended that the final year **Training Report** entitled **Proctored Examination System** carried out by the following student for the partial fulfillment of the degree of Bachelor of Technology in Electronics & Communication Engineering may be accepted for the evaluation of **Industrial Training (EC 781)** of "**Bachelor of Technology in Electronics and Communication Engineering**" of **Asansol Engineering College** under **Maulana Abul Kalam Azad University of Technology, West Bengal (MAKAUT)**.

Name	**Univ. Roll No.**
Sreeparna Ray	10800319054

Countersigned:

……	……
Mr. Amit Kumar Rai, AP, ECE	**Prof. (Dr.) Arunava De**
(Training & Placement Coordinator)	(Head of the Department)
ECE Department	ECE Department
Asansol Engineering College	Asansol Engineering College
Asansol - 713305	Asansol - 713305

Acknowledgement

The successful completion of this training mark the beginning of an ever - going learning experience of converting ideas and concepts into real life, practical system. This training was a quite a learning experience for me at each step. At the same time, it has given me confidence to work in professional setup. I feel the experience gained during the training will lead me to gain the bright prospect in the future.

First, I would like to give thanks to our Training & Placement Co-Ordinator, **Mr. Amit Kumar Rai**, for giving me the opportunity to do the training, which not only has increased our awareness about latest fields but also taught me the importance of team building.

With the deep sense of gratitude, I would like to express my sincere gratitude towards **Prof. (Dr.) Arunava De**, **Head of the Department of ECE, Asansol Engineering College** for providing us the requisite support for time completion of our work. I would also like pay my heartiest thanks and gratitude to all the teachers of the **Department of ECE, Asansol Engineering College** for various suggestions being provided in attaining success during the training.

I would like to express my earnest thanks to my other colleagues along with all technical staffs of the **Department of ECE, Asansol Engineering College** for their valuable assistance being provided during my project work.

Finally, I would like to express my deep sense of gratitude to my parents for their constant motivation and support throughout this training.

……

Name: Sreeparna Ray

Roll No: 10800319054

Table of Contents

Introduction :

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values. It is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, agriculture, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks. A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers, but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. Some implementations of machine learning use data and neural networks in a way that mimics the working of a biological brain. In its application across business problems, machine learning is also referred to as predictive analytics.

Recommendation engines are a common use case for machine learning. Other popular uses include fraud detection, spam filtering, malware threat detection, business process automation (BPA) and Predictive maintenance.

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data . While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars).

Deep learning vs. machine learning

If deep learning is a subset of machine learning, how do they differ? Deep learning distinguishes itself from classical machine learning by the type of data that it works with and the methods in which it learns.

Machine learning algorithms leverage structured, labeled data to make predictions—meaning that specific features are defined from the input data for the model and organized into tables. This doesn't necessarily mean that it doesn't use unstructured data; it just means that if it does, it generally goes through some pre-processing to organize it into a structured format.

Deep learning eliminates some of data pre-processing that is typically involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and it automates feature extraction, removing some of the dependency on human experts. For example, let's say that we had a set of photos of different pets, and we wanted to categorize by "cat", "dog", "hamster", et cetera. Deep learning algorithms can determine which features (e.g. ears) are most important to distinguish each animal from another. In machine learning, this hierarchy of features is established manually by a human expert.

Then, through the processes of gradient descent and backpropagation, the deep learning algorithm adjusts and fits itself for accuracy, allowing it to make predictions about a new photo of an animal with increased precision.

Machine learning and deep learning models are capable of different types of learning as well, which are usually categorized as supervised learning, unsupervised learning, and reinforcement learning. Supervised learning utilizes labeled datasets to categorize or make predictions; this requires some kind of human intervention to label input data correctly. In contrast, unsupervised learning doesn't require labeled datasets, and instead, it detects patterns in the data, clustering them by any distinguishing characteristics. Reinforcement learning is a process in which a model learns to become more accurate for performing an action in an environment based on feedback in order to maximize the reward.

How deep learning works

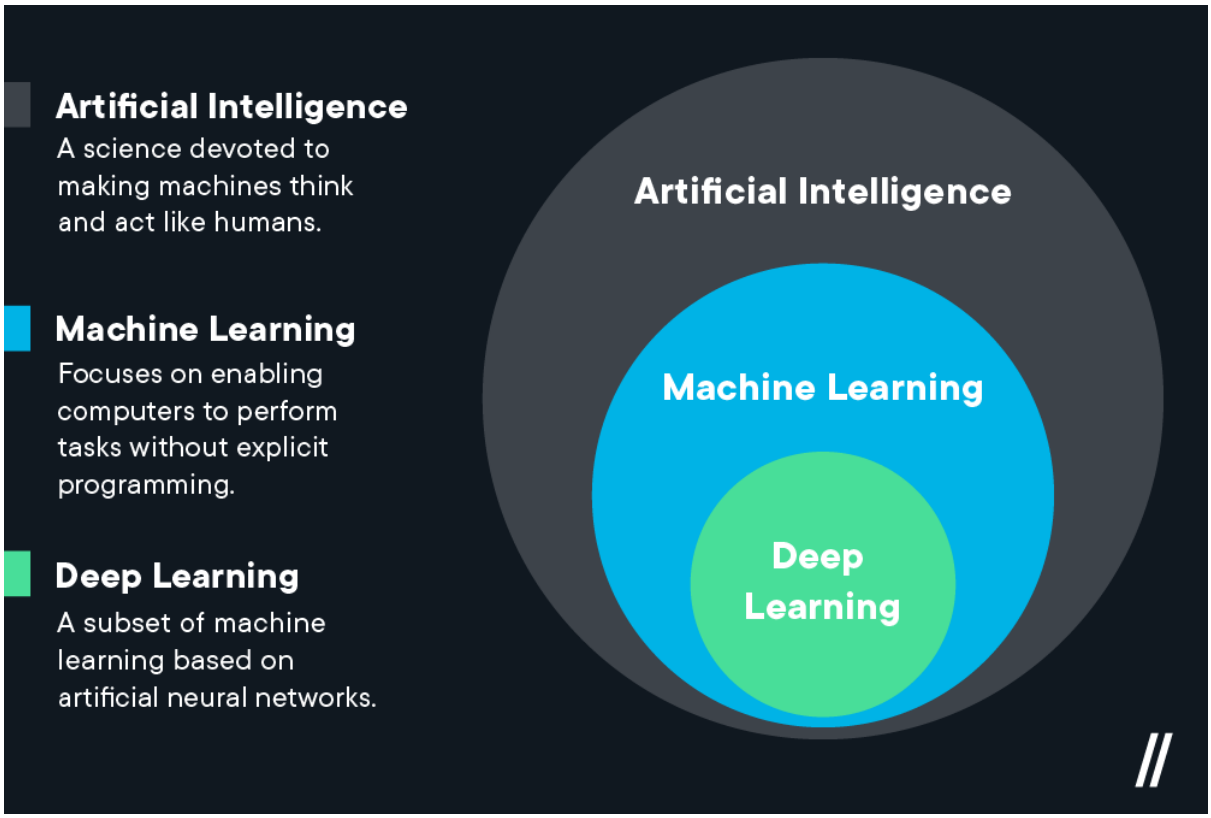
Deep learning neural networks, or artificial neural networks, attempts to mimic the human brain through a combination of data inputs, weights, and bias. These elements work together to accurately recognize, classify, and describe objects within the data.

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This progression of

computations through the network is called forward propagation. The input and output layers of a deep neural network are called *visible* layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.

Another process called backpropagation uses algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model. Together, forward propagation and backpropagation allow a neural network to make predictions and correct for any errors accordingly. Over time, the algorithm becomes gradually more accurate.

