PostureScore: An On-Device, Phase-Aware Scoring Pipeline for At-Home Rehabilitation

Ella Wang
Emma Willard School
Troy, New York, USA
ellawang0710@hotmail.com

Abstract

After surgery, many patients do their rehab exercises at home without knowing whether their form is correct. This project originated from the author's own post-surgical rehabilitation (ACL reconstruction, 2025; shoulder arthroscopy, six months later), which surfaced three engineering gaps in existing tools. Smartphone pose models like MediaPipe Pose only output landmarks; the tools that wrap them typically use one fixed target angle regardless of rehabilitation week; and many upload video to a server, which fellow patients met during rehabilitation were uncomfortable with. PostureScore is a prototype that addresses these three gaps in one mobile app. Its core is a scoring formula combining whether the patient holds the movement long enough and whether the movement is steady, with targets that change by rehabilitation week (wider tolerance early, narrower later). The privacy design sends no video at all; only session-summary numbers (total reps, average score, peak range of motion, and similar) reach the backend. The pipeline was measured at 38.5 fps on iPhone 16 and 31.0 fps on iPhone 14 and deployed to seven post-surgical patients over an 8- to 12-week window. The system ran across all 664 sessions and produced consistent per-rep scores. The work is reported as an engineering feasibility check on the prototype. It does not provide evidence that PostureScore improves clinical outcomes.

CCS Concepts

• **Human-centered computing** → **Ubiquitous and mobile computing**; • **Applied computing** → *Consumer health*; • **Computing methodologies** → *Activity recognition and understanding*.

Keywords

On-Device AI, Pose Estimation, Rehabilitation, Mobile Health, Movement Scoring, Privacy-Preserving Design

ACM Reference Format:
Ella Wang. 2026. PostureScore: An On-Device, Phase-Aware Scoring Pipeline for At-Home Rehabilitation. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 3)*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.67149/yhjs2024.5/17iljp8h

IJSCAR VOL. 3, ISSUE 3

1 Introduction

Post-surgical rehabilitation is mostly a self-directed activity that happens at home. After a procedure such as anterior cruciate ligament (ACL) reconstruction or shoulder arthroscopy, the patient is usually discharged within a day or two with a printed list of exercises and an outpatient appointment weeks out. The period between discharge and that appointment is the longest stretch of unsupervised rehabilitation, and it is also when adherence drops the most. A 2025 prospective cohort of 240 lower-limb orthopedic postoperative patients found that only 53.7% complied with the prescribed rehabilitation protocol at one-month follow-up [1], and a 2025 case-series review across six National Health Service trusts in the United Kingdom reported that 29% of ACL patients failed to complete their rehabilitation [2]. The exercises themselves are rarely the issue. The gap is that the patient has no objective signal between clinic visits about whether they are performing them correctly.

The obvious solution is to put a movement-sensing system in the patient's pocket. Smartphones have cameras and enough compute to run pose estimation locally. MediaPipe Pose (BlazePose [3]) runs at interactive frame rates on regular phones (natively and, as measured in §6, as WebAssembly inside a mobile WebView), and its 2D joint angles are close to motion-capture references on rehabilitation movements [4, 5]. This could give the at-home patient the missing feedback signal, but an off-the-shelf pose model does not get there on its own: wrapping it in a UI leaves three engineering gaps.

Gap 1: feedback resolution. A pose model returns landmarks, not a score. Tools that wrap MediaPipe in a rehab UI typically collapse the landmark stream to a binary "correct / incorrect" or green/red cue, which tells the user neither how far off they were nor anything about the movement's *duration*.

Gap 2: phase blindness. The same exercise has very different correct ranges in week 1 and week 12: a target set for the late phase is simply out of reach in the first weeks; one set for the early phase stops challenging patients as recovery progresses. Tools that ship a single fixed target per exercise therefore either discourage early-week patients or under-stretch late-week ones.

Gap 3: privacy boundary. Several existing tools upload video to a cloud service for processing. Fellow patients the author met during rehabilitation (§3.3) were uncomfortable with this, and the upload is unnecessary: pose estimation runs locally on a phone already capable of much heavier ML workloads.

PostureScore is a prototype that addresses these three gaps. The app is built in React + TypeScript, packaged for iOS using Capacitor, and runs MediaPipe Pose inside the phone's WebView so camera frames are processed locally. A small Node.js / PostgreSQL backend only ever sees session-level summary numbers, never frames or per-frame data. This paper makes three contributions: a scoring formula (§4) that turns a stream of pose landmarks into a 0–100 score per repetition by combining hold time and movement steadiness; a privacy design (§5) that keeps raw video in the phone's WebView process and only sends summary numbers off-device; and a set of design choices (early-week tolerance, per-rep voice feedback, on-device inference) that come directly from the author's prior rehabilitation experience (§3.3) rather than from the published literature. The system was evaluated on seven post-surgical patients over an 8- to 12-week window (§7); across 664 sessions the pipeline ran end-to-end without recorded failures.

2 Related Work

This work does not try to improve MediaPipe itself. It uses MediaPipe as a tool and focuses on what can be built on top of it: a scoring rule, week-by-week targets, and a privacy-preserving app. Three things from the literature shaped that choice.

Pose accuracy on phones. Several studies find MediaPipe Pose's 2D joint angles close enough to motion capture for rehab feedback when the camera is set up reasonably [4, 5], though accuracy depends on viewpoint [6, 7]. That justified using it off-the-shelf rather than training a new model; the trade-off is that the underlying angle accuracy cannot be improved here, only built upon.

Digital rehab tools and adherence. A broader literature has looked at whether digital rehab tools help patients stick with exercises and recover better [8, 9]. Commercial systems like Sword Health publish operational cost-of-care reports [10], but none of these papers publishes its actual scoring formula in enough detail to reproduce. That is why (3) and Table 3 are written out in full, so the next student can reuse them.

On-device vs. cloud privacy. Two things motivated keeping frames on the phone. The author's prior rehabilitation experience (§3.3) surfaced patient discomfort with video upload, and Apple's Privacy Manifest framework [11], mandatory since 1 May 2024, requires cloud-uploading apps to declare a lot more than on-device ones. Walker *et al.* also report that tool-level trust is a real barrier to ACL rehabilitation adherence [12].

3 System Design

3.1 Overall architecture

PostureScore is a four-layer system (Figure 1). The layout is designed so that raw video stays inside the WebView on the user's phone; no code path in the app writes a frame to disk or sends it off the device.

(L1) **UI layer** — a React + TypeScript app showing exercise selection, plan navigation, on-canvas overlays, and post-session summaries.

