International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 14, Issue 1, 2026

Posey: An Ai-Powered Virtual Fitness Trainer With Real-Time Posture Correction

A Hima Bindu[1], M Samyuktha[2], D Ujwala[3], G Yashaswini[4]

[1]Assistant Professor; Department Of Computer Science And Engineering Bhoj Reddy Engineering College For Women Telangana, India

[2,3,4]B.Tech Students ; Department Of Computer Science And Engineering Bhoj Reddy Engineering College For Women Telangana, India

Mail Id; bindu.avmm@gmail.com[1], samyukthamandali95@gmail.com[2], ujwaladubbaka@gmail.com[3], yashugm78@gmail.com[4]

ABSTRACT

Posey is an intelligent, AI-driven fitness training system designed to deliver real-time posture evaluation and exercise guidance using computer vision and machine learning techniques. The platform assists users in performing workouts correctly without requiring a physical trainer, thereby minimizing injury risks and improving exercise effectiveness. The system captures live video input through a webcam and applies pose estimation algorithms to identify key body joints and compute joint angles.Based on posture analysis, Posey provides immediate feedback through on-screen notifications and voice instructions, enabling users to correct their form during exercises such as squats, push-ups, lunges, and yoga postures. The system also incorporates repetition counting, accuracy assessment, performance tracking, and progress monitoring to increase user engagement and motivation.The application is developed using HTML, CSS, and JavaScript for the frontend, while Python handles backend processing. Libraries such as MediaPipe, OpenCV, and NumPy are utilized for pose detection and movement analysis. By combining real-time posture evaluation with intelligent feedback and gamification elements such as streak tracking and leaderboards, Posey encourages safe, consistent, and effective workouts. The system bridges the gap between professional fitness coaching and home-based training, making personalized fitness guidance more accessible and interactive.

Keywords *— Artificial Intelligence, Computer Vision, Pose Estimation, Real-Time Feedback, Fitness Training, Human Pose Detection, MediaPipe, Virtual Trainer.*

INTRODUCTION

Posey is a web-based application that utilizes artificial intelligence to provide real-time posture correction and fitness guidance. Performing exercises with improper form can lead to muscle strain and injuries. Posey addresses this issue by using computer vision and machine learning algorithms to analyze body movements through a webcam and detect posture inaccuracies. The system delivers instant feedback to help users adjust their posture during workouts.The platform allows individuals to exercise at home while receiving guidance similar to a personal trainer. Additionally, it tracks workout performance, monitors improvements, and provides actionable suggestions. Features such as posture alerts, live feedback, repetition counting, and progress analytics make Posey a safer and smarter solution for fitness training.

Scope

The scope of Posey involves the development of an AI-powered virtual fitness trainer capable of analyzing user posture using webcam-based computer vision. The system provides real-time corrective suggestions for exercises including squats, push-ups, lunges, and yoga poses. It incorporates repetition counting, form scoring, and rule-based posture evaluation to ensure accurate execution of workouts.

Furthermore, Posey includes performance analytics, streak tracking, and leaderboard-based gamification features to improve user motivation and engagement. The platform aims to enhance workout efficiency while promoting safe exercise practices.

Existing System

Most existing fitness applications rely primarily on pre-recorded instructional videos or basic motion tracking technologies. These systems typically lack real-time posture evaluation and do not provide personalized feedback. Many applications depend on manual input or wearable sensors, which can reduce convenience and accessibility.

Additionally, existing solutions often focus only on counting repetitions rather than evaluating exercise form. As a result, users may unknowingly perform exercises incorrectly, increasing the risk of injuries. The absence of adaptive feedback creates a gap between professional training and home-based workouts.

Proposed System

Posey introduces an AI-based virtual fitness trainer that enhances traditional workout approaches by providing an interactive and personalized experience. The system uses computer vision and webcam input to analyze body posture in real time and guide users during exercises such as squats, push-ups, lunges, and yoga poses. This ensures correct form, reduces injury risk, and improves workout effectiveness.Posey also tracks user performance, counts repetitions, and generates progress reports. Gamification elements such as streak tracking and leaderboards are integrated to encourage consistent engagement. Overall, the system provides a safe, efficient, and accessible at-home fitness solution.

