Federal University of Technology - Paraná – UTFPR
Academic Department of Electronics – DAELN – DAELN
Academic Department of Informatics – DAINF – DAINF
Computer Engineering
Integration Workshop 3 (ELEX22) – S72 – 2025/2

Technical report

PosturePal: Ballet Posture Monitor

Carlos Eduardo Obristo – carlosobristo@alunos.utfpr.edu.br
Ricardo Rodrigues Affonso – ricardoaffonso@alunos.utfpr.edu.br
Rodrigo Moliani Braga – rodrigomoliani@alunos.utfpr.edu.br
Sebastião Antonio Araujo Neto – sneto.2001@alunos.utfpr.edu.br
Waine Barbosa de Oliveira Junior – waine2000@gmail.com

November 2025

Abstract

The PosturePal project presents the development of an automated posture evaluation system designed specifically for ballet dancers. Traditional feedback methods, such as mirrors, instructors, or recorded videos, offer limited perspectives and lack objective, real-time assessment, often slowing down learning and increasing the risk of improper technique. PosturePal integrates mechanical motion, embedded electronics, and computer vision to deliver consistent and multi-angle posture analysis. A smartphone is mounted on a motorized cart that follows a 180º arc around the dancer, capturing images from several viewpoints. A mobile application processes the footage, evaluates posture accuracy on a 0–100 scale, and provides a personalized dashboard for progress tracking. Electronic components include an ESP32-based control system, DC motors with an H-bridge driver, an encoder for motion monitoring, and infrared sensors for end-of-rail detection. By combining accessible hardware and intelligent software, PosturePal offers dancers immediate, objective, and easy-to-understand feedback, enabling safer practice, faster improvement, and more independent training.

1 Introduction

In classical ballet, posture and alignment are fundamental skills that directly influence technique, performance quality, and long-term physical health [1]. Yet, despite their importance, dancers often struggle to evaluate their own posture objectively. Traditional tools, such as studio mirrors or video recordings, can be restrictive: mirrors show only a frontal perspective, videos require manual recording and later review, and constant instructor supervision is rarely feasible. In a training environment where precision matters and repetition is essential, these limitations impact both the learning process and the dancer's confidence.

The challenge becomes even more relevant as ballet training increasingly intersects with modern technology. While digital platforms have expanded access to instructional content, they rarely address real-time posture evaluation, a need that remains unmet for beginners and experienced dancers alike. A system capable of analyzing alignment automatically, consistently, and from multiple angles can significantly enhance the training experience.

The PosturePal project was created with this goal in mind: to provide dancers with an intuitive, automated, and data-driven tool for posture analysis. By combining mechanical design, embedded control, and mobile computer vision, the system offers a new way to observe technique, one that is objective, accessible, and synchronized with the dancer's movements.

1.1 Proposed Solution

To overcome the limitations of traditional feedback methods, PosturePal introduces an integrated solution built around three components: a motorized camera platform, an embedded control system, and a mobile application for posture evaluation.

Mechanically, the system features a smartphone mounted on a rail-based motorized car capable of traveling along a 180º semicircular path around the dancer. This motion allows the camera to capture the dancer from multiple angles without requiring manual repositioning. The smooth and repeatable movement ensures consistent recording conditions across training sessions.

Electronically, the car is driven by an ESP32 microcontroller connected to two DC motors through a motor driver. Infrared sensors placed at the ends of the rail detect physical barriers and signal the system to stop automatically, ensuring safe and precise operation. A rotary encoder tracks wheel rotation to monitor movement and detect potential faults, such as stalling or misalignment on the rail. Bluetooth communication allows the mobile application to remotely trigger the car's motion and synchronize video capture with analysis.

On the software side, the mobile application processes the recorded footage using computer vision algorithms capable of evaluating the dancer's posture. The system produces a numerical accuracy score, displays feedback in real time, and stores performance history in a personal dashboard. Users can also export reports summarizing their progress over multiple training sessions.

Together, these components create a unified platform that enhances the learning experience by providing objective, multi-angle posture evaluation. The solution supports independent practice, reduces reliance on instructor-only feedback, and promotes safer, more informed training for dancers at any level.

