Capstone Project

Phase A

BodyTrack

Real-Time Human Posture Evaluation Using Biomechanical Modeling and Machine Learning

Project ID:

25-2-D-4

Authors:

Elad Krauz

Email: elad.krauz@e.braude.ac.il

Uri Ziv

Email: uri.ziv@e.braude.ac.il

Supervisors:

Mrs. Elena Kramer

Dr. Dan Lemberg

Git Repository:

https://github.com/Eladkrauz/BodyTrack

Table of Contents

1 Abstract

BodyTrack is an Android real-time application that monitors and analyzes body posture during strength training exercises in the gym. The system will assist users in performing exercises effectively and safely by providing real-time feedback regarding their form and posture. Ineffective exercise form is a prevalent issue among gym users, and it may lead to injury, muscle imbalance, or suboptimal training outcomes. Many individuals who train independently or without professional help might be unaware they are doing exercises incorrectly.

BodyTrack solves this problem by combining computer vision, biomechanical modeling, and machine learning into an easy-to-use mobile solution. The app uses the smartphone's camera to record the user's exercise, track main body joints, and compare them with correct movement models of selected exercises. The app then gives instant voice and visual feedback whenever it detects incorrect posture or unsafe poses. In addition, the app provides an overview of the user's performance at the end of each session, providing feedback on form consistency, errors, and improvement recommendations.

With machine learning integrated with biomechanics, BodyTrack provides intelligent, context-aware form corrections. The real-time posture analysis and movement evaluation are performed on a remote Python-based server, enabling access to advanced machine learning models and high-precision processing. The Kotlin-based mobile app captures exercise video input and communicates securely with the cloud backend to receive immediate posture feedback. This architecture ensures a responsive, user-friendly experience while leveraging powerful remote computing. With BodyTrack, users can train more confidently, avoid injury risks, and continuously improve their exercise technique over time.

2 Introduction

Strength training has become an essential part of many people's fitness routines, with exercises such as squats, deadlifts, and shoulder presses gaining popularity. While gym equipment is widely accessible, professional guidance is not always available. Many individuals work out alone, without a personal trainer or coach, which increases the risk of poor posture and incorrect exercise technique. Even minor mistakes in form, repeated over time, can lead to injury, joint stress, or long-term muscle imbalances.

This project began with a simple but impactful question: Can we help people train more safely and effectively using just their smartphones? The answer led to the development of BodyTrack, a mobile application designed to monitor exercise movements in real time and provide instant feedback. Acting as a digital personal coach, BodyTrack guides users in correcting posture and technique as they train.

At its core, BodyTrack is based on three fundamental principles: understanding body mechanics, detecting body position, and delivering immediate feedback. Biomechanical modeling defines the joint angles and movement patterns considered correct for different strength exercises. Using pose estimation, a computer vision technique, BodyTrack identifies the user's body joints in real time from the smartphone camera feed. This allows the system to analyze each frame and track how the user's posture changes while performing an exercise. By offering feedback during the workout rather than afterward, BodyTrack helps users make adjustments in the moment – just like a coach standing by their side.

The complete workflow of BodyTrack operates through a structured pipeline that ensures seamless posture evaluation and feedback. First, the mobile device captures live video frames using the rear camera. These frames are sent to a cloud-based server, where a Pose Estimation Module powered by models such as MediaPipe detects key body landmarks. The system then uses a Biomechanical Analysis Engine to calculate joint angles and determine the current movement phase, such as descent or ascent in a squat. If the detected posture does not align with biomechanical standards, the system's Error Detection Logic triggers corrective feedback. This feedback, in the form of visual overlays or audio cues, is transmitted back to the mobile app almost instantly. Users receive this guidance while training, allowing for real-time corrections. After the session, a Session Summary Generator compiles key statistics, including repetition counts, detected errors, and personalized improvement suggestions, all displayed clearly through the app's user interface.

BodyTrack's key advantage is its usability: users need only their Android phone – no special hardware or sensors. They simply select an exercise, position the camera, and begin training. The system handles everything else in the background, from posture tracking to feedback delivery.

By combining pose estimation, biomechanical rules, and real-time feedback in a single, accessible system, BodyTrack offers an intelligent solution for improving exercise performance and preventing injuries. It empowers users to train more safely and confidently, bridging the gap between professional coaching and independent workouts.

3 Literature Review

3.1 Technologies for Pose Estimation

MediaPipe Pose, developed by Google, is a widely adopted real-time pose estimation framework that detects 33 anatomical landmarks across the full human body. It operates through a two-stage pipeline [1]: the first stage uses a lightweight detection model to localize the region of interest (typically the full-body bounding box), and the second stage applies a regression-based convolutional neural network (CNN) to directly estimate the (x, y) coordinates and visibility scores of each joint. Unlike models such as OpenPose that rely on heatmap-based key point detection followed by spatial association algorithms (will be discussed in the next section), MediaPipe Pose uses coordinate regression to simplify the inference process. Its neural network backbone is based on a ResNet-like architecture, optimized for real-time inference and efficient computation, making it particularly suitable for latency-sensitive applications like fitness tracking and posture correction. To enhance temporal stability and reduce noise between frames, MediaPipe includes internal smoothing filters and coordinate normalization techniques. It supports both 2D output and approximate 3D (Z-axis) estimation from a monocular RGB camera, allowing applications to infer depth without requiring additional hardware. The framework is available across multiple platforms via APIs for Python, JavaScript, Android (via TensorFlow Lite), and iOS, enabling integration in both mobile and cloud-based systems. Its modular design, cross-platform support, and low computational cost have contributed to its growing use in domains such as physical therapy, sports analysis, and interactive applications.

OpenPose, from Carnegie Mellon University, is a bottom-up pose estimation system designed for high-accuracy, multi-person detection. The core algorithm predicts heatmaps for key point locations and Part Affinity Fields (PAFs) – vector fields that encode limb direction and association [2]. A greedy parsing algorithm then connects key points into complete skeletons. This method allows OpenPose to resolve complex scenes with overlapping individuals. The underlying architecture is based on multi-stage CNNs, often implemented with VGG or ResNet backbones. Although computationally demanding, OpenPose offers superior accuracy and dense landmark coverage (like hands, feet, face), making it ideal for research and clinical settings. However, its resource requirements limit its practicality for real-time mobile use.

PoseNet is a pose estimation model developed as part of TensorFlow.js and TensorFlow Lite for browser and mobile platforms [3]. Using a single convolutional neural network, PoseNet directly regresses 2D locations of 17 body key points from RGB images. Unlike other systems that first detect bounding boxes, PoseNet simplifies the pipeline with a single-pass approach at the cost of some anatomical accuracy for real-time performance. It is effective with lightweight skeletons like MobileNet and ResNet-50 and thus can be deployed on devices with less hardware. PoseNet might not have detailed data like 3D data or specific joint sets. Still, it is easy to implement and integrate and thus perfect for lightweight apps and initial experimentation in fitness tracking and pose estimation.

3.2 Biomechanical Analysis Techniques

Joint angle measurement is one of the primary biomechanical methods used to measure the quality of posture and detect movement faults. After calculating significant joint coordinates through pose estimation, vectors are created between successive points, and angles are calculated using vector trigonometry. The dot product formula is used as the standard:

$$\Theta = arccos\left(\frac{\vec{A}\cdot\vec{B}}{\|\vec{A}\|\cdot\|\vec{B}\|}\right)$$

This method helps measure limb positions with precision, for example, the angle of the thigh and calf when one performs a squat. These are then compared to standard models of correct form to establish if the movement is within acceptable biomechanics. This rule-based comparison forms the foundation of real-time evaluation in mobile apps where consistent monitoring of joints is essential for consistent feedback [4].

Kinematic analysis extends static joint measurements by incorporating the time dimension of movement. It quantifies displacement, velocity, and acceleration of body segments as a function of time via numerical differentiation. Given a time series of joint positions x(t), velocity v(t) and acceleration a(t) are calculated as:

$$v(t) = \frac{dx(t)}{dt}, \qquad a(t) = \frac{dv(t)}{dt}$$

These measures help identify conditions like asymmetrical motion, unstable pacing, or compensation patterns during exercise. For this project, kinematic analysis aids in identifying sudden changes to movement flow, allowing for the identification of wrong or incomplete movements that would not be evident from static angles alone. Practical implementations apply sliding window algorithms or low-pass filters to filter out noise and provide real-time responses.