REQUIREMENT ANALYSIS

Functional Requirements

The functional requirements of the Posey system define the essential operations needed to support real-time posture correction and workout guidance. The application includes both user-side and system-side functionalities. From the user perspective, individuals should be able to register and log into the platform by providing basic personal information such as name, age, weight, and fitness goals. After authentication, users can select exercises available within the application and begin workout sessions using their device webcam. During the session, the system displays repetition counts and accuracy metrics in real time while providing corrective feedback through visual prompts and voice guidance. Additionally, users can review their performance and track progress over time to monitor improvements.

From the system perspective, the application captures live video input from the webcam and processes it using computer vision techniques. The system detects key body landmarks through pose estimation algorithms and calculates joint angles to evaluate posture accuracy. Based on movement phases, the system counts repetitions and identifies incorrect form. Real-time feedback is delivered in both visual and audio formats to help users maintain correct posture. The system also records workout session data, including accuracy scores and repetition counts, and stores them for future analysis and progress tracking.

Non-Functional Requirements

The non-functional requirements ensure the overall quality, performance, and usability of the Posey system. The application must provide real-time posture detection and feedback with minimal latency to maintain a smooth workout experience. Scalability is essential so that the system can support multiple users simultaneously while efficiently handling video streams and user data. The interface should be intuitive and easy to use, enabling users to interact with the system comfortably during exercise sessions. Reliability is another key requirement, ensuring continuous operation with minimal interruptions and stable performance during prolonged usage. The system must also maintain security by protecting user information and safeguarding personal and performance-related data. Furthermore, compatibility is required to ensure that the application functions effectively across various devices, including laptops and desktops, and supports multiple web browsers equipped with webcam functionality.

Software Resources

The software resources required for Posey include the tools, frameworks, and programming environments necessary for implementation. The system is developed using web technologies such as HTML, CSS, and JavaScript to create an interactive frontend interface. Python is used for backend processing and integration of machine learning components. Computer vision and pose estimation are implemented using libraries such as MediaPipe and OpenCV, while numerical computations are handled using NumPy. MongoDB is utilized for storing user profiles, workout metrics, and session data. The development and testing environment operates on Windows 11, which supports all required dependencies and tools. These software resources collectively provide a robust platform for building, testing, and deploying the application efficiently.

Hardware Resources

The Posey system requires standard computing hardware capable of handling real-time video processing. A system equipped with an Intel i5 processor or equivalent is recommended to ensure smooth execution of pose detection algorithms. A minimum of 8 GB RAM is required to manage simultaneous processing tasks efficiently. Storage capacity of at least 512 GB is suggested for storing application files, datasets, and user data. Additionally, an integrated or external high-definition webcam is necessary for capturing live video input, which is essential for posture detection and exercise monitoring.

Software Process Model

The development of Posey follows a structured software process model to ensure systematic implementation. Among the various models such as Waterfall, V-Model, Incremental, Spiral, and Agile, the Agile model is most suitable for this project. Agile supports iterative development, allowing the system to evolve through continuous testing and feedback. Since Posey involves AI-based posture detection and real-time performance evaluation, frequent refinements are required to improve accuracy and usability. Agile promotes collaboration among developers, enables quick adaptation to requirement changes, and supports incremental delivery of features. This approach ensures flexibility, improves software quality, and facilitates efficient development of the Posey system.