2 Project Specification

As system specifications of the PosturePal project establish the functional, mechanical, electronic, and software requirements necessary for reliable arc-based

camera movement and accurate real-time posture analysis. These requirements define the constraints for structural stability, sensing, motion control, and interaction with the mobile application, as well as the computational processes used for pose detection, metric extraction, and user-level feedback. They ensure that the mechanical platform operates safely along the 180° rail, that the embedded electronics provide consistent control and sensor monitoring, and that the software stack delivers a complete evaluation workflow, from video capture to scoring and report generation. Tables below summarize all initial requirements defined for the system. Some specifications, particularly those related to absolute position tracking along the arc, were later classified as non-mandatory and intentionally omitted after practical testing demonstrated that relative motion and end-stop detection were sufficient for the intended functionality.

2.1 Mechanical requirements

Types	Number	Description
FR	1	The camera must travel along a rail in a 0–180° arc, with speed set by firmware and smooth motion (no visible jerks).
FR	2	The rail must include basic end-stop switches to prevent overtravel and allow manual homing at startup.
FR	3	The rail must include limit switches or optical sensors at 0° and 180° to prevent overtravel.
FR	4	The system must expose to the application basic states (IDLE / MOVING / ERROR) for monitoring.
FR	5	The camera carriage must be securely fixed to the rail to avoid derailment during movement.
FR	6	The system must ensure the camera maintains a stable orientation while moving along the rail.
FR	7	The rail must support the weight of the camera and carriage with a safety factor of at least 2× the total load.
FR	8	The system must include cable management (e.g., cable chain or flexible wiring path) to prevent tangling or obstruction during motion.

2.2 Hardware requirements

Types	Number	Description
FR	1	The system must operate using a standard smartphone camera (Android/iOS) mounted on the mobile support.
FR	2	The system must measure the movement velocity with a error tolerance of 30%.

FR	3	The system must operate with a battery and without a wire connected to the power socket.
FR	4	The system must detect the limits of the mechanical arc.
FR	5	The system must move the mechanical installation using motors capable of reaching a speed of at least 0.4m/s.
FR	6	The system must track where it is in the arc with a precision of 10 degrees.
FR	7	The system values for its tracking must be calibrated in its initialization.
FR	8	The system must control its power through an on/off button.
FR	9	The system must control its speed, reaching the target speed in at most 5s.
FR	10	The system must be able to go to a given position in the arc.
FR	11	The system must be able to move in both directions.
FR	12	The system must receive control commands from the smartphone.

2.3 Software requirements

Types	**Number**	**Description**
FR	1	The application must contain a metrics dashboard with results of each training session, including performance history.
FR	2	The application must allow users to create and manage a personal profile.
FR	3	The application must provide secure user login and logout functionality.
FR	4	The application must be able to access the device's camera to capture a live video stream of the user.
FR	5	The application must process the video stream in real-time using MediaPipe to detect and track 3D body pose landmarks.
FR	6	The system must analyze postural alignment (torso, head, shoulders, hips) and flag deviations.
FR	7	The system must calculate joint angles (torso, head, shoulders, hips) and compare them with reference values.
FR	8	The system must calculate body symmetry metrics between the left and right sides.
FR	9	The system must provide a verdict for each position/segment, "ideal," "acceptable," or "needs correction", and assign a score (0–100).
FR	10	The application must allow the user to select a specific ballet exercise for focused analysis.

FR	11	The system will have at least 5 poses for the user to select.
FR	12	The application must automatically generate a detailed performance report after each session concludes.
FR	13	The report must include quantitative metrics such as an overall posture score, frequency of specific errors, and stability scores.
FR	14	The system must identify and discard frames with low landmark confidence scores.
FR	15	The user must be able to export and share their session reports (e.g., as a PDF).
FR	16	The application must provide visual playback of captured videos with body landmarks overlaid for feedback.
FR	17	The application must generate spoken real-time feedback for the user (e.g., “Right arm too low,” “Torso leaning to the left”).
FR	18	The application must store session history and display progress over time through charts and graphs.
FR	19	The application must allow users to repeat the analysis of a pose until the desired score is achieved.
FR	20	The application must provide a calibration routine before the first training session to ensure correct camera positioning and lighting.
FR	21	The system must allow the user to delete session data and reports permanently from the device.
FR	22	The application must provide basic error messages in plain language (e.g., “Low light detected, please improve lighting”).
FR	23	The application must generate a 3D model of the dancer.

