In [1]:

```
!pip install mediapipe opencv-python
```

```
Collecting mediapipe
  Downloading mediapipe-0.8.8-cp38-cp38-win_amd64.whl (45.8 MB)
Collecting opencv-python
  Downloading opencv_python-4.5.3.56-cp38-cp38-win_amd64.whl (34.9 MB)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\sit
e-packages (from mediapipe) (1.19.2)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\li
b\site-packages (from mediapipe) (3.3.2)
Requirement already satisfied: attrs>=19.1.0 in c:\programdata\anaconda3
\lib\site-packages (from mediapipe) (20.3.0)
Collecting protobuf>=3.11.4
  Downloading protobuf-3.18.1-cp38-cp38-win_amd64.whl (912 kB)
Collecting absl-py
  Downloading absl_py-0.14.1-py3-none-any.whl (131 kB)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-
packages (from mediapipe) (1.15.0)
Collecting opencv-contrib-python
  Downloading opencv_contrib_python-4.5.3.56-cp38-cp38-win_amd64.whl (4
1.8 MB)
```

In [56]:

```
import cv2
import mediapipe as mp
import numpy as np
mp_draw = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
```

Calculate angle

In [57]:

```
def calculate_angle(a,b,c):
    a = np.array(a) # First
    b = np.array(b) # Mid
    c = np.array(c) # End

    radians = np.arctan2(c[1]-b[1], c[0]-b[0]) - np.arctan2(a[1]-b[1], a[0]-b[0])
    angle = np.abs(radians*180.0/np.pi)

    if angle >180.0:
        angle = 360-angle

    return angle
```

Squats AI Gym Instructor

In [59]:

```
cap = cv2.VideoCapture(0)

# Squats counter variables
counter = 0
timer = 30
stage = None
start = False #whether user has started exercising


## Setup mediapipe instance
with mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5) as pose:
    while cap.isOpened():
        ret, frame = cap.read()

        # Recolor image to RGB since mediapipe only accepts image in this format
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        image.flags.writeable = False

        # Make detection
        results = pose.process(image)

        # Recolor back to BGR
        image.flags.writeable = True
        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

        # Extract landmarks from image
        try:
            landmarks = results.pose_landmarks.landmark

            # Get (x,y) coordinates
            hip = [landmarks[mp_pose.PoseLandmark.RIGHT_HIP.value].x,landmarks[mp_pose.Pose
            knee = [landmarks[mp_pose.PoseLandmark.RIGHT_KNEE.value].x,landmarks[mp_pose.Po
            ankle = [landmarks[mp_pose.PoseLandmark.RIGHT_ANKLE.value].x,landmarks[mp_pose.
            nose = [landmarks[mp_pose.PoseLandmark.NOSE.value].x,landmarks[mp_pose.PoseLand

            # Calculate angle
            angle1 = calculate_angle(hip, knee, ankle)
            angle2 = calculate_angle(nose,knee,hip)
            angle3 = calculate_angle(nose,hip,knee)
            # 3 angles used to maximise accuracy of posture and strictness of our program

            # Display corrections on live feed to guide users to correct posture
            if angle1>120 and angle1<160 and stage == "up" and counter>0:
                cv2.rectangle(image, (5,130), (300,250), (255,255,255), -1)
                cv2.putText(image, 'Go Lower!', (12,200),
                            cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0,0,255), 1, cv2.LINE_AA)
            if angle2>50 and angle3<120 and stage == "up" and counter>0:
                cv2.rectangle(image, (5,260), (300,380), (255,255,255), -1)
                cv2.putText(image, 'Back Bent!', (12,330),
                            cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0,0,255), 1, cv2.LINE_AA)
            if angle1 < 120 and angle1 > 70 and angle2 < 50 and angle3<120 and angle3>60 :
                if counter == 0:
                    start = True
                stage = "down"
            if angle1 > 150 and angle3 > 150 and stage =="down":
                stage="up"
                counter +=1

        except:
```

```
            pass


        # when our program senses the user starting the exercise, it will display the follo
        if timer>0 and start == True:
            cv2.putText(image, 'Beginning Exercise', (20,240),
                        cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0,255,0), 2, cv2.LINE_AA)
            timer-=1

        #display counter on live feed
        cv2.putText(image, str(counter),
                    (10,100),
                    cv2.FONT_HERSHEY_SIMPLEX, 3, (255,255,255), 2, cv2.LINE_AA)

        #display stage on live feed
        cv2.putText(image, 'STAGE', (65,12),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv2.LINE_AA)
        cv2.putText(image, stage,
                    (60,60),
                    cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 2, cv2.LINE_AA)


        # display joints and connections
        mp_draw.draw_landmarks(image, results.pose_landmarks, mp_pose.POSE_CONNECTIONS,
                               mp_draw.DrawingSpec(color=(245,117,66), thickness=2, circle
                               mp_draw.DrawingSpec(color=(245,66,230), thickness=2, circle
                                )

        cv2.imshow('Webcam Feed', image)

        if cv2.waitKey(10) & 0xFF == ord('x'):
            break

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

In [ ]: