ABSTRACT

This project presents a real-time **Gender and Age Detection System** using Python and deep learning techniques. The system utilizes **OpenCV** for face detection and employs pre-trained **Convolutional Neural Networks (CNNs)** to classify the **gender (Male/Female)** and estimate the **age group** (e.g., 0–2, 4–6, 8–12, etc.) of the individual.

The system aims to deliver accurate demographic predictions with minimal latency, making it suitable for applications such as personalized marketing, surveillance, human-computer interaction, and smart kiosks.

The project demonstrates:

- How to integrate pre-trained deep learning models in Python
- How to process live webcam input in real time
- How to display predicted demographic information directly on screen

The proposed system is efficient, scalable, and suitable for further expansion into fields like emotion recognition and behavior analytics.

Index

CHAPTER 1

INTRODUCTION

Gender And Age Detection

Face analysis is an essential and challenging area in computer vision. Understanding facial attributes such as age and gender helps machines interact more naturally with humans. Applications include security surveillance, demographic analysis for marketing, human-computer interaction, and entertainment.

Traditionally, face recognition systems focused on identifying individuals, but modern systems have expanded to extract additional demographic information from faces, such as age groups and gender classification.

1.1 Objection of the project:

The aim of this project is to develop a **real-time system** that detects a person’s face from a video stream or an image and accurately predicts their gender and age group. This involves:

2 Detecting faces using computer vision techniques
3 Extracting relevant features from the face region
4 Applying deep learning models for classification
5 Displaying results intuitively

Modules and their Description

1.2.3 PROJECT OBJECTIVE:

Accurate age and gender detection can revolutionize various sectors by enabling:

- Personalized customer experiences in retail through targeted advertising
- Enhanced security through demographic profiling
- Data collection for sociological research
- Improvements in user experience for interactive applications

1.2.4 PROJECT SCOPE:

The scope of the project is confined to store the image and store in the database. When a person has to be identified the images stored in the database are compared with the existing details.

1.2.5 PROJECT OVERVIEW:

This project focuses on:

- Implementing pre-trained CNN models for gender and age detection
- Using OpenCV for face detection and image preprocessing
- Real-time inference on webcam input
- Building a user-friendly interface for displaying results

CHAPTER -2

SOFTWARE PROJECT PLAN

This chapter discusses about that time schedule for the project and it contains the various phases of the project.

The Various Phases of the Project:

S.NO	TASK	DURATION
1	**Requirement Specification**	**10 Day's**
2	**Requirement document specification**	**10 Day's**
3	**Design analysis**	**20 Day's**
4	**Design Documentation**	**15 Day's**
5	**Design Review**	**20 Day's**
6	**Coding**	**15 Day's**
	Total	**90 Day's**

CHAPTER-3

SOFTWARE REQUIREMENTS

3.1 EXISTING SYSTEM

The objective is to design and implement a system that can accurately detect and classify the gender and estimate the age group of a person from facial images or live video input in real time.

3.2 PROPOSED SYSTEM

To overcome the drawbacks that were in the existing system we develop a system that will be very useful for any investigation department. Here the program keeps track of the record number of each slice during the construction of identifiable human Gender and Age and calculate maximum number of slices of the similar record number. Based on this record number the program retrieves the personal record of the suspect (whose slice constituted the major parts of the constructed human Gender and Age) on exercising the “locate” option.