(L2) **Native shell** — Capacitor [13] wraps the app in an iOS WebView and handles camera permission, the iOS privacy manifest, and HTTPS. Using Capacitor (instead of writing native Swift) lets the same code also run as a mobile-web app.

(L3) **Pose inference + scoring** — MediaPipe Pose v0.5 runs as WebAssembly inside the WebView; each frame is consumed and overwritten in place, and only the 33 per-frame landmarks cross into application code. The 20-frame smoothing, rep state machine, and scoring formula (3) are TypeScript modules in the same WebView context, grouped as L3 because they all touch per-frame data. L1 receives, read-only, the per-frame landmarks (to draw the on-canvas overlay) together with the per-rep score and the end-of-session summary; it never writes back to the frame or pose buffers.

(L4) **Backend** — a small Node.js + PostgreSQL backend storing week-indexed plans, the per-week standards (§4.6), and one scalar summary record per session (fields under G2 in §5). It never sees frames, landmarks, or per-frame angles; the schema has no field for them.

3.2 Component inventory

Table 1 lists the principal components and the maximum data scope each touches; §5 spells out the trust model behind the on-device/off-device split.

3.3 Patient-Informed Design Rationale

Three design choices in PostureScore come from my own ACL (June 2025) and shoulder (December 2025) post-surgical rehabilitation rather than from the literature. All hand-authored values in Table 3 were reviewed by the clinical advisor (Acknowledgements) before deployment.

Wider angle tolerance in early weeks (15° at week 1, narrowing to 10° at week 12; full table in §4.6). During my early ACL recovery, small deviations triggered "incorrect" feedback in the apps I tried, which made sessions demoralizing and cut my practice frequency. Week-1 tolerance is set wide for an engagement reason rather than a clinical one. Clinically, a narrower tolerance would be more correct, but early in recovery the main limiter on practice is whether a session feels achievable, so the wide setting trades strict scoring for engagement.

Per-repetition voice feedback. During shoulder rehabilitation I found summary feedback at session end insufficient to correct mid-session form drift: by the time I learned rep 4 was wrong, I had already done reps 5–10. The system therefore speaks the score immediately when each rep terminates (§4.4), with a 2-second debounce to avoid stacked utterances on fast reps.

Keeping video on the device, no exceptions. Fellow patients I met during my rehab at Fu's Orthopedic Hospital (Hangzhou) repeatedly said they were uncomfortable with

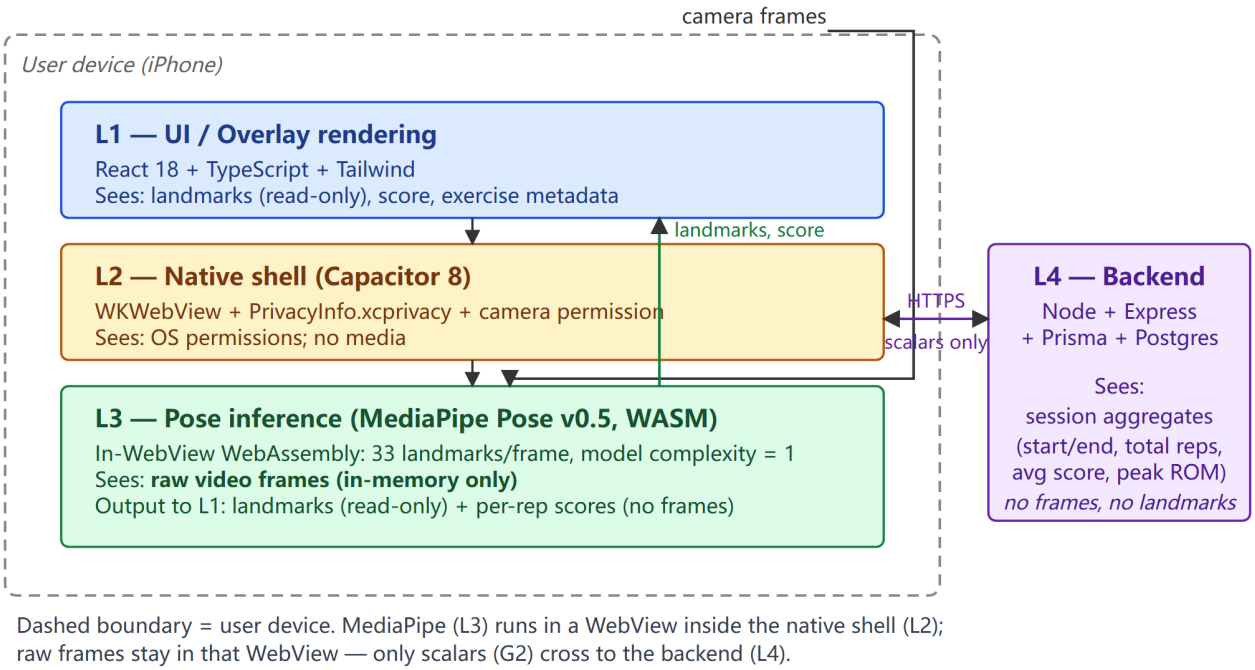

Figure 1: Four-layer architecture and privacy boundary; the dashed line marks the user's device (§5).

Table 1: Component inventory.

Layer	Component	Technology	Data scope at runtime
L1	UI / overlay rendering	React 18 + TypeScript + Tailwind	Landmarks (read-only), score, exercise metadata
L1	TTS feedback	Web Speech API (`speechSynthesis`)	Score string only
L2	Native shell	Capacitor 8 (iOS)	OS permissions; no media
L2	Privacy manifest	`PrivacyInfo.xcprivacy` (NSPrivacyTracking=false)	Static metadata only
L3	Pose inference	MediaPipe Pose v0.5 (WASM)	Raw frames (in-memory only)
L3	Angle smoothing	20-frame moving average (TypeScript)	Smoothed angle stream
L3	Rep state machine + scoring	Custom (TypeScript)	Per-rep summaries
L4	API	Node.js + Express + Prisma	Session aggregates only
L4	Database	PostgreSQL	Session aggregates only

apps that uploaded video to remote servers. For this reason, L3 runs WebAssembly inference inside the WebView instead of processing frames on a server; §5 spells out the implications.

