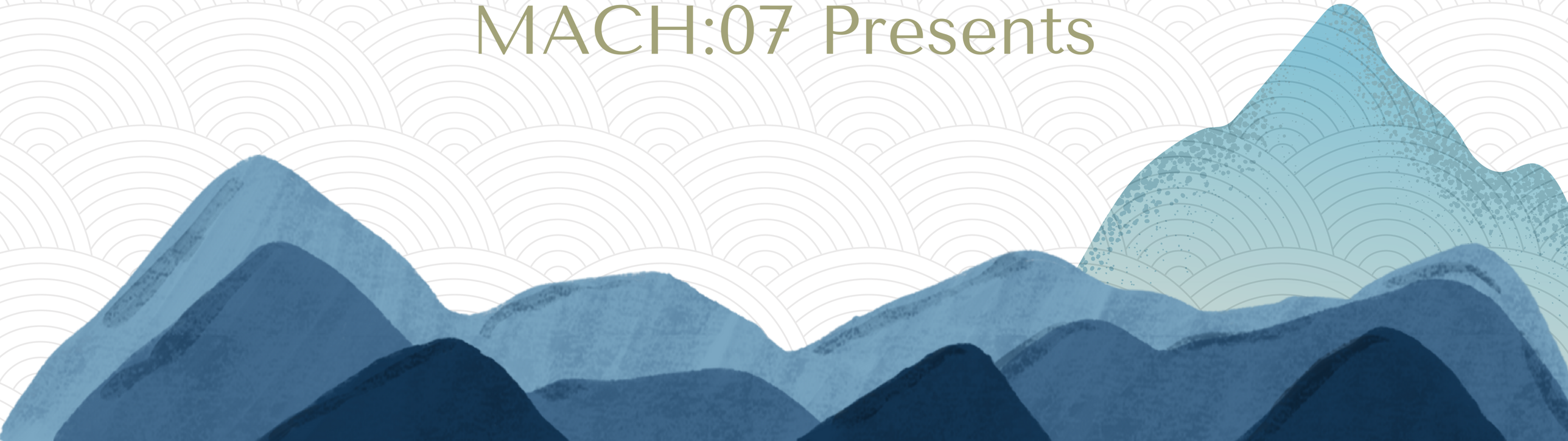
ZENVISION

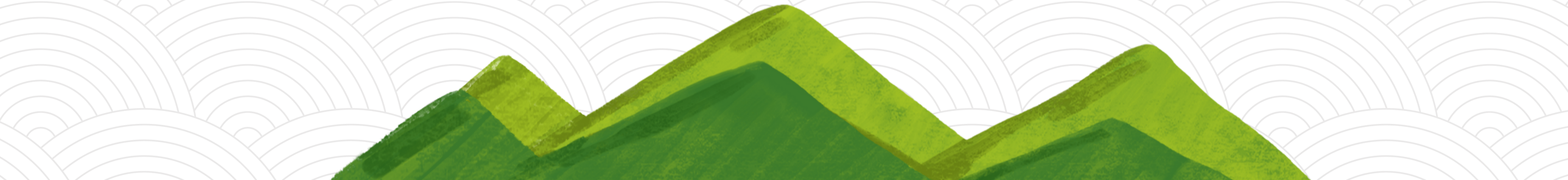
MACH:07 Presents



“YUJ”