CHAPTER 4

SOFTWARE REQUIREMENTS SPECIFICATION

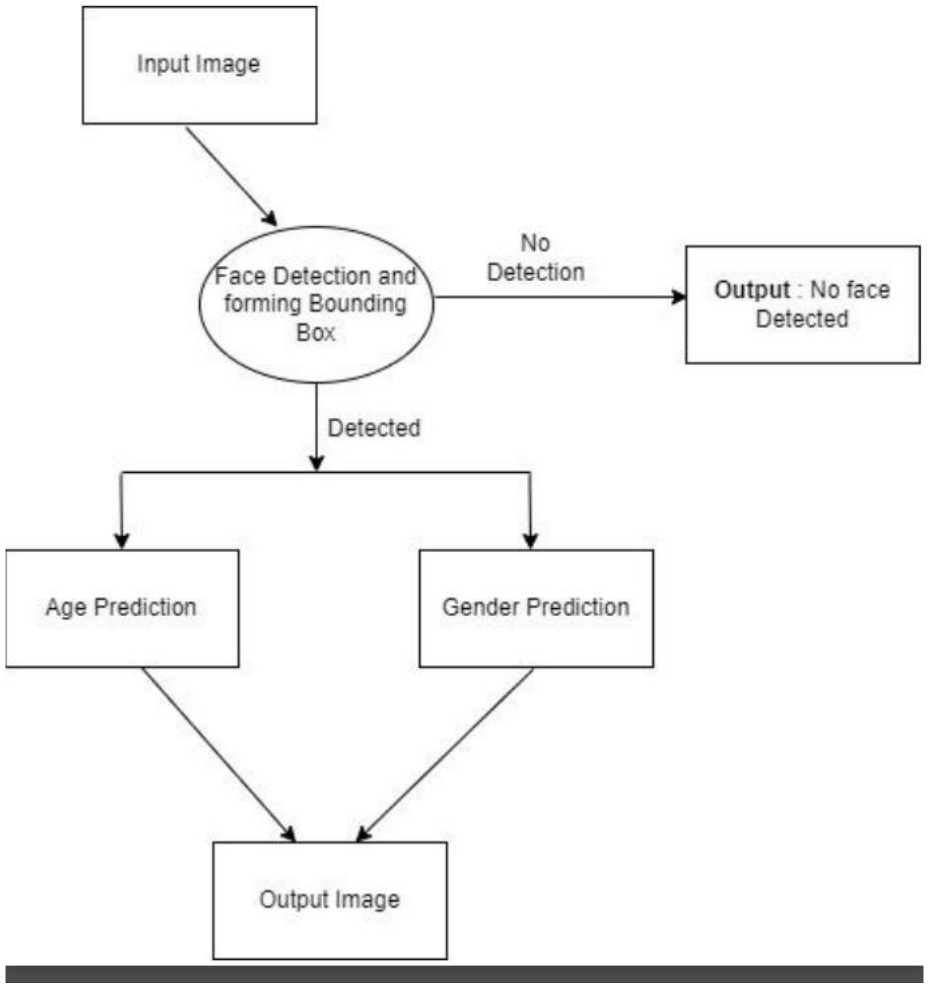
Software Requirements Specification (SRS) is the starting point of the software development activity. Little importance was given to this phases in the **early** days of software development. The emphasis was first on coding and then shifted to design.

As systems grew more complex, it become evident that the goal of the entire system cannot be easily comprehended. Hence need for the requirements analysis phase arose. Now, for large software systems, requirements analysis is perhaps the most difficult activity and also the most error prone.

Some of the difficulty is due to the scope of this phase. The software project is imitated by the client needs. In the beginning these needs are in the minds of various people in the client organization. The requirement analyst has to identify the requirements by tacking to these people and understanding their needs. In situations where the software is to automated a currently manuals process, most of the needs can be understood by observing the current practice.

The SRS is a means of translating the ideas in the minds of the clients (the output) into formal document (the output of the requirements phase). Thus the output of the phase is a set of formally specified requirements, which hopefully are complete and consistent, while the input has none of these properties.

4.1 Functional Requirements



4.2 Performance Requirements

The project must the end user requirements. Accuracy and fast must be imposed in the Project.

The project is development as easy as possible for the sake of end user. The project has to be developed with view of satisfying the future requirements and future enhancement.

The tool has been finally implemented satisfying the needs specified by the company. As per the performance is concerned this system said is performing This processing as well as tine taken to generate well reports where also even when large amount of data was used.

4.3 Interface requirements

4.3.1 Hardware Interface

The stranded input device like keyboard and mouse are to get input. The output will be generated and display in the monitor. The reports can also be exported to a csv document are text file. The stranded printer in used to take outputs.

4.3.2 Software Interface

The design part and interface id done the front end python as a backend of the project.

4.4 Operational requirements

The database or databases that are being failed over to the stand by server cannot be used for anything else. But databases on the standby server not being used for failover can still be used normally.

When it comes time for actual failover, you much one of two things to make your application work either rename the standby server the same name as the failed production server(and the IP address),or re-point your user’s applications to new standby server in some cases,neither of this option is practical.

4.5 Resource Requirements

4.5.1 Hardware Requirements

PROCESSOR	:	PENTIUM i5
RAM	:	8 GB
MONITOR	:	15” COLOR
HARD DISK	:	500 GB
CD DRIVE	:	LG 52X
KEYBOARD	:	STANDARD 102 KEYS
MOUSE	:	3 BUTTONS

4.5.2 Software Requirements

OPERATING SYSTEM	:	Windows 10/11
ENVIRONMENT	:	VS Code
LANGUAGE	:	Python

4.6 Security Requirements

Web application are available via network access, it is a difficult. If not possible, to limit the population of the end-user who may access the applications? In order to product sensitive connect and provide secure mode be implemented throughout the infrastructure that the supports web application and within the application itself.

