Experiment-7.py

```python
# 18BCS6201-CV Practical-7 (Gauri Prabhakar) (AI-ML-2)(B)
# Aim: To implement face detction using HAAR cascades  using mediapipe in python and OpenCV.

# Importing necessary modules.
import cv2
import mediapipe as mp
# Importing 'mp.solutions.face_detection' to a variable 'mp_face_detection'.
mp_face_detection = mp.solutions.face_detection
# We will use 'drawing_utils' to draw the key points.
mp_drawing = mp.solutions.drawing_utils


# Importing the cascade file into the variable 'face_cascade'
face_cascade = cv2.CascadeClassifier(r"C:\Users\gauri\Desktop\OpenCV Media\haarcascade_frontalface_default.xml")
# Importing the image into the variable 'img'.
img = cv2.imread(r"C:\Users\gauri\Desktop\OpenCV Media\lena.png")
# Reading the image and converting it to RGB and storing it in the variable 'gray'.
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# Now we find the faces in the image.
# If faces are detected, we return the positions of detected faces as Rect(x,y,w,h).
# Once we get these locations, we can create a ROI(Rectangular region of interest) for the face
# and apply eye detection on this ROI (since eyes are always on the face !!! ).
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
# Returning the co-ordinates of the ROI.
print(faces)

# Drawing a rectangle on the ROI.
for (x,y,w,h) in faces:
    img = cv2.rectangle(img,(x,y),(x+w,y+h),(31,79,254),2)

# Rendering the video with effective Facetracking to the console by using the function '.imshow()'.
cv2.imshow('Face Detection on Image using Mediapipe',img)

# Setting up '.waitkey()' to wait for a specific time until any key is pressed and break the loop.
cv2.waitKey(0)
# Destroying all windows.
cv2.destroyAllWindows()


# Creating a variable to store the video using the '.VideoCapture()' function.
cap = cv2.VideoCapture(r"C:\Users\gauri\Desktop\OpenCV Media\To Plan or Not to Plan_.mp4")

# Specifying the detection and tracking confidence uisng 'mp_pose.pose()'.
with mp_face_detection.FaceDetection(
    model_selection=0, min_detection_confidence=0.5) as face_detection:
  # While the video is running.
  while cap.isOpened():
    # Capturing the video frame by frame using the '.read()' method.
    success, image = cap.read()

    # To improve performance, marking the image as not writable and passing by reference.
    image.flags.writeable = False
    # Reading the frames and converting them to RGB.
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    # Detecting faces in the frame using the function 'face_detection.process()'.
    results = face_detection.process(image)

    # Drawing the pose annotations on the image.
    image.flags.writeable = True

    # Reading the frames and converting them to RGB.
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

    # If a face is detected we draw a rectangle on the ROI.
    if results.detections:
      for detection in results.detections:
        mp_drawing.draw_detection(image, detection)

    # Rendering the video with effective Facetracking to the console by using the function '.imshow()'.
    cv2.imshow('MediaPipe Face Detection',image)

    # Setting up '.waitkey()' to wait for a specific time until any key is pressed and break the loop.
    # '.waitkey(1)' displays a frame for 1ms after which it moves to the next frame in the video.
    # Setting 'x' as the quitting button.
    if cv2.waitKey(5) & 0xFF == ord('x'):
      break

# Releasing the variable/object 'cap'.
cap.release()
```