2.4 Risk management

In parallel with the requirements definition, the team conducted a risk analysis to identify the main threats to the project schedule and to the quality of the final prototype. The risks were grouped into mechanical, hardware, software and logistical categories, and each item received likelihood and impact ratings together with a mitigation strategy. Typical examples included insufficient motor torque to move the carriage along the rail, instability of the hose-based structure, limitations of the mobile device for real-time processing, delays in the delivery of electronic components and temporary unavailability of team members. Mitigation actions ranged from redesigning the mechanical structure (e.g., adopting a dual-hose guide and stronger motors) to defining fallback procedures such as manually walking around the dancer with the smartphone in case the rail system was not ready. Several of these mitigation measures were actually applied during development and are reflected in the design decisions reported in the following sections.

3 Development

3.1 Mechanical Development

The mechanical structure of the PosturePal system was developed to provide smooth and stable motion for the camera along a 180° arc around the dancer. The project began with the construction of the camera carriage, built on a 45 cm × 45 cm wooden base chosen for its rigidity and durability. Two DC motors with traction wheels were mounted underneath the board, supported by wooden spacers to achieve the correct height. A PVC pipe was fixed vertically at the center of the carriage using metal brackets, serving as the mast for the smartphone and ensuring that the camera remained at the required elevation to capture the dancer consistently from all angles.

To guide the movement, the first rail prototype consisted of a flexible hose mounted onto several wooden segments cut at matching angles to form a 2-meter-radius arc. This modular approach made transportation and installation easier, and early tests confirmed that the carriage could follow the curved path without major issues. However, after integrating the full system, the team identified limitations with this mechanism: the single-hose guidance did not provide enough lateral constraint, causing small misalignments and occasional derailment.

To address this, the rail system was redesigned using two parallel hoses instead of one. The spacing between the hoses created a physical channel, and the carriage was fitted with two vertically mounted screws holding ball bearings at their tips. These bearings ran between the hoses and constrained the carriage laterally as the motors propelled it forward. This configuration provided a significant improvement in stability, reducing friction, preventing derailment, and ensuring that the camera followed the arc smoothly throughout the trajectory. The redesign also maintained the advantages of easy installation and low-cost materials, meeting the project's mechanical requirements for reliability, transportability, and safety.

Figure 3 shows the initial carriage construction and the final dual-hose guiding mechanism. Also, the Figure 4 shows the full rail trajectory.

Figure 1: *
Carriage base with PVC support.

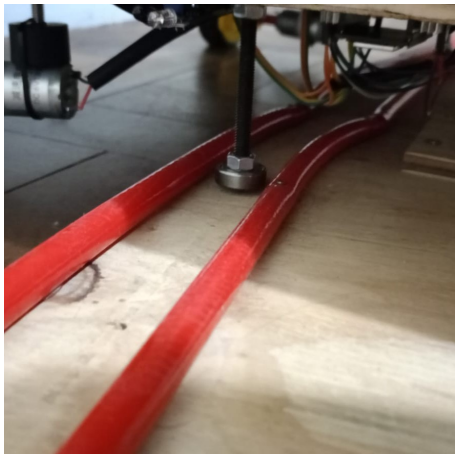

Figure 2: *
Final dual-hose rail guiding mechanism.

Figure 3: Mechanical structure: initial carriage (left) and final rail system (right).

Figure 4: Full rail demonstration

3.2 Electronic Development

The electronic system of the PosturePal was designed to provide reliable motor control, sensor feedback, and communication with the mobile application while remaining compact enough to fit beneath the carriage structure. The core of the system is an ESP32 microcontroller [2], chosen for its integrated Bluetooth

capabilities and sufficient processing power for real-time control. Two DC motors are driven through an L298N dual H-bridge driver[3], allowing independent control of direction and speed. A rotary encoder [4] is attached to one of the motor shafts to measure wheel rotation, enabling the firmware to estimate velocity and detect abnormal motion conditions.

During development, the original design used two Hall effect sensors as endstop detectors, but the final system replaced them with two reflective infrared (IR) sensors [5]. These sensors were mounted facing downward toward small wooden barriers placed at both ends of the rail. When the carriage reaches an endpoint, the IR sensor detects the barrier and triggers an immediate stop, ensuring safe operation without requiring physical contact. All electronic components—including the ESP32, the H-bridge, cabling, and supporting circuitry—were installed on the underside of the wooden base to protect them and improve stability. Only the battery pack remained on the upper side for accessibility and space management. The interactions between these hardware components and the control logic are further detailed in the firmware section.