Web Application have become heavy integrated with critical corporate and database.

E-commerce application extracts and then store sensitive customer information.

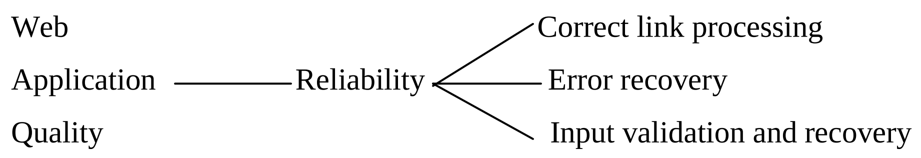
4.7 Design Requirements

To create project, add base masters and masters to the project, assign behaviors to the master, create and assign behavior sets, and then apply, test and validate those behaviors. It also shows how to create and build a stencil to hold the shapes.

4.8 Quality and Reliability Requirements

A software component that is developed for reuse would be correct and contain no defects. In reality, formal verification is not carried out routinely, and defects can add to occur.However, with each reuse, defects are found eliminated, and a components qualify improve as a result. Over time the components virtually defect free.

Software reliability is defined in statical term as" the probability of faultier-free operation of a computer program in a specified environment for specified tine". The software quality and reliability, failure is nonconformance to software requirements. Failure can be only anything or catastrophic. One failure can be corrected within seconds while another requirements week even mouths to correct. Complicating the issue even further, the correction of the one failure may in fact result in the introduction of the errors that ultimately result in other failure.



CHAPTER-5

SYSTEM ANALYSIS

System Analysis

System analysis involves examining the technical and functional aspects of the gender and age detection system to ensure it meets user needs, operates efficiently, and is feasible for implementation.

1. Feasibility Study

a. Technical Feasibility

- Utilizes well-established technologies like OpenCV, TensorFlow/Keras, and CNNs.
- Requires only basic hardware (e.g., webcam, PC with CPU/GPU).
- Can run in real-time for small-scale applications.

b. Operational Feasibility

- Easy-to-use interface.
- Detects and displays results in real-time.
- Can be integrated with attendance systems, surveillance, marketing, etc.

c. Economic Feasibility

- Open-source libraries reduce development cost.
- Requires minimal investment in hardware.
- Suitable for both academic and commercial applications.

CHAPTER 6

SYSTEM DESIGN

6.1 INPUT DESIGN

Input design is the process of converting user-originated inputs to a computer-based format. Input design is one of the most expensive phases of the operation of computerized system and is often the major problem of a system.

In the project, the input design is made in various window forms with various methods. This project consist of Encryption is the conversion of data into a form, called a cipher text, that cannot be easily understood by unauthorized people. Decryption is the process of converting encrypted data back into its original form, so it can be understood.

6.2 OUTPUT DESIGN

Output design generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application. In any system, the output design determines the input to be given to the application.

6.3 INTERFACE DESIGN

The ODBC (Open Database Connectivity) interface is a pure .NET to execute SQl statement. The ODBC provides a set classes and interfaces that can be used by developers to write database applications. Basic ODBC interactions in its simplest form, can be broken down into four steps:

1. Open a connection to the database.
2. Execute a SQL statement
3. Process the result
4. Close the connection to the database

6.4 TABLE AND DATABASE DESIGN:

6.4.1 Normalization:

Normalization is the process of strutting relational database schema such that most ambiguity is removed. The stage of normalization are referred to as forms and progress from the least restrictive(first normal form)through the most restrictive(Fifth normal form), generally , most database designers do not attempt to implement anything higher then normal form of Boyce code Normal Form.

6.4.1.1FIRST NORMAL FORM:

A relation is said to be in First normal form (INF) if and each attributed of the relation is atomic. More simply, to be INF, each column must contain only a single value and each now contain in the same column.

6.4.1.2 SECOND NORMAL FORM:

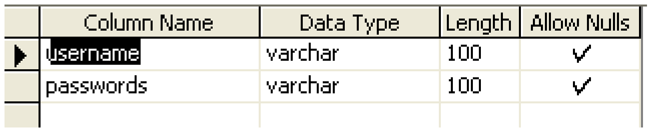
In the Second normal Form, a relation must first fulfill the requirement to be in first Normal Form. Additional, each donkey attribute in the relation must be functionality dependent upon the primary key**.**

6.4.1.3 THIRD NORMAL FORM:

A table is said to be in third normal form and every non key attribute is functionality dependent only on the primary key. This normalization process is applied to this system and the normalized tables are given in the above section.

