APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 7 No. 1 (2026) 1339-1355
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Athlete Training Management System (ATMS)

Syasya Farzana Mustafa[1], Suziyanti Marjudi[1]*

[1] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: suziyanti@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2026.07.01.072

Article Info

Received: 13 June 2025
Accepted: 18 June 2026
Available online: 30 June 2026

Keywords

Athlete Training Management System (ATMS), Centralized digital platform Training schedules, Web-based system, Agile methodology

Abstract

Centralized management is essential in upkeeping the efficiency of athlete management, current reliance on manual methods such as physical log books and spreadsheets at university sports centers leads to redundancy, inefficiency, and errors. This study aims to design, develop, and test a web-based Athlete Training Management System (ATMS) that centralizes training-related activities for athletes, coaches, and team managers. Agile methodology has been used to develop ATMS. The system utilizes object-oriented design principles and incorporates features like digital log books, automated reminders, and training schedules. Findings demonstrate that the ATMS significantly improves communication, reduces redundancy, and organize training management processes. The discussion highlights the potential of a centralized digital platforms to enhance efficiency in sports training and suggests further integration of advanced analytics to support data-driven decision-making in athlete development.

1. Introduction

The athlete training management system (ATMS) involves the systematic planning, monitoring, and organization of training activities for athletes to improve performance and track progress. Traditionally, this process is typically overseen by coaches, and team managers who handle various aspects of athlete development, such as workout schedules, performance data, and attendance. A systematic management is essential for ensuring that athletes follow structured training routines.

Currently, the management of athlete training activities at the university sport center is primarily done using manual methods. Coaches and team managers rely on physical log books and spreadsheets to record training schedules, attendance, and performance data. This data is often scattered across various documents, and not a centralized digital platform for organizing information. Individuals involved in this process include athletes, coaches, and team managers. While some teams may use basic software for attendance or scheduling, the overall process is still largely dependent on paper records and manual updates.

The current manual approach on managing the athletes training comes with several challenges. Physical log books and scattered documents are prone to errors, and information can be lost, redundant or misinterpreted. Retrieving and updating training data takes time, leading to disorganizations in communication between coaches and athletes. Furthermore, the lack of a central system makes it difficult to monitor athlete progress, track attendance over time, and ensure that athletes are following their training regimens consistently. These issues result in wasted resources, miscommunication, and less effective training programs overall.

The proposed athlete training management system (ATMS) should be able to address these issues by providing a central, digital platform for managing athlete training activities. The system will allow coaches to create and manage training schedules, record performance, and track athlete attendance in real-time. Athletes will have access to their personalized schedules and performance reports, allowing them to monitor their own progress. Digital systems driven by big data improve training performance and reduce inefficiencies in

traditional methods [1]. The system will also include reminders and notifications to ensure athletes are informed of upcoming training sessions. An integration of a digital systems in sports training enhances efficiency and addresses the limitations of manual methods [2]. By digitizing and centralizing the training management process, the athlete training management system will reduce redundancy, improve communication, and enhance the overall effectiveness of training programs.

This paper consists of five sections, with the first, Section 1, describing the background of the project. Section 2 provides the literature review of the project, offering insights into related works and concepts while Section 3 elaborates on the methodology used to develop the proposed system, including its analysis and design. The results from developing the proposed system are given in Section 4 and lastly, Section 5 concludes the project by summarizing key findings and suggesting potential improvements for future work.

2. Related Work

The case study examines athlete training management at Universiti Tun Hussein Onn Malaysia (UTHM), where data for training activities, attendance, and performance is currently handled manually using physical logbooks and basic spreadsheets. This obscure, manual process creates challenges in data organization and accessibility, making it difficult for coaches and managers to track athlete progress consistently and efficiently.

The proposed Athlete Training Management System (ATMS) will influence web-based Information Systems technology to centralize and streamline the management of training schedules, attendance, and performance data. This technology enables users to access and update information in real-time, ensuring that training activities and athlete progress are correctly recorded. This technology allows users to access and update information in real time, ensuring accurate recording of training activities and athlete progress [3].

A web-based approach is ideal for this system, as it offers flexibility, accessibility, and enhancing data security. In comparison to the traditional method, a web-based system can be accessed on many devices and from any location, allowing coaches and athletes to stay updated on training activities even while offsite. Additionally, a central data storage in a secure environment reduces the risk of data loss or duplication, supporting reliable performance tracking and communication [4]. A web-based approach makes it possible for users to access data from anywhere throughout various platforms.

A web-based approach involves creating a system or application that is accessible through a web browser, eliminating the need for users to install additional software on their devices [5]. These systems rely on technologies such as HTML, CSS, and JavaScript for frontend development, while backend frameworks or languages like PHP, Node.js, or Python handle server-side operations [6]. The data entered or managed within the system is stored in an integrated database, hosted securely on a server. A web-based approach also ensures that the security of the data could be well monitored. A key importance of a web-based approach would be ensuring that the data is not tampered by any individual as said web-based systems must prioritize addressing key vulnerabilities such as broken access control, injection attacks, and cryptographic failures, as outlined in OWASP's Top Ten security risks [7].

The Athlete Training Management System (ATMS) takes inspiration from existing platforms like TeamSnap, SportsEngine, and Hudl, which offer functionalities relevant to training management, such as scheduling, attendance tracking, and performance analysis. TeamSnap provides features like team scheduling, communication tools, and attendance tracking, facilitating efficient team coordination and improving athlete communication and training organization. SportsEngine focuses on registration, scheduling, and team communication tools, with its registration functionality adaptable for athlete profile management in ATMS, while its scheduling tools align with the system's needs. Hudl specializes in performance analysis through video analysis, data sharing, and feedback, offering valuable tools for tracking athlete progress that could be extended in ATMS to provide detailed analytics for both athletes and coaches. These features from existing systems can be adapted and enhanced in ATMS to address current challenges at Universiti Tun Hussein Onn Malaysia (UTHM).

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1339-1355

Table 1 *Comparison with Existing System*

System	TeamSnap	SportsEngine	Hudl	ATMS
Login	Yes	Yes	Yes	Yes
Profile Management	No	No	Yes	Yes
Attendance statistics	Yes	Yes	No	Yes
Session Scheduling	Yes	Yes	No	Yes
Performance Tracking	No	No	Yes	Yes
Notification Reminder	No	No	Yes	Yes

3. Methodology

Methodology is the foundation of a project. It is vital in project development as it shapes how the system is built, tested, and delivered. Selecting the appropriate software development methodology is crucial to the success of any system, as it guides how requirements are gathered, designs are created, and development is managed throughout the project.