The above describes the simplest type of deep neural network in the simplest terms. However, deep learning algorithms are incredibly complex, and there are different types of neural networks to address specific problems or datasets.



Training Objective :

The main objective of this training is to make students comfortable with tools and techniques required in handling large amounts of datasets. We will also cover various deep learning methods in NLP, Neural Networks etc. Several libraries and datasets publicly available will be used to illustrate the application of these algorithms. This will help students in developing skills required to gain experience of doing independent research and study.

As corona virus hit our everyday life we were made to stay at home in lockdown, and everyday work was going online and work from home. From shopping online to ordering food , the whole industry was running from home in online mode.

As a matter of fact examinations that were previously held offline was also being taken in online mode from computers.

But in online mode examinations can't be invigilated properly due to absence of teachers or invigilators.

So The concept is to make a examination system for Online application, where it can be invigilated using AI (Deep learning model) for any kind of malpractice. In the web application system the web-cam of the client system will take pictures with a certain interval of time. Those pictures will be sent to the backend server for verification and detection if the pictures contains a human facing forward on his/her computer screen or not.

It can detect things such as :

1. No presence of human.
2. Fake Image of a human.
3. Can classify between real Human from Human like objects.

Feasibility Study :

Economic Feasibility:

Economic analysis is most frequently used for evaluation of the effectiveness of the system. More commonly known as cost/benefit analysis the procedure is to determine the benefit and saving that is expected from a system and compare them with costs, decisions is made to design and implement the system. This part of feasibility study gives the top management the economic justification for the new system .This is an important input to the management the management, because very often the top management does not like to get confounded by the various technicalities that bound to be associated with a project of this kind. A simple economic analysis that gives the actual comparison of costs and benefits is much more meaningful in such cases. In the system, the organization is most satisfied by economic feasibility .Because, if the organization implements this system, it need not require any additional hardware resources as well as it will be saving lot of time.

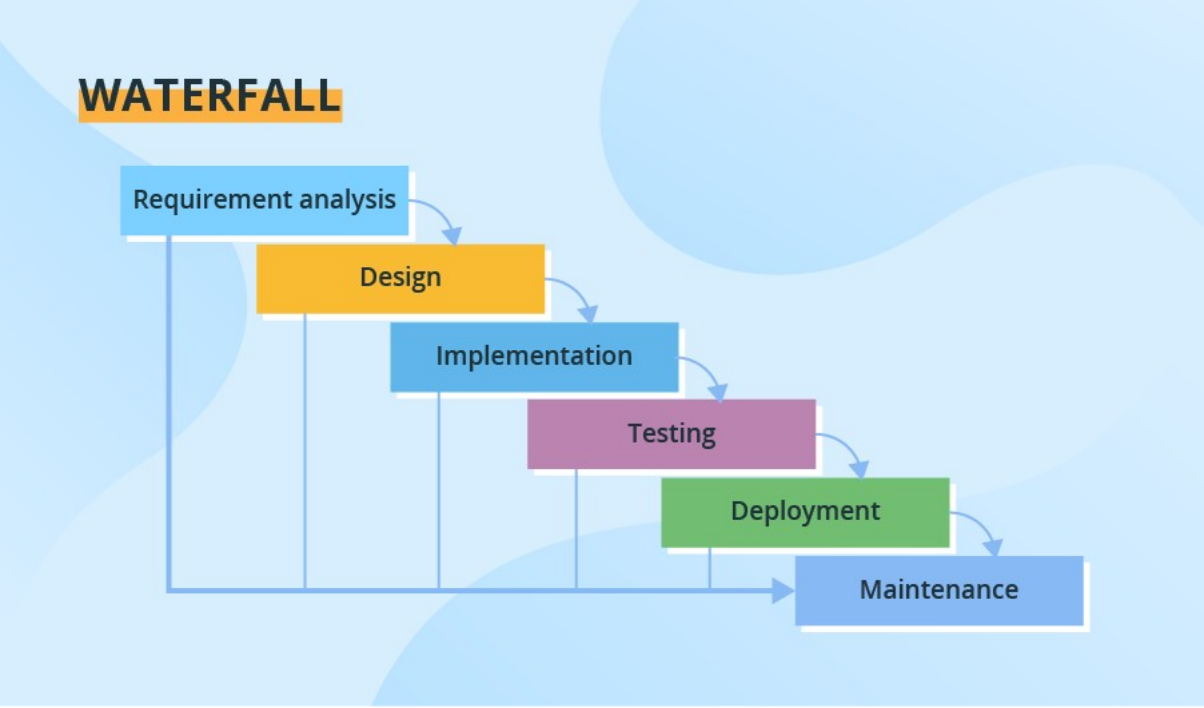
Technical Feasibility:

Technical feasibility centres on the existing manual system of the test management process and to what extent it can support the system. According to feasibility analysis procedure the technical feasibility of the system is analysed and the technical requirements such as software facilities, procedure, inputs are identified. It is also one of the important phases of the system development activities. The system offers greater levels of user friendliness combined with greater processing speed. Therefore, the cost of maintenance can be reduced. Since, processing speed is very high and the work is reduced in the maintenance point of view management convince that the project is operationally feasible.

Behavioural Feasibility:

People are inherently resistant to change and computer has been known to facilitate changes. An estimate should be made of how strong the user is likely to move towards the development of computerized system. These are various levels of users in order to ensure proper authentication and authorization and security of sensitive data of the organization.

Software Engineering paradigm applied :



The waterfall model depicts the software development process in a linear sequential flow; due to this, it is also referred to as a linear-sequential life cycle model, which indicates that any development process steps can start only after the previous one has finished. The stages are always done in this order and never overlap. Before moving on to the next step, the developer must finish the present one. The model is called a waterfall because it progresses from one phase to the next logically.

Requirement Gathering and Analysis: This phase captures all feasible needs for the system to be created and documents them in a requirement specification document. The customer and software developer collaborate to document all software's functionalities and performance. To further grasp the requirements, teams organize brainstorming sessions and get a clear walkthrough of the requirement. These requirements are then processed to perform a detailed analysis on how to proceed. The requirements are assessed for feasibility to ensure that they are testable. It also allows us to decide on the product's hardware or software needs, the planning, developed, and captured along the process.
Example: Based on your analysis on the requirements provided by the client, you may ask questions like does the new reservation system support multiple languages? How many users are expected to be active on the application in one minute ? Etc.

System Design: This is an important stage in the model. It's in charge of analyzing the data you gathered in the first phase to be used in subsequent coding phases. It also aids in the overall architecture of the system design by providing a clear picture of the hardware and software needs. In this phase, the requirement specification is broadly researched and verified. It also aids in transforming the SRS document into the software product's functional design and development. A Software Design Document is used to document all of this effort.
Example: The senior members of the team collaborate to discuss on high level design, low level design and system architecture of the software. This would include discussions on redundant backup and failover capabilities so that the system is accessible at all times.

Implementation: The output of the system design phase is fed into the implementation phase as part of the sequential flow. It is initially divided into small programs known as units used in subsequent testing and implementation phases. Unit testing is a technique that involves developing each unit in the implementation phase and then testing each module in isolation. Following that, these modules are tested by writing additional code to see how the modules interact with each other and with the flow of intermediate output. Because the SDD contains all of the information required by software developers, the implementation or coding phase goes smoothly if this document is complete.
Example: Based on the complexity of the application, the developer incorporate features like security checks, audit logging, etc for smooth functioning of the application.

Integration and Testing: This phase takes the designs and advancements from the previous processes as input. For various tests, they are integrated into a module or system. The testing environment is subjected to a continuous software check to see whether any flows or errors in the design or code exist. Alpha testing (done by the development team), beta testing (performed by a specified group of users), and acceptance testing (performed after delivery by the customers and decides for the acceptance or rejection of the software based on results and feedback) are all examples of system testing.
Example: For the project, testers with reservation domain knowledge were also employed so that they could test the application from a domain standpoint. The application's security was put to the test by security testing teams.