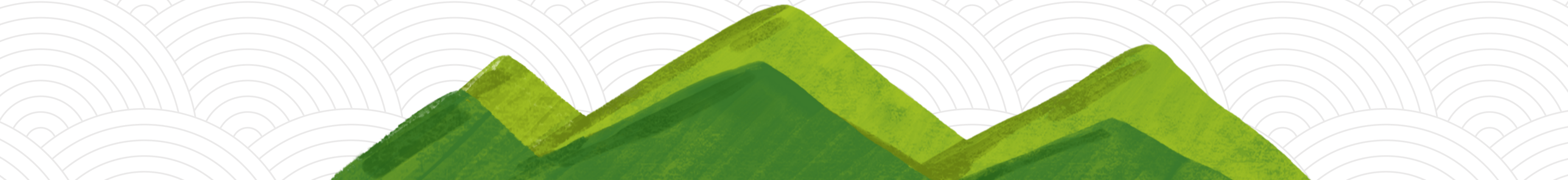
THE SANSKRIT NOUN YOGA IS DERIVED FROM THE ROOT YUJ (युज्) "TO ATTACH, AND JOIN, TO BIND TOGETHER; OR TO UNITE. ITS AIM IS TO YOKE OR CREATE

UNION OF THE BODY, MIND, SOUL, AND UNIVERSAL CONSCIOUSNESS. THIS PROCESS OF UNITING THE PHYSICAL, MENTAL, EMOTIONAL, AND SPIRITUAL ASPECTS OF OURSELVES IS WHAT ALLOWS YOGIS TO EXPERIENCE DEEP STATES OF FREEDOM, PEACE AND SELF-REALIZATION.



PROBLEM STATEMENT

MANY YOGA PRACTITIONERS STRUGGLE TO MAINTAIN PROPER FORM DURING POSES, LEADING TO POTENTIAL INJURIES OR REDUCED BENEFITS. TRADITIONAL METHODS OF RECEIVING FEEDBACK, SUCH AS ATTENDING CLASSES OR HIRING A PERSONAL INSTRUCTOR, CAN BE EXPENSIVE AND TIME-CONSUMING.



APPROACH TO SOLUTION:

We have divided this problem into the following Parts

1.

Data collection

Collect a large dataset of images of individuals performing various yoga poses.

2.

Pose Detection

Use computer vision techniques to detect the user's body and identify the pose being performed.

3.

Pose Analysis

Employ machine learning models to analyze the pose and identify any deviations from the ideal form.

APPROACH TO SOLUTION:

4.

Web Application Development

Develop a web application using Flask that integrates the pose detection, analysis, and feedback generation components.

5.

User Interface

Design a user-friendly interface that allows users to upload videos of their yoga poses and receive feedback.

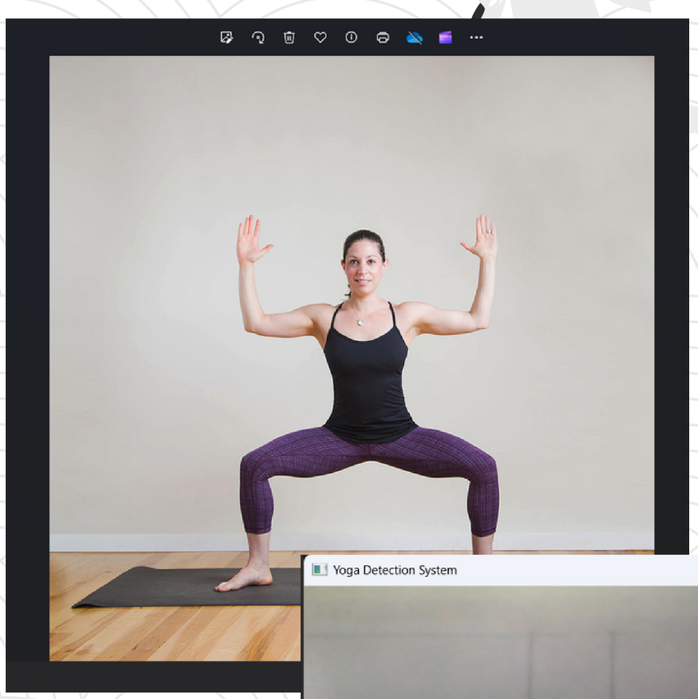
1.