TABLE DESIGN:

Admin Login Table



Column Name	Data Type	Length	Allow Nulls
username	varchar	100	✓
passwords	varchar	100	✓

6.4.2 Database Design:

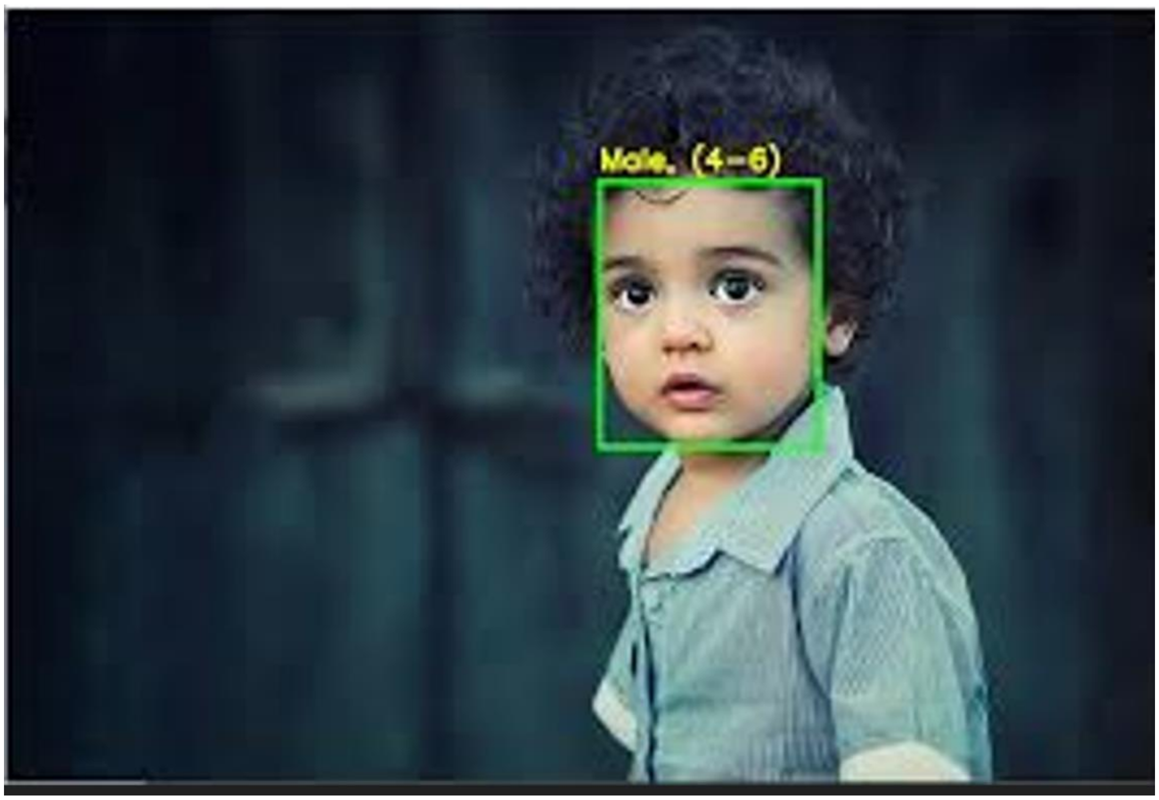
The database design is a must for any application developed especially more for the data store projects. Since the chatting method involves storing the message in the table and produced to the sender and receiver, proper handling of the table is a must.

In the project, login table is designed to be unique in accepting the username and the length of the username and password should be greater than zero

The complete listing of the tables and their fields are provided in the annexure under the title ‘Table Structure’.