3.4 User Interface Workflow

A session runs in five steps, shown for shoulder forward flexion in Figure 3. First the patient picks an exercise (Figure 3A). The screen shows the current rehabilitation week, and that week selects the $(\theta_t, \tau, \mathrm{H})$ row from the standards table (§4.6); in week 1 the row is a 60° target with $\tau = 15°$ and a 3 s hold. The camera then opens (Figure 3B). MediaPipe runs in the WebView, and the overlay draws the skeleton, marks the three landmarks for the measured angle (hip–shoulder–elbow for shoulder forward flexion), and shows the live angle, the target, and the rep count. The patient raises the arm into the target zone and holds while the state machine of §4.2 tracks it. When the rep ends, the app shows and speaks the score (Figure 3C), for example "78 points, next", after the 2 s debounce of §6. At the end a summary screen lists the totals (Figure 3D): reps, sets, average score, peak ROM, ROM achievement, voice-correction and compensation counts, and duration. These are exactly the G2 scalars sent to the backend (§5); no frame or per-frame value is shown or stored. The

same five steps repeat throughout the rehabilitation course. Only the targets change from week to week (§4.6); the steps do not.

4 The PostureScore Algorithm

This section describes the algorithm that turns a stream of MediaPipe pose landmarks into a single 0–100 score per repetition. The pipeline has four stages: joint angle extraction (§4.1), repetition state-machine detection (§4.2), hold-time and steadiness accumulation (§4.3), and the composite scoring function (§4.4). The function shape is justified in §4.5, and §4.6 gives the table that changes the targets across rehabilitation weeks. This work does not contribute a new pose model; it uses MediaPipe Pose v0.5 off-the-shelf. The new piece is the scoring layer on top of it.

4.1 Joint angle extraction

For each frame, MediaPipe Pose returns 33 landmarks with image-normalized 2D coordinates and a per-landmark visibility score in [0, 1]. For each tracked exercise the system picks three landmarks (A, B, C); for shoulder forward flexion on the right side these are the right hip, right shoulder, and right elbow (as shown in Figure 3B: R.Hip→R.Shldr→R.Elbow), with the angle measured at the shoulder vertex. It then computes the planar joint angle at vertex B using the law of cosines:

$$\theta = \frac{180}{\pi} \arccos\left(\frac{|AB|^2 + |BC|^2 - |AC|^2}{2\,|AB|\,|BC|}\right) \qquad (1)$$

The argument is clamped to $[-1, 1]$ before the inverse cosine to absorb floating-point drift near the cone limits. Frames in which any of the three landmarks has visibility below 0.5 are dropped. To suppress single-frame jitter without introducing perceptible latency, raw angles are smoothed with a 20-frame moving average; the resulting smoothing-window duration depends on the device's achieved frame rate and is quantified in §6. Some exercises measure a complementary angle (knee flexion is naturally 180° minus the inner angle); each exercise is tagged `increasing` or `decreasing`, and decreasing angles are flipped before §4.2, so the state machine is direction-agnostic.

4.2 Repetition state machine

Each repetition is detected by the four-state machine in Figure 2. A user begins in *Idle*. When the smoothed angle θ enters the target zone $[\theta_t, \theta_t + \tau]$ (θ_t is the prescribed target angle, τ the tolerance), the machine moves to *Holding* and a per-rep timer starts. τ is single-sided: the entry zone is $[\theta_t, \theta_t + \tau]$ and the rep-end threshold is $\theta_t - \tau$, so despite the $\pm$ notation the tolerance is not symmetric. A single τ sets both the entry-zone width and the rep-end margin, keeping each (week, movement) standard to one value. Decoupling it into a separate entry width and rep-end margin is a straightforward extension; because the rep-end margin sets how far the angle must fall to end a rep, it would shape the descent counted in `totalTime` and hence the steadiness ratio, giving the score-geometry analysis (§7) an extra parameter to vary. The upper bound $\theta_t + \tau$ makes overshooting count as leaving the zone, so it does not score higher. This caps credit and discourages forcing a joint past its prescribed range. For maximal-range movements the upper bound would be worth revisiting. A watchdog resets the machine to *Idle* and discards the in-progress rep if it has not completed within a timeout T_{idle} (30 s; see below). Without it, a rep stalled in the band $[\theta_t - \tau, \theta_t)$ could sit in *Drift* indefinitely, and because later reps are reachable only through *Idle*, every one of them would go uncounted. While *Holding*, two quantities are tracked: a continuous-hold timer (time in the zone) and a maximum-hold register (the longest hold so far). Leaving the zone moves the machine to *Drift* and commits the timer to the register if it is larger. The rep ends and the machine moves to *Complete* when θ falls below $\theta_t - \tau$. At that point the score (§4.3–4.4) is computed and spoken, and the machine returns to *Idle*. Re-entering the zone from *Drift* restarts the timer without starting a new rep, so a brief overshoot-and-recover is not penalized twice (the longer prior run is preserved; Algorithm 1).

The watchdog timeout is $T_{\text{idle}} = 30$ s. In the deployment the slowest completed reps ran about 20–25 s. Setting the cutoff at 30 s gives a genuinely slow hold room to finish, and still clears a stalled rep within a few seconds. (The ˜30 s trajectory in Figure 2B is a drawn illustration, not a deployment maximum.) The timeout is a single fixed value applied to every exercise. A fast wrist movement and a slow shoulder hold have different natural durations, so a per-exercise timeout would be a reasonable extension.

4.3 Hold-time accumulation and steadiness ratio

When *Complete* is reached, the algorithm computes two summary quantities. Let `finalHold` be the larger of the most recent continuous-hold value and the maximum-hold register, and let `totalTime` be the wall-clock elapsed from rep start to rep end. The *steadiness ratio* (the term "steadiness" is used because (3) rewards sustained holding rather than total time in zone) is

$$e = \min\left(\frac{\text{finalHold}}{\text{totalTime}},\ 1\right) \in [0, 1] \qquad (2)$$

with `totalTime` lower-bounded at 0.01 s to prevent division by zero. In words, e is the user's longest unbroken hold relative to total rep time: a user who reaches the target and holds it cleanly has $e \approx 1$, while one who oscillates around the boundary keeps restarting the hold timer (§4.2), so even their longest span stays small relative to the rep, giving $e \ll 1$. The two factors in (3) capture different things: whether the longest hold reached the prescribed duration H, and whether it was a high fraction of total rep time. Two timing details make the accumulation reproducible: the hold timer advances by the real inter-frame interval Δt (so a gap is never back-counted), and an occluded frame (any of the

