# PRD For Pose Estimation Using Skeleton or 3D Model

| ID Number | Name |
|---|---|
| 2023204002 | Shisha Chhotray |
| 2023204012 | Shaik Nayeem Aslam |
| 2023202019 | Shantanu Phatak |
| 2023201031 | Shalu Kumari |

**Project Number : 16, Performed under: Charu Sharma**

**Pose Estimation using Skeleton or 3D Model is a system that aims to accurately estimate the pose or position of a person or object in an image or video**.

**Overview:**
Developing a system or algorithm that can effectively perform pose estimation using either a skeleton-based approach or a 3D model-based approach. A pose detection system is an augmented reality application that takes inputs from video or image to accurately recognize and estimate various human poses across areas like dance, yoga, health & wellness etc.

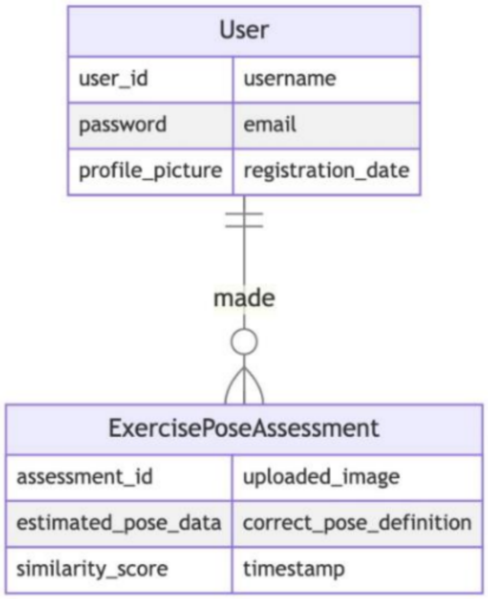
**Aim** : Accurately estimate the pose or position of a person or object in an image or video
**Objective**:
• To design a web-based application that take inputs from an image or video to estimate pose
• To build AI/ML models to detect, estimate and predict poses accurately
• To provide a seamless user experience for the user with intuitive UI

**Repo Link**: https://github.com/ShisaC/16_Pose-Estimation-Devsquad

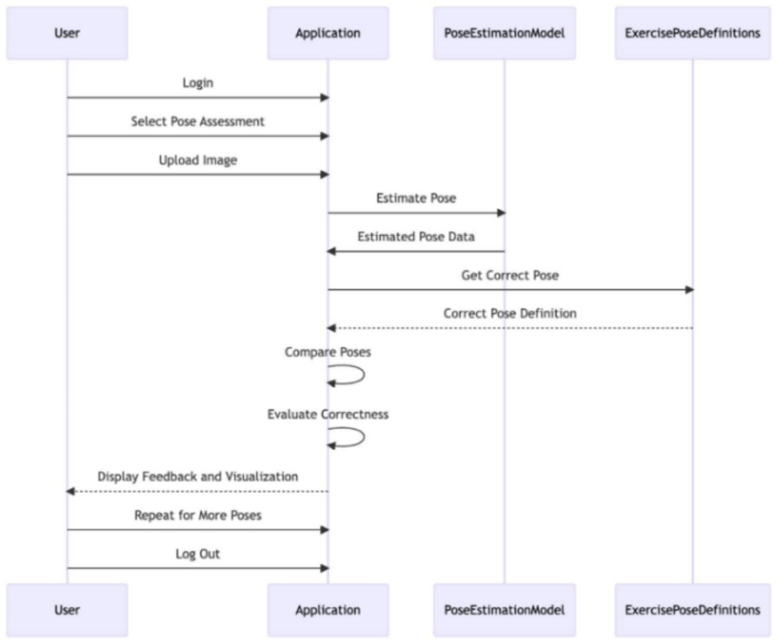
**Targeted Users:**
1.Fitness trainers, health professional, and individuals who want to monitor and form the progress of their
fitness
2.Individuals like guardians, caretakers or researchers who want to recognize sign languages of disabled
3.Dance tutors, instructors or learners who are looking to monitor and improve dance moves
4.Rehabilitation or physical therapists who want to monitor the movements and activities of patients
5.Automotive designers who want provide driving assistance based on human poses to avoid accidents