Figure 7 shows the final electronic schematic and the assembled electronics mounted on the carriage.

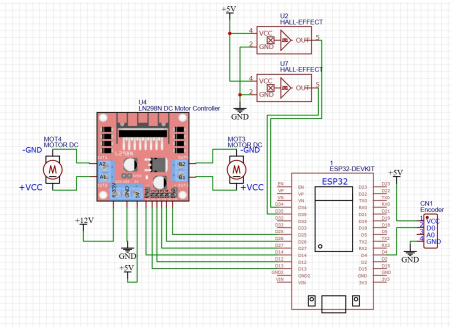

Figure 5: *
Electronic schematic of the ESP32, motor driver, encoder, and IR sensors.

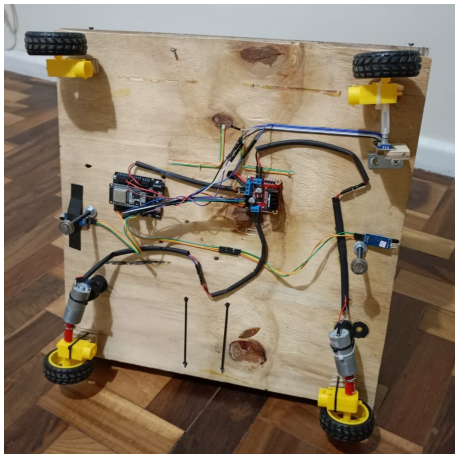

Figure 6: *
Final assembly of the electronics under the carriage base.

Figure 7: Electronic system design and physical implementation.

3.3 Firmware Development

The firmware of the PosturePal system was designed to coordinate all real-time control tasks required for safe and consistent motion along the rail. It runs on an ESP32 microcontroller and integrates motor actuation, velocity sensing, infrared-based end detection, and Bluetooth communication with the mobile

application.

At its core, the firmware uses a Bluetooth-based command interface in which the app sends a single movement command. Rather than requiring directional control from the app, the ESP32 internally alternates between left and right movement on consecutive activations. This approach simplifies the application logic while ensuring the carriage completes its full 180° traversal in both directions with no additional user input.

Motor control is implemented using an L298N H-bridge, driven through two hardware PWM channels for smooth speed modulation. A rotary encoder attached to one of the wheels generates pulses that are counted via an interrupt service routine, allowing the firmware to estimate instantaneous velocity. These velocity readings are used to detect abnormal conditions such as wheel slip or stall. If the encoder reports zero motion for longer than a defined threshold while the system is attempting to move, the firmware immediately triggers an emergency stop to protect the mechanical structure.

End-of-rail detection is handled by two infrared reflective sensors mounted on the carriage. Wooden markers placed at both ends of the track reliably trigger these sensors when the carriage reaches either extremity. To avoid false positives caused by noise or momentary reflections, the firmware requires a minimum number of consecutive sensor activations before accepting an end-stop event. Once one of these sensors is triggered, the system applies an electronic brake by forcing both motor terminals to a high state, bringing the carriage to a rapid and controlled stop.

The firmware also includes a lightweight PID controller supporting closed-loop velocity regulation. Although the final system operates at maximum power for practical reasons, the controller remains integrated to ensure stability and to allow future extensions such as adaptive speed control. The control loop processes encoder data, computes proportional–integral–derivative corrections, and translates them into PWM updates at regular intervals.

Communication with the mobile app follows a simple protocol. The ESP32 listens for a movement command and, upon finishing the traversal or detecting a stop condition, transmits a “stop” message back to the application. This signal is used by the app to conclude the posture-analysis process, synchronizing the physical motion with the computer-vision pipeline.

Overall, the firmware ensures that all safety, motion, and communication requirements are fulfilled through a combination of real-time sensing, closed-loop control, and robust state management. Its modular design makes the system reliable during continuous operation while supporting future improvements without significant architectural changes.

3.4 Software development

3.4.1 User Interface

The user interface of the PosturePal system was developed using React Native, providing a cross-platform mobile application with consistent layout and behavior across devices. The interface functions as the main access layer to the system, enabling users to authenticate, initiate an evaluation, review past sessions, and consult performance statistics.