There are several methodologies used in system development, such as the traditional System Development Life Cycle (SDLC), Waterfall, and Rapid Application Development (RAD). However, considering the involvement of multiple stakeholders which includes athletes, coaches, and team managers, and the potential for evolving requirements, the Agile methodology is the most suited approach for this project. Agile enables continuous feedback, iterative development, and flexibility, making it ideal for systems like the Athlete Training Management System (ATMS) where adaptability and real-time collaboration are essential.

3.1 Methodology Chosen and System Workflow

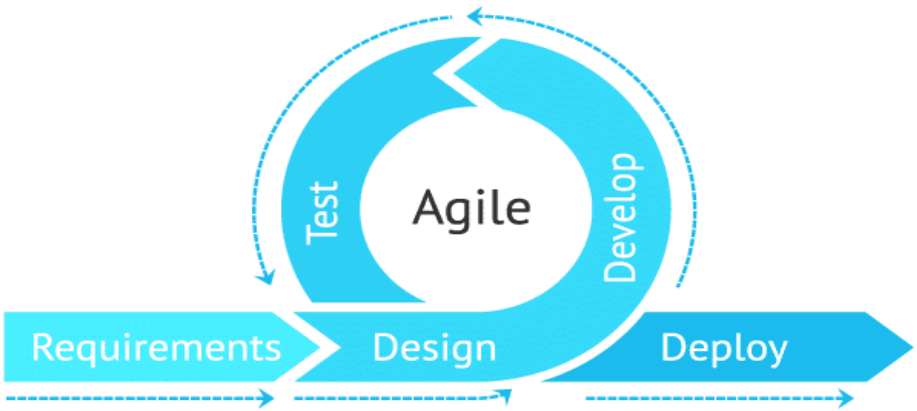


Fig 1. *Agile method cycle*

The Agile Model, as illustrated in Figure 1, is typically divided into five main phases: Requirements, design, development, testing, and deployment. Each phase involves specific tasks aimed at producing valuable outcomes for system development. Unlike a linear model, Agile adopts an iterative approach, where phases are revisited multiple times during the project. This iterative nature allows for continuous improvement of the system, incorporating feedback from previous cycles into each new iteration. As a result, the system can adapt and evolve to meet changing requirements and incorporate feedback effectively. Table 2 below outlines the overall workflow for the development of the proposed system.

Table 2 *Software development activities and their tasks*

Phase	Task	Output
Requirements	Propose the project, determine the schedule, activities, and outputs, collect requirements from stakeholders, analyze collected information, and review existing systems.	-Project proposal -Gantt chart -Comparison table with similar existing system -flowchart -Entity Relationship Diagram (ERD).
Design	Design the system architecture, database schema, data dictionary, and user interface.	-System architecture -Database schema -Data dictionary -User interface design.
Development	Develop and code the front-end and back-end of the system, link the system with the database.	-Fully developed system modules -Integrated system database.
Testing	Conduct unit testing, modular testing, debugging, and test the connection to the database for real-time data updates.	Tested system modules and functioning system.
Deployment	Deliver the completed system to end-users and provide ongoing maintenance through updates as needed.	Fully developed and maintained system ready for use.

4. System Analysis and Design

This section explores systematic analysis and design of the Athlete Training Management System (ASTMS). The objective is to define system requirements and create a structure design.

4.1 System Requirements

System Requirement Analysis is the process of identifying, documenting, and analyzing the needs and expectations of stakeholders to ensure that the system being developed aligns with its intended purpose. This involves gathering functional requirements, which describe what the system should do, and non-functional requirements, which focus on qualities like performance, reliability, security, and usability. The Table 3 and table 4 shows the system's functional and non-functional requirements.

Table 3 *Functional Requirement for ATMS*

Module	Function
Registration and Login	• The system must allow users to log in using a valid username and password. • Role-based access must be implemented for Athletes, Coaches, Team Managers, and Admins. • Successful login should redirect users to their respective dashboards.
Admin Management	• Admins must be able to create, update, and delete user accounts. • Admins should have the ability to assign or modify user roles and permissions.

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1339-1355

Table 3: (cont)

Module	Function
Coach Management	• Create, update, and manage training schedules.
	• Input record and review athlete attendance.
	• Evaluate athlete performance and provide feedback.
	• Send notifications to athletes.
Team Manager Management	• View upcoming training schedules and attendance records.
	• Export attendance and performance reports.
	• Send or receive notifications regarding athlete activities.
Athlete Management	• Allow athletes to create/update personal profiles and training goals.
	• View training schedules and performance reports.
	• View notification that is assigned
Attendance Management	• Automatically record attendance during training.
	• Input attendance logging (by coach).
	• Allow access to attendance records by coaches and team managers.
Notification	• Send automated reminders to athletes.
	• Notify users of upcoming training, feedback, or important updates.
Report Generation	• Generate and export attendance/performance reports.
	• Reports accessible by coaches and team managers.

Table 4 *Non-Functional requirements*

No	Requirement	Description
1	Performance	The system should be always usable
2	Operational	The loading time required for a website is no more than 2 minutes
3	Security	The system should securely store sensitive data such as login credentials and performance records.
4	Cross-Browser Compatibility	The system should be able to work on any web browser
5	Scalability	The system must support an increasing number of users without performance degradation.
6	Maintainability	The system should allow easy updates and maintenance with minimal disruptions.

4.2 System Analysis

After identifying and defining the system's requirements, the system architecture is analyzed through three key modeling approaches that help visualize its structure, processes, and data flow. A Use Case Diagram outlines how each user role (admin, coach, team manager, athlete) interacts with the system. It helps clarify the system's functionality from the user's perspective. Lastly, to reflect the object-oriented nature of the system, a UML Class

Diagram was constructed to model the structure of the system in terms of classes, their attributes, and relationships. This diagram supports a clear understanding of the system's architecture and promotes modularity and reusability.

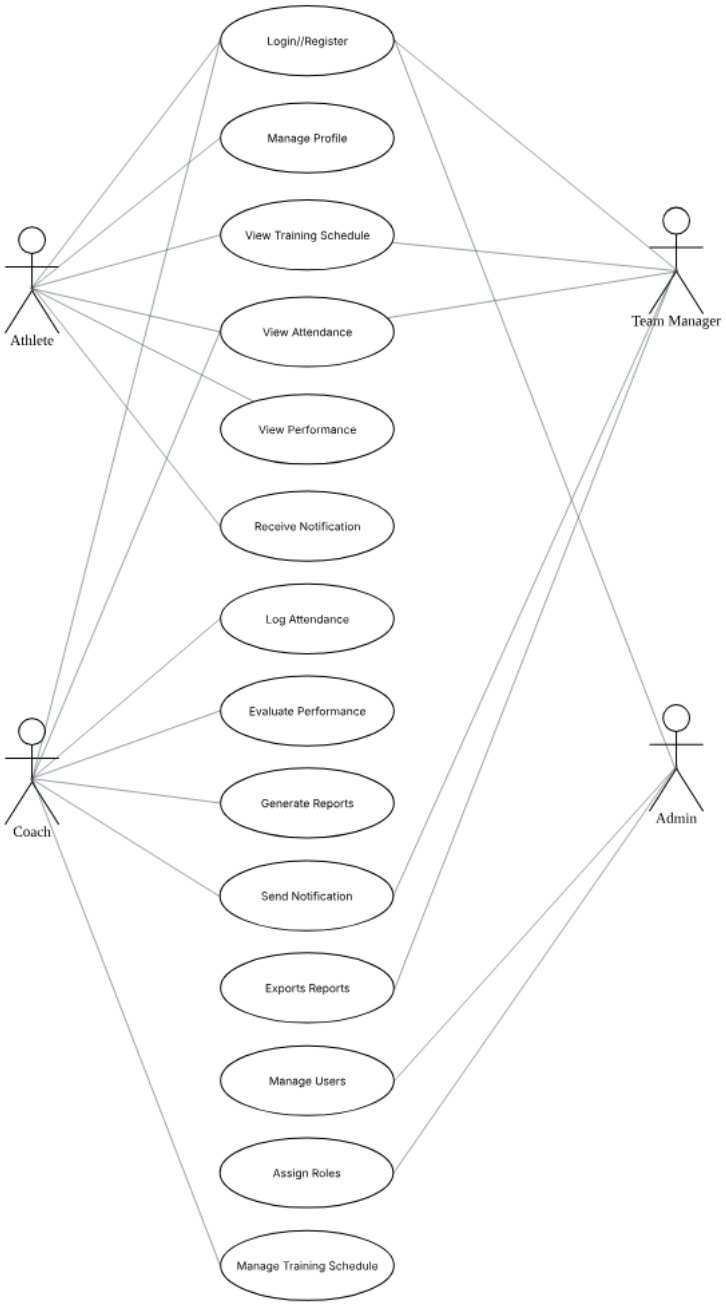


Fig 2. *Use Case Diagram for ATMS System*

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1339-1355

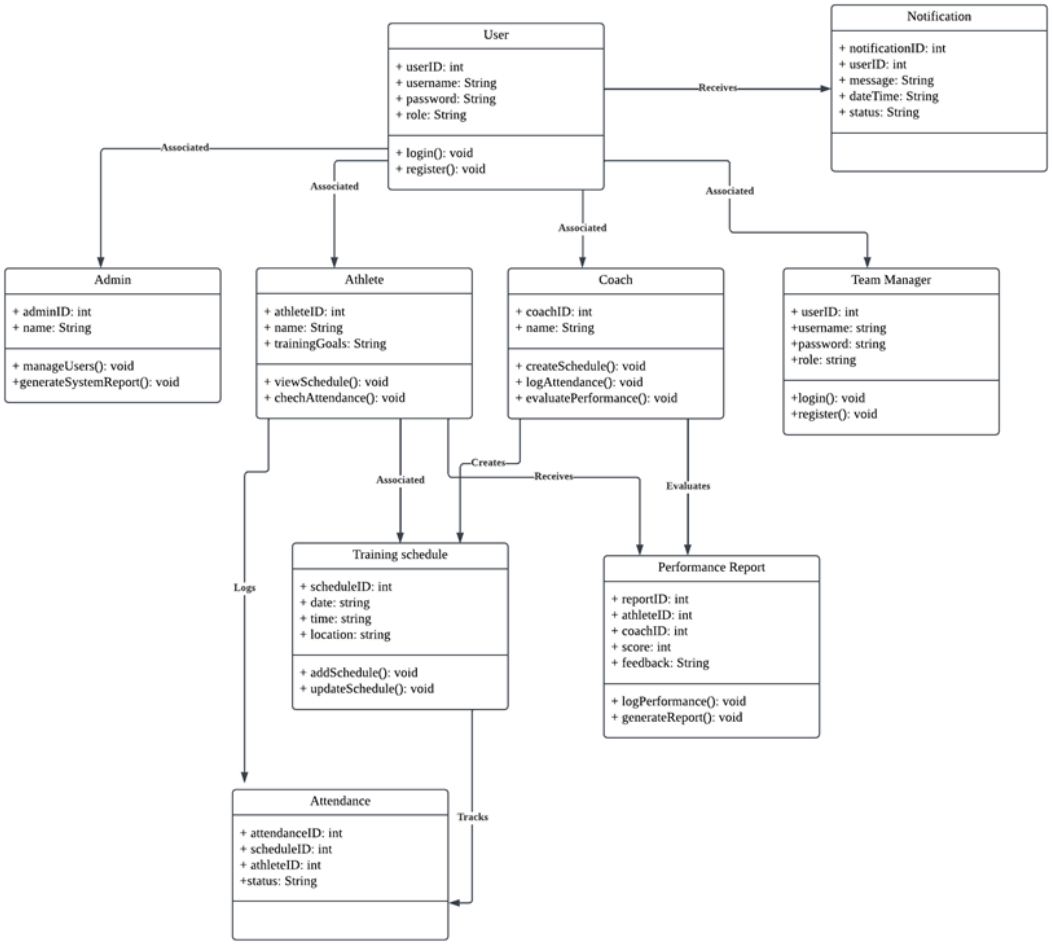


Fig. 3 *UML Class Diagram for ATMS*

Together, these diagrams provide a comprehensive understanding of the system's processes, user interactions, and architectural structure. They not only illustrate how data flows within the system and how various user roles engage with its functionalities, but also offer a clear representation of how components are organized in an object-oriented environment. By bridging both functional and structural views, these models serve as a foundation for effective system development, ensuring that the design aligns with both user expectations and technical requirements.

4.3 System Design

After analyzing the users' requirements, the project progresses into the design phase. This phase focuses on creating the system architecture to visualize the structure and workflow of the system, ensuring that all components interact seamlessly. The system architecture defines how different modules, databases, and interfaces communicate with each other, which is critical for system implementation. In software development, meticulous attention to both user interface (UI) and database design is paramount. A well-crafted UI acts as the intermediary between the user and the system, enabling efficient and intuitive interactions. Research underscores that user-centered design produces applications that are not only effective but also engaging, thereby enhancing user satisfaction [8]. On that subject matter, a properly structured databases prevent anomalies and redundancies, helping maintain data quality over time. With the increasing complexity of software and data, automated database design processes are essential to ensure scalability and efficient management [9].

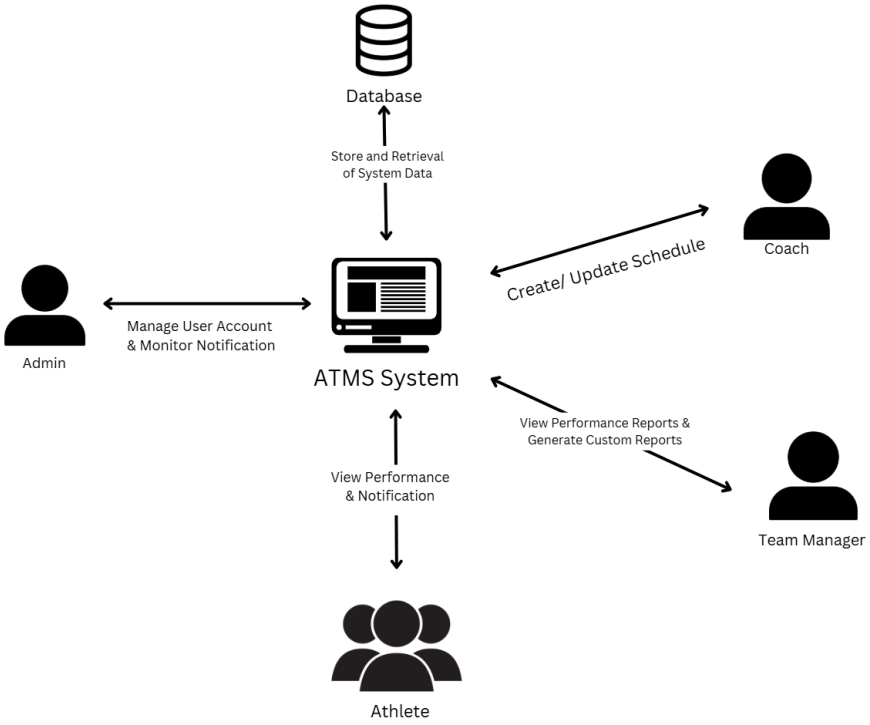


Fig 4. *System architecture for ATMS*

In the diagram above, the Athlete Training Management System (ATMS) is designed to accommodate multiple types of users: administrators, coaches, team managers, and athletes. Administrators have high-level access for managing user accounts, monitoring notifications, and overseeing the overall system. Coaches are responsible for creating and updating training schedules, which are stored in the centralized database. Team managers interact with the system to view performance reports and generate custom evaluations for athletes, ensuring streamlined monitoring and analysis. Athletes use the system to view their training schedules, performance updates, and receive notifications regarding their progress and upcoming sessions.

The ATMS serves as the central hub, processing user requests and communicating with the database to store or retrieve crucial data, such as training schedules, attendance records, and performance reports. This architecture ensures efficient data flow between the system and its users, enabling accurate, real-time updates and facilitating effective training management for all stakeholders. The centralized design improves collaboration, reduces redundancy, and enhances the overall efficiency of athlete training operations.
The relational schemas for the database are listed as follows:

I. Tbl_user (userID, username, password, role)
II. Tbl_athlete (athleteID, userID, name, trainingGoals)
III. Tbl_coach (coachID, userID, name)
IV. Tbl_team_manager (managerID, userID, name)
V. Tbl_admin (adminID, userID, name)
VI. Tbl_training_schedule (scheduleID, coachID, date, time, location)
VII. Tbl_attendance (attendanceID, scheduleID, athleteID, status)
VIII. Tbl_performance_report (reportID, athleteID, coachID, score, feedback, evaluationDate)
IX. Tbl_notification (notificationID, userID, message, dateTime, status)

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1339-1355

The ATMS interfaces are designed with simplicity and functionality in mind, ensuring that users can easily navigate and interact with the system. While the layout prioritizes clarity and ease of use over aesthetic elements, it effectively supports workflow efficiency and ensures accessibility across different devices. An example of the interface designs is shown below.

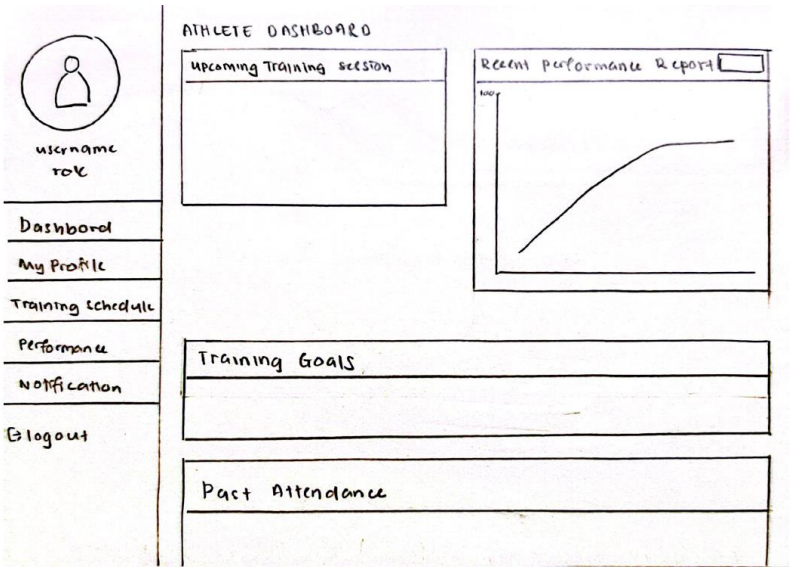


Fig. 5 *Sketch of ATMS Dashboard*

5. Implementation

This section discusses the implementation of ATMS. The implementation of log in modules, user modules, report module and notification module, would be described here.

5.1 Implementation for Log in/Registration

```php
// Handle Login
if (isset($_POST['login'])) {
    $username = $_POST["username"];
    $password = $_POST["password"];

    $stmt = $conn->prepare("SELECT id, password, role FROM users WHERE username = ?");
    $stmt->bind_param("s", $username);
    $stmt->execute();
    $stmt->store_result();
    $stmt->bind_result($id, $hashed_password, $role);
    $stmt->fetch();

    if ($stmt->num_rows > 0 && password_verify($password, $hashed_password)) {
        $_SESSION["user_id"] = $id;
        $_SESSION["username"] = $username;
        $_SESSION["role"] = $role;

        switch ($role) {}
        exit();
    } else {
        $login_error = "Invalid username or password";
    }
    $stmt->close();
}
```

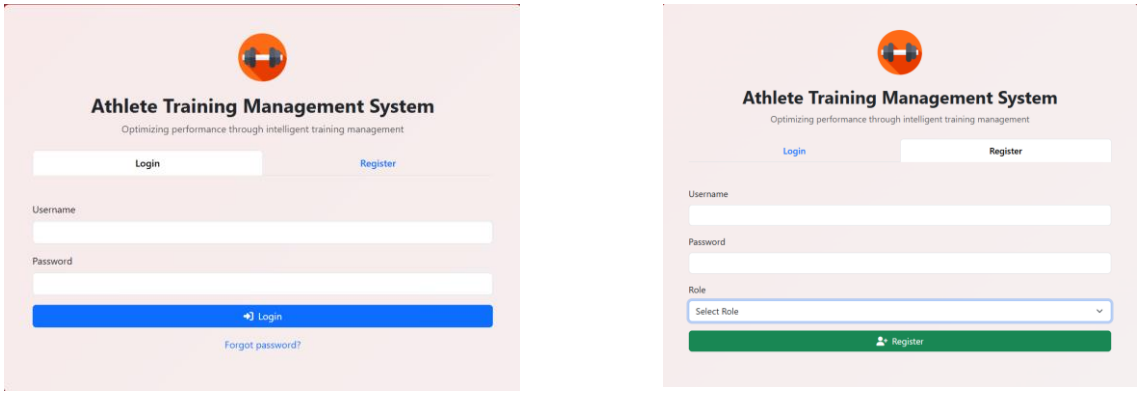
(a)

```php
// Handle Registration
if (isset($_POST['register'])) {
    $username = $_POST["reg_username"];
    $password = password_hash($_POST["reg_password"], PASSWORD_BCRYPT);
    $role = $_POST["reg_role"];

    $stmt = $conn->prepare("INSERT INTO users (username, password, role) VALUES (?, ?, ?)");
    $stmt->bind_param("sss", $username, $password, $role);

    if ($stmt->execute()) {

        // insert to other tables based on role
        $user_id = $stmt->insert_id;

        switch ($role) {}

        $registration_success = "Registration Successful! You can now login.";
    } else {
        $registration_error = "Error registering user. Username may already exist.";
    }
    $stmt->close();
}
?>
```

(b)

Fig. 6 Login Page Coding (a) Registration Page Coding (b)

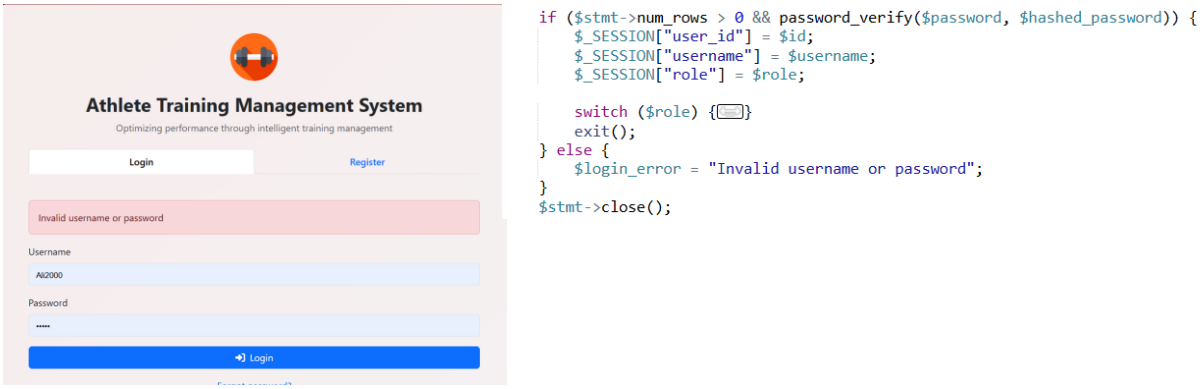
In figure 6 the system implements secure login and registration with PHP backend and Bootstrap frontend. The login code in Figure 6(a) uses prepared statements and password_verify() for secure authentication, redirecting users to role-specific dashboards. The registration code in Figure 6(b) features bcrypt password hashing and prepared statements, with form validation for username, password, and role selection. Both components employ session management and client-side validation while preventing SQL injection, following security best practices throughout the authentication workflow.



(a) **(b)**

Fig. 7 *Login Page (a) Registration Page (b)*

Figure 7(a) shows the user interface for the login page, where users must input their username and password to gain access to their respective dashboards. Figure 7(b) displays the registration page, requiring users to enter a username, password, and select their role (Athlete, Coach, Team Manager, or Admin) before submitting their details to create an account. Both interfaces are designed for simplicity and ease of use, ensuring a smooth authentication process.



(a) **(b)**

Fig. 8 *Error message in login page (a) Error message coding (b)*

Figure 8(a) shows the user interface's error response when invalid credentials are submitted, displaying "Invalid username or password" in a Bootstrap alert component for immediate user feedback. Figure 8(b) presents the secure PHP backend implementation that validates credentials against database records using prepared statements, triggering this error message when authentication fails.

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1339-1355

5.2 Implementation for Admin Management Module

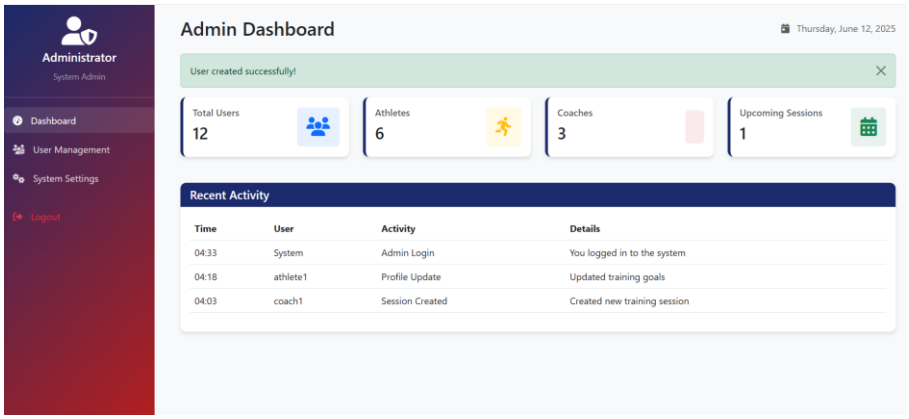


Fig. 9 *Admin Dashboard*

Figure 9 shows the admin dashboard overview. It displays the recent activity that has happened and what the system has logged. It displays the total user in the system, number of athletes, coaches and upcoming session that might take place for the team. From the dashboard, admin can also access the user management and the system settings.

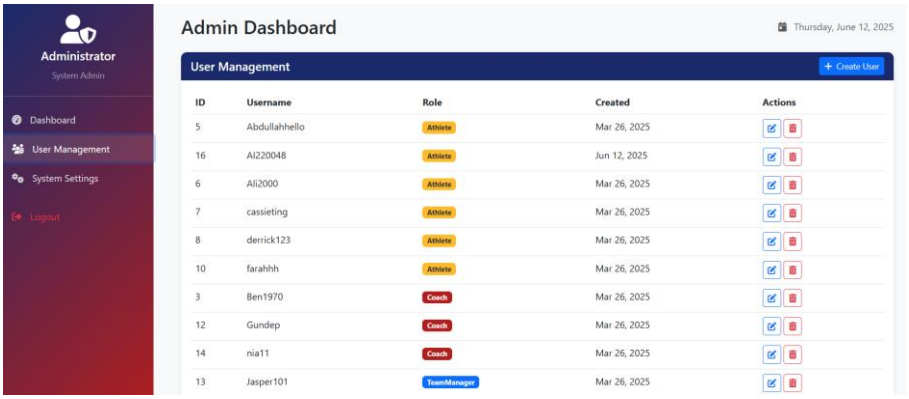


Fig. 10 *User management in Admin Dashboard*

Figure 10 shows the user management page that can be accesed by the admin. Admins are able to create, update user role, and delete user. The username, role, date created and actions is also displayed. For action, the admin can delete or edit the role for the user.

5.3 Implementation for Coach Management Module

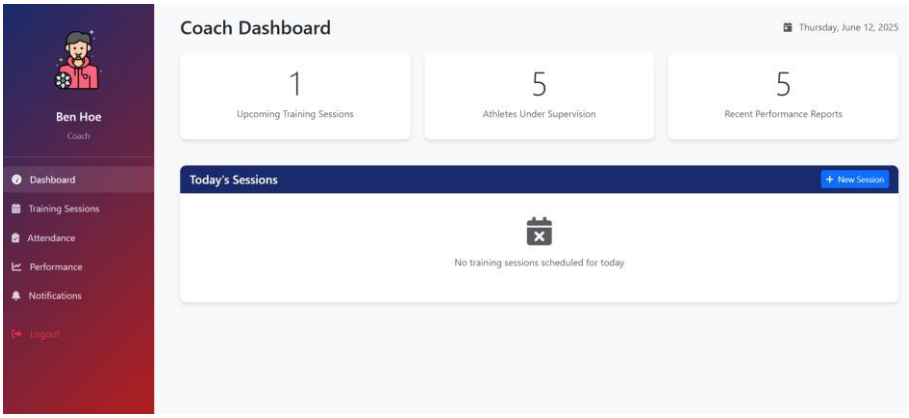
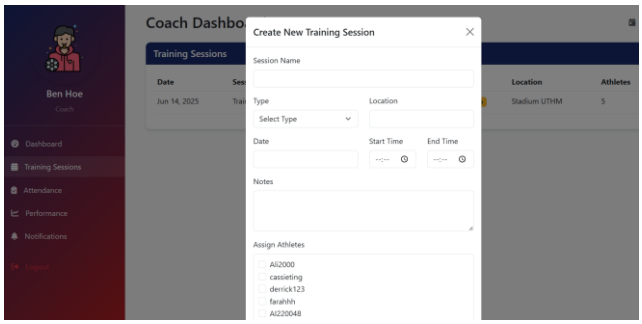


Fig. 11 *Coach Dashboard*

Figure 11 shows the coach dashboard where coaches can create new training session, access training session, access attendance, input performance and view and send notification. The dashboard display upcoming training session, number of athletes under supervision, and recent performance report.

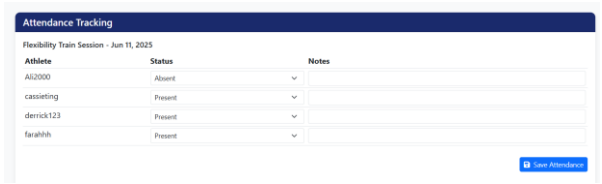



```php
// Handle training session creation
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['create_session'])) {
    $session_name = $_POST['session_name'];
    $session_type = $_POST['session_type'];
    $session_date = $_POST['session_date'];

    date_default_timezone_set('Asia/Kuala_Lumpur');
    $today = new DateTime();
    $selected_date = DateTime::createFromFormat('Y-m-d', $session_date);

    if ($selected_date < $today->setTime(0, 0)) {
        $error_msg = "You cannot create a training session for a past date.";
    } elseif ($selected_date->format('Y-m-d') === $today->format('Y-m-d')) {
        // Same day – validate time
        $current_time = new DateTime();
        $start_time_obj = DateTime::createFromFormat('H:i', $_POST['start_time']);

        if ($start_time_obj < $current_time) {
            $error_msg = "Start time cannot be in the past.";
        }
    }
```

(a) (b)

Fig. 12 *Create New Training Session page (a) code for Create Training Session (b)*

Figure 12(a) shows the interface for the creating training session. Figure 12(b) The code handles the creation of a new training session by capturing details such as the session name, type, date, time, location, and notes from a POST request. It inserts these into the training_sessions table along with the coach's ID. If successful, it then assigns selected athletes to the session by adding entries to the athlete_training_sessions table and sends each athlete a notification about their new session.




```php
// Handle attendance update
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['update_attendance'])) {
    $session_id = $_POST['session_id'];
    $attendance_data = $_POST['attendance'];

    foreach ($attendance_data as $assignment_id => $status) {
        $stmt = $conn->prepare("UPDATE athlete_training_sessions SET attendance_status = ? WHERE id = ?");
        $stmt->bind_param("si", $status, $assignment_id);
        $stmt->execute();
        $stmt->close();
    }

    $success_msg = "Attendance updated successfully!";
}
```

(a) (b)

Fig .13 *Attendance Tracking (a) Attendance Tracking code (b)*

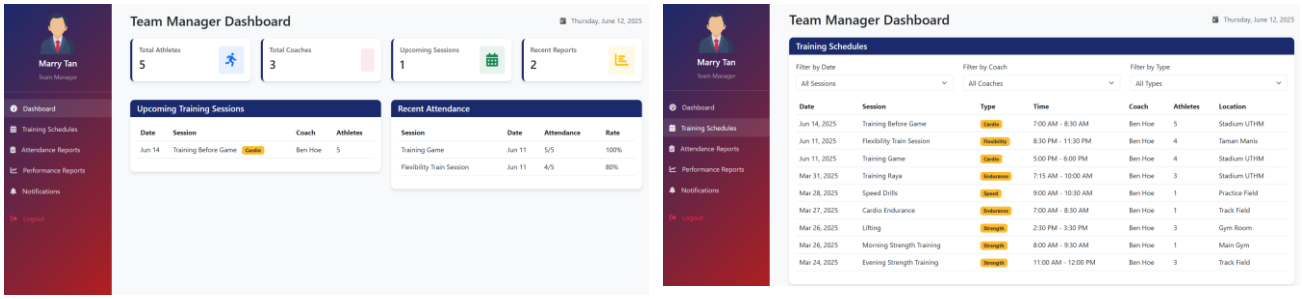
Figure13(a) displays a list of athletes for a session with checkboxes or dropdowns to mark presence (e.g., "Present," "Absent," "Late"). Figure13(b) The code processes attendance updates by looping through submitted attendance_data and updates the athlete_training_sessions table accordingly. A success message confirms the changes.

```php
// Handle performance report creation
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['create_report'])) {
    $athlete_id = $_POST['athlete_id'];
    $report_date = $_POST['report_date'];
    //$overall_score = $_POST['overall_score'];
    $strength_score = $_POST['strength_score'];
    $endurance_score = $_POST['endurance_score'];
    $speed_score = $_POST['speed_score'];
    $flexibility_score = $_POST['flexibility_score'];
    $recovery_score = $_POST['recovery_score'];
    $comments = $_POST['comments'];

    //Auto-calculate overall score
    $overall_score = ($strength_score + $endurance_score + $speed_score + $flexibility_score + $recovery_score) / 5;

    $stmt = $conn->prepare("INSERT INTO performance_reports (athlete_id, coach_id, report_date, overall_score, strength_score, endurance_score, speed_score, flexibility_score, recovery_score,
        comments)
                            VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)");
    $stmt->bind_param("iisddddddds", $athlete_id, $coach['id'], $report_date, $overall_score, $strength_score, $endurance_score, $speed_score, $flexibility_score, $recovery_score, $comments);

    if ($stmt->execute()) {
        $report_id = $stmt->insert_id;

        // Create notification
        $message = "You have a new performance report available. Overall score: " . $overall_score . "%";
        $stmt2 = $conn->prepare("INSERT INTO notifications (recipient_id, sender_id, title, message, notification_type, related_id)
                                 VALUES ((SELECT user_id FROM athletes WHERE id = ?), ?, 'Performance Report', ?, 'Performance', ?)");
        $stmt2->bind_param("iiisi", $athlete_id, $coach_id, $message, $report_id);
        $stmt2->execute();
        $stmt2->close();

        $success_msg = "Performance report created successfully!";
```

Fig. 14 *Code for Performance Report*

Figure 14 shows the code for performance reporting. It handles the submission of a report via a POST request by retrieving performance metrics from the form, calculating an overall score based on five criteria (strength, endurance, speed, flexibility, and recovery), and inserting the data into the performance_reports database table using a prepared statement. After the report is saved, a notification is generated and sent to the corresponding athlete to inform them of the newly available performance report.

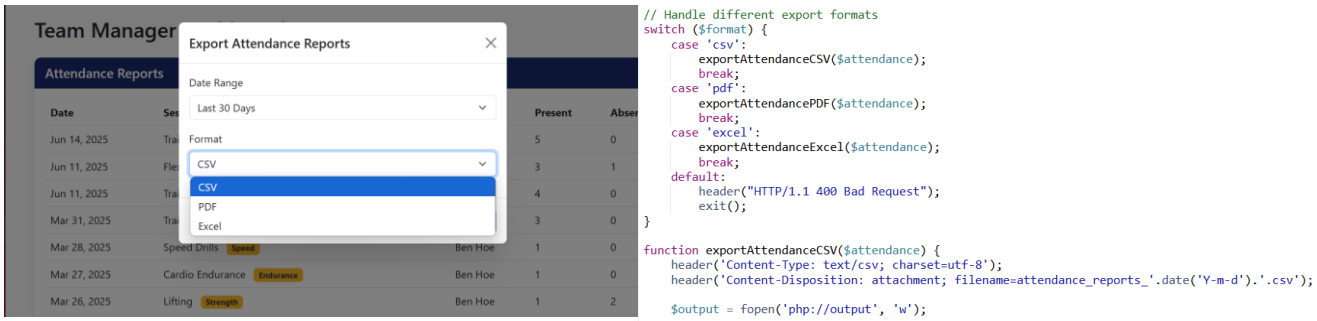
5.4 Implementation for Team Manager Management Module