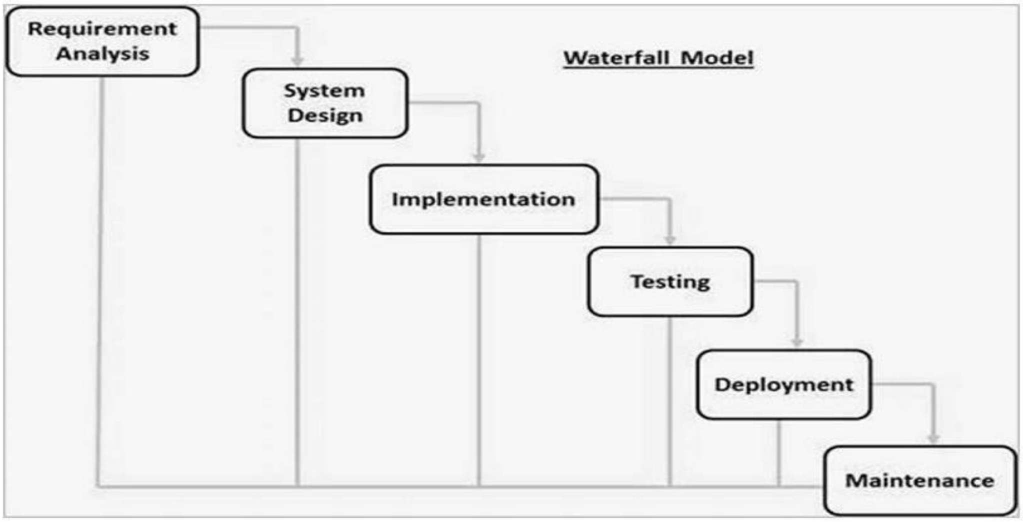


Fig.1 Waterfall Model

DESIGN

System design focuses on planning and structuring the Posey application to ensure efficient implementation and deployment. The design process begins by identifying system objectives and defining data acquisition, processing, and storage mechanisms. Posey adopts a layered architecture that separates functionality into presentation, application, data, and analytics components. The presentation layer manages user interaction, live posture visualization, and feedback display. The application layer performs core operations such as pose detection, joint angle calculation, repetition counting, and posture validation. The data layer handles storage of user profiles, session records, and performance metrics. The analytics layer generates insights such as accuracy trends, workout consistency, and progress reports. The design also includes selecting suitable algorithms, defining evaluation metrics, and ensuring a user-friendly interface for seamless interaction.

Architecture

The architecture of the Posey system defines the components involved and the interaction between them. It provides a structured representation of how requests are processed and how data flows across modules. The architecture consists of software architecture and technical architecture, each addressing different aspects of system design. These architectural components ensure efficient processing, maintainability, and scalability of the application.

Software Architecture

The software architecture of Posey focuses on designing system modules to achieve high performance and reliability. Real-time posture detection requires continuous video processing, which demands efficient handling of computational tasks. The architecture ensures that pose detection, feedback generation, and data storage modules operate seamlessly. It also allows developers to identify potential issues during development and refine algorithms to improve accuracy. By structuring the system into modular components, the architecture enhances maintainability and supports future upgrades. This design helps reduce processing delays and ensures stable performance during workout sessions.

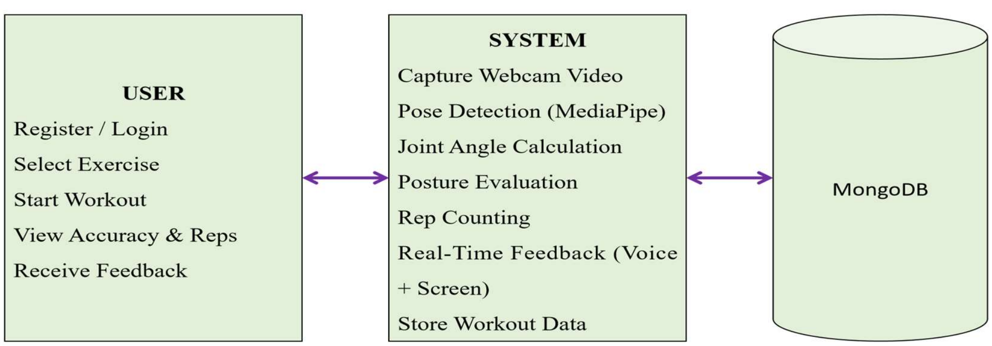


Fig.2 Software Architecture

Technical Architecture

The technical architecture provides a detailed blueprint of how hardware, software, and data components interact within the Posey system. It defines the arrangement of system layers, including the presentation layer for user interaction, the application layer for business logic and pose detection, and the data layer for storing information. The architecture also incorporates integration mechanisms for connecting libraries, APIs, and database systems. Additionally, infrastructure components such as servers and runtime environments are considered to ensure scalability and reliability. A well-structured technical architecture improves maintainability, enhances performance, and allows the system to adapt to future enhancements and increasing user demands.