Deployment of the system: After passing all of the tests, the software is installed on the user's PC (released to the market). Installation, migration, and support of the entire system to the user or customer environment are part of the deployment process. This will make the

customers happier, which will reduce maintenance costs, and results will be more accurate as a result of the improved output. The modules' interactions with each other and the system are tested at this phase. While performing this step, you must ensure that the environment is up and running and check that the test exit conditions were fulfilled. It is also critical to perform sanity checks on the environment after the application has been deployed to ensure that it is functioning properly.

Example: To get the reservation system up and operating on the production servers, the teams requires to coordinate with network and IT administrative teams for smooth deployment process.

Maintenance: The core aspect of any product development cycle is providing support to your clients through good maintenance and regular checks. The waterfall model follows the same approach by having maintenance as the last phase. This process occurs shortly after installation and includes making the necessary modifications to the product or system and increasing, changing, or modifying aspects linked to system performance difficulties. In the client environment, some complications arise. Patches are published to address these vulnerabilities. To improve the product, newer versions of the product are produced.

Example: A reservation system consists of various sections that are responsible for performing separate functions like booking feature, payment feature, filtering feature, etc. And its important to keep these sections up to date because any failure in the live application can cause serious issues.

Hardware and Software Requirements : (For building the system)

Hardware -

1. Computer Processor – intel i3-8100 or AMD Ryzen 3 1200 and Above

2. RAM - 8GB DDR4 2400 MHz and above

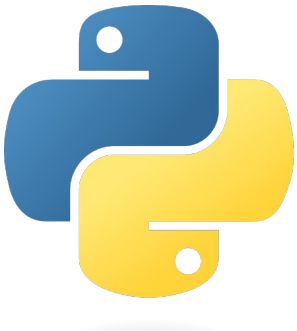
3. GPU – GTX 1050 or AMD RX580 and Above

Software -

1. Programming language – Python 3

2. Libs – Numpy, Pandas, Matplotlib, Tensorflow, Keras, Scikit-image, ImageDataGenerator.

3. Flask, HTML, CSS, JavaScript



Dataset :

We have used a dataset “Human vs Non human” . Where there are pictures of human faces and other category is for non human objects like cat, dog, car, plane etc.

Human Objects

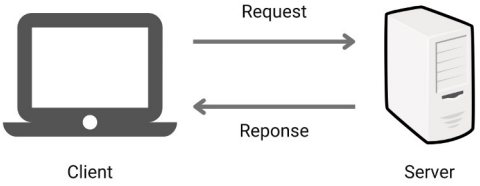



Non–Human Objects





System Architecture :



The software system uses client-server architecture for communicating with backend server. And neural network for building the deep learning model.

Application workflow :

1. The Source code for neural network model has been developed in google colab, as It is easier to find libs available.

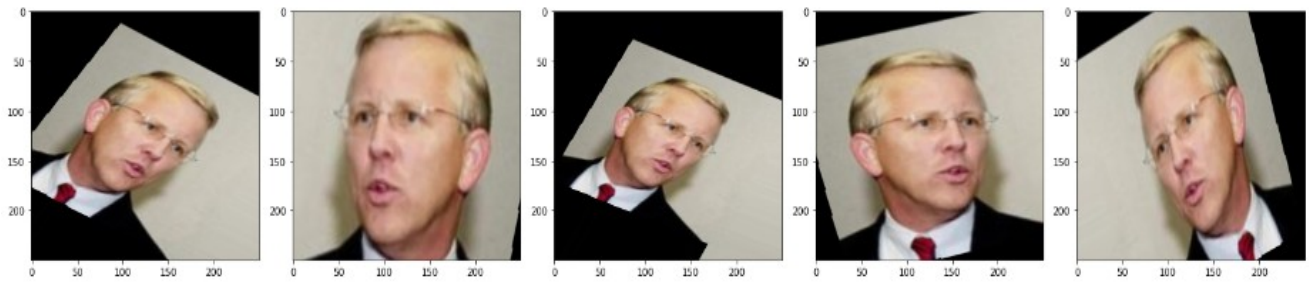
2. The dataset has been uploaded to a google drive and afterwards imported to google colab, and accordingly extracted.

```
from google.colab import drive
drive.mount('/content/gdrive',force_remount=True)
```

```
Mounted at /content/gdrive
```

```
!unzip /content/gdrive/MyDrive/archive.zip
```

3. Now for better results with different types of human position in front of camera, the dataset images has been augmented using ImageDataGenerator from keras. Eg : Rotation, shearing, zoom etc



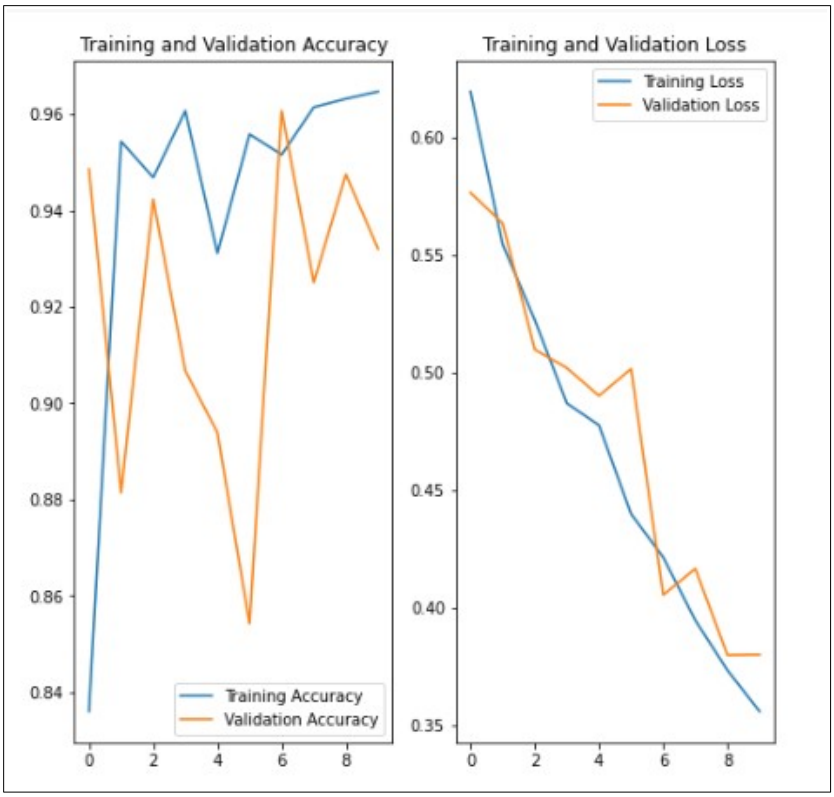
4. The Model has been created using keras 2D convolutional layer with maxpooling for input layer and has a single hidden layer of 512 neurons , and one output layer with two alternatives.

```
model=tf.keras.models.Sequential(layers=[
        tf.keras.layers.Conv2D(filters=32,padding='same',kernel_size=(3,3),activation='relu',input_shape=(Img_shape,Img_shape,3)),
        tf.keras.layers.MaxPooling2D(2),
        tf.keras.layers.Conv2D(filters=64,padding='same',kernel_size=(3,3),activation='relu'),
        tf.keras.layers.MaxPooling2D(2),
        tf.keras.layers.Conv2D(filters=128,padding='same',kernel_size=(3,3),activation='relu'),
        tf.keras.layers.MaxPooling2D(2),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(units=512,activation='softmax'),
        tf.keras.layers.Dense(units=2)
])
```

```
Epoch 1/10
41/41 [==============================] - 125s 3s/step - loss: 0.6196 - accuracy: 0.8360 - val_loss: 0.5766 - val_accuracy: 0.9486
Epoch 2/10
41/41 [==============================] - 115s 3s/step - loss: 0.5547 - accuracy: 0.9543 - val_loss: 0.5636 - val_accuracy: 0.8814
Epoch 3/10
41/41 [==============================] - 115s 3s/step - loss: 0.5225 - accuracy: 0.9469 - val_loss: 0.5099 - val_accuracy: 0.9423
Epoch 4/10
41/41 [==============================] - 114s 3s/step - loss: 0.4871 - accuracy: 0.9607 - val_loss: 0.5021 - val_accuracy: 0.9067
Epoch 5/10
41/41 [==============================] - 114s 3s/step - loss: 0.4777 - accuracy: 0.9311 - val_loss: 0.4903 - val_accuracy: 0.8939
Epoch 6/10
41/41 [==============================] - 114s 3s/step - loss: 0.4400 - accuracy: 0.9558 - val_loss: 0.5017 - val_accuracy: 0.8542
Epoch 7/10
41/41 [==============================] - 115s 3s/step - loss: 0.4216 - accuracy: 0.9516 - val_loss: 0.4055 - val_accuracy: 0.9607
Epoch 8/10
41/41 [==============================] - 116s 3s/step - loss: 0.3948 - accuracy: 0.9615 - val_loss: 0.4167 - val_accuracy: 0.9251
Epoch 9/10
41/41 [==============================] - 116s 3s/step - loss: 0.3735 - accuracy: 0.9632 - val_loss: 0.3800 - val_accuracy: 0.9475
Epoch 10/10
41/41 [==============================] - 115s 3s/step - loss: 0.3560 - accuracy: 0.9647 - val_loss: 0.3802 - val_accuracy: 0.9321
```