## SYSTEM DESIGN FLOW, SEQUENCE AND ARCHITECTURE

ERD diagram (Entity-relationship) :



Sequence diagram:



## Use Cases:

**1. Fitness trainers, health professionals, and individuals monitoring fitness progress:**

- Exercise Form Correction
- Progress Tracking
- Virtual Training Sessions

**2. Guardians, caretakers, or researchers recognizing sign languages:**

- Sign Language Recognition
- Assistive Devices
- Educational Tools

**3. Dance tutors, instructors, or learners monitoring and improving dance moves:**

- Dance Routine Analysis
- Choreography Planning
- Performance Evaluation

## Technology Stack

Here's a suggested tech stack for building this system:
**Backend**:
• Programming Language: Python
• Web Framework (if developing a web application): Flask, Django, or FastAPI
• Deep Learning Framework: TensorFlow, PyTorch
• Image Processing Libraries: OpenCV
• Pose Estimation Model: OpenPose, PoseNet, or a custom-designed model
• Exercise Pose Comparison: Custom code or mathematical libraries like NumPy

**Frontend (for web applications):**
• JavaScript, HTML, CSS
• Frontend Framework: React, React Native
• Visualization: Libraries like D3.js (for drawing pose comparisons)
Database (for user data and feedback, if needed):
• Database System: MongoDB
• Object-Relational Mapping (ORM): SQLAlchemy (for Python)

**Deployment and Hosting:**
• Hosting: AWS or Google Cloud or Azure (for web applications)
User Authentication (if needed):
• Authentication Library: OAuth, Firebase Authentication

## Functional Flow and Logic

**Step 1: User Authentication (Sign-In Page):**

- User accesses the web page.
- The web page includes a sign-in mechanism for user authentication.
- Upon successful authentication, the user gains access to the pose estimation functionality.

**Step 2: Pose Estimation**

Authenticated users can now have two ways to estimate pose

1. Manual estimation with input from webcam
2. Upload a dataset containing pairs of images for pose comparison (Automatic Estimation)

We utilize the MediaPipe pose estimation library to train the model to train the model using the dataset uploaded by the user or poses detected through webcam and save the trained model for later use in pose estimation.

**Manual Estimation**

- Select the desired exercise
- Perform it infront of webcam
- If the pose matches with selected exercise, count of repetition increases (if pose doesn't match or idle count won't be displayed)

**Automatic Estimation**

- Select one reference image and upload
- Upload the image to compare and click pose Estimation Request
- The uploaded images are processed using the mediapipe's trained pose estimation model.
- Extract relevant pose features or keypoints from the images.
- Compare the pose features/key points obtained from the two images.
- Calculate the accuracy or similarity score based on the comparison.

## Step 3: Output

**Result Display:**

- Display the pose accuracy result to the user on the web page.
- Provide visualizations or metrics indicating the similarity or differences in the poses.

**Data Storage in MongoDB:**

- Store the pose estimation results and associated metadata in MongoDB.
- This can include history logs like user information, timestamps, and accuracy scores.

## Feature Set

1. **Pose estimation through webcam :** lets users select the desired exercise to perform interact with webcam, where it displays count if performed accurately ( detects and calculates accuracy)
2. **Comparison Of Images :** lets users upload images to system one for reference and one for comparison to calculate similarity
3. **Pose Similarity Score :** calculates score based on similarity of images to give user a visual indication of accuracy of pose implementation and degree correction
4. **History log:** Every time a user performs comparison through manual or automatic, the activities are stored in history tab for future references

## Final Output Images

The overall output will be an web app/mobile app that lets users detect their poses and define the position based on input in format of video/image as per user convenience