The application follows a modular screen structure composed of authentication, exercise selection, evaluation, and results visualization. After login, the user is presented with a dashboard summarizing recent evaluation metrics, historical scores, and weekly progress trends. Exercises are listed with descriptors and difficulty levels, allowing users to navigate directly to the analysis workflow.

During an evaluation session, the application displays the live camera preview together with concise instructions related to the selected ballet position. After the firmware signals the end of the movement, the app processes the recorded frames and presents a score along with categorized feedback. Users can also export a structured PDF report containing the session analysis.

All user data, including profiles, historical evaluations, and settings, is stored locally using an SQLite database, ensuring offline operation and fast access to historical records.

A representative interface example is shown in Figure 10, illustrating the visual structure adopted throughout the application.

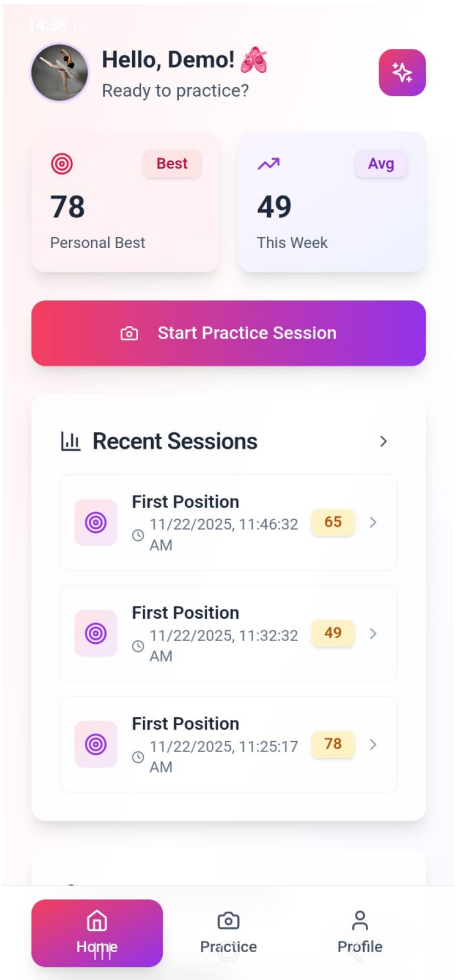

Figure 8: *
Home Screen

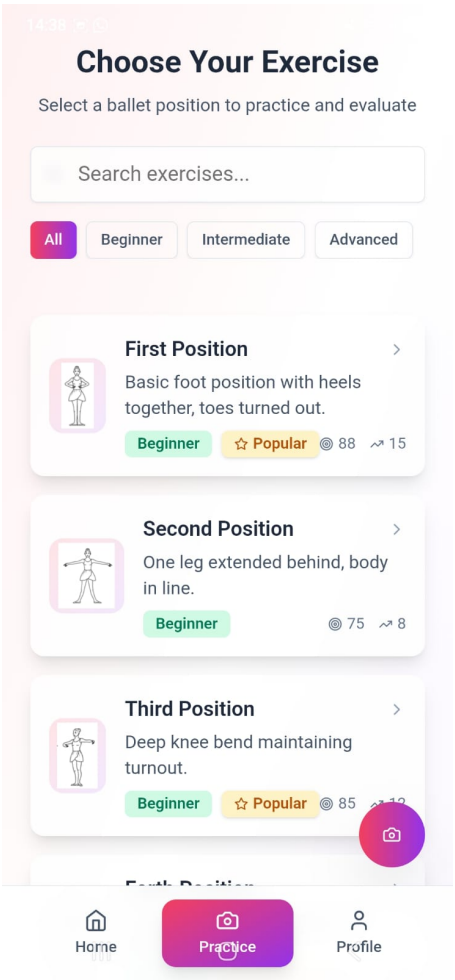

Figure 9: *
Exercises Screen

Figure 10: Examples of the interface screen of the PosturePal mobile application.

3.4.2 Posture Analysis Pipeline

The posture-analysis module is responsible for evaluating the user's execution of each ballet position using anatomical landmarks extracted from the smartphone camera. The system operates on-device using MediaPipe Pose [6, 7], which detects 33 keypoints with normalized (x, y) coordinates and an optional visibility score. All computed metrics—angles, alignments, and posture deviations—are derived directly from these landmarks.