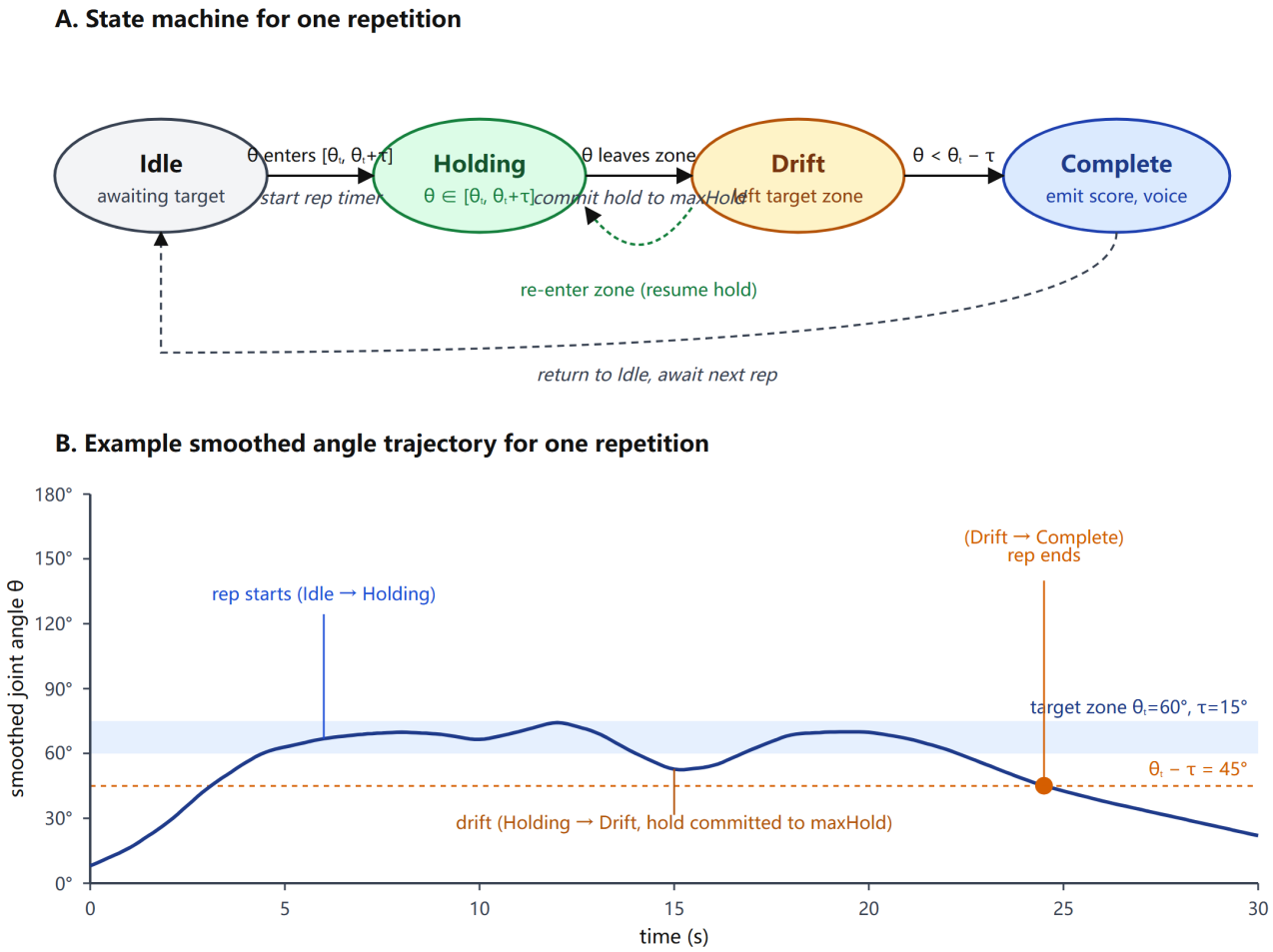

Figure 2: (A) The four-state per-repetition machine (§4.2): Idle → Holding → Drift → Complete. (B) Example smoothed-angle trajectory ($\theta_t = 60°$, $\tau = 15°$): the blue band is the target zone $[\theta_t, \theta_t + \tau]$, the dashed line the rep-end threshold $\theta_t - \tau$. The 30 s span is a constructed illustration (deployed reps ran ≈20–25 s, below the $T_{\text{idle}} = 30$ s watchdog; §4.2).

three landmarks below visibility 0.5) is skipped before the timer updates; `totalTime`, by contrast, is wall-clock ($t - t_{\text{start}}$) and includes occluded spans, so heavy occlusion lowers e. Time the camera could not verify does not count as a steady hold.

4.4 The composite scoring function

The final per-rep score is then

$$\text{score} = 100 \cdot \min\left(\frac{\text{finalHold}}{H},\ 1\right) \cdot \sqrt{e} \qquad (3)$$

where H is the prescribed phase-hold in seconds (§4.6). The `min(·, 1)` cap means holding longer than required earns no bonus. Without the cap, users could lengthen holds to inflate the score. Scores are rounded to integers and averaged per session for reporting; Algorithm 1 summarizes the per-rep loop. The $H = 0$ branch in Algorithm 1 is retained for reach-target movements that do not require a sustained hold. The current deployment standards in Table 3 use nonzero hold durations.

4.5 Choice of penalty function

PostureScore uses $\sqrt{e}$ as the steadiness penalty. The alternatives are plain e and the harsher e^2. Table 2 shows the difference: a steadiness of 0.25 scores 50 under $\sqrt{e}$ but 25 under linear e. All three penalties are monotonic in e, so the choice preserves the ordering of repetitions when the hold-time achievement factor is equal. It does not generally preserve rankings when repetitions differ in both hold-time achievement and steadiness. The penalty

```
Algorithm 1 - PostureScore per-rep scoring loop.
Input:  landmark stream {L_t} with frame times t; target theta_t; tolerance tau; phase-hold H; rep timeout T_idle = 30 s
Output: per-rep score s in {0, ..., 100}
state <- Idle; timer <- 0; maxHold <- 0; t_start <- null; t_prev <- null
for each frame at time t:
    dt <- (t - t_prev) if t_prev != null else 0;  t_prev <- t       # real inter-frame interval, advanced every frame
    if state != Idle and (t - t_start) > T_idle:                   # watchdog: abandon a stalled rep
        state <- Idle; continue                                    # partial rep discarded, no score
    if any of (A, B, C) in L_t has visibility < 0.5: continue      # occluded frame: timer not advanced
    theta_smooth <- rolling mean of last 20 angles((A,B,C))      # eqs. (1)
    in_zone <- theta_smooth in [theta_t, theta_t + tau]
    if state = Idle and in_zone:
        state <- Holding; t_start <- t; timer <- 0; maxHold <- 0
    elif state = Holding:
        if in_zone: timer <- timer + dt
        else:       state <- Drift; maxHold <- max(maxHold, timer); timer <- 0
    elif state = Drift:
        if   in_zone:                state <- Holding             # restart span
        elif theta_smooth < theta_t - tau:     state <- Complete
    if state = Complete:
        finalHold <- max(timer, maxHold)
        totalTime <- max(t - t_start, 0.01)                        # wall clock; includes any occlusion
        e <- min(finalHold / totalTime, 1)                         # eq. (2)
        achievement <- 1 if H = 0 else min(finalHold / H, 1)
        s <- round(100 * achievement * sqrt(e))                          # eq. (3)
        emit s; state <- Idle
```

Table 2: Score multiplier under three steadiness penalties at representative e (hold-time factor = 1, so each value is the full 0–100 score).