CHAPTER 7

CODING



```
 import cv2



video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier('style.xml')
video_capture.set(3, 480) #set width of the frame
video_capture.set(4, 640) #set height of the frame
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744,
114.895847746)
age_list = ['(0, 2)', '(4, 6)', '(8, 12)', '(15, 20)', '(25, 32)', '(38, 43)',
'(48, 53)', '(60, 100)']
gender_list = ['Male', 'Female']
```

```
def load_caffe_models():
age_net = cv2.dnn.readNetFromCaffe('deploy_age.prototxt',
'age_net.caffemodel')
gender_net = cv2.dnn.readNetFromCaffe('deploy_gender.prototxt',
'gender_net.caffemodel')
return(age_net, gender_net)

def video_detector(age_net,gender_net):
font = cv2.FONT_HERSHEY_SIMPLEX #Type of font
while True:
check,frame = video_capture.read()
frame=cv2.flip(frame,1)
#converted our Webcam feed to Grayscale.**< Most of the operations
in OpenCV are done in grayscale>**
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(gray,
1.2,10,cv2.CASCADE_SCALE_IMAGE,(30,30))
print(check)
print(frame)
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
```

```
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_V
ALUES,swapRB=True)#**
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2,
cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
video_capture.release()
cv2.destroyAllWindows()

def main():
age_net, gender_net = load_caffe_models()  # load caffe models (age
& gender)
video_detector(age_net, gender_net)  # prediction age & gender

if __name__ == "__main__":
main()
```

```
import cv2



video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier('style.xml')
video_capture.set(3, 480) #set width of the frame
video_capture.set(4, 640) #set height of the frame
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
age_list = ['(0, 2)', '(4, 6)', '(8, 12)', '(15, 20)', '(25, 32)', '(38, 43)', '(48, 53)', '(60, 100)']
gender_list = ['Male', 'Female']

def load_caffe_models():
age_net = cv2.dnn.readNetFromCaffe('deploy_age.prototxt', 'age_net.caffemodel')
gender_net = cv2.dnn.readNetFromCaffe('deploy_gender.prototxt',
'gender_net.caffemodel')
return(age_net, gender_net)

def video_detector(age_net,gender_net):
font = cv2.FONT_HERSHEY_SIMPLEX #Type of font
while True:
check,frame = video_capture.read()
frame=cv2.flip(frame,1)
#converted our Webcam feed to Grayscale.**< Most of the operations in OpenCV are
done in grayscale>**
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(gray,
1.2,10,cv2.CASCADE_SCALE_IMAGE,(30,30))
print(check)
print(frame)
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
```

```
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_VALUES,swapR
B=True)#**
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2, cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
video_capture.release()import cv2


video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier('style.xml')
video_capture.set(3, 480) #set width of the frame
video_capture.set(4, 640) #set height of the frame
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744,
114.895847746)
```

```
age_list = ['(0, 2)', '(4, 6)', '(8, 12)', '(15, 20)', '(25, 32)', '(38, 43)',
'(48, 53)', '(60, 100)']
gender_list = ['Male', 'Female']

def load_caffe_models():
age_net = cv2.dnn.readNetFromCaffe('deploy_age.prototxt',
'age_net.caffemodel')
gender_net = cv2.dnn.readNetFromCaffe('deploy_gender.prototxt',
'gender_net.caffemodel')
return(age_net, gender_net)

def video_detector(age_net,gender_net):
font = cv2.FONT_HERSHEY_SIMPLEX #Type of font
while True:
check,frame = video_capture.read()
frame=cv2.flip(frame,1)
#converted our Webcam feed to Grayscale.**< Most of the operations
in OpenCV are done in grayscale>**
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(gray,
1.2,10,cv2.CASCADE_SCALE_IMAGE,(30,30))
print(check)
print(frame)
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
```

```
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_V
ALUES,swapRB=True)#**
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2,
cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_V
ALUES,swapRB=True)#**
```

```
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2,
cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
video_capture.release()
cv2.destroyAllWindows()

def main():
age_net, gender_net = load_caffe_models()  # load caffe models (age
& gender)
video_detector(age_net, gender_net)  # prediction age & gender

if __name__ == "__main__":
main()
```

```
import cv2
```

```
video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier('style.xml')
video_capture.set(3, 480) #set width of the frame
video_capture.set(4, 640) #set height of the frame
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
age_list = ['(0, 2)', '(4, 6)', '(8, 12)', '(15, 20)', '(25, 32)', '(38, 43)', '(48, 53)', '(60, 100)']
gender_list = ['Male', 'Female']

def load_caffe_models():
age_net = cv2.dnn.readNetFromCaffe('deploy_age.prototxt', 'age_net.caffemodel')
gender_net = cv2.dnn.readNetFromCaffe('deploy_gender.prototxt',
'gender_net.caffemodel')
return(age_net, gender_net)

def video_detector(age_net,gender_net):
font = cv2.FONT_HERSHEY_SIMPLEX #Type of font
while True:
check,frame = video_capture.read()
frame=cv2.flip(frame,1)
#converted our Webcam feed to Grayscale.**< Most of the operations in OpenCV are
done in grayscale>**
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(gray,
1.2,10,cv2.CASCADE_SCALE_IMAGE,(30,30))
print(check)
print(frame)
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
```

```
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_VALUES,swapR
B=True)#**
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2, cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
video_capture.release()import cv2


video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier('style.xml')
video_capture.set(3, 480) #set width of the frame
video_capture.set(4, 640) #set height of the frame
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744,
114.895847746)
age_list = ['(0, 2)', '(4, 6)', '(8, 12)', '(15, 20)', '(25, 32)', '(38, 43)',
'(48, 53)', '(60, 100)']
```

```
gender_list = ['Male', 'Female']

def load_caffe_models():
age_net = cv2.dnn.readNetFromCaffe('deploy_age.prototxt',
'age_net.caffemodel')
gender_net = cv2.dnn.readNetFromCaffe('deploy_gender.prototxt',
'gender_net.caffemodel')
return(age_net, gender_net)

def video_detector(age_net,gender_net):
font = cv2.FONT_HERSHEY_SIMPLEX #Type of font
while True:
check,frame = video_capture.read()
frame=cv2.flip(frame,1)
#converted our Webcam feed to Grayscale.**< Most of the operations
in OpenCV are done in grayscale>**
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(gray,
1.2,10,cv2.CASCADE_SCALE_IMAGE,(30,30))
print(check)
print(frame)
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
```

```
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_V
ALUES,swapRB=True)#**
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2,
cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_V
ALUES,swapRB=True)#**
# Predict Gender
gender_net.setInput(blob)
```

```
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2,
cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
video_capture.release()
cv2.destroyAllWindows()

def main():
age_net, gender_net = load_caffe_models()  # load caffe models (age
& gender)
video_detector(age_net, gender_net)  # prediction age & gender

if __name__ == "__main__":
main()
```

```
import cv2


video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier('style.xml')
```

```
video_capture.set(3, 480) #set width of the frame
video_capture.set(4, 640) #set height of the frame
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
age_list = ['(0, 2)', '(4, 6)', '(8, 12)', '(15, 20)', '(25, 32)', '(38, 43)', '(48, 53)', '(60, 100)']
gender_list = ['Male', 'Female']

def load_caffe_models():
age_net = cv2.dnn.readNetFromCaffe('deploy_age.prototxt', 'age_net.caffemodel')
gender_net = cv2.dnn.readNetFromCaffe('deploy_gender.prototxt',
'gender_net.caffemodel')
return(age_net, gender_net)

def video_detector(age_net,gender_net):
font = cv2.FONT_HERSHEY_SIMPLEX #Type of font
while True:
check,frame = video_capture.read()
frame=cv2.flip(frame,1)
#converted our Webcam feed to Grayscale.**< Most of the operations in OpenCV are
done in grayscale>**
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(gray,
1.2,10,cv2.CASCADE_SCALE_IMAGE,(30,30))
print(check)
print(frame)
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
```

```
print(face_img)
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_VALUES,swapR
B=True)#**
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2, cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
video_capture.release()import cv2


video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier('style.xml')
video_capture.set(3, 480) #set width of the frame
video_capture.set(4, 640) #set height of the frame
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744,
114.895847746)
age_list = ['(0, 2)', '(4, 6)', '(8, 12)', '(15, 20)', '(25, 32)', '(38, 43)',
'(48, 53)', '(60, 100)']
gender_list = ['Male', 'Female']
```

```
def load_caffe_models():
age_net = cv2.dnn.readNetFromCaffe('deploy_age.prototxt',
'age_net.caffemodel')
gender_net = cv2.dnn.readNetFromCaffe('deploy_gender.prototxt',
'gender_net.caffemodel')
return(age_net, gender_net)

def video_detector(age_net,gender_net):
font = cv2.FONT_HERSHEY_SIMPLEX #Type of font
while True:
check,frame = video_capture.read()
frame=cv2.flip(frame,1)
#converted our Webcam feed to Grayscale.**< Most of the operations
in OpenCV are done in grayscale>**
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(gray,
1.2,10,cv2.CASCADE_SCALE_IMAGE,(30,30))
print(check)
print(frame)
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_V
ALUES,swapRB=True)#**
```

```
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2,
cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break

video_capture.release()
cv2.destroyAllWindows()

def main():
age_net, gender_net = load_caffe_models()  # load caffe models (age
& gender)
video_detector(age_net, gender_net)  # prediction age & gender

if __name__ == "__main__":
main()
```