The pipeline begins with a preprocessing step that verifies landmark visibility. A point is considered valid when its detection confidence exceeds 0.5. Missing or unreliable points produce maximum penalties for their correspond-

ing metrics, since these elements cannot be reliably evaluated.

Once validated, the module computes several geometric descriptors. Joint angles are obtained via a three-point formulation: for points a, b, and c, the angle at b is computed as

$$\theta = \left|\arctan 2(c_y - b_y,\, c_x - b_x) - \arctan 2(a_y - b_y,\, a_x - b_x)\right|,$$

converted to degrees and normalized to the 0°–180° interval. This formulation is used for knee flexion (hip–knee–ankle), arm extension (shoulder–elbow–wrist), and arm height (elbow–shoulder–hip).

In addition to angles, the algorithm evaluates alignment metrics using vertical or horizontal differences between symmetric points. Shoulder and hip alignment are computed using the absolute difference in their y-coordinates, while spine alignment uses the horizontal deviation between the nose and the computed midpoint of the hips. A Euclidean distance function is also available for auxiliary measurements when needed.

For each computed metric, the system compares the measured value to predefined acceptable biomechanical ranges obtained from the literature [8, 9, 10]. These ranges represent the expected execution window for each joint in classical ballet positions. Deviations from the expected range generate penalties through a linear model:

$$\text{penalty} = \left\lfloor \frac{\text{excess}}{\text{range}} \cdot P_{\max} \right\rfloor,$$

where *excess* is the amount by which the measurement falls outside the normative interval, *range* is the allowable width of that interval, and $P_{\max}$ is the maximum penalty for that metric. The implementation automatically handles both increasing and decreasing target intervals (e.g., "smaller is better" metrics). All penalties are saturated at $P_{\max}$.

Throughout the analysis, the system evaluates the key posture components relevant to the selected exercise. Shoulder and hip alignment are checked first, followed by spine alignment, knee angle correctness, arm extension, and arm height. Each metric yields a numeric penalty and a normalized severity measure in $[0, 1]$ for later aggregation. This approach supports both numerical scoring and generation of user-facing feedback.

The global posture score begins at 100 and is reduced by the accumulated penalties. After all metrics are processed, the score is clamped to the 0–100 interval. Alongside the numerical result, the algorithm outputs a structured set of messages describing both correct and incorrect components detected in the frame, enabling fine-grained feedback such as *"Shoulders misaligned"*, *"Knee angle too closed"*, or *"Arms positioned at correct height"*. A secondary stage then generates a summary message based on the final score, grouping the evaluation into qualitative categories ranging from poor to excellent.

This pipeline provides a deterministic and interpretable scoring system, combining geometric analysis of MediaPipe landmarks with biomechanical reference ranges to quantify posture quality in real time during each ballet exercise.

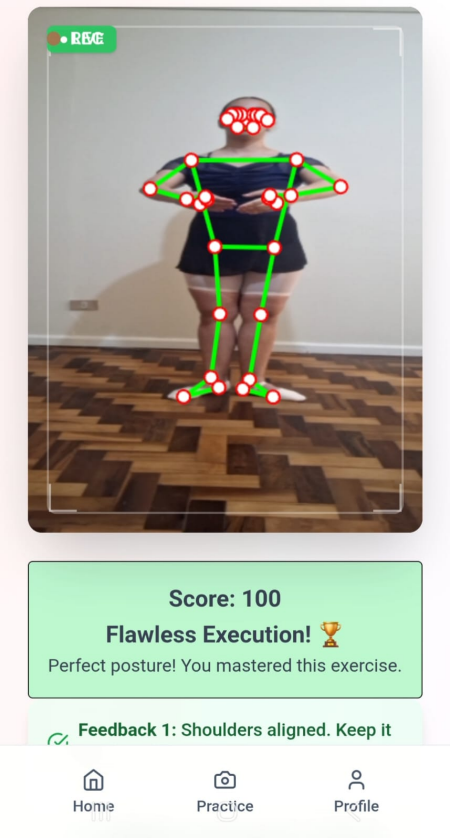

Figure 11: *
Good score.