Steadiness e	Linear e	$\sqrt{e}$ (used)	e^2
0.25	25	50	6
0.50	50	71	25
0.81	81	90	66
1.00	100	100	100

function therefore determines how harshly low-steadiness repetitions are scored. The square root gives partial credit for brief instability and was selected as a design choice rather than learned from data. An empirical comparison across the three penalties would need the per-rep `finalHold` and `totalTime`, which the deployment did not store (§5).

4.6 Phase-aware standards

The triple (θ_t, τ, H) that parameterizes (3) is not constant across a rehabilitation course. It is read from the `AiCoach PhaseStandard` table at session start, indexed by the user's current rehabilitation week. Table 3 shows three representative entries: shoulder forward flexion is targeted at 60° with $\tau = 15°$ and a 3-second hold in week 1, reaches 80° / $\tau = 15°$ / 3 s by week 3, and 170° / $\tau = 10°$ / 3 s by week 12 (approximately full physiological range). Both the target angle θ_t and the tolerance τ change over the rehabilitation course (Table 3). The rising θ_t makes the zone harder to reach, while the narrowing τ (from 15° to 10° by week 12) works the other way. Tightening the parameters therefore has a *non-monotonic* effect on (3) instead of uniformly making it harder to satisfy. The §7 score trajectory must be read with this coupling in mind. PostureScore is "phase-aware" because of this split between a *fixed scoring function* and *time-varying parameters*, neither of which touches the algorithm. Correction. The values in Table 3 represent the actual configuration used during the February–April 2026 pilot study, and the §7 results were generated under that configuration. An earlier manuscript version contained preliminary documentation values that were inadvertently presented as deployment parameters; these values have been corrected here.

Table 3: Representative phase-aware standards (`AiCoachPhaseStandard`). *Tol* = tolerance τ (both the zone width above θ_t and the rep-end margin below it, §4.2); *Hold* = phase-hold *H*.

Week	Movement	Target angle θ_t	Tol τ	Hold H
1	Shoulder forward flexion	60°	15°	3 s
3	Shoulder forward flexion	80°	15°	3 s
12	Shoulder forward flexion	170°	10°	3 s
1	Knee flexion	80°	15°	1 s
6	Knee flexion	110°	12°	1 s
12	Knee flexion	135°	10°	1 s

5 Privacy Architecture

For any home rehab app that uses computer vision, one question dominates the privacy discussion: *does video leave the device?* For PostureScore the answer is no. This was a deliberate design decision. The privacy design rests on three guarantees and an explicit trust model, organized around the WebView boundary shown in Figure 1.

(G1) Raw video frames remain in WebView memory. The camera frame buffer is consumed by the WASM pose-inference module on the same animation tick that produces it, and is overwritten by the next frame. Under no code path in PostureScore is a frame serialized to local storage, IndexedDB, the filesystem, or the network. (G1) is enforced by the app's source code, not by the OS or WebView runtime; see the trust model below.

(G2) Only session summary numbers cross the device-to-server boundary. What leaves the device over authenticated HTTPS is one session record with these scalar

fields: session start/end time, total duration, total reps, total sets, average score, peak ROM angle, target ROM angle, ROM achievement ratio, voice correction count, compensation event count, and a voice-enabled flag. Per-frame landmarks, per-frame angles, and per-rep scores are not in the payload. The Prisma model `AiCoachExerciseSession` has no field for landmarks or per-frame angle data, so the schema itself enforces this.

(G3) The system is not used for advertising tracking. The iOS privacy manifest declares `NSPrivacyTracking = false`, lists no tracking domains, and declares only the data categories the app actually uses (account email, fitness/health data, basic usage analytics, user notes); the sensitive APIs declared are all there for app functionality, not cross-app identification.

Trust model. PostureScore trusts the device OS and WebView runtime: a malicious ROM, a jailbroken device, or a tampered browser engine could exfiltrate frames before the WASM module consumes them, and the architecture does not detect this; nor does it stop a malicious user uploading frames out-of-band. The only network defense is TLS in transit. The system applies no differential privacy, federated learning, or encryption beyond HTTPS. The privacy design works by limiting *what* the app sends rather than by obscuring it. The §7 aggregates are used only for summary statistics, never for training. DP and federated learning are left as future work, to be revisited if cross-user personalization is ever pursued.

Logging for offline analysis. The deployment stored only session aggregates (G2), so the per-rep numbers needed to re-score sessions (§4.5, §7) were never kept. Storing only aggregates was a storage decision, and the privacy design does not require it. A future build could add a few de-identified per-repetition values such as `finalHold` and `totalTime`. Those are plain numbers, so logging them would make the analyses possible with no video leaving the device. A minimal clinical study would build on this data infrastructure: it would recruit a prospective cohort, have an independent physical therapist score the recorded movements against (3), and obtain full independent ethics-board approval before making any efficacy claim.

6 Implementation

Pose model and smoothing. MediaPipe Pose v0.5 runs as WebAssembly inside the WebView, with `modelComplexity = 1` (the medium model), built-in landmark smoothing on, segmentation off, and detection/tracking thresholds at 0.5. The system uses complexity 1 instead of the faster `complexity = 0` (lite). The lite model is also real-time (Table 4), but its lighter network yields noisier landmarks, which would degrade joint-angle precision relative to the τ tolerances in Table 3. The raw 3-point angle from (1) is pushed onto a 20-frame circular buffer; at 31–38 fps (Table 4) this gives ˜525–650 ms of smoothing, which damps per-frame jitter without noticeably lagging the voice feedback. Because the window smooths over brief transitions in both directions, it can bias `finalHold` in either direction by at most approximately one smoothing window (˜0.65 s). In particular, it may over-estimate `finalHold` by masking brief out-of-zone dips or under-estimate it by attenuating brief in-zone entries. The effect is small for the slow reps here but would grow for faster ones, which bounds the movement speeds for which (3) is valid. Low-visibility frames (any landmark < 0.5) are dropped, with an on-screen warning to reposition the camera.