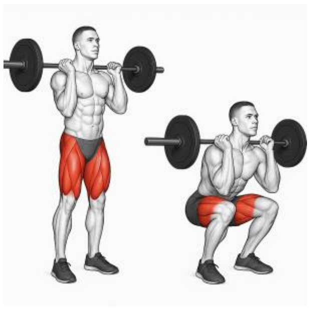
Motion capture (MoCap) [1] systems provide ground truth for posture analysis and are the benchmark for biomechanical testing. Traditional optical MoCap employs reflective markers to designate anatomical landmarks used by multiple infrared cameras. Sound algorithms are then used to triangulate each joint position with sub-millimeter accuracy. Inertial MoCap incorporates IMU (Inertial Measurement Unit) sensors by utilizing gyros and accelerometers worn on the body, which apply kinematic techniques for tracking motion. Markerless systems are also available, and vision-based models (OpenPose, MediaPipe) are used to reconstruct the skeleton without markers placed on the body. Calibration and synchronization algorithms are critically crucial for MoCap accuracy. Due to prohibitive costs and complexity, MoCap cannot be identified as a solution to everyday mobile use. Still, MoCap is a gold standard for validating mobile posture estimation algorithms and is required for training machine learning model pipelines for this project.

3.3 Biomechanical Reference Models for Strength Exercises

This section provides detailed biomechanical reference models for the exercises supported in the initial version of BodyTrack. Each model outlines the phases of movement, expected joint angles, and posture checkpoints throughout the execution of a single repetition. These models serve as the basis for the real-time evaluation and feedback logic implemented in the system.

3.3.1 Squat – Phases of Movement and Joint Angle Ranges

The squat is a compound lower-body exercise involving coordinated movement at the hip, knee, and ankle joints [4]. The goal is to lower the body while maintaining proper alignment and joint angles, then return to the original upright position. This movement pattern requires core stability, joint mobility, and control over load distribution.



Phase 1: Starting Position

- **Posture:** The individual stands upright with feet positioned shoulder-width apart and toes slightly pointed outward (~5°–15°), arms extended forward or crossed at the chest for balance.
- **Key Joint Angles:**
    - **Hip Angle:** ~180° (neutral standing, full extension)
    - **Knee Angle:** ~180° (neutral standing)
    - **Ankle Angle:** ~90° (shins vertical)
    - **Spine/Trunk Angle:** Erect posture, slight lumbar curve maintained

Phase 2: Descent (Eccentric Phase)

- **Initiation:** Movement begins by driving the hips backward, followed by simultaneous flexion at the hips and knees. The trunk naturally leans forward to maintain balance.
- **Descent Control:** The descent should be smooth and continuous, with the knees tracking in line with the feet.
- **Target Joint Angles (at maximum descent):**
    - **Hip Flexion:** 70°–90°
    - **Knee Flexion:** 80°–100°

    - **Ankle Dorsiflexion:** 60°–80°
    - **Trunk Forward Tilt (from vertical):** 30°–45°
- **Positional Requirements:**
    - Heels remain in contact with the ground.
    - Knees do not extend beyond the toes significantly.
    - Pelvis remains neutral (no excessive posterior pelvic tilt).

Phase 3: Bottom Position (Transition Point)

- **Posture:** The thighs reach a position where they are parallel or slightly below parallel to the floor.
- **Stillness:** The bottom position is brief and should not include a "bounce" or loss of control.
- **Checkpoint:** This is the reference point used to determine squat depth validity.
- **Minimum Requirement:** Knee flexion must reach at least 100° (visual cue: hip crease below knee level).

Phase 4: Ascent (Concentric Phase)

- **Initiation:** Drive upward by extending the hips and knees simultaneously while keeping the spine neutral and core engaged.
- **Joint Angle Progression:**
    - **Hip Angle:** Moves from ~80° to ~180°
    - **Knee Angle:** Moves from ~90° to ~180°
    - **Ankle Angle:** Returns from ~70° to ~90°
- **Posture Control:**
    - Trunk angle should gradually return to vertical.
    - Movement should remain symmetric with no lateral deviation of knees or torso.

Form Validation Rules for Implementation

To be considered a correct repetition:

- **Descent must reach depth:** Knee flexion angle ≤ 100°.
- **Symmetry between limbs:** Angle deviation between left and right knees/hips should be ≤ 10°.

- **Trunk tilt should not exceed:** 45° from vertical.
- **Heels must stay down:** No frame during the descent or ascent should show ankle extension beyond 100°.
- **Controlled pace:** No abrupt acceleration/deceleration between frames; smooth curve in joint angle vs. time graph.
- **Knee tracking:** Inward movement (valgus) during descent flagged if medial shift exceeds 10° from foot orientation.

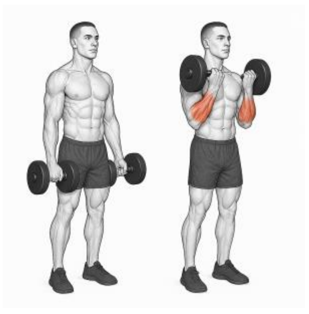
Common Errors and Detection Logic

Error Type	Biomechanical Indicator	Detection Strategy
Shallow Squat	Knee angle > 110° at lowest point	Check knee angle over full frame window
Knee Valgus	Distance between knees reduces; knees collapse inward	Monitor X-axis displacement of knees
Trunk Collapse	Trunk lean > 45° from vertical	Angle between hip-shoulder and vertical
Heel Lift	Ankle angle > 100°, indicating plantarflexion (heel raised)	Ankle Y-position elevated relative to foot base
Asymmetric Descent	Left and right knee angles differ > 10° during key descent frames	Compare joint angles per side over time

This model defines the biomechanical standards against which squat execution is evaluated in BodyTrack. All logic and detection rules are based on these measurable criteria, allowing the system to reliably identify and report incorrect movement patterns in real time [5].

3.3.2 Biceps Curl – Phases of Movement and Joint Angle Ranges

The biceps curl is an isolation exercise focused on the elbow joint and the biceps brachii. Proper execution involves elbow flexion and extension with minimal movement in the shoulders, torso, or wrists [4].



Phase 1: Starting Position

- **Posture:** The user stands upright with arms fully extended down, palms facing forward, holding dumbbells or a barbell. Feet are hip-width apart.
- **Key Joint Angles:**
  - **Elbow Angle:** ~180° (full extension)
  - **Shoulder Angle:** ~0° (arms by the side)

    - **Torso Angle:** Upright and stable

Phase 2: Concentric Curl (Lifting Phase)

- **Initiation:** Elbow flexion begins with minimal shoulder involvement.
- **Target Joint Angles at Peak:**
    - **Elbow Flexion:** 45°–60° (not touching the shoulders)
    - **Shoulder Movement:** Should remain within ±10°
- **Requirements:**
    - Elbows remain fixed close to the torso.
    - Wrists stay neutral (not flexed or extended).
    - No backward torso swing or hip extension.

Phase 3: Peak Contraction

- **Posture:** Arms are flexed, weights near shoulder height, with maximum biceps engagement.
- **Stillness:** Brief hold of 0.5–1 second.

Phase 4: Eccentric Phase (Lowering)

- **Execution:** Elbows extend smoothly back to starting position.
- **Joint Angle Return:**
    - **Elbow Angle:** Returns from ~50° to ~180°
    - **Torso:** Remains upright
- **Control:** Movement is slow and controlled without letting gravity drop the weight.

Form Validation Rules for Implementation
To be considered a correct repetition:

- **Full range elbow motion:** Flexion ≤ 60°, extension ≥ 170°
- **Minimal shoulder displacement:** ≤ ±10°
- **Stable elbow position:** X-axis movement of elbow joint minimal during rep
- **No torso swing:** Angle between hip and shoulder remains constant
- **Neutral wrist posture:** No wrist extension > 20°

Common Errors and Detection Logic

Error Type	Biomechanical Indicator	Detection Strategy
Partial Range	Elbow angle does not reach ≤ 60° or extend past 160°	Track max/min elbow angles across rep
Shoulder Involvement	Shoulder angle changes > 10° during curl	Compare shoulder joint Y-position over time
Torso Swing	Change in hip-shoulder angle ≥ 15° during lift	Monitor torso lean throughout rep
Elbow Drift	Lateral displacement > 5–10 cm during curl	Track elbow X-position relative to torso
Wrist Curl	Wrist joint angle exceeds 20° flexion or extension	Measure angle between forearm and hand

This biceps curl model defines precise joint-based criteria for correct movement execution and supports real-time evaluation in BodyTrack's feedback system [5].

3.3.3 Lateral Shoulder Raise – Phases of Movement and Joint Angle Ranges

The lateral shoulder raise is an isolation exercise that targets the middle deltoid. It involves abduction of the arm in the frontal plane and requires shoulder joint control with stable torso posture. Proper execution minimizes compensation from the trapezius, torso, or momentum [4].

Phase 1: Starting Position

- **Posture:** The user stands upright with dumbbells at their sides, arms straight or slightly bent, palms facing the body.
- **Key Joint Angles:**
  - **Shoulder Abduction:** ~0°
  - **Elbow Flexion:** ~10°–20° (slight bend for joint safety)
  - **Torso:** Upright, no lateral tilt

Phase 2: Abduction Phase (Lifting)

- **Initiation:** The movement starts with shoulder abduction, lifting the arms laterally to shoulder height.
- **Target Joint Angles at Peak:**
  - **Shoulder Abduction:** 80°–90° (arms parallel to the floor)
  - **Elbow Flexion:** Maintained at 10°–20°

  - **Torso Tilt:** < 5° lateral shift
- **Requirements:**
  - Arms lift evenly, maintaining symmetry.
  - No leaning, swinging, or shrugging.
  - Wrists stay neutral and aligned with the forearms.