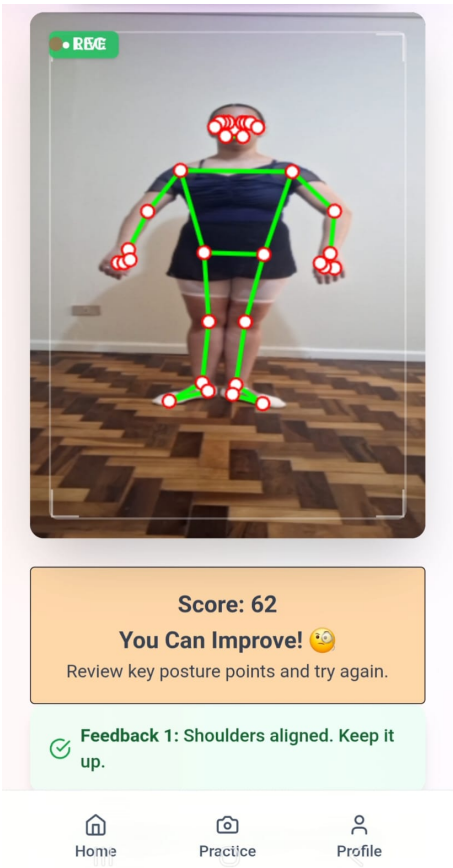

Figure 12: *
Bad score.

Figure 13: Different scores

4 Results

4.1 Budget

Component	Estimated Cost (BRL)
Wood Plank	59
PVC Tube	59
Wheels	80
"Cantoneira"	32
Hose (20m)	46
12V Battery	96
V Wheel	32
"Abraçadeiras"	20
ESP32	20
DC Motors	164
L298N Motor Driver	50
IR Sensors (x2)	40
Rotary Encoder	20
Total	**948.00**

Table 4: Estimated material cost.

4.2 Schedule

The project timeline was impacted primarily by unforeseen mechanical challenges encountered early in development. The second deliverable, in particular, accumulated a significant delay due to repeated failures involving the initial motors, issues with torque insufficiency, and the need to redesign key structural components. These setbacks created a cascading effect, increasing the workload required in subsequent phases and resulting in all deliverables exceeding their estimated hours. Despite the accumulated delays, the team adapted effectively by redistributing tasks, iterating on mechanical prototypes, and refining the control system to ensure the final functionality met all project requirements. Table 5 summarizes the worked hours and deviations relative to the original estimates, illustrating how the early-stage mechanical difficulties shaped the overall project timeline.

Deliverable	Expected Hours	Worked Hours	Delay (h)	Status
Deliverable 1	35	37.5	1.5	Delayed
Deliverable 2	45	59	14	Delayed
Deliverable 3	60	62	2	Delayed
Deliverable 4	41	45	4	Delayed
Deliverable 5	81	82	1	Delayed
Deliverable 6	53	55	2	Delayed

Table 5: Project Work Progress

4.3 Functional requirement completion

The PosturePal system was evaluated against the mechanical, hardware, and software requirements defined in the project specification. Overall, the system successfully met all mandatory functional requirements, with only optional tracking-related functions left unimplemented.

From the mechanical perspective, the final structure supports smooth motion along the 180° arc, maintains stable camera orientation, prevents derailment through the dual-hose guiding mechanism, and integrates reliable end-stop sensing using infrared detectors. The carriage withstands the load with the expected safety margin, and all structural requirements related to installation, portability, and continuous operation were fulfilled.

Regarding hardware, the ESP32-based control module, encoder feedback, motor driver, and battery system satisfied all mandatory constraints. The system accurately measures velocity, detects rail limits, operates wirelessly, and moves bidirectionally at the required speed. The only requirements not implemented were those involving absolute position tracking along the arc and moving the carriage to a specific position. These were explicitly marked as non-mandatory elements in the specification, and the final design intentionally excluded them to simplify mechanical and firmware complexity after practical testing showed that relative motion and end-detection already provided the needed behavior for the application workflow.

On the software side, all critical functionality was delivered. The mobile application includes user authentication, exercise selection, real-time posture detection with MediaPipe, angle and alignment evaluation, scoring, session history, local storage via SQLite, and PDF report generation. A simple 3D visualization module was also implemented: the system reconstructs the dancer's pose by plotting the MediaPipe landmarks in a 3D coordinate grid and connecting them via joint relationships. This allows the user to inspect their posture from different angles even without computing a full volumetric model, fulfilling the essential requirements for 3D feedback.