Voice feedback and compensation detection. Per-rep scores are spoken via the Web Speech API in Chinese or English with a 2-second debounce so fast reps do not stack utterances; users can toggle voice off and that flag is recorded per session. Compensation detection is rule-based and reuses the smoothed landmark stream from the scoring pipeline (§4.1). As a representative rule, trunk-lean during shoulder forward flexion is measured from the trunk axis between the mid-shoulder point S = midpoint(landmarks 11 and 12) and the mid-hip point P = midpoint(landmarks 23 and 24). The trunk's angular deviation from image vertical is

$$\beta = |\arctan2(S_x - P_x,\ P_y - S_y)| \times \frac{180}{\pi} \qquad (4)$$

where coordinates are MediaPipe's image-normalized values (y increasing downward, so $P_y - S_y > 0$ for an upright posture and $\beta \approx 0$). β is smoothed with the same 20-frame moving average as the joint angles, and is taken relative to a per-rep baseline β_0 set at rep start (Idle → Holding). The baseline cancels a user's neutral standing posture or a slightly tilted camera. A trunk-lean event is counted when $|\beta - \beta_0|$ exceeds 15° for at least 0.5 s while all four trunk landmarks stay visible (visibility ≥ 0.5). A 15° / 10° hysteresis band stops one long lean from counting several times: the event registers at 15° and does not re-arm until $|\beta - \beta_0|$ drops back below 10°. Only the per-session count reaches the backend (G2, §5); no per-frame trunk angles are stored. A single camera cannot see front-to-back lean, so the rule covers side-to-side (coronal) lean only. The thresholds were hand-authored and reviewed by the clinical advisor (Acknowledgements); a learned classifier is left for future work (§8).

On-device performance. Fps and p95 latency were measured on iPhone 16 (A18) and iPhone 14 (A15) via `performance.now()` in MediaPipe's `onResults` callback, averaging several 40–60 s sessions per Table 4 cell. MediaPipe dominates inference; the scoring layer adds under 1 ms/frame. At the default `complexity = 1`, iPhone 16 runs 38.5 fps and iPhone 14 31.0 fps, well above the real-time threshold. The 16–25% inter-device gap informs the §8 L3 warning about older hardware.

Adding new exercises. Fifteen exercises are currently active across four joints (shoulder, knee, hip, ankle); adding one needs only a config row (three landmarks, rep direction, compensation rules) plus per-week rows in `AiCoachPhaseStandard` (Table 3), with no code change.

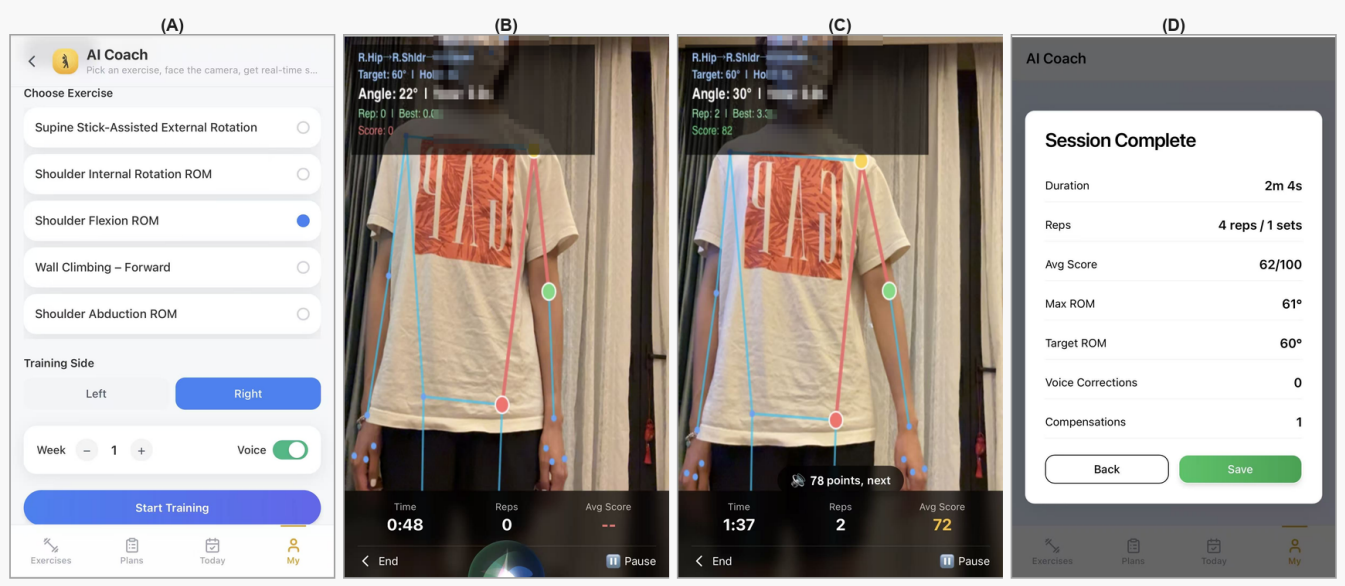

Figure 3: The five-step session workflow (§3.4) for shoulder forward flexion: (A) exercise selection; (B) live capture with skeleton overlay; (C) per-rep score, displayed and spoken; (D) session summary (G2 scalars only, no frames). ***Screenshots on iPhone 16; the subject's face in (B) and (C) is anonymized.***

Table 4: End-to-end performance on two reference devices across all three `modelComplexity` settings. Each cell averages several 40–60 s exercise sessions; *fps* is the mean frame-rate, *p95 lat* the 95th-percentile per-frame inference latency.

`modelComplexity`	iPhone 16 — fps	iPhone 16 — p95 lat (ms)	iPhone 14 — fps	iPhone 14 — p95 lat (ms)
0 (lite)	52.8	19.9	44.3	25.0
1 (medium, default)	38.5	30.2	31.0	33.9
2 (heavy)	10.6	96.8	7.9	140.3

7 Field Evaluation

The system was tested with seven post-surgical patients to see whether it could run reliably outside the author's own testing. The deployment was a feasibility check, **not a clinical trial**. The results show only that the app worked end-to-end on real phones for several weeks, produced reasonable (non-random) scores, and respected the privacy boundary of §5. Whether a higher PostureScore actually corresponds to better clinical recovery is a separate question (§8 L5).

Scale. Between February and April 2026 PostureScore was distributed as a private build to seven post-surgical patients (ACL-reconstruction and shoulder-arthroscopy, ages 19–30) recruited through Fu's Orthopedic Hospital, Hangzhou. Each followed an 8- to 12-week home plan; the app recorded 664 scoring sessions.