5. Finally compiling and training the model for 10 epochs.

For practical uses the model will be saved for deployment in a backend server, whererequests from client will be served. The model has been created with multiple layers using Convolution and max-pooling,and dropping off some nodes with poor performance. It has been trained for 10 epochs resulting in a validation accuracy of about 94%.

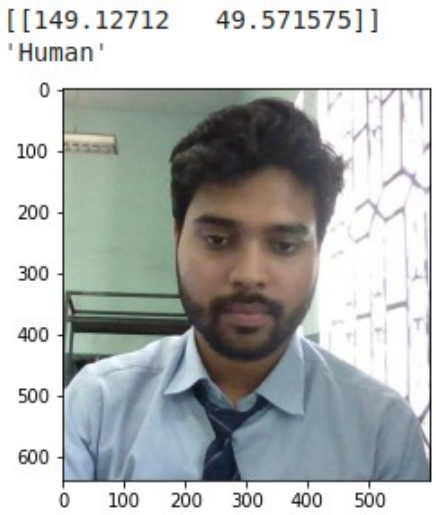


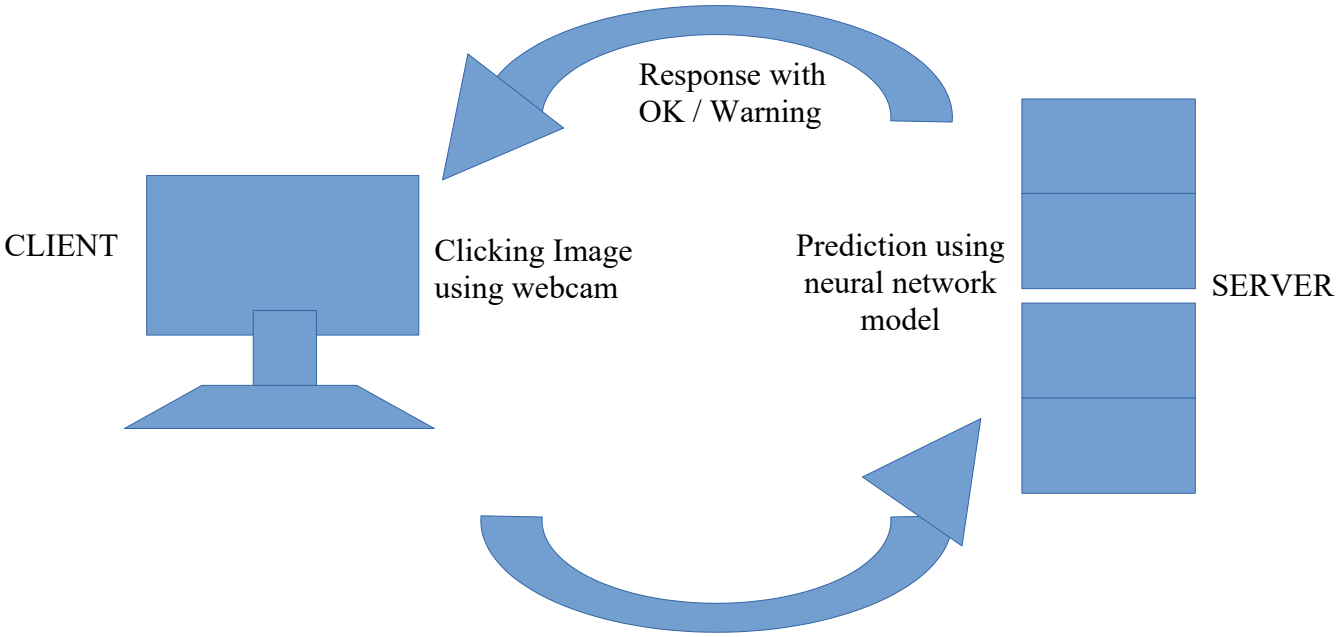
6. The user opens the application in a browser, when the user click on start exam, the browser asks for webcam permission and clicks a photo every 3s.

7. The image is converted into base64 URL data and being sent to the backend using javascript.

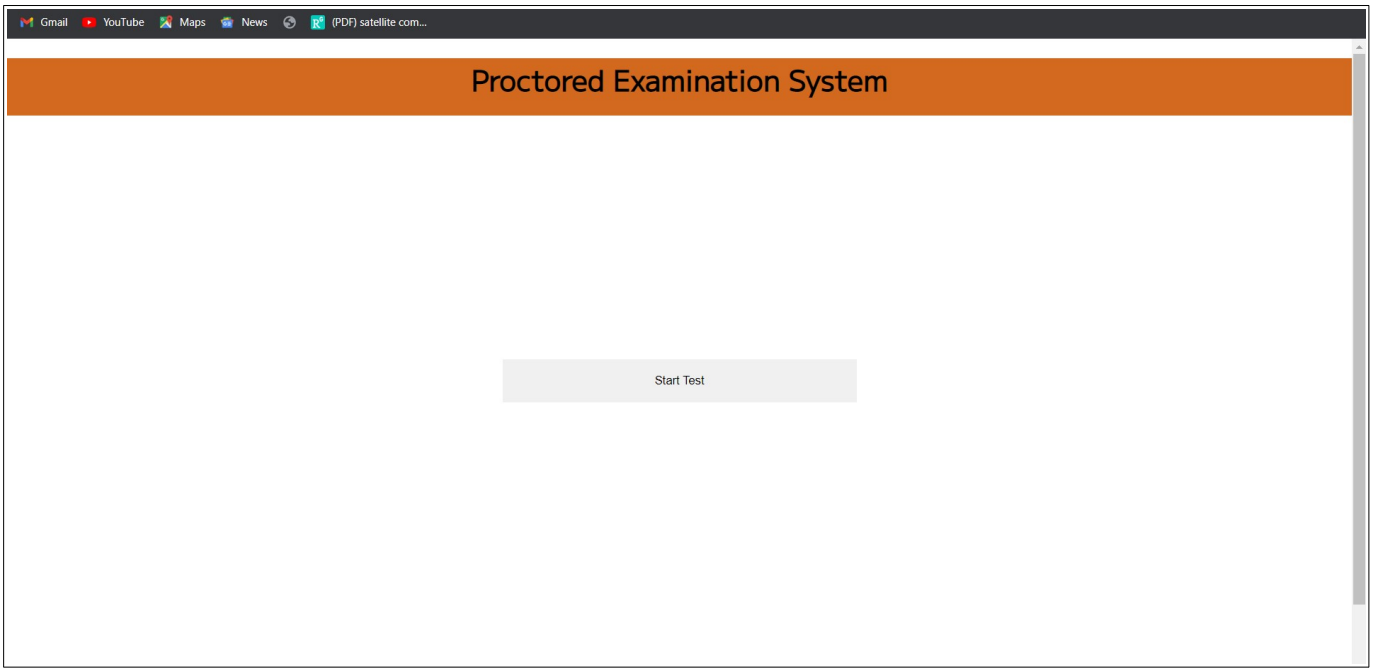
8. In the backend a python server made using Flask is listening on a port, receives the image data and converts back to image.