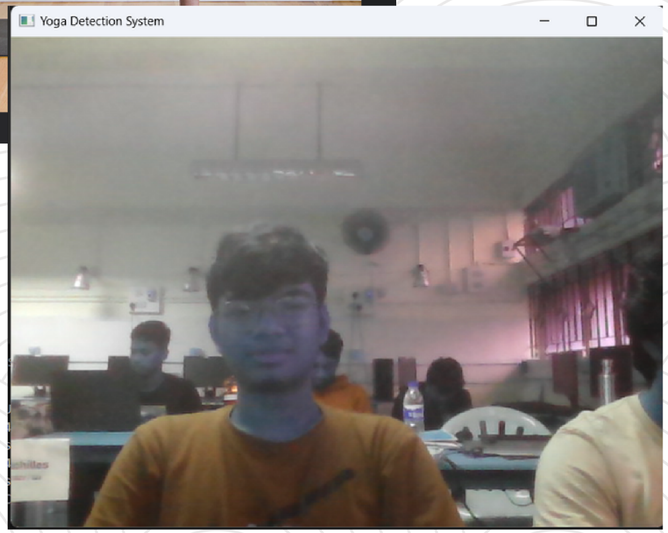
Data collection

Collect a large dataset of images of individuals performing various yoga poses.

data_model.py > ...

```python
img_dir = 'DATASET/DOWNDOG'
def inFrame(lst):
    if lst[28].visibility > 0.6 and lst[27].visibility > 0.6 and lst[15].visibility>0.6 and lst[16].visibility>0.6:
        return True
    return False

name = input("Enter the name of the Asana : ")

holistic = mp.solutions.pose
holis = holistic.Pose()
drawing = mp.solutions.drawing_utils

X = []
data_size = 0

for filename in os.listdir(img_dir):
    lst = []
    sample_image = cv2.imread(os.path.join(img_dir, filename))
    img_copy = sample_image.copy()
    frm = cv2.flip(img_copy, 1)
    res = holis.process(cv2.cvtColor(frm, cv2.COLOR_BGR2RGB))

    if res.pose_landmarks and inFrame(res.pose_landmarks.landmark):
        for i in res.pose_landmarks.landmark:
            lst.append(i.x - res.pose_landmarks.landmark[0].x)
            lst.append(i.y - res.pose_landmarks.landmark[0].y)
A = np.array(lst).reshape(33, 2)
        X.append(lst)
        data_size = data_size+1

print(X)
#np.savetxt(f"{name}.csv", X, delimiter=",")
np.save(f"{name}.npy", np.array(X))
```