System stability. All 664 sessions ran through §4's pipeline without crashing: every session produced a non-null average score, peak ROM angle, and total-rep count, and there were no recorded MediaPipe inference failures or runtime crashes. Because a session is recorded only once its summary reaches the backend, the count is a lower bound on stability: a client-side crash before upload would leave no record. Voice feedback was on by default and toggleable per session (the flag is recorded). Three of the seven users had shoulder arthroscopy and used a shoulder library shaped by the author's prior rehabilitation experience (§3.3).

Score trajectory across rehabilitation phases. As a basic check that (3) was producing reasonable values in real use, each user's session-average scores (the `avgScore` field in G2) were grouped by plan-week (early 1–4, mid 5–8, late 9 and beyond; plan-week is the user's own progression through their 8- to 12-week plan, not calendar week) and averaged within user, then across users (equal weight per user). The cohort means were 57 / 65 / 68 (sample SD 3.9 / 5.0 / 5.1; Figure 4). Two of seven users contributed no late-phase data (one's plan ended before the late phase; the other practiced mostly in the first half), so the late mean is n = 5 versus n = 7. Among the five users present in all three phases, each scored higher late than early, so dropout does not explain the rise. The rise should not be read as clinical improvement. Because the scoring standards (θ_t, τ) also tighten each week,

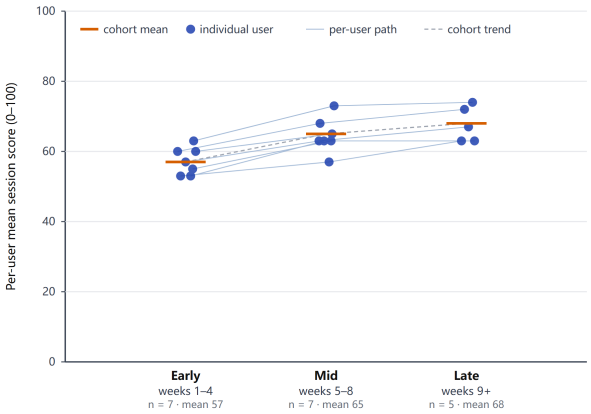

Figure 4: Mean session scores across rehabilitation phases (dots, per-user means; bars, cohort means; thin lines, within-subject paths; n = 7 early/mid, n = 5 late); a feasibility check of (3), not evidence of clinical efficacy (§7).

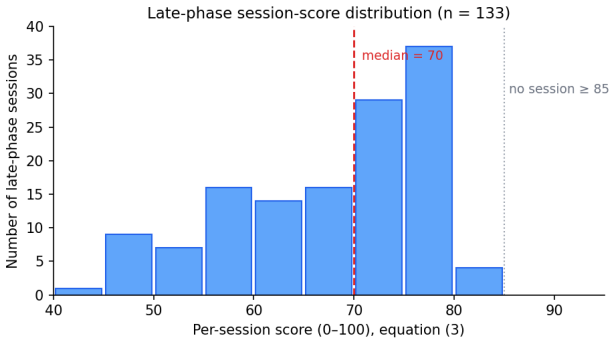

Figure 5: Per-session score distribution across the 133 late-phase sessions (5-point bins; §7).

the score geometry shifts independently of movement quality (§4.6): narrowing τ shortens totalTime (tending to raise the steadiness ratio e) while also lowering finalHold, so the net parameter effect on (3) is non-monotonic instead of a clean directional bias, and real change cannot be separated from parameter effects even in sign. Separating them would require re-scoring the same repetitions under a fixed (θ_t, τ), which needs the per-rep `finalHold` and `totalTime` the deployment did not retain (§5). The de-identified session-level dataset is publicly archived [14].

Score distribution and ceiling effects. The $\sqrt{e}$ factor is concave (§4.5), so it compresses scores near the top and could blur already-good repetitions if real scores piled up there. Figure 5 shows they did not. Across the 133 late-phase sessions the scores span 42–81, with a median of 70 and most in the 55–80 band; the highest was 81 (per-session; per-user means are lower, Fig. 4), well short of 85 and the 100 ceiling. The concavity therefore had no real effect here, since the scores never reached the flat part of $\sqrt{e}$. At the same time, 16 of the 133 sessions fell below 55, so the low end still separates. Because nothing reached a ceiling, saturation cannot explain the late-phase rise in the cohort means.

Ethics. All seven participants gave written informed consent at sign-up; the consent explicitly authorized de-identified, aggregate results to be used both for product evaluation and for academic publication. The data reported here are session-level summary numbers, not video and not landmarks. Before deployment, the Research Administration Office of Fu's Orthopedic Hospital (Hangzhou), the institution that referred the participants, reviewed this work and determined in writing that, because it analyzes only de-identified secondary records arising from ordinary product use rather than an interventional protocol, it did not require formal institutional ethics-committee review. Consistent with that determination, PostureScore was offered as a rehabilitation aid that users chose to use rather than as an interventional study; the in-app rehabilitation protocol (exercise selection, reference angles, tolerance thresholds, and per-week progression; Table 3) was reviewed for safety before deployment by the clinical advisor named in the Acknowledgements, who also referred the seven participants. Participation was entirely voluntary and did not affect any participant's clinical care or treatment. The scope of this determination (a not-required finding issued by the hospital's research administration office rather than full prospective ethics-board approval) is noted as a limitation (§8 L6).

8 Discussion and Limitations

Seven limitations are stated upfront, grouped as clinical, technical, and design, so readers can judge the contributions in §§4–5 against what they do not establish. The original labels (L1–L7) are retained to preserve cross-referencing with the previous review response.

Clinical limitations.

L4 — Deployment scale. Seven users show the pipeline runs end-to-end in the wild but cannot establish general robustness; all came from one hospital and were aged 19–30.

L5 — No external validation against a clinician. These results do not prove a higher PostureScore means better clinical recovery.

L6 — Ethics clearance was a not-required determination, not full board approval. The clearance was a written "no formal review required" determination issued before deployment by the research administration office of Fu's Orthopedic Hospital (the same institution that referred the participants), on the basis that the study uses only de-identified secondary data (§7); it was not a full prospective ethics-board protocol, and the determining office is not independent of the deployment, a conflict of interest for any future clinical claim. Any future study that makes clinical claims, or that collects data prospectively as research rather

than as ordinary product use, should obtain full, independent institutional ethics approval first.

Technical limitations.

L2 — The system trusts the camera angle. A user who positions the phone so a movement *looks* like it reaches the target can game the score, and (3) does not detect this; the rule-based compensation detection in §6 catches only a few common patterns, not a clever camera placement.