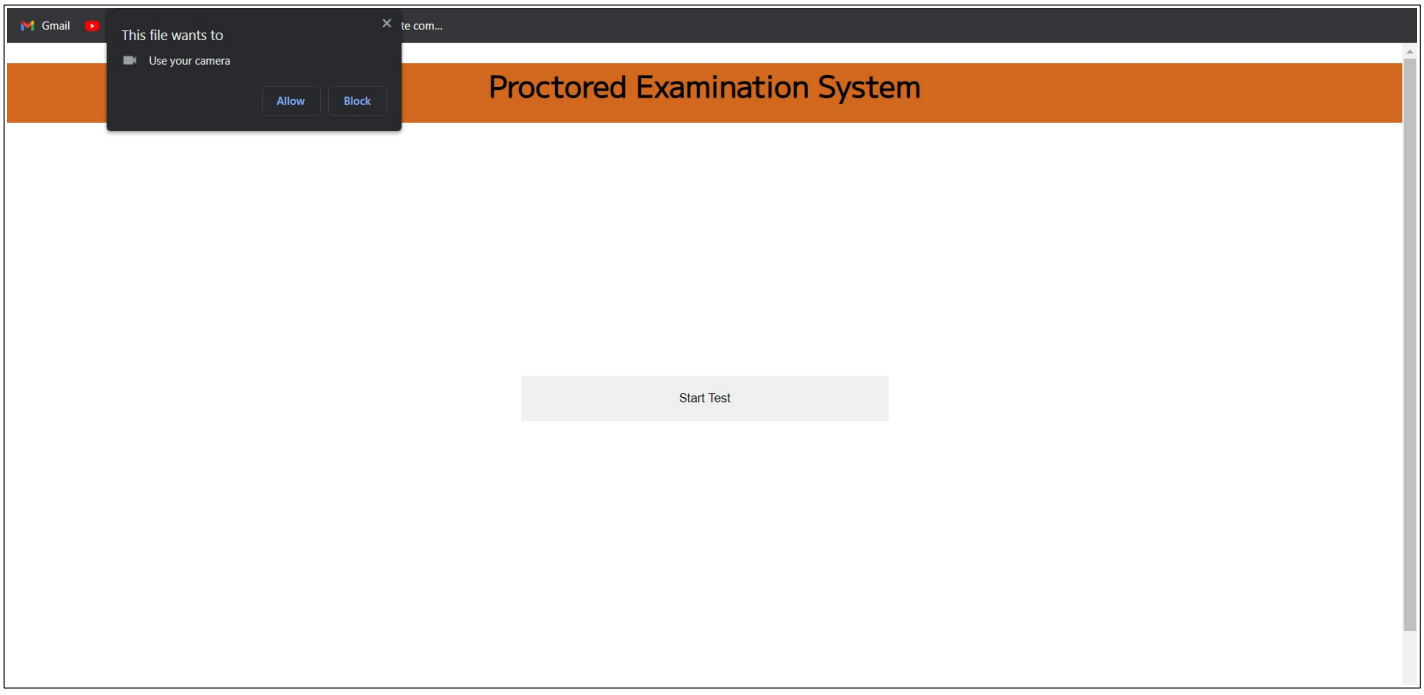
9. The ML model is loaded and used for prediction on the image, if the model finds absence of human face then it sends a warning to the frontend user, otherwise it ignores.

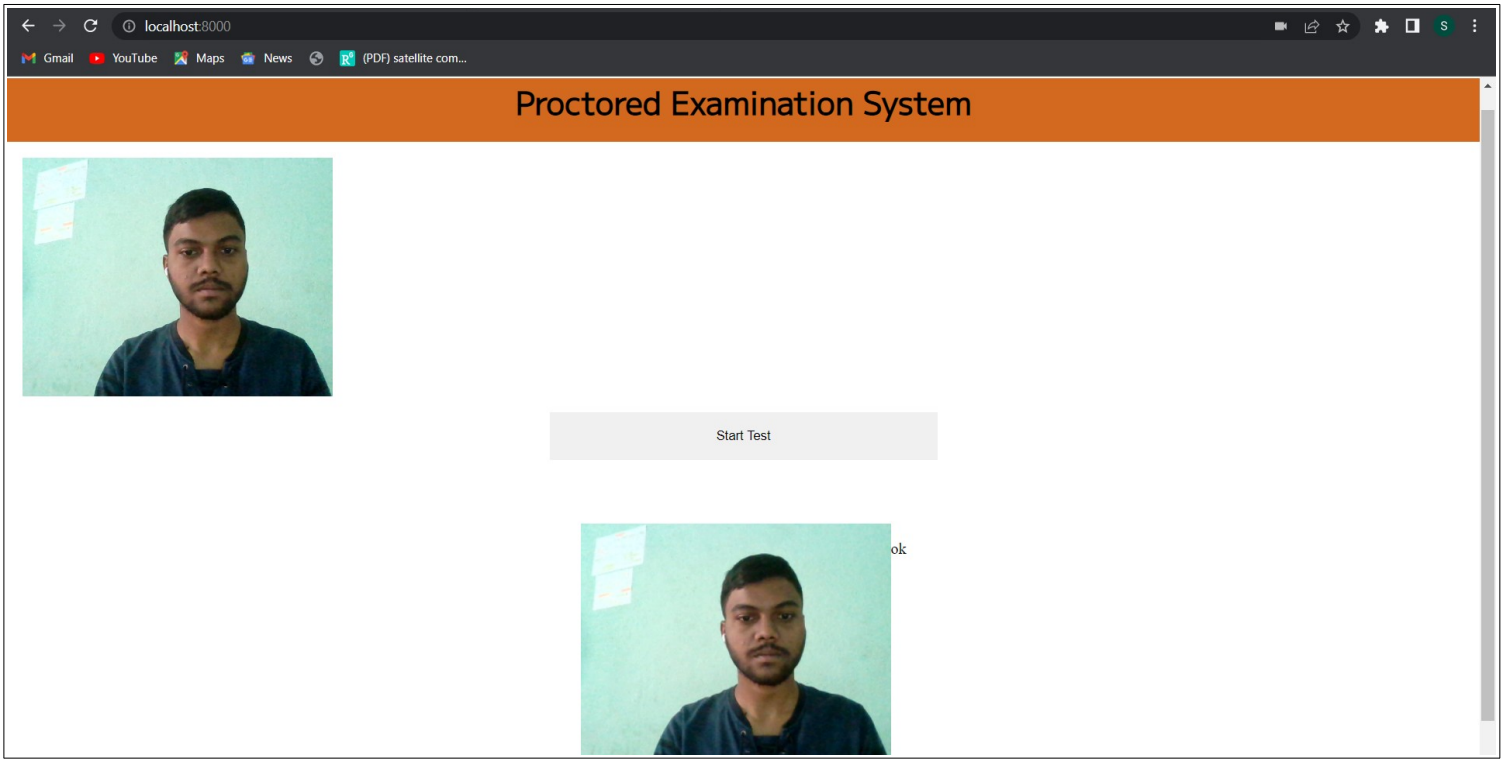


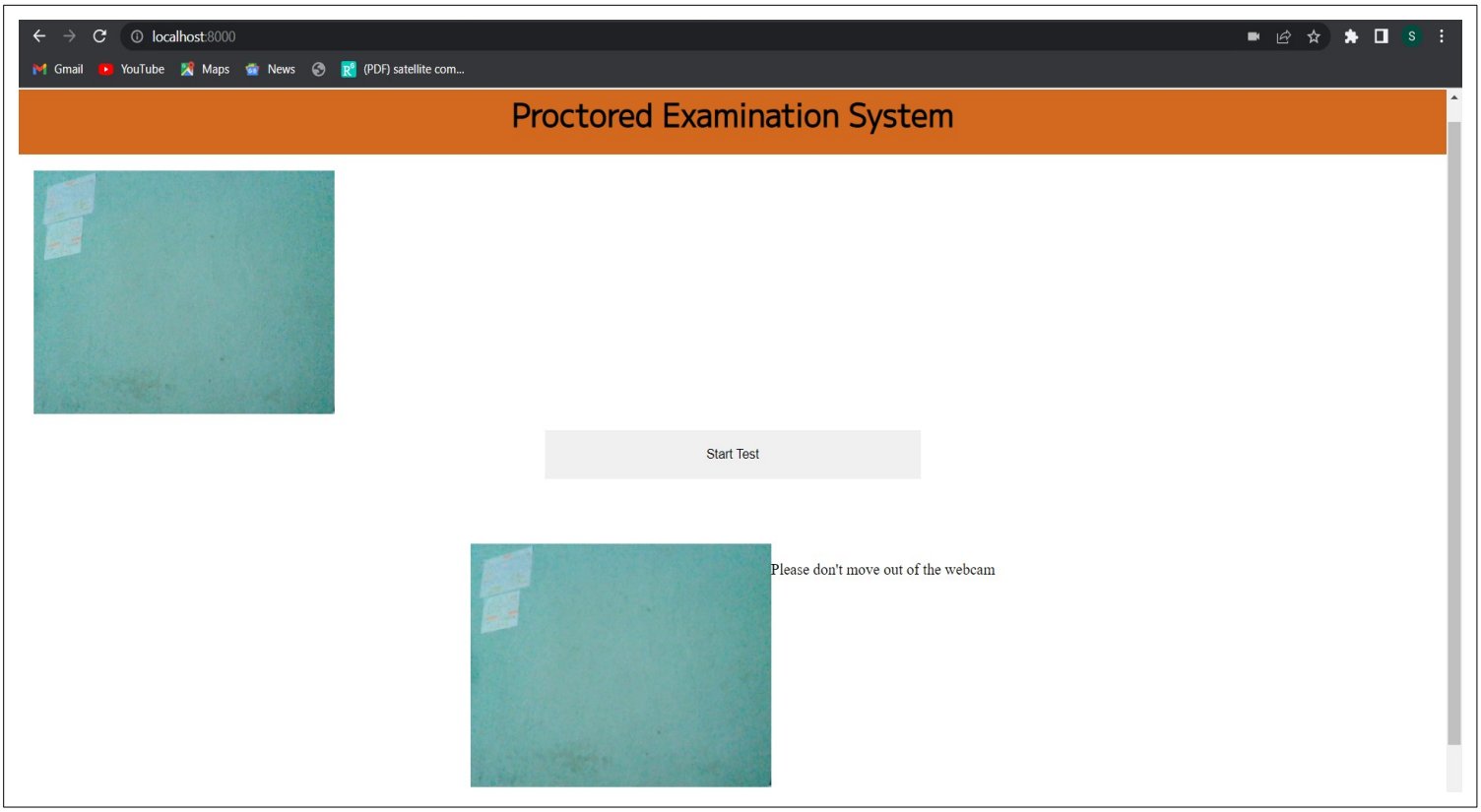


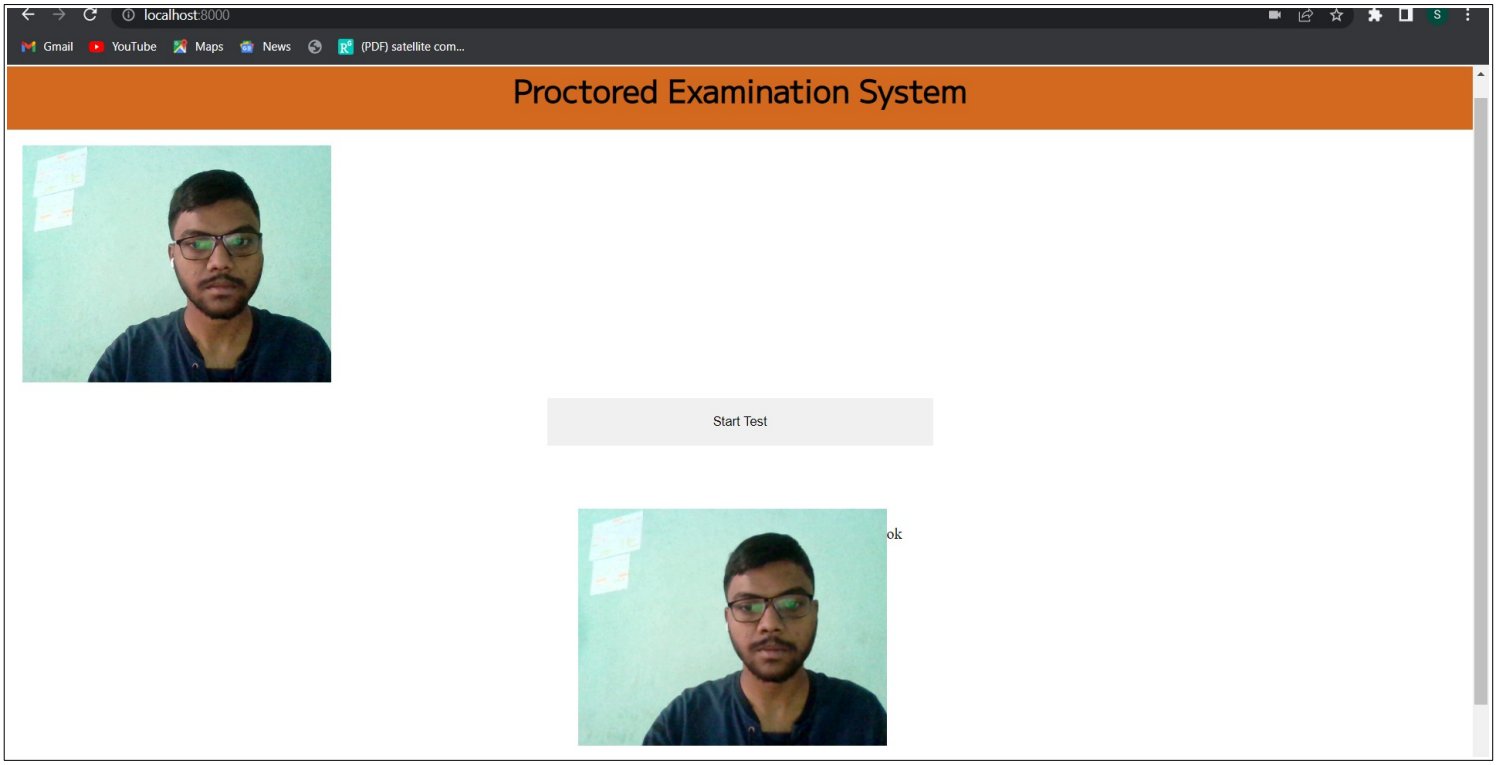
IMAGE/SCREENSHOT OF APPLICATION :






localhost:8000
Gmail
YouTube
Maps
News
(PDF) satellite com...
Proctored Examination System
Start Test
ok


localhost:8000
Gmail
YouTube
Maps
News
(PDF) satellite com...
Proctored Examination System
Start Test
Please don't move out of the webcam