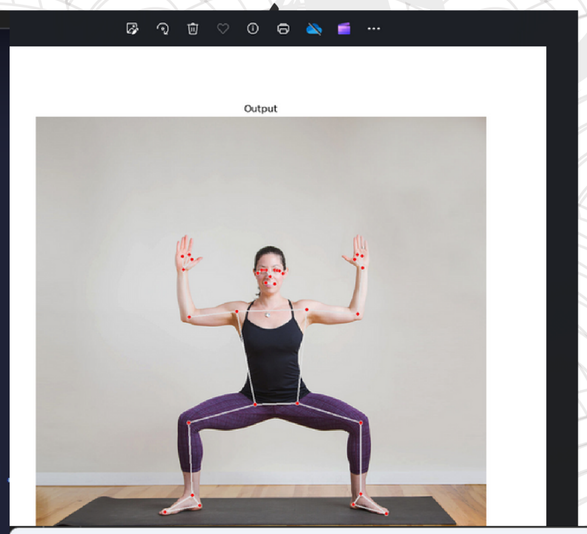


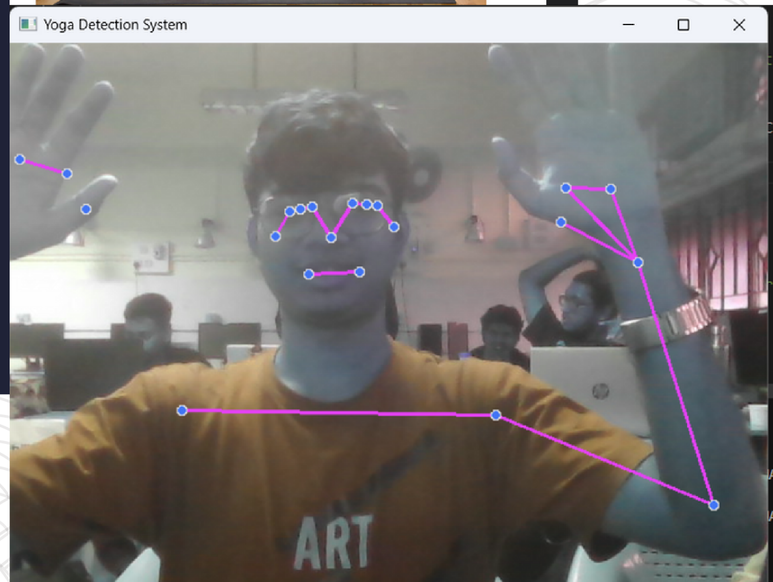
2.

Pose Detection

Use computer vision techniques to detect the user's body and identify the pose being performed.

```
cap = cv.VideoCapture(0)
with mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5) as pose:
    while cap.isOpened():
        ret, frame = cap.read()
        image = cv.cvtColor(frame, cv.COLOR_BGR2RGB) #Recolor Image
        image.flags.writeable = False
        results = pose.process(image)

        image.flags.writeable = True
        image = cv.cvtColor(frame, cv.COLOR_RGB2BGR) #Recoloring it back to BGR

        try:
            landmarks = results.pose_landmarks.landmarks
            shoulder = [landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x,landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value]
            elbow = [landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x,landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].y]
            wrist = [landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].x,landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].y]

            # Calculate angle
            angle = calculate_angle(shoulder, elbow, wrist)

            # Visualize angle
            print(angle)
            #cv.putText(image, str(angle), tuple(np.multiply(elbow, [640, 480]).astype(int)), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255,
        except:
            pass
        mp_drawing.draw_landmarks(image, results.pose_landmarks, mp_pose.POSE_CONNECTIONS) #Drawing Landmarks
```





3.

Pose Analysis

Employ machine learning models to analyze the pose and identify any deviations from the ideal form.