import cv2

```
video_capture = cv2.VideoCapture(0)
faceCascade = cv2.CascadeClassifier('style.xml')
video_capture.set(3, 480) #set width of the frame
video_capture.set(4, 640) #set height of the frame
MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
age_list = ['(0, 2)', '(4, 6)', '(8, 12)', '(15, 20)', '(25, 32)', '(38, 43)', '(48, 53)', '(60, 100)']
gender_list = ['Male', 'Female']

def load_caffe_models():
age_net = cv2.dnn.readNetFromCaffe('deploy_age.prototxt', 'age_net.caffemodel')
gender_net = cv2.dnn.readNetFromCaffe('deploy_gender.prototxt',
'gender_net.caffemodel')
return(age_net, gender_net)

def video_detector(age_net,gender_net):
font = cv2.FONT_HERSHEY_SIMPLEX #Type of font
while True:
check,frame = video_capture.read()
frame=cv2.flip(frame,1)
#converted our Webcam feed to Grayscale.**< Most of the operations in OpenCV are
done in grayscale>**
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(gray,
1.2,10,cv2.CASCADE_SCALE_IMAGE,(30,30))
print(check)
print(frame)
# Draw a rectangle around the faces
for (x, y, w, h) in faces:
#Make rectangle on face
#print(x,y,w,h)#print The coordinates of face
```

```
cv2.rectangle(frame, (x, y), (x + w, y + h), (255,255,0), 2)
# Get Face as Matrix and copy it
face_img = frame[y:y + h, h:h + w].copy()
print(face_img)
blob=cv2.dnn.blobFromImage(face_img,1,(244,244),MODEL_MEAN_VALUES,swapR
B=True)#**
# Predict Gender
gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()]
# Predict Age
age_net.setInput(blob)
age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
overlay_text = "%s %s" % (gender, age)
cv2.putText(frame, overlay_text, (x, y), font, 1, (255, 255, 255), 2, cv2.LINE_AA)
cv2.imshow('frame', frame)
key=cv2.waitKey(1)
if key == 27:
break
video_capture.release()
cv2.destroyAllWindows()

def main():
age_net, gender_net = load_caffe_models()  # load caffe models (age & gender)
video_detector(age_net, gender_net)  # prediction age & gender

if __name__ == "__main__":
main()
```

```
cv2.destroyAllWindows()

def main():
age_net, gender_net = load_caffe_models()  # load caffe models (age & gender)
video_detector(age_net, gender_net)  # prediction age & gender

if __name__ == "__main__":
main()
```