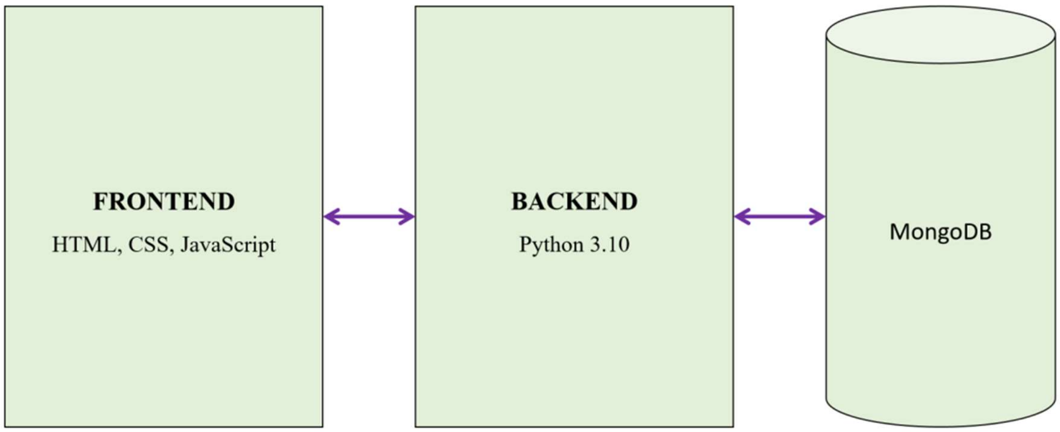


Fig.3 Technical Architecture

IMPLEMENTATION

Technologies

The Posey system is implemented using a combination of web technologies, programming frameworks, and machine learning libraries to enable real-time posture detection and feedback. The technology stack is divided into frontend and backend components, each responsible for specific functionalities within the system.

Frontend

The frontend of the Posey application is developed using HTML, CSS, and JavaScript. These technologies provide a responsive and interactive interface that allows users to perform workouts and view posture feedback in real time. HTML structures the application layout, CSS enhances visual presentation, and JavaScript enables dynamic interaction between the user and the system. The frontend includes workout dashboards, posture visualization windows, accuracy displays, and progress tracking panels. These components ensure smooth navigation and improve user engagement by presenting information clearly during exercise sessions.

Backend

Python is used as the primary backend technology to manage system operations and execute core functionalities. It facilitates communication between the user interface and processing modules, handles user input, and controls the logic for posture detection and feedback generation. The backend integrates multiple libraries to perform real-time analysis. MediaPipe is utilized for human pose estimation and landmark detection, while OpenCV manages video capture and image processing. NumPy supports numerical computations such as joint angle calculations, and Matplotlib is used for generating performance visualization graphs. These tools collectively enable accurate posture analysis and efficient feedback generation.

MongoDB is used as the database for storing system data. It maintains user profiles, workout session records, performance metrics, timestamps, and progress information. The document-based structure of MongoDB allows flexible storage of dynamic fitness-related data. This ensures efficient retrieval and updating of records, supporting scalable and reliable data management within the application.

Pseudocode Description

The Posey system follows a structured workflow for real-time posture detection and workout monitoring. Initially, required libraries such as Flask, OpenCV, MediaPipe, and NumPy are loaded, and the application server is initialized. The system establishes a connection with the MongoDB database and loads the trained machine learning model. After initialization, the camera module is activated to capture live video input.

The authentication module allows users to register and log into the system. User details are validated, passwords are securely stored, and session tokens are generated for authorized access. Once logged in, users can start workout sessions by selecting a specific exercise. The system resets counters and activates posture monitoring.

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

During video processing, each frame is converted into RGB format and passed to the pose detection module. MediaPipe extracts body landmarks, which are then used to compute joint angles for different body parts such as knees, hips, elbows, and shoulders. These angles are processed by the machine learning model to classify posture as correct or incorrect and calculate accuracy scores.

The workout logic tracks repetitions for dynamic exercises and hold duration for static poses. Based on posture evaluation, the feedback module generates corrective instructions or positive reinforcement messages. The processed frame, along with repetition count, accuracy, and feedback, is streamed to the frontend in real time.

At the end of the session, performance data including accuracy, repetitions, duration, and timestamps are stored in the database. The analytics module then calculates performance trends such as average accuracy, weekly progress, total workout time, and recent activity. Finally, the Flask server exposes routes for controlling workout sessions, streaming video, and retrieving analytics, allowing the application to operate efficiently.

TESTING

Software testing is conducted to verify that the Posey system meets functional requirements and performs accurately without defects. Since the application relies on real-time posture detection and feedback, even minor errors can affect user experience and system reliability. Testing ensures that pose estimation, repetition counting, and feedback generation operate correctly. It also improves system stability, enhances data security, and increases user satisfaction. By identifying defects early, testing reduces maintenance costs and ensures the delivery of a reliable fitness training application.