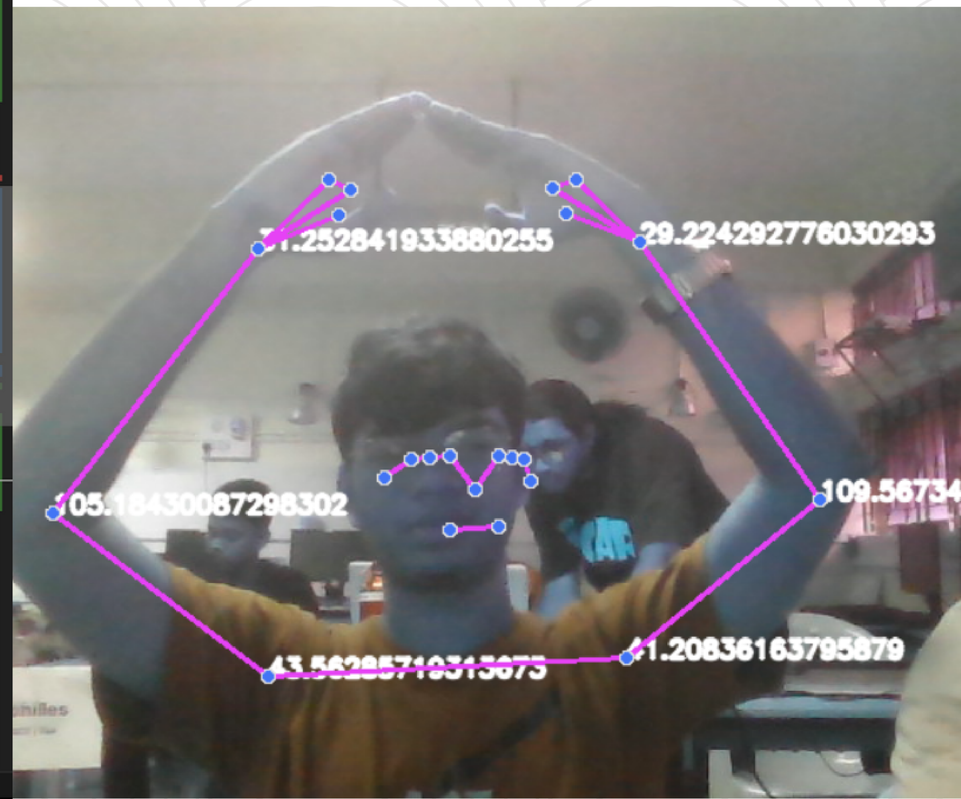
Go Run Terminal Help ZenVision

Zen.py M 00000000.jpg

Zen.py > ...

```
        landmarks = results.pose_landmarks.landmark
        #COLLECTING LEFT SIDE INFORMATION AND DISPLAYING
        left_shoulder = [landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x,landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].y]
        left_elbow = [landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x,landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].y]
        left_wrist = [landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].x,landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].y]
        #LEFT ELBOW
        le = calculateAngle(left_shoulder, left_elbow, left_wrist)
        cv.putText(image, str(le), tuple(np.multiply(left_elbow, [640, 480]).astype(int)), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv.
        #LEFT WRIST
        lw = calculateAngle(left_elbow, left_wrist, left_shoulder)
        cv.putText(image, str(lw), tuple(np.multiply(left_wrist, [640, 480]).astype(int)), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv.
        #LEFT SHOULDER
        ls = calculateAngle(left_wrist, left_shoulder, left_elbow)
        cv.putText(image, str(ls), tuple(np.multiply(left_shoulder, [640, 480]).astype(int)), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2,
        #COLLECTING RIGHT SIDE INFORMATION AND DISPLAYING
        right_shoulder = [landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].x,landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].y]
        right_elbow = [landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].x,landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].y]
        right_wrist = [landmarks[mp_pose.PoseLandmark.RIGHT_WRIST.value].x,landmarks[mp_pose.PoseLandmark.RIGHT_WRIST.value].y]
        #RIGHT ELBOW
        rs = calculateAngle(right_wrist, right_elbow, right_shoulder)
        cv.putText(image, str(rs), tuple(np.multiply(right_elbow, [640, 480]).astype(int)), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv
        #RIGHT WRIST
        re = calculateAngle(right_shoulder, right_wrist, right_elbow)
        cv.putText(image, str(re), tuple(np.multiply(right_wrist, [640, 480]).astype(int)), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv
        #RIGHT SHOULDER
        rw = calculateAngle(right_elbow, right_shoulder, right_wrist)
        cv.putText(image, str(rw), tuple(np.multiply(right_shoulder, [640, 480]).astype(int)), cv.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2,
    except:
        pass
    mp_drawing.draw_landmarks(image, results.pose_landmarks, mp_pose.POSE_CONNECTIONS,mp_drawing.DrawingSpec(color=(245,117,66), thickness=2, ci


    #print(angle)
    # landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].visibility
    # landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value]
    # landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value]
    cv.imshow("Yoga Detection System", image)

    #DETECTION
```