Phase 3: Peak Position

- **Posture:** Arms held momentarily at shoulder height, parallel to the ground.
- **Control:** No jerking or sudden drops; brief pause of 0.5 seconds.

Phase 4: Adduction Phase (Lowering)

- **Execution:** Arms are lowered slowly to the starting position under control.
- **Joint Angle Return:**
  - **Shoulder Abduction:** Returns to ~0°
  - **Elbow Flexion:** Maintained at ~10°–20°
- **Requirements:**
  - Avoid letting arms fall passively.
  - Movement should mirror the lifting phase in tempo.

Form Validation Rules for Implementation:
To classify a valid repetition:

- **Shoulder Abduction:** Peak angle ≥ 80°, return to ≤ 10°
- **Symmetry:** Left and right arms within ±10° of each other during lift
- **Torso Stability:** No trunk deviation > 5° during entire rep
- **Elbow Control:** Maintain 10°–20° bend without significant variation
- **Controlled Tempo:** Consistent angular velocity; avoid sudden changes in speed

Common Errors and Detection Logic

Error Type	Biomechanical Indicator	Detection Strategy
Incomplete Raise	Shoulder angle < 80° at peak	Track peak shoulder abduction per arm
Uneven Arms	Shoulder angle difference > 10° between arms	Compare left vs. right shoulder angle
Torso Lean	Lateral torso angle > 5° during lift	Track trunk angle from vertical

Elbow Straightening	Elbow angle decreases < 5° or extends completely	Monitor elbow flexion angle throughout
Arm Swing	Acceleration spikes, trunk/hip velocity spikes detected	Derivative of shoulder angle > threshold
Shoulder Shrug	Elevation of shoulder joint > baseline by > 5 cm	Monitor Y-axis movement of shoulder landmarks

This model provides the biomechanical framework used by BodyTrack to evaluate lateral shoulder raises. Real-time detection logic uses these joint metrics and positional constraints to determine movement accuracy and provide feedback [5].

3.4 Machine Learning in Movement Evaluation

3.4.1 Machine Learning Models for Posture Classification and Anomaly Detection

Machine learning forms the foundation of intelligent posture assessment systems. Supervised learning is the most typical paradigm applied in this application. Prototypes are trained with labeled video data to classify types of movement or spot incorrect forms. Convolutional neural networks (CNNs) have been established for spatial feature capturing, whereas temporal patterns employ Recurrent Neural Networks (RNNs), mainly Long-Short-Term Memory (LSTM) networks. This allows capturing temporal change, which may characterize upward and downward motion, as in the case of a squat [4].

In practice, pose sequences are converted to feature vectors (such as joint angles or relative positions) and used to train classifiers, decision trees, support vector machines (SVM), or within a more comprehensive architecture like deep learning frameworks. Once trained, models can be served in production environments using frameworks like TensorFlow, PyTorch, or ONNX Runtime. For example, transfer learning simplified model generality (such as COCO and Human3.6M) to domain-specific contexts with little labeled data.

Anomaly detection algorithms can also be employed utilizing clustering algorithms (like k-means) or autoencoders to detect anomalies in movement from learned movement patterns. These algorithms are helpful for detecting dangerous forms, compensatory efforts due to fatigue, and erratic repetitions. Machine learning can also afford personalization so the system can update thresholds and feedback based on individual characteristics and performance history, a key advantage in adaptive coaching focused on fitness.

3.4.2 Injury Prevention and Performance Optimization Using Machine Learning

One of the significant applications of machine learning in the analysis of human movement is its role in preventing injury and improving athletic performance. Machine learning models can identify minor changes in movement patterns [4] – i.e., asymmetrical behaviors of joints, unstable trajectories, and fatigue onset – by examining the data of poses and movements over time, which have minor changes that can lead to injury development during training but will be told early enough to prevent an injury from developing [6].

Injury prediction models typically utilize time series data of pose estimations to analyze joint angles, balance patterns, and movement velocity. After that, Recurrent Neural Networks (RNN), specifically LSTM networks, are employed to recognize the movement patterns that increase injury risk (like knee valgus during squats, lateral sway of the trunk during lunge). Anomaly detection methods, like clustering or reconstruction error methods in autoencoders, can also identify unsafe or abnormal movement patterns without explicitly defined labels for every risk case.

Aside from injury prevention, performance optimization is another area where machine learning offers benefits in real-time. They can detect motions and models to understand biomechanically efficient movements, supporting the individual in developing safe and effective movement patterns. Recently, the literature has been exploring reinforcement learning approaches so that feedback and goals [6] may change based on the motor learning process of the individual. Therefore, machine learning will ultimately keep the injury risk to the user at something below 1% and boost the potential for continuous skills development and long-term athlete progression.

3.5 Real-Time Feedback in Mobile Fitness Applications

Real-time feedback systems create an exercise activity with coaching by providing real-time feedback on user movement. This approach leverages video input and real-time pose estimation and means to evaluate proper form using rules-based and learned models to offer means of feedback [5]. When inconsistency occurs, the real-time feedback system can render feedback using user interface (UI) alerts or audio instructions. Audio is preferred as feedback using the Android Text-to-Speech (TTS) to allow users to receive corrections without change during the exercise experience.

To minimize latency, optimal video frame rate processing pipelines and inference engines (like TensorFlow) are made efficient so that the overall feedback loop, including camera input, processing, and user notification, does not exceed the 'real-time' feeling. There are a lot of time and motivation factors for how algorithms make decisions on the delivery of feedback, such as using sliding-window filters, threshold triggers, and confidence scoring mechanisms to regulate feedback over the individual exercise phase to avoid providing too much or repeated prompts. Successful real-time feedback systems are ultimately based on human-computer interaction (HCI) principles, designed to be clear, timely, motivating, and not overwhelming. During this project, real-time feedback will be produced whenever joint angles are outside the expected ranges or when a movement phase is missed to provide both correction and advancement of the exercise in real-time.

3.6 Existing Applications and Tools for Movement Evaluation

Several commercial and academic tools exist with features that evaluate movement, but these tools vary in accuracy, scope, and accessibility. Freeletics and Nike Training Club provide tailored workouts but do not offer real-time posture tracking, taking advantage of timetables and self-reporting from the participant. Kaia Health focuses on physical therapy and has a smartphone camera-based movement sensing tool with some limited visual feedback; however, it is primarily based on clinical exercises and doesn't even attempt full-body pose tracking.

More advanced systems like Mirror, Tonal, and Peloton Guide include a camera and utilize AI, emphasizing performance more. The system shows the user when they are a rep behind scheduled synchronous workouts and provides motion matching with an avatar or trainer visible on the screen. These systems typically include rep counting and potentially show some comparison via machine learning models for motion analysis but don't usually do biomechanical evaluations or joint angle analysis. Users often have difficulty ownership of the hardware used [5].

In contrast, our project is a real-time posture evaluation tool that uses only a smartphone camera as input, and machine learning analysis as output. Our system bridges the gap between lightweight fitness tools and expensive hardware platforms by combining a relatively efficient pose estimation tool (using MediaPipe [1]), our joint angle calculations, and rule-based feedback into one complete system that can be accessed by a user. Through this, we provide a scalable evaluation of movement that supports safety and performance for everyday users.

4 Problem Definition

Strength training exercises require precise control of posture and technique to be safe and effective. However, many gym-goers cannot access real-time professional feedback while they train. Even minor deviations from the correct form can, over time, cause joint stress, muscular imbalances, and injury.

Currently, most solutions require in-person coaching or retrospective video analysis, which does not help users correct their form during the workout. There is a clear need for a system to monitor exercise form live and provide immediate, actionable guidance to the user.

<u>The specific challenges our project aims to address include:</u>

- **Lack of real-time feedback:** Users often cannot know if their form is incorrect during a set. [7]
- **Accessibility:** Not everyone can access personal trainers or professional coaching during workouts.
- **Injury prevention:** Poor form over time can lead to significant injuries that might have been prevented with early correction.
- **User convenience:** The solution must work efficiently with minimal setup, using only devices users already have (like smartphones).

<u>Our solution, BodyTrack, addresses these challenges by:</u>

- Using pose estimation to track the body in real-time through the device's camera.
- Applying biomechanical rules to detect improper joint angles, movement patterns, and posture faults.
- Giving immediate feedback via visual overlays and audio cues to guide the user.
- Providing a session summary to reinforce correct habits and highlight improvement areas.

By merging computer vision technologies [5] with practical exercise knowledge, we aim to create a system that evaluates and improves the quality of exercise performance while making training safer and more effective for everyone.

5 Engineering Process

5.1 Process

5.1.1 Development Approach

The development of BodyTrack is based on a modular and structured approach aimed at creating a real-time, reliable, and user-friendly mobile application for posture analysis during fitness exercises [1]. The system is built for Android smartphones and communicates with a remote server that performs advanced pose estimation and movement analysis. This architecture enables the mobile app [7] to remain lightweight and responsive while leveraging powerful machine learning models hosted in the cloud.

The project follows a logical division into several independent yet connected modules, distributed between the mobile application and the cloud server:

- **Pose Estimation Module (Backend):** Responsible for detecting and tracking body landmarks from video frames sent by the mobile app. This module uses a server-side pose estimation framework [1] (such as MediaPipe Pose) to extract joint coordinates.
- **Biomechanical Analysis Module (Backend):** Applies predefined biomechanical models to the estimated poses. It evaluates joint angles, postures, and movement patterns to determine whether they align with safe and effective standards for the selected exercise [4].
- **Feedback Engine (Backend):** Based on the biomechanical evaluation, this module generates real-time feedback instructions. The feedback is then transmitted to the app to help users correct errors immediately during training [6].
- **Session Summary Generator (Backend):** At the end of each session, the server compiles a report summarizing the number of correct and incorrect repetitions, common movement faults, and personalized suggestions for improvement.
- **User Interface (UI) Layer (Mobile App):** The Android application provides a minimalistic and intuitive interface [8]. Users can select exercises, start a session, receive real-time feedback from the server, and view post-session summaries – all while the computational logic runs in the background on the cloud.

Development is carried out using Android Studio with Kotlin for the mobile client and Python for the backend server. The mobile application handles user interface, camera capture, and communication, while machine learning and pose estimation are processed on the remote server using Python frameworks such as MediaPipe and Flask.

The development process is placed on:

- **Real-time performance:** Ensuring fast and smooth feedback through efficient communication between the app and the server, with minimal delay.
- **System robustness:** Handling typical challenges such as partial body visibility, varying movement speeds, and unstable lighting conditions.

- **User privacy and security:** Ensuring secure data transmission to the backend and protecting user information through encrypted communication protocols.

This approach enables us to build a lightweight yet powerful solution that delivers accurate feedback in real time, offering gym users a practical and intelligent tool for safer and more effective training sessions [4].

5.1.2 Motivation

Strength training has become an essential part of many people's daily lives, whether for health, fitness, or athletic goals. While correct exercise technique is well documented and promoted, improper form during training remains a widespread problem. It can lead to muscle imbalances, joint stress, or preventable injuries. Many gym-goers unknowingly develop faulty movement patterns, often due to a lack of professional supervision or feedback.

BodyTrack was developed to bridge the gap between the need for real-time form correction and the ability to access it anywhere using just a smartphone. We believe that by combining fundamental biomechanical principles with modern machine learning and computer vision techniques, it's possible to deliver intelligent feedback that would traditionally require the guidance of a professional trainer.

The system is designed to help users execute movements with proper technique, engage the right muscles, and minimize unnecessary joint stress. This not only improves the safety and efficiency of workouts but also supports long-term strength gains, postural awareness, and injury prevention.

We also recognize the need for solutions that balance convenience, accessibility, and data security. By using a lightweight mobile interface connected to a secure remote server, BodyTrack enables high-quality, real-time analysis without compromising the user experience.

Ultimately, our motivation is to empower anyone, regardless of location or experience level, to train smarter, safer, and more effectively by leveraging the intelligent technologies available today.

5.1.3 Development Plan

BodyTrack development will follow specific logical stages, ensuring a structured and efficient workflow. Each stage builds on the previous one, allowing for continuous progress from research to a ready-to-use application.

Stage 1: Research and Planning

- Study biomechanical models for selected exercises (squats, biceps curls, lateral raises).
- Review and choose the most appropriate pose estimation framework (MediaPipe Pose).

- Define and finalize the system’s functional and non-functional requirements.

<u>Stage 2: System Design</u>

- Design the complete system architecture, including the core modules: Pose Detection, Biomechanical Analysis, Feedback Engine, and User Interface.
- Prepare all necessary UML diagrams.
- Design a fundamental data structure for tracking session results and feedback.

<u>Stage 3: Core Implementation</u>

- Implement the pose estimation module using the selected framework.
- Develop the biomechanical analysis logic, including joint angle tracking and exercise-specific movement models.
- Build a real-time feedback engine that provides audio and visual guidance during workouts.

<u>Stage 4: User Interface Development</u>

- Design and build a clean, user-friendly Android application.
- Integrate the live camera view, pose tracking overlays, feedback alerts, and session summary screen.

<u>Stage 5: Integration and Testing</u>

- Integrate all modules into a complete, operational system.
- Perform extensive functionality testing to verify real-time detection accuracy and feedback timing.
- Test application performance across different Android devices to ensure stability and responsiveness.

<u>Stage 6: Final Refinement and Documentation</u>

- Refine the user experience based on internal usability tests.
- Complete full project documentation, including the Engineering Book, architecture diagrams, and a short user guide.

5.1.4 Constraints

During the development and implementation of BodyTrack, several technical and practical constraints must be considered:

<u>Real-Time Processing Requirements:</u>

- The system must provide immediate feedback during exercise execution, which requires fast and efficient communication between the mobile app and the server.
- Both backend inference and client response must be optimized to minimize delay between user movement and received feedback.

Mobile Device and Connectivity Constraints:

- The Android application must run smoothly on a range of smartphones, maintaining responsiveness despite variations in hardware capabilities.
- Stable internet connectivity is required to enable real-time server-based pose analysis and feedback.
- In cases of connectivity loss or delay, the app must handle interruptions gracefully without disrupting the user experience.

Camera and Environment Variability:

- The system's accuracy depends on the camera's ability to capture the user's full body during movement.
- Lighting conditions, background clutter, and camera positioning can impact pose detection and must be addressed through both guidance to the user and backend algorithm robustness.

Exercise Selection Scope:

- The initial version focuses on a defined set of exercises: squat, biceps curl, and lateral shoulder raise.
- Expanding to new exercises will require developing and validating additional biomechanical reference models and analysis logic.

User Privacy and Data Handling:

- While real-time video frames are transmitted to the server for analysis, no videos or images are stored.
- All data transfer is encrypted, and the system is designed to avoid collecting personally identifiable information.
- Privacy policies and minimal data retention practices must be followed to ensure user trust and regulatory compliance.

5.1.5 Interview Questions and Research Basis

To evaluate BodyTrack's real-world needs and ensure it addresses issues that gym users face, we conducted interviews and questioning at Holmes Place Kiryon, one of the largest fitness centers in Haifa District.

Our goals with the interviews were to understand the common problems associated with exercise technique, the access issues related to professional supervision, and whether users are invested in the need for a real-time tool to correct form.

Summary of Key Questions Asked:

- Do you feel confident that your exercise form is correct during training sessions?
- Have you ever experienced discomfort or injury that you believe was related to poor exercise posture?

- Would you find value in receiving real-time feedback about your technique while training?
- How often do you receive guidance from trainers during your workouts?
- Would you prefer a mobile application that offers immediate posture correction feedback?

Findings from the Interviews:

- Many participants reported uncertainty about their form, especially when performing complex exercises like squats and deadlifts.
- Several users mentioned minor injuries or persistent joint discomfort, which they suspected could have been avoided with better technique.
- Most users expressed strong interest in an application that could provide real-time guidance without requiring the presence of a trainer.
- There was a general preference for a simple, private, and easily accessible solution that could operate directly on their smartphones.

These findings support the need for BodyTrack as a practical tool to improve training quality, reduce injury risks, and make professional-level feedback accessible to independent gym-goers.

5.2 Product

5.2.1 Project Architecture

The architecture of BodyTrack follows a modular client–server model designed for real-time posture evaluation during strength training exercises. The system is composed of two major components: a lightweight Android application (client) written in Kotlin, and a remote Python-based backend server responsible for pose estimation, biomechanical analysis, and real-time feedback generation [4][5].

The core system is divided into three logical layers, distributed between the client and server:

Input Layer (Client):

- **Component:** Device Camera
- **Function:** Captures live video frames of the user during exercise and sends them to the server for analysis.
- **Output:** Raw video data or compressed frame sequences.

This layer initiates the posture analysis workflow and depends on correct user positioning, good lighting, and unobstructed visibility to ensure high-quality input.

Analysis Layer (Server):

- **Component A: Pose Estimation Module**

    - Receives video frames from the client and extracts key body landmarks using a real-time pose estimation model (MediaPipe Pose).
    - Outputs X, Y coordinates of joints per frame.
- **Component B: Biomechanical Evaluation Engine**
    - Analyzes joint angles, ranges of motion, and movement phases (descent, peak, ascent).
    - Compares detected movement with biomechanical reference models for each selected exercise.
- **Component C: Error Detection Logic**
    - Applies predefined thresholds and rules to detect posture faults (incomplete range, poor alignment).
    - Generates error alerts to be sent to the client.

This layer contains the system's intelligence and performs all heavy computations server-side using Python-based machine learning and analysis libraries.

<u>Presentation Layer (Client):</u>

- **Component A: User Interface (UI)**
    - Provides users with a clean and intuitive interface to select exercises, start training sessions, and receive feedback.
    - Displays analysis results and feedback in a clear and engaging format.
- **Component B: Feedback Module**
    - Receives alerts from the backend in real time and presents audio or visual feedback to the user.
    - Displays a session summary including rep count, success rate, common faults, and improvement suggestions.

This layer ensures a smooth user experience while keeping all complex computations offloaded to the cloud backend.

5.2.2 Software Package Structure

The architecture of BodyTrack follows a modular, distributed design based on a client–server model. The system is divided into logical packages that group related functionality and responsibilities across both the mobile application and the remote backend server.

- **<u>Mobile App (Client – Kotlin)</u>**
    - Input & Capture Package – Handles video input acquisition from the device's camera and sends frame data to the server for analysis.

    - User Interface & Feedback Package – Manages user interaction, including exercise selection, session initiation, and displaying real-time feedback and post-session summaries. Receives alerts and evaluation results from the backend and presents them using visual and audio cues.
- **Cloud Server (Backend – Python)**
    - Analysis & Evaluation Package – Receives video frames from the mobile client, performs pose estimation, calculates joint angles, detects movement phases, and identifies form faults. Returns feedback results to the app in real time.

This package structure separates the real-time user interface and data acquisition handled by the mobile app from the computationally intensive posture evaluation logic performed on the cloud backend. This modularity improves scalability, maintainability, and performance while allowing each component to evolve independently.

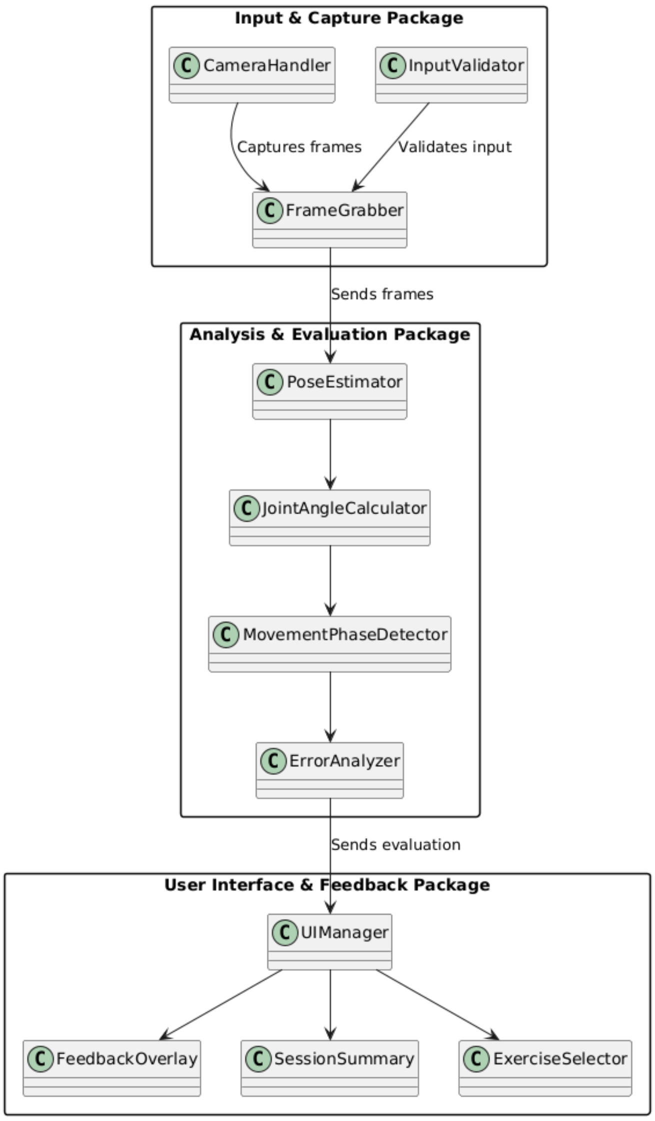


Figure 1 - BodyTrack Package Diagram: *Modular overview of BodyTrack's core software components and data flow.*

Package Descriptions for BodyTrack:

Input & Capture Package:
This package is responsible for initializing the video input pipeline on the user's device. It manages the camera, handles frame extraction [9], and ensures the captured input is valid before being sent for analysis.

- **CameraHandler** – Interfaces with the device's camera to initialize and control video input.
- **InputValidator** – Verifies that the user is visible in the frame with adequate lighting and full-body visibility.
- **FrameGrabber** – Extracts video frames in real time and sends them to the backend server for further processing.

Analysis & Evaluation Package:
This package represents the core computational logic hosted on the cloud server. It processes the incoming frames, extracts posture-related features, evaluates movement correctness, and identifies form deviations [4].

- **PoseEstimator** – Uses pose estimation models to identify and track body keypoints in each frame.
- **JointAngleCalculator** – Calculates angles between key joints to quantify movement and alignment.
- **MovementPhaseDetector** – Segments the exercise into phases (descent, peak, ascent) for contextual evaluation.
- **ErrorAnalyzer** – Compares extracted movement data to biomechanical models and detects posture or technique faults.

User Interface & Feedback Package:
This package handles the user interaction layer of the mobile app. It manages exercise selection, displays real-time feedback, and presents a summary of the training session [8].

- **UIManager** – Orchestrates the user interface flow across all training stages.
- **ExerciseSelector** – Allows users to choose the exercise to be evaluated.
- **FeedbackOverlay** – Displays visual markers and cues on top of the live camera feed to guide user corrections.
- **SessionSummary** – Presents a final overview including rep count, error analysis, and suggestions for improvement.

5.2.3 Process Flow: Real-Time Posture Evaluation

The activity diagram below illustrates the detailed operational workflow of the BodyTrack system. The workflow is organized vertically to reflect the step-by-step interaction, including frame-level processing, decision points, and feedback delivery during the session. This flow ensures continuous evaluation and guidance while the user performs strength training exercises.

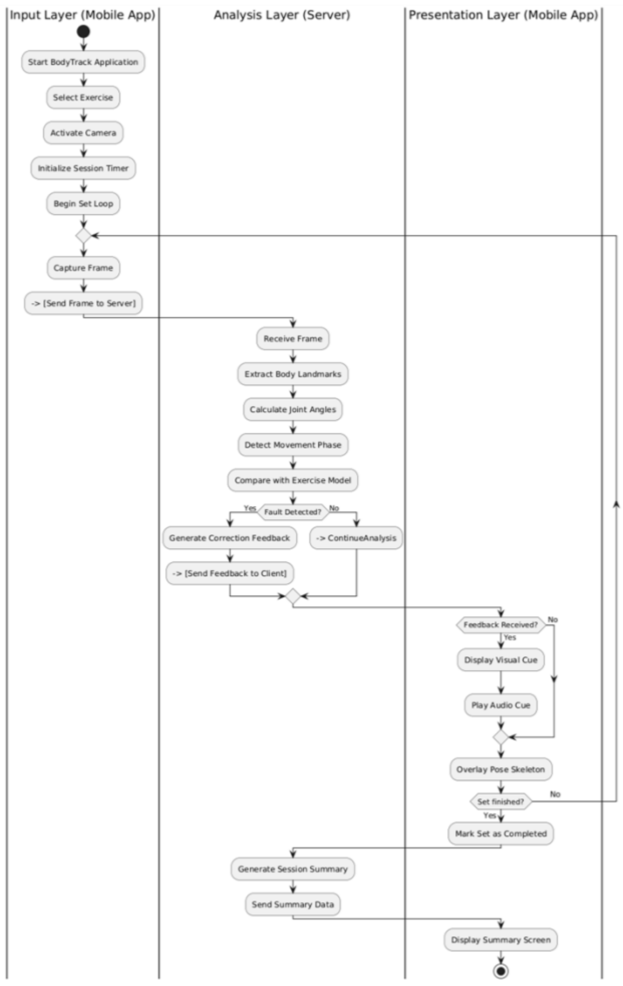


Figure 2 - BodyTrack Activity Diagram: *Detailed flow of actions between the mobile client and backend during a training session.*

Input Layer (Mobile App):

This layer handles user interaction and video capture:

- **Start BodyTrack Application** – The user launches the app.
- **Select Exercise** – The user chooses the specific strength exercise to evaluate.
- **Activate Camera** – The app activates the phone's rear camera and prepares for recording.
- **Initialize Session Timer** – A timer begins to manage and segment the session to sets.
- **Begin Set Loop / Capture Frame** – The app continuously captures video frames.
- **Send Frame to Server** – Each frame is sent to the backend for processing.

Analysis Layer (Server):

This layer runs in the cloud and performs all analysis:

- **Receive Frame** – The backend server receives the frame from the mobile client.
- **Extract Body Landmarks** – Pose estimation is performed to identify joints and key body points.

- **Calculate Joint Angles** – The system computes relevant joint angles based on keypoints.
- **Detect Movement Phase** –Identifies the current movement stage (descent, ascent).
- **Compare with Exercise Model** – The movement is checked against biomechanical standards.
- **Fault Detected?** – A decision point checks if posture is incorrect or unsafe.
- **Generate Corrective Feedback** – If needed, corrective feedback is prepared.
- **Send Feedback to Client** – Feedback is transmitted to the mobile app in real time.

<u>Presentation Layer (Mobile App):</u>
This layer delivers real-time guidance and post-session output:

- **Feedback Received?** – The app checks for feedback from the server.
- **Display Visual Cue / Play Audio Cue** – Errors are shown on screen or played as sound instructions.
- **Overlay Pose Skeleton** – A pose skeleton is shown live to help users align properly.
- **Mark Set as Completed** – After finishing the repetition set, the user or system ends the session.
- **Generate Session Summary (Server)** – A final report is created based on all collected data.
- **Send Summary Data / Display Summary Screen** – The summary is sent back to the app and shown to the user with stats, rep count, and suggestions.

5.2.4 Technical Design, Frameworks, and Development Approach

The BodyTrack system is designed as a modular and distributed architecture, separating real-time user interaction on the mobile app from computationally heavy posture evaluation performed on a remote backend server. This structure improves system performance, keeps the client lightweight, and enables the use of advanced machine learning libraries that are impractical on mobile hardware. Below is a detailed description of each component, including the tools, technologies, and implementation considerations used.

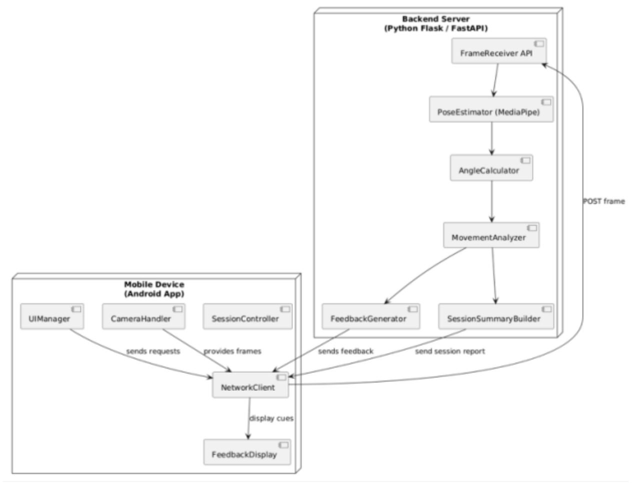


Figure 3 – BodyTrack Deployment Diagram: *the architecture of the system.*

Mobile Application (Client Side):

UI Layer [8][5]:

- **Implementation**: Built in **Kotlin** using **Jetpack Compose** for declarative UI or standard XML-based layouts.
- **Responsibilities**: Managing screens for exercise selection, live camera view, visual feedback overlay, and session summary display.
- **Libraries Used**:
  - **Jetpack Compose / Android Views**
  - **Navigation Component** (for screen flow)
- **Notes**:
  - UI must remain responsive despite constant frame streaming.
  - Feedback display must be timed correctly with server feedback, so UI components subscribe to live events.

Camera & Frame Capture [9][10]:

- **Implementation**: Managed using **CameraX** API.
- **Responsibilities**: Capturing real-time frames from the device's rear camera and preparing them for network transmission.
- **Libraries Used**:
  - **Android CameraX** (optimized for consistency across devices)
- **Notes**:
  - Frame resolution and frequency need to balance between accuracy and performance.
  - Some pre-processing (cropping or compression) is done before sending frames to reduce load.

Networking Module [5]:

- **Implementation**: Built using **Retrofit** or **OkHttp** for RESTful HTTP communication.
- **Responsibilities**: Handling secure data transfer between client and server.
- **Libraries Used**:
  - **Retrofit** (for JSON serialization)
  - **Gson** or **Moshi** (for data parsing)
- **Notes**:
  - All data is sent over **HTTPS** to ensure encryption.
  - Reconnection logic is required in case of packet loss or lag.

Feedback Receiver & Visual/Auditory Alerts [4][8][5]:

- **Implementation**: Observes backend responses and updates the UI with immediate feedback.
- **Libraries Used**:
  - **LiveData** or **StateFlow** for reactive updates
  - **MediaPlayer** (for audio cues)

- **Notes**:
    - Latency must be monitored and optimized to ensure real-time response.

Session Summary Viewer:

- **Implementation**: Displays data from the backend report.
- **Features**: Total reps, fault count, success rate, improvement tips.
- **Notes**:
    - This component must parse structured server-side summary responses into a friendly UI format.

<u>**Backend Server (Python Flask or FastAPI)**</u>

Frame Receiver API [4][7][5]:

- **Implementation**: Built using **Flask** or **FastAPI**.
- **Responsibilities**: Accepts incoming POST requests with image frame data.
- **Libraries Used**:
    - **Flask** / **FastAPI**
    - **Pillow** or **OpenCV** (for image processing)
- **Notes**:
    - Frame rate must be balanced to prevent overloading.
    - Security: implement input validation, size checks, and timeout mechanisms.

Pose Estimation Engine [2][1][9]:

- **Implementation**: Uses **MediaPipe** (Python API).
- **Responsibilities**: Extracts 2D keypoints of body joints from image input.
- **Libraries Used**:
    - **MediaPipe**, **OpenCV**, **NumPy**
- **Notes**:
    - MediaPipe is chosen for speed.
    - Preprocessing includes resizing, normalization, and grayscale conversion.

Biomechanical Analyzer [4]:

- **Implementation**: Custom Python logic based on vector math and kinematic rules.
- **Responsibilities**: Calculates joint angles, detects movement phase, and validates against biomechanical ranges.
- **Libraries Used**:
    - **NumPy** (for vector operations)
    - **SciPy** (for smoothing, filtering if needed)
- **Notes**:
    - Different thresholds are used for different exercises (squat vs. bicep curl).
    - Real-time windowing to detect motion phases (start, descent, bottom, ascent).

Error Detection & Feedback Generator [6]:

- **Implementation**: Matches detected patterns against error rules and generates text/audio flags.
- **Responsibilities**:
    - Real-time detection of common faults (shallow squat, bent back)
    - Sends structured JSON feedback
- **Notes**:
    - Feedback priority must be managed (warn about back alignment before minor angle drift).

Session Summary Generator:

- **Implementation**: Logs session data, aggregates rep statistics, generates report.
- **Output Includes**: Total reps, success rate, most common faults, improvement suggestions.
- **Libraries Used**:
    - **Pandas** or simple in-memory structures
    - **JSON** for response formatting
- **Notes**:
    - Reports are returned as lightweight JSON objects, parsed by the mobile UI.

5.2.5 Core Algorithmic Methodologies

BodyTrack uses a sequence of well-established algorithms and techniques to detect user posture, analyze movement, and deliver corrective feedback in real time. This section explains the logic and methodology behind the system's main algorithmic processes. The goal is to provide a clear, accessible understanding of how the app evaluates posture and performance – without focusing on implementation code or mathematical derivations.

1. **Pose Estimation**
   At the heart of BodyTrack is a pose estimation model that extracts key body points from camera frames. This is performed on the backend using MediaPipe Pose (Python API) [1], which provides real-time inference and detects 33 body landmarks such as shoulders, elbows, hips, knees, and ankles.
   - The model takes a 2D image and returns the (x, y) coordinates of each landmark.
   - Each point is accompanied by a visibility score that helps filter out noisy or occluded data.
   - This process forms the foundation for later biomechanical analysis.

2. **Joint Angle Calculation**
   Once key points are identified, the system calculates joint angles to determine body alignment [4]. For example:
   - Knee angle: measured between hip, knee, and ankle
   - Elbow angle: measured between shoulder, elbow, and wrist

This is done using simple vector geometry (cosine-based angle calculation). The joint angles are crucial for identifying posture correctness and determining which phase of the exercise the user is in. We use NumPy to process and calculate angles efficiently.

3. **Movement Phase Detection**
   Each exercise is broken into distinct phases [3][4][5]. For example, a squat has:
   - **Descent**: user lowers their body
   - **Bottom**: knees reach minimum angle
   - **Ascent**: user returns to starting position

   We identify these phases by observing how joint angles change over time:
   - If the knee angle is decreasing, the user is descending.
   - If the angle stabilizes at a low value, they are at the bottom.
   - If it increases, they are ascending.

   These changes are detected in a frame-by-frame stream. To make this stable, we apply a sliding window approach that tracks a short history of angles to reduce false detections.

4. **Error Detection Logic**
   Each exercise has defined biomechanical rules. For example:
   - A squat should go below 90 degrees in the knee.
   - The back should remain straight (no excessive forward lean).

   If a calculated angle violates one of these rules:
   - An error is recorded
   - The system triggers real-time feedback

   This logic is implemented using simple if/else decision trees [4] that compare calculated values to defined thresholds (can be found in literature review section).

5. **Feedback Generation**
   When an error is detected, the system generates:
   - A **textual message** (like "Bend your knees more")
   - A **visual marker** (highlighting the joint in question)
   - An **audio cue** (such as short beep or spoken alert)

   These are sent back to the mobile app through the network layer and rendered immediately using the feedback display module [5].

6. **Session Summary Aggregation**
   At the end of a session, the system analyzes all recorded repetitions to generate a personalized summary:
   - Number of correct/incorrect reps
   - Most common mistake

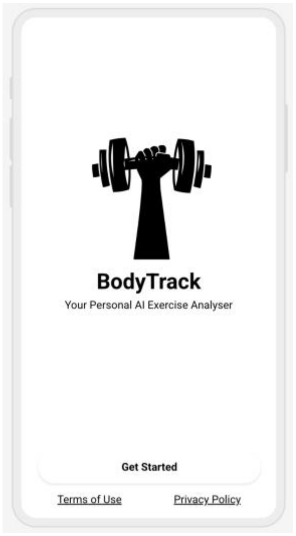
- Overall consistency score

This summary is generated in the backend and sent to the app as a structured JSON object. It is displayed in the session summary screen. We use simple Python data structures to count repetitions and classify them as valid or faulty, based on whether errors were triggered during each rep.

5.2.6 User Interface

The current user interface is a prototype that represents the intended visual and functional flow of the BodyTrack mobile application. The interface is tailored for clarity, simplicity, and usability during workout sessions, with a focus on minimizing distractions and guiding users effectively through each step of the process.

The prototype includes key screens which illustrate how users will interact with the app - from choosing a workout to receiving feedback.

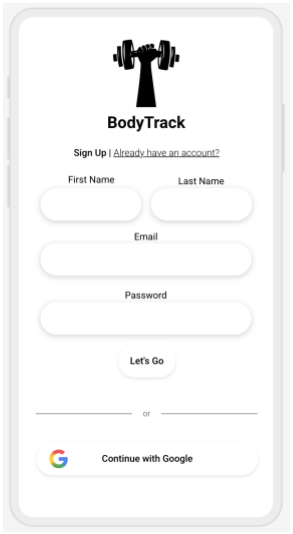

BodyTrack

Your Personal AI Exercise Analyser

Get Started

Terms of Use | Privacy Policy

Welcome Screen


BodyTrack

Sign Up | Already have an account?

First Name | Last Name

Email

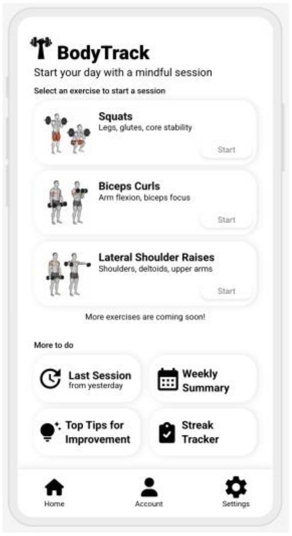
Password

Let's Go

or

Continue with Google

Login Screen


BodyTrack

Start your day with a mindful session

Select an exercise to start a session

Squats — Legs, glutes, core stability — Start

Biceps Curls — Arm flexion, biceps focus — Start

Lateral Shoulder Raises — Shoulders, deltoids, upper arms — Start

More exercises are coming soon!

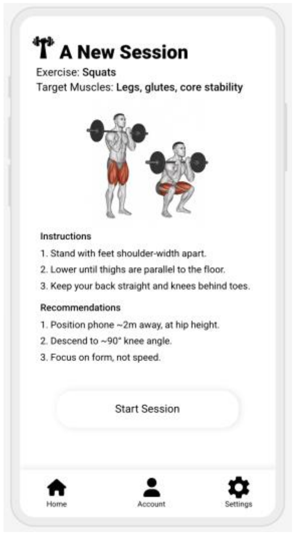
More to do

Last Session from yesterday | Weekly Summary

Top Tips for Improvement | Streak Tracker

Home | Account | Settings

Home Screen


A New Session

Exercise: Squats

Target Muscles: Legs, glutes, core stability

Instructions

1. Stand with feet shoulder-width apart.
2. Lower until thighs are parallel to the floor.
3. Keep your back straight and knees behind toes.

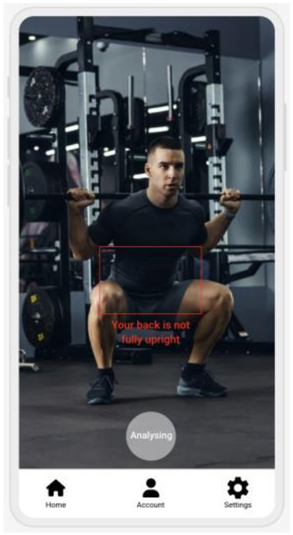
Recommendations

1. Position phone ~2m away, at hip height.
2. Descend to ~90° knee angle.
3. Focus on form, not speed.

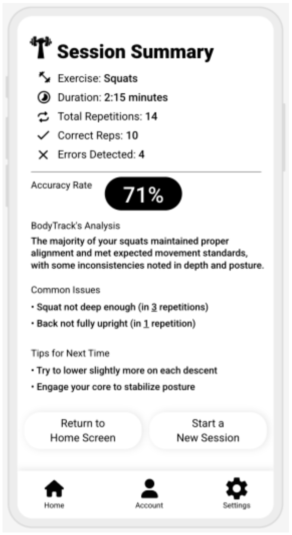
Start Session

Home | Account | Settings

Session Screen



Tracking Screen


Session Summary

Exercise: Squats

Duration: 2:15 minutes

Total Repetitions: 14

Correct Reps: 10

Errors Detected: 4

Accuracy Rate 71%

BodyTrack's Analysis

The majority of your squats maintained proper alignment and met expected movement standards, with some inconsistencies noted in depth and posture.

Common Issues

- Squat not deep enough (in 3 repetitions)
- Back not fully upright (in 1 repetition)

Tips for Next Time

- Try to lower slightly more on each descent
- Engage your core to stabilize posture

Return to Home Screen | Start a New Session

Home | Account | Settings

Summary Screen

5.2.7 Requirements

Functional Requirements

- The system shall capture real-time video using the Android device’s camera.
- The system shall allow the user to select the type of exercise (squats, biceps curls).
- The system shall transmit video frames to a remote server for pose analysis.
- The server shall perform preprocessing on received frames (resize, normalize).
- The server shall apply pose estimation to extract key body landmarks from each frame.
- The server shall calculate joint angles and detect movement phases using biomechanical rules.
- The server shall compare the user’s motion against reference models for the selected exercise.
- The system shall detect posture or movement faults based on predefined biomechanical thresholds.
- The system shall return real-time feedback (audio/text/visual) to guide the user during the exercise.
- The system shall allow the user to start, pause, resume, or end an exercise session via the interface.

Non-Functional Requirements

- The system shall have an intuitive interface requiring minimal setup prior to each session.
- Feedback shall be delivered in real time with minimal latency between movement and response.
- The app interface shall remain clear, non-intrusive, and easy to understand without prior training.
- The mobile application shall be compatible with mid-range Android smartphones.
- Audio feedback shall use a clear, natural-sounding voice.
- The system shall stream and process video continuously during an active session.
- On-screen guidance shall assist the user in positioning, camera setup, and lighting.
- All feedback and alerts shall be displayed in a non-distracting manner during the workout.
- The system shall detect suboptimal conditions (poor lighting, partial visibility) and inform the user.
- Users shall be able to pause or disable feedback without terminating the session.
- The application shall only request necessary permissions (like camera output).
- The prototype shall initially support English language output.
- The system shall handle minor issues (like brief signal loss or camera feed disruption) gracefully.
- System initialization (such as model loading and session start) shall complete in a reasonable time.

6 Expected Achievements

The main goal of this project is to design and implement a mobile application that helps users perform gym exercises safely and effectively by monitoring their posture and giving real-time feedback. BodyTrack aims to reduce the risk of exercise-related injuries and support users in improving their movement quality, even without professional supervision. The application will combine pose estimation, biomechanical modeling, and machine learning to provide immediate corrections during training sessions and detailed summaries afterward [4][6][5].

Real-Time Posture Detection and Feedback:

- Track the user's body movements during exercise using the device's camera.
- Identify incorrect joint angles or unsafe postures based on biomechanical rules.
- Provide immediate audio and visual feedback when mistakes are detected, allowing users to correct themselves during the exercise.

Exercise Model and Biomechanical Reference:

- Implement exercise-specific motion models for selected gym exercises (squats, curls, lateral raises).
- Define the correct range of joint angles and posture stability needed for each exercise.
- Detect deviations from the optimal movement pattern and classify errors (incomplete squats, asymmetric knees, back rounding, etc.).