Dimensions of Testing

Testing of the Posey system is performed across multiple dimensions. These include testing at different layers such as user interface, backend processing, and database storage. Various scales of testing are applied, including unit testing, integration testing, and system testing. Different testing types such as functional testing, performance testing, accuracy validation, and security testing are also conducted. Both manual and exploratory testing approaches are used to ensure thorough validation of system functionality.

Stages of Testing

Unit testing focuses on verifying individual components of the Posey system. Functions such as pose detection, angle calculation, and feedback generation are tested independently to ensure correct operation. White-box testing techniques are used to validate internal logic and confirm that each module performs as expected.

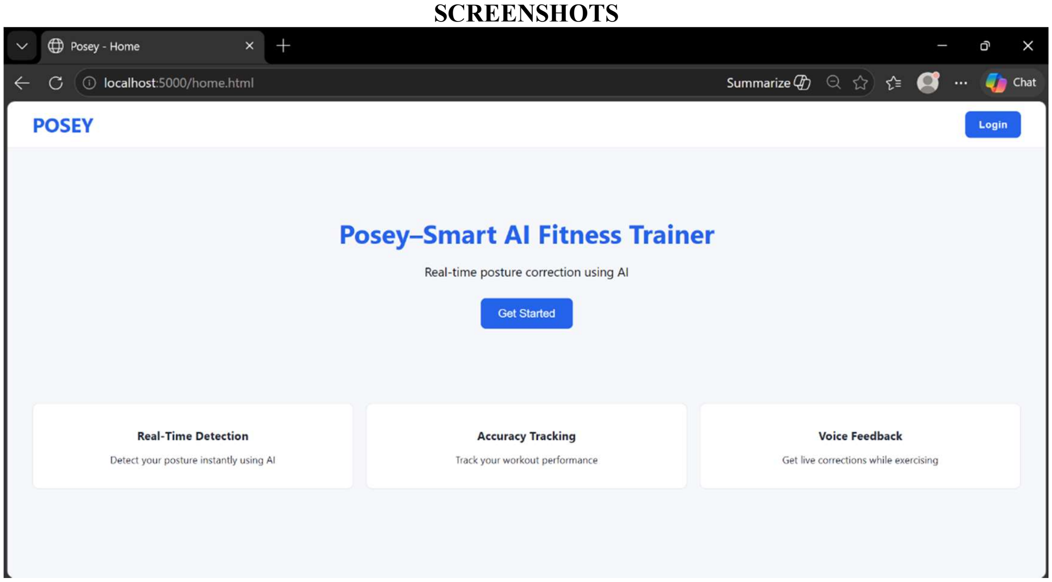
Integration Testing

Integration testing evaluates how different modules interact with each other. In the Posey system, testing is performed between the frontend interface, backend processing, pose detection module, machine learning model, and database. This stage ensures that data flows correctly between components and that feedback is generated and displayed properly.