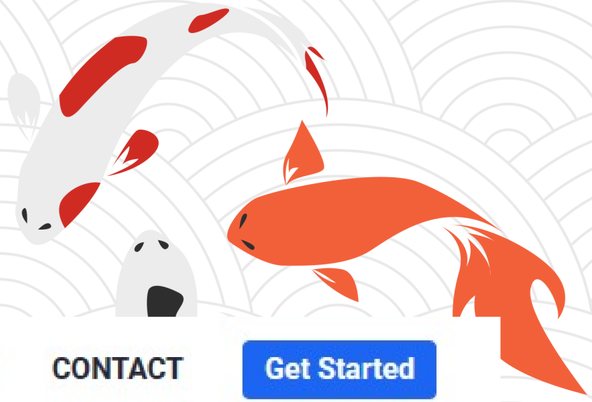
Ln 105, Col 37 Spaces: 4 UTF-8 CRLF Python 3.11.7 ('base': conda) Go Live Prettier



4.

Web Application Development

Develop a web application using Js and React that integrates the pose detection, analysis, and feedback generation components.



ZenVision

HOME SERVICE ABOUT CONTACT Get Started

Personal Guide Help You Build Healthy Life.

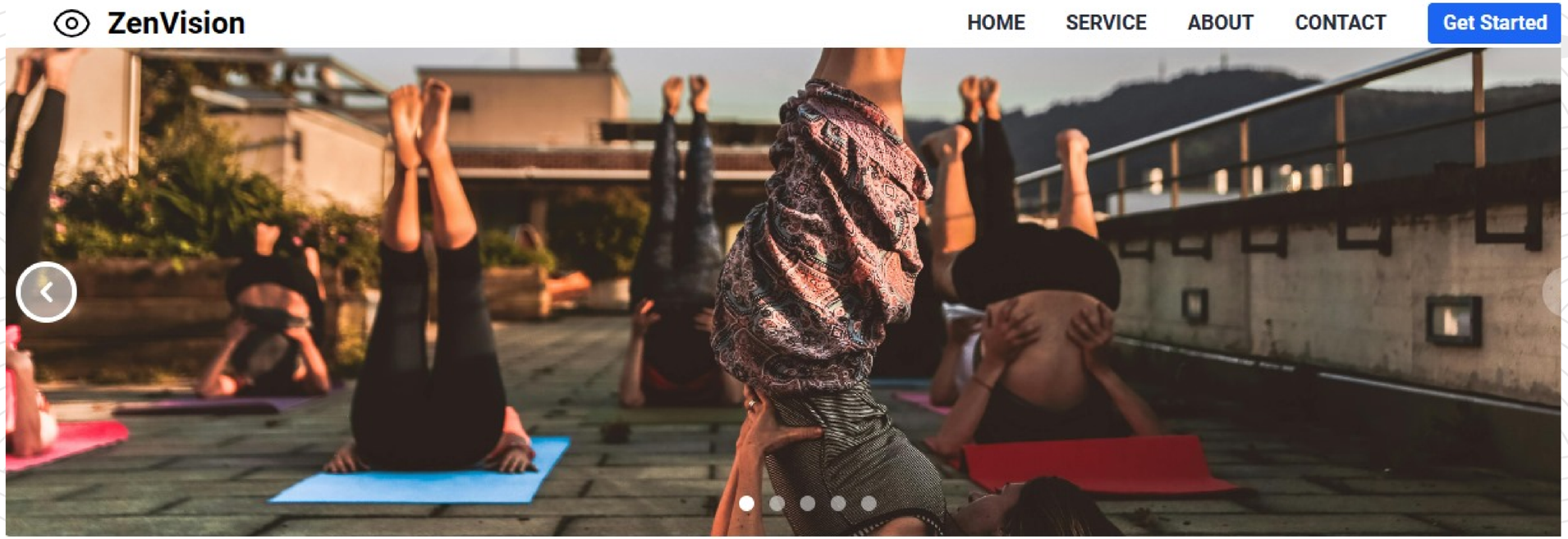
🙏 Yoga helps us connect with our true selves by focusing on mindfulness, breath awareness, and acceptance. 🌞

Try It Free → Learn More

5.

User Interface

Design a user-friendly interface that allows users to upload videos of their yoga poses and receive feedback.