The only optional software features left unimplemented were those related to high-precision position tracking on the mechanical side. All core analysis requirements—such as angle calculation, symmetry evaluation, frame filtering by

confidence, feedback generation, and metric visualization—were fully implemented.

Altogether, the system satisfies the project's essential goals, delivering a fully functional mechanical–electronic–software platform capable of performing automated ballet-posture evaluation with consistent multi-angle capture and robust real-time analysis.

5 Conclusions

The development of the PosturePal system integrated mechanical design, embedded control, and computer-vision techniques to create a functional tool for automated ballet-posture evaluation. Throughout the project, each subsystem underwent several iterations as practical challenges emerged.

A major difficulty encountered early in the mechanical phase involved the initial motors, which lacked the necessary torque to move the carriage reliably along the rail. This limitation caused delays in testing and required the acquisition and integration of stronger motors. After these adjustments, the mechanical and control layers became stable enough to support the continuous movement required for video capture.

Another challenge was achieving reliable guidance along the 180° trajectory. The first rail prototype—based on a single guiding hose—presented alignment issues and occasional derailment. The redesign using two parallel hoses, combined with ball-bearing guides, produced a robust and low-cost solution that met the stability and safety requirements. This improvement was key to ensuring smooth and repeatable motion during evaluations.

On the electronic side, the transition from Hall sensors to infrared reflective sensors simplified end-detection and improved consistency. The ESP32 firmware evolved from a simple PWM driver to a structured state machine with encoder feedback, emergency stop logic, and automatic direction switching, allowing the system to operate reliably with minimal interaction from the mobile application.

In the software layer, the system achieved a complete application capable of capturing video, processing biomechanical metrics using MediaPipe, evaluating joint angles, comparing them with reference values, and generating structured feedback and historical reports. A simple but effective 3D reconstruction tool was also implemented: instead of producing a full volumetric mesh, the system renders all body landmarks in a 3D grid and connects them according to anatomical structure. This enables users to inspect their posture spatially and understand misalignments that are not obvious in 2D images.

Despite the challenges, the final system meets the primary objectives of the project: providing dancers with an accessible, objective, and multi-angle platform for posture evaluation. The combination of hardware automation and vision-based analysis enables a training experience that is consistent and independent, extending the traditional feedback tools typically available in ballet

instruction.

The project also establishes a foundation for future work, including improvements in absolute position tracking, expanded pose libraries, or more advanced 3D visualization techniques. Nonetheless, the completed system demonstrates the feasibility and value of integrating embedded electronics and computer vision into artistic training environments.

References

[1] H. Weighart et al. Insights on ten weeks of classical ballet training and postural stability. *Journal of Aging Physical Activity*, 2020. Acess: https://pmc.ncbi.nlm.nih.gov/articles/PMC7039482/.

[2] Espressif Systems. Esp32 technical reference manual, 2023. Access: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf.

[3] STMicroelectronics. L298 datasheet, 2010. Access: https://www.st.com/resource/en/datasheet/l298.pdf.

[4] Keyes / Components101. Rotary encoder ky040. Access: https://www.handsontec.com/dataspecs/module/Rotary%20Encoder.pdf.

[5] Keyes / Components101. How to use sensor fc-51 ir: Examples, pinouts, and specs. Access: https://docs.cirkitdesigner.com/component/794e0aa1-9872-489b-bdc1-5f95d749086a/sensor-fc-51-ir.

[6] Google AI Edge / MediaPipe Team. Official mediapipe solutions guide. Acess: https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker?hl=pt-br.

[7] Camillo Lugaresi et al. Mediapipe: A framework for building perception pipelines. Acess: https://arxiv.org/pdf/1906.08172.

[8] H. W. de Oliveira et al. On the track of the ideal turnout: Electromyographic and kinematic analysis of the five classical ballet positions. *PLOS ONE*, 2020. Access: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0230654.

[9] Clare M. K. et al. Relationship between hip external rotation and turnout angle for the five classical ballet positions. *Journal of Orthopaedic & Sports Physical Therapy*, 1998. Access: https://www.jospt.org/doi/epdf/10.2519/jospt.1998.27.5.339.

[10] A. E. Dogan et al. The anatomical analysis of basic stance in ballet. *Nigde Journal of Physical Education and Sport Sciences*, 2022. Access: https://dergipark.org.tr/en/download/article-file/2155239.