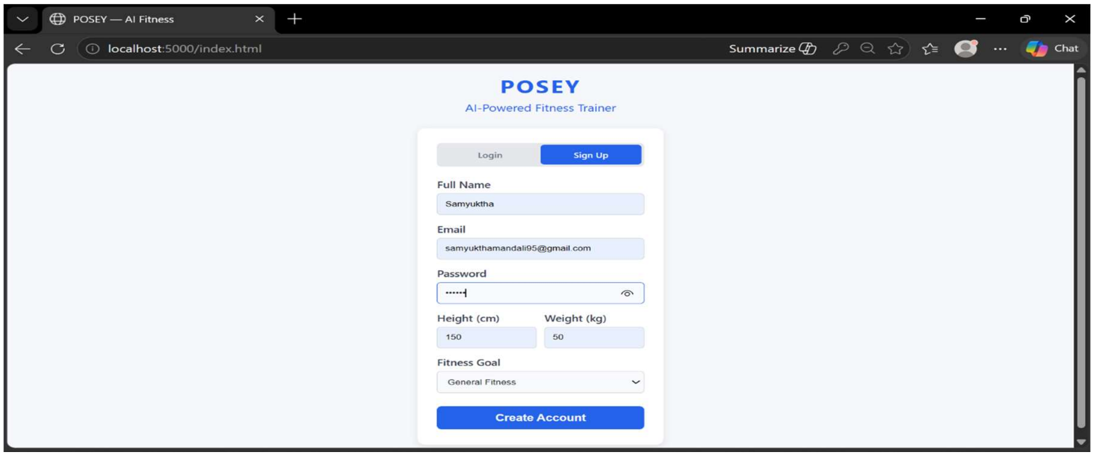
Types of Testing

Black-box testing is performed to evaluate system functionality without examining internal code. This approach validates inputs and outputs across different testing levels. White-box testing is conducted to analyze internal logic and code structure, particularly during unit testing. Techniques such as statement coverage, branch coverage, and path coverage are used to ensure thorough validation of program execution.