Real-Time Cloud-Based Processing:

- Perform all pose estimation, posture evaluation, and movement analysis on the cloud using advanced machine learning models.
- Return real-time feedback to the user's device with minimal latency, ensuring responsiveness during training.
- Maintain a lightweight mobile app while offloading heavy computation to the backend.
- Ensure user data is handled securely through encrypted communication between the app and the server.

User Session Summary and Improvement Suggestions:

- After completing an exercise session, generate a simple report summarizing the following:
    - Total repetitions performed
    - Number and types of detected errors

    - Specific advice on how to improve form

- Help users track their progress over time and build better exercise habits [8].

User Interface Simplicity and Accessibility:

- Design an intuitive interface with a precise workout flow:
    - Exercise selection
    - Live camera feedback screen
    - Real-time alerts
    - End-of-session summary
- Focus on minimizing user setup and cognitive load, making the app accessible for beginners and experienced users.

Project Success Measurements:

- Accuracy of posture detection compared to professional trainer evaluation.
- Real-time responsiveness (minimal delay between movement and feedback).
- Positive user experience feedback regarding clarity, usefulness, and usability.
- Observable improvement in user form over multiple sessions.

7 Verification Plan

Our testing strategy will target five major components of the BodyTrack system:

1. **Video Capture & Transmission**
2. **Pose Estimation and Biomechanical Analysis (Server)**
3. **Real-Time Feedback System**
4. **Session Summary & Reporting**
5. **User Interface (UI) and Navigation**

Each component will be tested through a combination of **manual testing, simulation of edge cases**, and **data-driven checks** to validate accuracy, responsiveness, and user flow. The aim is to ensure that BodyTrack operates reliably across a wide range of user environments and hardware capabilities, while maintaining low latency and accurate feedback.

Test Table:

ID	Module	Tested Function	Expected Result
1	Video Capture	Start camera and begin live video session	Camera activates and begins capturing frames in real-time
2	Input Handling	Frame sent to server correctly	Frame reaches server endpoint with correct resolution and structure
3	Pose Estimation Engine (Server)	Extract joint landmarks from video frame	All key joints are identified with correct X/Y coordinates
4	Biomechanical Model Evaluation	Detect incorrect squat posture (for example - knees too far forward)	Detected and flagged by backend engine with correct error type
5	Real-Time Feedback	Display warning when posture is incorrect	Feedback appears on screen within 1 second of error
6	Real-Time Feedback	Play audio cue for incorrect motion	Correct audio file plays instantly when a fault is identified
7	Session Control	Start, pause, and end session	Session timer responds correctly and session state changes as expected
8	Summary Generation	Count total and correct repetitions	Final summary shows accurate counts and error breakdown
9	Summary Reporting	Generate analysis summary after session	Feedback includes accuracy score, issue breakdown, and improvement tips
10	UI/UX	Navigate between app screens	User transitions smoothly without glitches or delays
11	UI Feedback	User receives error message if lighting is too low	Warning message appears with suggestion to improve lighting

12	Permissions	App asks for camera and audio permission on first use	Android permissions dialog appears appropriately
13	Network Resilience	Handle short network disconnection during session	App recovers gracefully and continues session when connection is restored
14	Compatibility	Run on mid-range Android device (around 4GB RAM)	App loads and operates within acceptable speed and memory range
15	Localization	Display interface in default English	All labels, instructions, and feedback appear in clear English

8 AI Tools & Prompts Used in the Process

While preparing the BodyTrack system, multiple AI tools were employed to enhance the quality, precision, and efficiency of various development and documentation phases. These tools were not used to generate final outputs directly but to guide architectural decisions, refine system components, validate algorithmic approaches, and support UI design exploration.

The tools were chosen and used responsibly, in a way that complements human expertise and aligns with academic standards. Below is a breakdown of the AI platforms and the types of prompts and tasks they were applied to:

8.1 Tools and Purposes

1. **OpenAI ChatGPT 4**
   - **Role:** Expert consultant during architecture and system design phase.
   - **Usage:**
     - Clarified distinctions between on-device and cloud-based pose estimation pipelines and helped evaluate trade-offs.
     - Suggested appropriate layered architecture separation (Input Layer, Analysis Layer, Presentation Layer), reinforced through academic examples.
     - Advised on data flow modeling between mobile client and server.
   - **Representative Prompt Examples:**
     - “What architectural patterns are commonly used in cloud-connected Android apps involving video processing and real-time analysis?”
     - “List pros and cons of using TensorFlow Lite vs. TensorFlow server-side when working with pose estimation for mobile fitness.”
     - “What diagram types best represent a system with a mobile app frontend and cloud backend?”
2. **GitHub Copilot**
   - **Role:** Autocomplete and code assistance during prototyping experiments.
   - **Usage:**

        - Supported early proof-of-concept snippets for real-time camera input and JSON transmission.
        - Provided code suggestions for Android camera activation and UI layout in Jetpack Compose.
    - **Representative Guidance:**
        - Suggested lifecycle-safe implementations for CameraX usage on Android.
        - Highlighted platform-specific quirks for rendering overlay feedback in Kotlin.
3. **PlantUML Model Generator**
    - **Role:** Visualization of system components and processes.
    - **Usage:**
        - Used iteratively to construct Package, Class, Use Case, and Activity Diagrams.
        - Templates and rules provided guidance on UML best practices.
    - **Prompt Guidance:**
        - “Generate a layered activity diagram for a mobile fitness app with real-time server processing.”
        - “What should be included in a class diagram for a fitness posture evaluation app?”
4. **Perplexity AI**
    - **Role:** Source verification and academic literature reference suggestions.
    - **Usage:**
        - Assisted in finding relevant studies on biomechanics, pose estimation accuracy, and feedback efficacy.
        - Provided summarized comparisons of ML frameworks (MediaPipe vs. OpenPose) with citation trails.
    - **Prompt Examples:**
        - “Scientific articles comparing BlazePose and OpenPose in accuracy and real-time performance.”
        - “Biomechanical modeling of squats: what angles and ranges are considered optimal?”
5. **Canva AI Design Assistant**
    - **Role:** UI sketching and ideation companion.
    - **Usage:**
        - Helped brainstorm logical structuring of screens and user flows before final Figma design.
        - Suggested accessible text hierarchies and feedback prompt visibility.
    - **Prompt Direction:**
        - “How should a feedback interface look for an AI-powered exercise app to avoid cognitive overload?”

8.2 Reflection and Integration

The integration of AI tools into BodyTrack's planning and prototyping did not replace original work, but instead enhanced it by providing rapid access to knowledge, testing architectural soundness, and evaluating design decisions under constraints. All prompts were crafted with critical oversight, and AI-generated outputs were reviewed and edited before being incorporated into the official system design and documentation.

As such, AI tools were instrumental in shaping the **design rationality, language clarity**, and **visual expression** of the BodyTrack system, demonstrating their growing value in modern software engineering workflows.

9 References

[1] S. Dill, A. Rösch, M. Rohr, G. Güney, L. De Witte, E. Schwartz, and C. Hoog Antink
“Accuracy evaluation of 3D pose estimation with MediaPipe Pose for physical exercises”
Current Directions in Biomedical Engineering, vol. 9, no. 1, pp. 563–566, 2023.

[2] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh
“OpenPose: Realtime multi-person 2D pose estimation using part affinity fields”
Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), pp. 2292–2299, 2019.

[3] A. Kendall, M. Grimes, and R. Cipolla
“PoseNet: A convolutional network for real-time 6-DOF camera relocalization”
Proc. IEEE Int. Conf. Comput. Vis. (ICCV), pp. 2938–2946, 2015.

[4] F. Roggio, S. Di Grande, S. Cavalieri, D. Falla, and G. Musumeci
“Biomechanical posture analysis in healthy adults with machine learning: applicability and reliability”
Sensorsi, vol. 24, no. 9, Art. no. 2929, 2024.

[5] H. Kotte and A. Kumar
“Real-time posture correction in gym exercises: A computer vision-based approach for performance analysis, error classification, and feedback”
Proc. 3rd Int. Workshop on Multimodal Immersive Learning Systems (MILeS) CEUR Workshop Proc., vol. 3499, pp. 63–70, 2023.

[6] R. Maharjan, S. Mendu, M. Mariani, S. Abdullah, and J. P. Hansen
“Exploring user engagement with real-time verbal feedback from an exoskeleton-based virtual exercise coach”
Digital Health, vol. 10, pp. 1–17, 2024.

[7] J. Liang, **“Research on fitness app design and user engagement”**
Proc. Int. Conf. Inf. Sci. Syst. (ICISS), Web of Proceedings, pp. 50–55, 2019.

[8] J. Liu, L. Wang, and H. Zhou
“The application of human-computer interaction technology fused with artificial intelligence in sports moving target detection education for college athletes”
Front. Psychol., vol. 12, Art. no. 677590, 2021.

[9] Google Developers, **“Pose detection”**, 2024.

[10] TensorFlow, **“MoveNet: Ultra fast and accurate pose detection model”**, 2024.