CHAPTER 8

SYSTEM TESTING

System testing involves user training system testing and successful running of the developed proposed system. The user tests the developed system and changes are made according to their needs. The testing phase involves the testing of developed system using various kinds of data.

An elaborate testing of data is prepared and the system is tested using the test data. While testing, errors are noted and the corrections are made. The corrections are also noted for the future use. The users are trained to operate the developed system.

TESTING:

System testing is the stage of implementation that is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. System testing makes logical assumption that if all the parts of the system are correct, then the goal will be successfully achieved. A series of testing are done for the proposed system before the system is ready for the user acceptance testing.

The following are the types of Testing:

1. Unit Testing
2. Integration Testing
3. Validation Testing
4. Verification testing
5. User acceptance testing

8.1 UNIT TESTING

The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. Then the web form level testing is made. For example storage of data to the table in the correct manner.

In the company as well as seeker registration form, the zero length username and password are given and checked. Also the duplicate username is given and checked. In the job and question entry, the button will send data to the server only if the client side validations are made.

The dates are entered in wrong manner and checked. Wrong email-id and web site URL (Universal Resource Locator) is given and checked.

8.2 INTEGRATION TESTING

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus the system testing is a confirmation that all is correct and an opportunity to show the user that the system works.

8.3 VALIDATION TESTING

The final step involves Validation testing, which determines whether the software function as the user expected. The end-user rather than the system developer conduct this test most software developers as a process called "Alpha and Beta Testing" to uncover that only the end user seems able to find.

8.4 VERIFICATION TESTING

Verification is a fundamental concept in software design. This is the bridge between customer requirements and an implementation that satisfies those requirements.

This is verifiable if it can be demonstrated that the testing will result in an implementation that satisfies the customer requirements.

Inadequate testing or non-testing leads to errors that may appear few months later. This will create two problems

- ✓ Time delay between the cause and appearance of the problem.
- ✓ The effect of the system errors on files and records within the system.

8.5 USER ACCEPT TESTING

User acceptance testing of a system is the key factor of the success of any system. The system under study is tested for the user acceptance by constantly keeping in touch with the prospective system users at any time of developing and making changes whenever required.