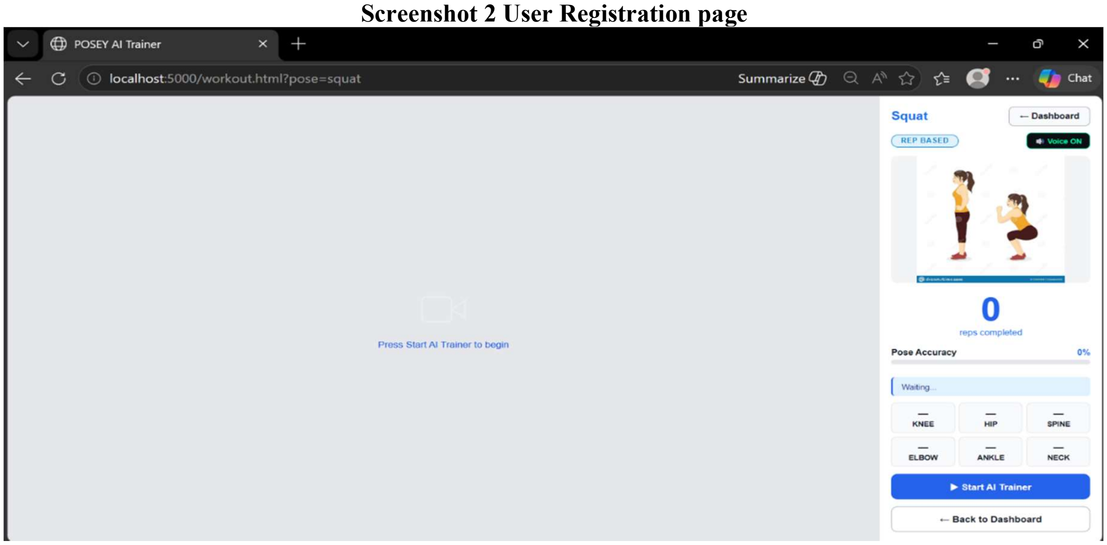
SCREENSHOTS



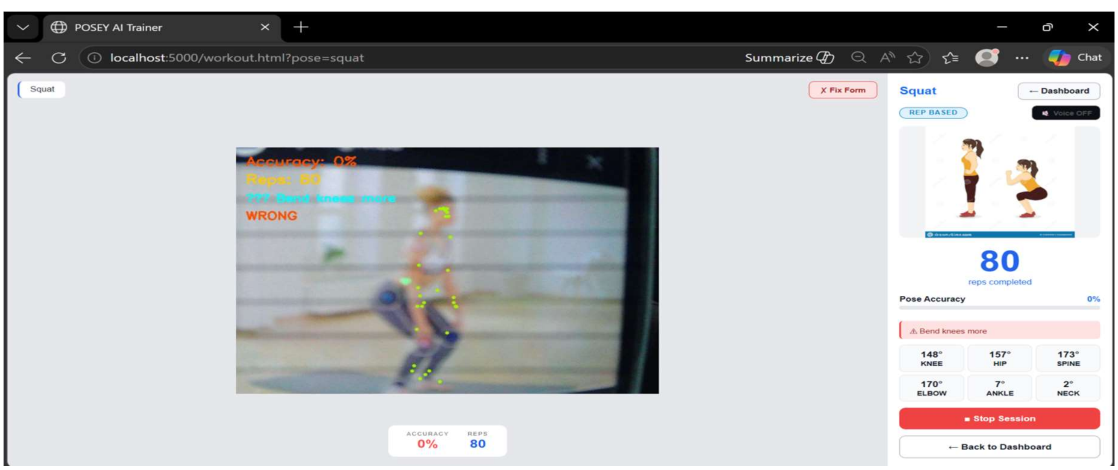
Screenshot 1 Posey Home Page Interface



Screenshot 2 User Registration page

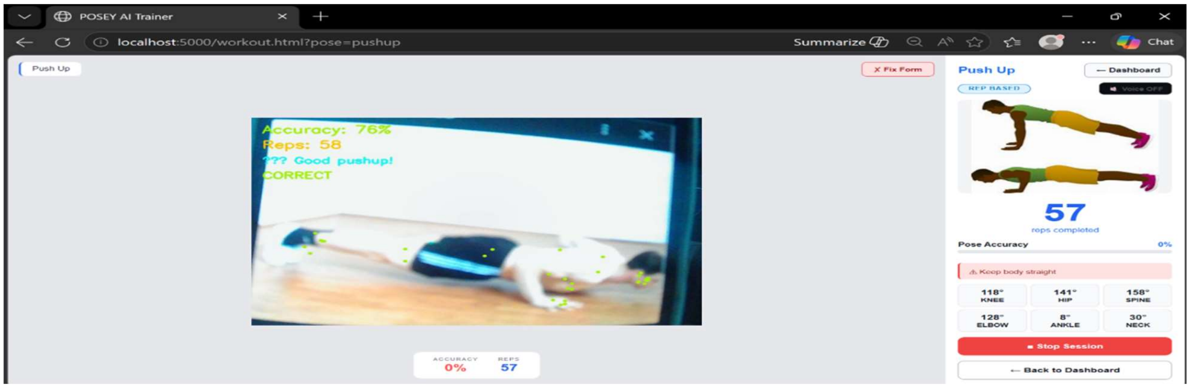


Screenshot 3 initial Exercise Session Screen



Screenshot 4 Incorrect Squat Detection with Feedback

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026



Screenshot 5 Correct Push-Up Detection with Feedback

Conclusion

Posey simplifies fitness training through an AI-powered digital platform. It allows users to perform at-home workouts with real-time posture detection, personalized guidance, and performance tracking. The system analyzes movements using video processing to ensure correct form and reduce the risk of injury. Users receive instant feedback and progress insights, improving consistency and motivation. By combining accessibility with professional-level training support, Posey enhances workout accuracy, safety, and overall fitness effectiveness.

Future Scope

Posey can be enhanced by incorporating advanced deep learning techniques for more accurate exercise detection and posture analysis. It can be developed into a mobile application to improve accessibility and user convenience. Additional features like personalized workout plans, real-time voice guidance, and integration with wearable devices can help track health metrics more effectively. The system can also include cloud-based progress monitoring for better data management and insights. Security and performance can be further improved, making Posey a more intelligent and complete virtual fitness assistant.

REFERENCES

1) Toshev, A., and Szegedy, C., “DeepPose: Human Pose Estimation via Deep Neural Networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1653–1660, 2024.
2) Shotton, J., Fitzgibbon, A., Cook, M., et al., “Real-Time Human Pose Recognition in Parts from Single Depth Images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1297–1304, 2023.
3) Mendez, M. E., Park, S. H., and Lee, J., “Human Pose Estimation and Activity Recognition Using Deep Learning: A Review,” *IEEE Access*, vol. 10, pp. 120456–120478, 2022.
4) Iqbal, M. I., and Gall, E., “PoseCoach: Real-Time Physical Exercise Feedback Using Pose Estimation and Rule-Based Evaluation,” in *Proceedings of the International Conference on Artificial Intelligence and Computer Engineering (ICAICE)*, pp. 320–325, 2021.
5) Singh, A., and Kumar, P., “AI-Powered Virtual Trainer for Posture Correction Using Computer Vision,” *International Journal of Advanced Research in Computer Science (IJARCS)*, vol. 12, no. 3, pp. 155–160, 2021.
6) Cao, Z., Hidalgo, G., Simon, T., Wei, S., and Sheikh, Y., “OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 43, no. 1, pp. 172–186, 2021.
7) Toshev, A., and Szegedy, C., “Deep Neural Network-Based Human Pose Estimation Techniques,” *IEEE CVPR*, pp. 1653–1660, 2024.