Need Help?

Contact our Customer Support that is always ready to help you with any possible questions, problems or information.

support@zenvision.io

KEY FEATURES

2.

Beginner and User friendly UI

1.

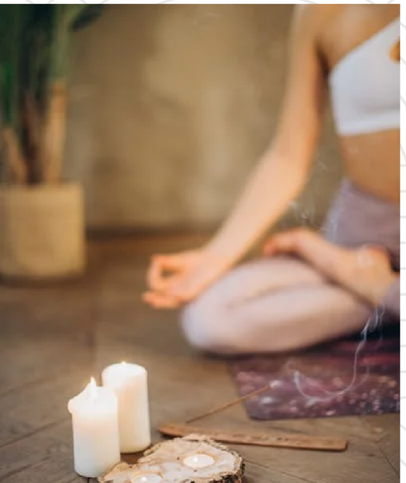
Live Image detection

3.

Customizable Workouts

VISION

Empowering yoga practitioners worldwide to enhance their practice through accessible and personalized feedback, ensuring safe and effective execution of poses for optimal physical and mental well-being.



MISSION

Our mission is to revolutionize the way yoga practitioners receive feedback on their form by providing an affordable, convenient, and personalized solution. Through innovative technology and expert guidance, we aim to empower individuals to deepen their practice, prevent injuries, and unlock the full benefits of yoga, anytime and anywhere.

FUTURE GOALS

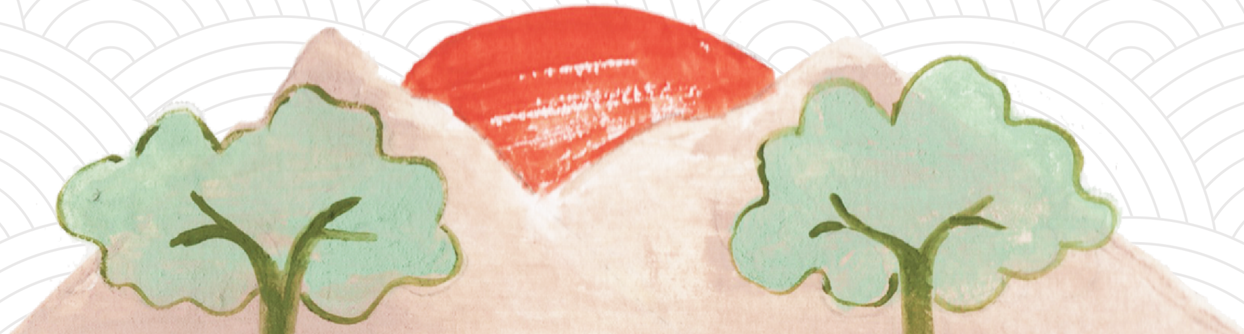
1. Personalized Feedback and Guidance:
Develop advanced algorithms that provide personalized feedback on poses, tailored to the individual's body type, flexibility, and experience level.
Integrate motion capture technology to accurately track body movements and identify areas for improvement.

2. Virtual Reality and Augmented Reality Integration:
Create immersive VR/AR experiences that guide users through poses with real-time feedback and virtual instructors. Allow users to practice yoga in virtual environments, providing a safe and private space for learning.

3. Gamification and Motivation:
Introduce gamified elements to make yoga practice more engaging and motivating. Set challenges, track progress, and reward users for consistency and improvement.

4. Health and Wellness Integration:
Partner with healthcare providers to integrate the AI yoga assistant into holistic health and wellness programs. Monitor users' progress and provide insights into their overall fitness and well-being.

5. Accessibility and Affordability:
Make the AI yoga assistant widely accessible through mobile apps and online platforms. Offer subscription models or tiered pricing to ensure affordability for all users.



CONCLUSION

Through the development and implementation of an ML-powered yoga instructor, we have demonstrated the transformative potential of technology in enhancing the practice of yoga.