Terminal

```
import base64
import tensorflow as tf
from skimage import transform
import numpy as np
import matplotlib.pyplot as plt
from flask import Flask, render_template, request
from PIL import Image
model=tf.keras.models.load_model('version_1.h5')

app=Flask(__name__,template_folder='.')
@app.route("/")
def index_view():
    return render_template("index.html")
@app.route("/predict",methods=['POST','GET'])
def predict_model():
    if request.method=='POST':
        url=request.get_json()
        img_data=url['data'].split(",")
        img_base=base64.b64decode(img_data[1])
        file=open('img.jpg','wb')
        file.write(img_base)
        file.close()
        img=Image.open(r"img.jpg")
        img=img.crop((130,15,300,200))
        img=np.array(img).astype('float32')/255
        img=transform.resize(img,(250,250,3))
        img=np.expand_dims(img,axis=0)
        predictions=model.predict(img)
        class_names=["Human","Not Human"]
        if np.argmax(a=predictions)==0:
```

Documents/DeepLearning/source files/app.py 1,1 Top

Future Scope of Improvements :

1. A better and larger dataset will help the model to train better and show improved results in predicting. Custom dataset will be more helpful like different angles of human face images, with glasses on, etc.

2. More optimizing can be done on the client-server model. New features like blocking the user from the webapp, and a full new set invigilation system can be added.

Bibliography :

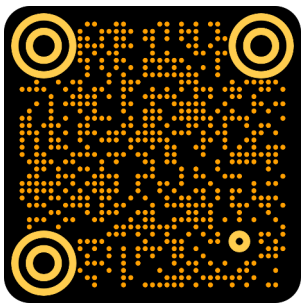
link : https://w3schools.com

link : https://colab.research.google.com

link: https://www.kaggle.com/datasets

Source Code

Dataset



Code :

SERVER APPLICATION – app.py

```
import base64
import tensorflow as tf
from skimage import transform
import numpy as np
import matplotlib.pyplot as plt
from flask import Flask, render_template, request
from PIL import Image
model=tf.keras.models.load_model('version_1.h5')

app=Flask(__name__,template_folder='.')
@app.route("/")
def index_view():
    return render_template("index.html")
@app.route("/predict",methods=['POST','GET'])
def predict_model():
    if request.method=='POST':
        url=request.get_json()
        img_data=url['data'].split(",")
        img_base=base64.b64decode(img_data[1])
        file=open('img.jpg','wb')
        file.write(img_base)
        file.close()
        img=Image.open(r"img.jpg")
        img=img.crop((130,15,300,200))
        img=np.array(img).astype('float32')/255
        img=transform.resize(img,(250,250,3))
        img=np.expand_dims(img,axis=0)
        predictions=model.predict(img)
        class_names=["Human","Not Human"]
```

```
43.         if np.argmax(a=predictions)==0:
44.             return "ok"
45.         else:
46.             return "Please don't move out of the webcam"
47.
48.  if __name__=="__main__":
49.      app.run(debug=True,port=8000)
```

index.html

```
1.  <!doctype html>
2.  <html>
3.
4.  <head lang="en">
5.      <title>Home Page</title>
6.      <link rel="preconnect" href="https://fonts.googleapis.com">
7.      <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
8.
9.      <link href="https://fonts.googleapis.com/css2?family=M+PLUS+1&display=swap"
10.  rel="stylesheet">
11.
12.     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
13.     <meta name="viewport" content="width=device-width,initial-scale=1.0,maximum-
        scale=1.0,user-scalable=no">
14.     <style type="text/css">
15.                     body{
16.                             margin: auto;
17.                     }
18.                     #top-strip{
19.                        margin: auto;
20.                        width: auto;
21.
22.                        height: 4rem;
23.
24.                        text-align: center;
25.                        background: chocolate;
26.                        font-family: 'M PLUS 1';
```

```
27.                 }
28.                 #test{
29.                     text-align: center;
30.                     height: 2rem;
31.                 }
32.                 #test>button{
33.                     padding: 1rem;
34.                     width: 25rem;
35.                     outline: none;
36.                     border: none;
37.                 }
38.
39.                 #camera{
40.                     display: flex;
41.                     justify-content: inline;
42.                 }
43.
44.                 #camera>video{
45.                     padding: 1rem;
46.                 }
47.
48.                 #canva{
49.                     display: flex;
50.                     justify-content: center;
51.                     padding: 5rem;
52.                 }
53.
54.     </style>
```