L3 — WebView inference performance ceiling. Much older hardware (e.g. iPhone 11 / A13) would run markedly slower than the tested devices, stretching the 20-frame smoothing window and lagging the voice feedback; such hardware was not available to test.

L7 — The steadiness ratio can penalize good form. Because `totalTime` includes the descent out of the zone (§4.3), a slow, controlled eccentric lowering (which is clinically desirable) lowers e and hence the score.

Design limitation.

L1 — Targets reviewed by a rehab doctor, not learned from data. The (θ_t, τ, H) triples in Table 3 were hand-authored and reviewed by one rehabilitation specialist (Acknowledgements); the tolerances chosen for young-adult ACL and shoulder patients would likely need redoing for older patients or children.

Each limitation points to a next step: a larger multi-site cohort (L4), independent therapist scoring with an agreement statistic such as ICC (L5), and full independent ethics approval before any clinical-claim study (L6); on the technical side, a second camera view or inertial cross-check against camera-angle gaming (L2), benchmarking on older A11–A13-class hardware with a WebGL/Metal inference path (L3), and scoring the eccentric phase on its own tempo (L7); and refitting the hand-authored targets from multi-age data with a mixed-effects model (L1). The clinician-scoring (L5) and eccentric (L7) studies both depend on the per-rep logging in §5.

9 Conclusion

PostureScore is an on-device, phase-aware rehab-scoring app for home use after surgery. Its core is the scoring formula (3), combining hold time and movement steadiness into a 0–100 score, together with the week-indexed standards (Table 3) that shift targets across a rehabilitation course. The privacy design (§5) keeps raw video in the phone's WebView and sends only session summaries, enforced at the schema level. The early-week tolerance, per-rep voice feedback, and on-device-only inference come from the author's prior rehabilitation experience (§3.3). Across 664 sessions on seven patients the pipeline ran end-to-end. This is an engineering feasibility check rather than a test of clinical efficacy (§7, §8 L5). The algorithm is specified in full (Algorithm 1, equations (1)–(3), and the Table 3 schema), so it can be reimplemented without the original source; implementation code is available from the author on reasonable request.

Acknowledgements

This project came out of my own experience recovering from ACL surgery (June 2025) and shoulder arthroscopy (December 2025), and several of its design choices come directly from problems I ran into during those rehabs. I thank Z. Wang for technical discussions on system architecture, and Dr. Youzhang Huang (Fu's Orthopedic Hospital, Hangzhou) for reviewing the rehab protocol and referring the seven users in §7.

References

[1] J. J. Abraham, S. Kumar, and K. K. Vedavyasa Acharya, "Determinants of patient compliance with post-operative rehabilitation after lower limb orthopaedic surgery: a prospective study," *BMC Musculoskelet. Disord.*, vol. 26, no. 1, art. 1065, 2025. doi: https://doi.org/10.1186/s12891-025-09320-5

[2] N. J. Maher, C. Brogden, A. C. Redmond *et al.*, "Disparity in anterior cruciate ligament injury management: a case series review across six National Health Service trusts," *BMC Musculoskelet. Disord.*, vol. 26, no. 1, art. 363, 2025. doi: https://doi.org/10.1186/s12891-025-08572-5

[3] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-device real-time body pose tracking," *arXiv*:2006.10204, 2020. URL: https://arxiv.org/abs/2006.10204

[4] W. Simoes, L. Reis, C. Araujo, and J. Maia Jr., "Accuracy assessment of 2D pose estimation with MediaPipe for physiotherapy exercises," *Procedia Comput. Sci.*, vol. 251, pp. 446–453, 2024. doi: https://doi.org/10.1016/j.procs.2024.11.132

[5] S. Akturk, F. B. Derdiyok, and K. Serbest, "Markerless joint angle estimation using MediaPipe with a rapid setup for joint moment calculation," *Multimed. Tools Appl.*, vol. 85, no. 1, art. 38, 2026. doi: https://doi.org/10.1007/s11042-026-21256-z

[6] S. Dill, A. Ahmadi, M. Grimmer, D. Haufe, M. Rohr, Y. Zhao, M. Sharbafi, and C. Hoog Antink, "Accuracy evaluation of 3D pose reconstruction algorithms through stereo camera information fusion for physical exercises with MediaPipe Pose," *Sensors*, vol. 24, no. 23, art. 7772, 2024. doi: https://doi.org/10.3390/s24237772

[7] R. I. Hamilton, Z. Glavcheva-Laleva, M. I. H. Milon, Y. Anil, J. Williams, P. Bishop, and C. Holt, "Comparison of computational pose estimation models for joint angles with 3D motion capture," *J. Bodywork Mov. Ther.*, vol. 40, pp. 315–319, 2024. doi: https://doi.org/10.1016/j.jbmt.2024.04.033

[8] M. MohammadNamdar, M. L. Wilson, K.-P. Murtonen, E. Aartolahti, M. Oduor, and K. Korniloff, "How AI-based digital rehabilitation improves end-user adherence: rapid review," *JMIR Rehabil. Assist. Technol.*, vol. 12, no. 1, art. e69763, 2025. doi: https://doi.org/10.2196/69763

[9] C. D'Amore, J. C. Reid, M. Chan, S. Fan, A. Huang, J. Louie, A. Tran, S. Chauvin, and M. K. Beauchamp, "Interventions including smart technology compared with face-to-face physical activity interventions in older adults: systematic review and meta-analysis," *J. Med. Internet Res.*, vol. 24, no. 10, art. e36134, 2022. doi: https://doi.org/10.2196/36134

[10] Sword Health, "The cost of digital physical therapy vs. traditional PT: an employer's guide," industry report, 3 Oct. 2025. URL: https://swordhealth.com/articles/digital-physical-therapy-costs-vs-traditional-pt-costs *(cited as operational evidence; not peer-reviewed.)*

[11] Apple Inc., "Privacy manifest files," *Apple Developer Documentation*, 2024. URL: https://developer.apple.com/documentation/bundleresources/privacy_manifest_files

[12] A. Walker, W. Hing, and A. Lorimer, "The influence, barriers to and facilitators of anterior cruciate ligament rehabilitation adherence and participation: a scoping review," *Sports Med. Open*, vol. 6, art. 32, 2020. doi: https://doi.org/10.1186/s40798-020-00258-7

[13] Ionic / Drifty Co., "Capacitor 8 documentation: cross-platform native runtime for web apps," 2024. URL: https://capacitorjs.com/docs

[14] E. Wang, "PostureScore deployment dataset (n=7, 664 sessions)," Open Science Framework, 2026. doi: https://doi.org/10.17605/OSF.IO/3N4Y8