SYSTEM IMPLEMENTATION

Implementation is the most crucial stage in achieving a successful system and giving the user's confidence that the new system is workable and effective. Implementation of a modified application to replace an existing one. This type of conversation is relatively easy to handle, provide there are no major changes in the system.

Each program is tested individually at the time of development using the data and has verified that this program linked together in the way specified in the programs specification, the computer system and its environment is tested to the satisfaction of the user. The system that has been developed is accepted and proved to be satisfactory for the user. And so the system is going to be implemented very soon. A simple operating procedure is included so that the user can understand the different functions clearly and quickly.

Initially as a first step the executable form of the application is to be created and loaded in the common server machine which is accessible to all the user and the server is to be connected to a network. The final stage is to document the entire system which provides components and the operating procedures of the system.

SCOPE FOR FUTURE DEVELOPMENT

Every application has its own merits and demerits. The project has covered almost all the requirements. Further requirements and improvements can easily be done since the coding is mainly structured or modular in nature. Changing the existing modules or adding new modules can append improvements. Further enhancements can be made to the application, so that the web site functions very attractive and useful manner than the present one.

CHAPTER 9

PROBLEMS FACED

When there is a clear goal in sight but no clear set of directions or means to attain that goal, then it is called a problem. Problems can be broken down into four aspects; goal, givens, means of transforming conditions, and obstacles.

Goal – the goal is the desired end state which the problem solving is being directed toward. The hope is to reach that end state and be able to assess whether or not you achieved what you wanted.

Givens- these are the objects, conditions, and constraints that accompany a problem, and can be either explicit or implicit.

Means of transforming conditions- there should be a way of changing the initial state of the problem. this is most usually a person's knowledge or skill level. For instance ,a computer programmer presented with a problem would utilize his or her knowledge of programming language to transform the state of the problem.

Obstacles- the problem should present a challenge. If there are no challenges involved and the situation can be easily solved then it is not so a problem so much as a routines task.

Every problem has a **problem faced**, which is the whole range of possible states and operators. only some of these states and operators will bring the person closer to the goal state. The problem starts at the **initial state** and **operators** are applied to change the state, creating a series of intermediate states that should hopefully lead to the final goal state

CHAPTER-10

FUTURE PLANS

Every application has its own merits and demerits. The project has covered almost all the requirements. Further requirements and improvements can easily be done since the coding is mainly structured or modular in nature. Changing the existing modules or adding new modules can append improvements. Further enhancements can be made to the application, so that the web site functions very attractive and useful manner than the present one.

CONCLUSION

It is concluded that the application works well and satisfy the users. The application is tested very well and errors are properly debugged. The site is simultaneously accessed from more than one system. Simultaneous login from more than one place is tested.

The application works according to the restrictions provided in their respective system. Further enhancements can be made to the application, so that the application functions very attractive and useful manner than the present one. The speed of the transactions become more enough now.

BIBLIOGRAPHY

1. **Lutz, Mark.**
   Learning Python (5th ed.). O'Reilly Media, 2013.
   A comprehensive guide to learning Python, covering both basic and advanced concepts.
2. **Matthes, Eric.**
   Python Crash Course: A Hands-On, Project-Based Introduction to Programming (2nd ed.). No Starch Press, 2019.
   Focuses on practical projects like games and web apps using Python.
3. **Sweigart, Al.**
   Automate the Boring Stuff with Python: Practical Programming for Total Beginners. No Starch Press, 2015.
   Teaches how to automate everyday tasks using Python with real-world examples.
4. **Downey, Allen B.**
   Think Python: How to Think Like a Computer Scientist (2nd ed.). Green Tea Press, 2015.
   A great resource for understanding the logic behind Python programming.
5. **Raschka, Sebastian.**
   Python Machine Learning (3rd ed.). Packt Publishing, 2019.
   Ideal for those working on machine learning projects in Python.
6. **The Python Software Foundation.**
   Python Documentation.
   https://docs.python.org/3/
   The official documentation of Python; a reliable and up-to-date source.
7. **Real Python.**
   https://realpython.com/
   An excellent site with Python tutorials, project ideas, and code walkthroughs.
8. **Geeks for Geeks – Python Programming Language.**
   https://www.geeksforgeeks.org/python-programming-language/
   Contains thousands of Python examples, tutorials, and project ideas.
9. **GitHub – Python Projects Repository.**
   https://github.com/topics/python-project
   Open-source projects from the developer community, useful for reference and collaboration.
10. **Kumar, Yashavant, and BPB Publications.**
    Let Us Python. BPB Publications, 2019.
    Great for understanding the syntax and creating basic to intermediate level projects.