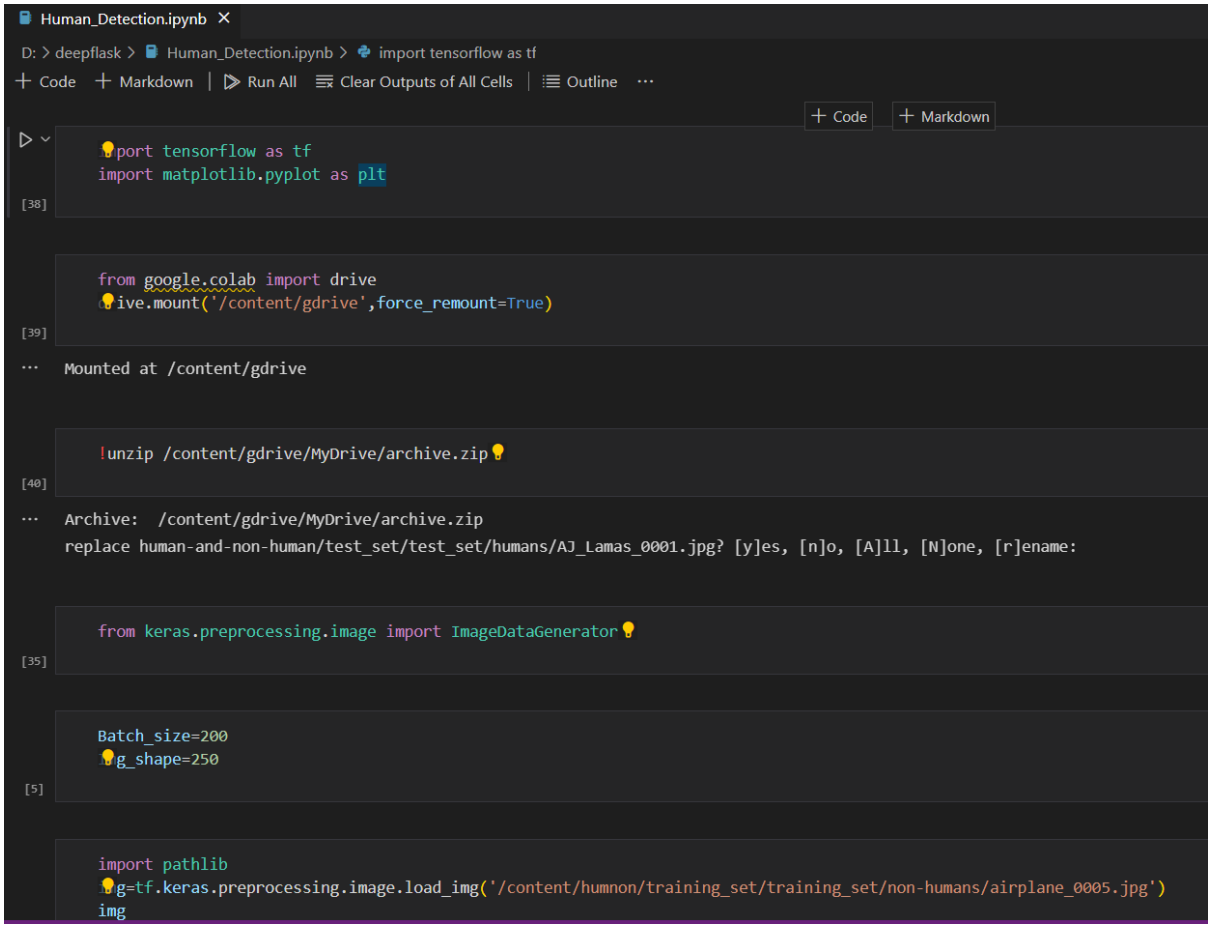
```
55.  </head>
56.    <body>
57.       <div class="container">
58.
59.          <div id="top-strip">
60.             <h1>Proctored Examination System</h1>
61.          </div>
62.          <div id="camera" >
63.             <video id="video" width="320" height="240" autoplay></video>
64.          </div>
65.          <div id="test">
66.             <button id="start-test">Start Test</button>
67.          </div>
68.
69.          <div id="canva">
70.             <canvas id="canvas" width="320" height="240"></canvas>
71.             <p id="stat"></p>
72.          </div>
73.       </div>
74.       <script type="text/javascript">
75.
76.          let camera_button = document.querySelector("#start-test");
77.          let video = document.querySelector("#video");
78.          let click_button = document.querySelector("#click-photo");
79.          let canvas = document.querySelector("#canvas");
80.
81.          camera_button.addEventListener('click', async function() {
```

```
82.             let stream = await navigator.mediaDevices.getUserMedia({ video: true, audio:
    false });
83.             video.srcObject = stream;
84.
85.             setInterval(()=>{
86.             canvas.getContext('2d').drawImage(video, 0, 0, canvas.width, canvas.height);
87.             let image_data_url = canvas.toDataURL('image/jpeg');
88.             evaluate(image_data_url);
89.             },5000);
90.         });
91.
92.         evaluate=(img_url)=>{
93.
94.             fetch('/predict',{
95.                 method:'post',
96.                 body:JSON.stringify({data:img_url}),
97.                 headers:{
98.                     'Content-Type':'application/json'
99.                 }
100.
101.            })
102.            .then(response=>response.text())
103.            .then(resolve=>{
104.                document.getElementById('stat').innerHTML=resolve
105.            })
106.            .catch(err=>console.log(err));
107.        }
108.
```

```
109.      </script>
110.   </body>
111.</html>
112.import base64
113.import tensorflow as tf
114.from skimage import transform
115.import numpy as np
116.import matplotlib.pyplot as plt
117.from flask import Flask, render_template, request
118.from PIL import Image
119.model=tf.keras.models.load_model('version_1.h5')
120.
121.app=Flask(__name__,template_folder='.')
122.@app.route("/")
123.def index_view():
124.   return render_template("index.html")
125.@app.route("/predict",methods=['POST','GET'])
126.def predict_model():
127.   if request.method=='POST':
128.      url=request.get_json()
129.      img_data=url['data'].split(",")
130.      img_base=base64.b64decode(img_data[1])
131.      file=open('img.jpg','wb')
132.      file.write(img_base)
133.      file.close()
134.      img=Image.open(r"img.jpg")
135.      img=img.crop((130,15,300,200))
136.      img=np.array(img).astype('float32')/255
137.      img=transform.resize(img,(250,250,3))
138.      img=np.expand_dims(img,axis=0)
139.      predictions=model.predict(img)
```

```
140.        class_names=["Human","Not Human"]
141.        if np.argmax(a=predictions)==0:
142.            return "ok"
143.        else:
144.            return "Please don't move out of the webcam"
145.
146.if __name__=="__main__":
147.    app.run(debug=True,port=8000)
```

Model training Code :



Human_Detection.ipynb

D: > deepflask > Human_Detection.ipynb > import tensorflow as tf

+ Code + Markdown | Run All | Clear Outputs of All Cells | Outline ···

```
from keras.preprocessing.image import ImageDataGenerator
```
[35]

```
Batch_size=200
g_shape=250
```
[5]

```
import pathlib
g=tf.keras.preprocessing.image.load_img('/content/humnon/training_set/training_set/non-humans/airplane_0005.jpg')
img
```
[6]

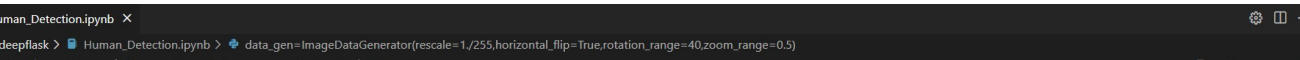
```
import pathlib
ain_dir=pathlib.Path('/content/humnon/training_set/training_set/')
val_dir=pathlib.Path('/content/humnon/test_set/test_set/')
```
[7]


```
data_gen=ImageDataGenerator(rescale=1./255,horizontal_flip=True,rotation_range=40,zoom_range=0.5)
ain_data_gen=data_gen.flow_from_directory(
    batch_size=Batch_size,
    directory=train_dir,
```

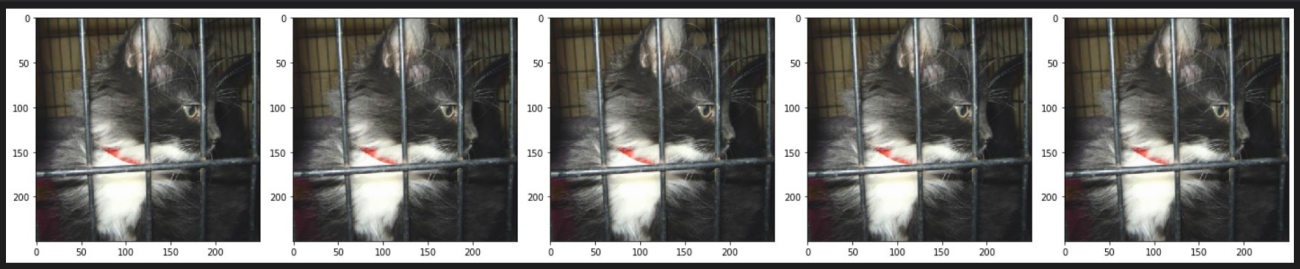
Human_Detection.ipynb

D: > deepflask > Human_Detection.ipynb > data_gen=ImageDataGenerator(rescale=1./255,horizontal_flip=True,rotation_range=40,zoom_range=0.5)

+ Code + Markdown | Run All | Clear Outputs of All Cells | Outline ··· Python 3.9.6 64-bit



```
aug_img=[val_data_gen[0][0][0] for i in range(5)]
otImages(aug_img)
```
[12] Python



```
model=tf.keras.models.Sequential(layers=[
        tf.keras.layers.Conv2D(filters=32,padding='same',kernel_size=(3,3),activation='relu',input_shape=(Img_shape,Img_shape,3)),
        tf.keras.layers.MaxPooling2D(2),
        tf.keras.layers.Conv2D(filters=64,padding='same',kernel_size=(3,3),activation='relu'),
        tf.keras.layers.MaxPooling2D(2),
        tf.keras.layers.Conv2D(filters=128,padding='same',kernel_size=(3,3),activation='relu'),
        tf.keras.layers.MaxPooling2D(2),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(units=512,activation='softmax'),
        tf.keras.layers.Dense(units=2)
```

Conclusion :

The project helped me to understand the concepts of machine learnin and deep learning , along with improving my concept in python programming. The project helped me working on a project and imrpoving my skillset which will be helpful for my future.