(a) **(b)**

Fig. 15 *Team Manager Dashboard (a) View of Traning Schedules (b)*

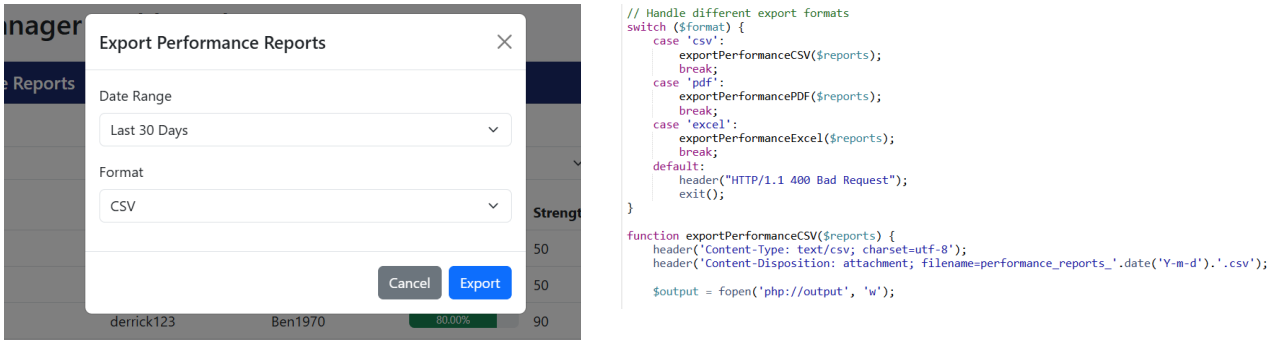
Figure 15(a) shows the team manager dashboard they can see total athletes, total coaches, upcoming sessions and recent report. From the dashboard they can acccess training schedules,attendance report, performance reportand notification. Figure 15(b) shows where the team manager can view the training schedules that has been created by the coaches.



(a) **(b)**

Fig. 16 *Export attendance report interface (a) code for export attendance report (b)*

Figure16(a) shows the interface of attendance report while in figure 16(b) shows the switch statement that exports training session data (dates, attendance counts) in CSV, PDF, or Excel. It calls functions like exportAttendanceCSV() to format the data accordingly. If an invalid format is requested, it returns a 400 Bad Request error.



(a) **(b)**

Fig. 17 *Export performance report interface (a) code for export performance report (b)*

The switch statement in Figure 17 (Performance Report) exports athlete performance data (scores, comments) in CSV, PDF, or Excel using functions like exportPerformanceCSV(). Invalid requests get a 400 error, matching the attendance report's error handling. Both follow the same structure but handle different data.

5.5 Implementation for Athlete Dashboard Module

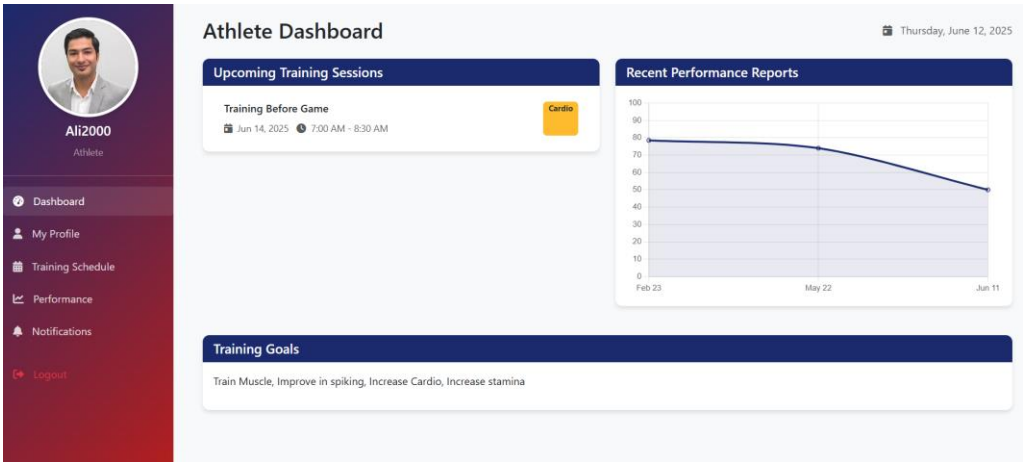
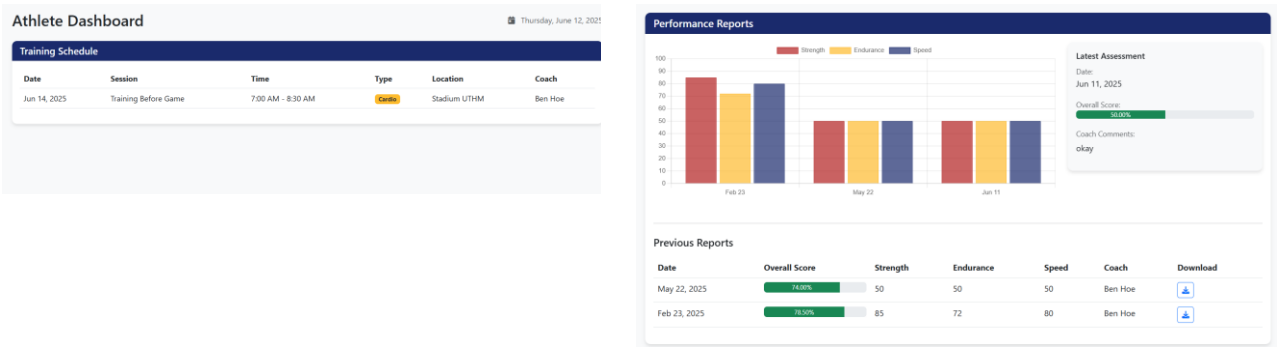


Fig. 18 *Athlete Dashboard*

Figure 18 shows the athlete dashboard where are able to view upcoming training session, recent performance report and their training goals. They can access their profile, view training schedules, view performance and view notification.



(a) (b)

Fig. 19 *Training Schedules that have been set for the athlete (a) View Performance Report (b)*

Figure 19(a) shows the training schedule interface that can be viewed by the athletes. They are able to see session name, time, type of training, location and the coach that assingn the session. In figure19(b) shows the interface of performance report that they can download according to the report date.

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1339-1355

5.6 Report module

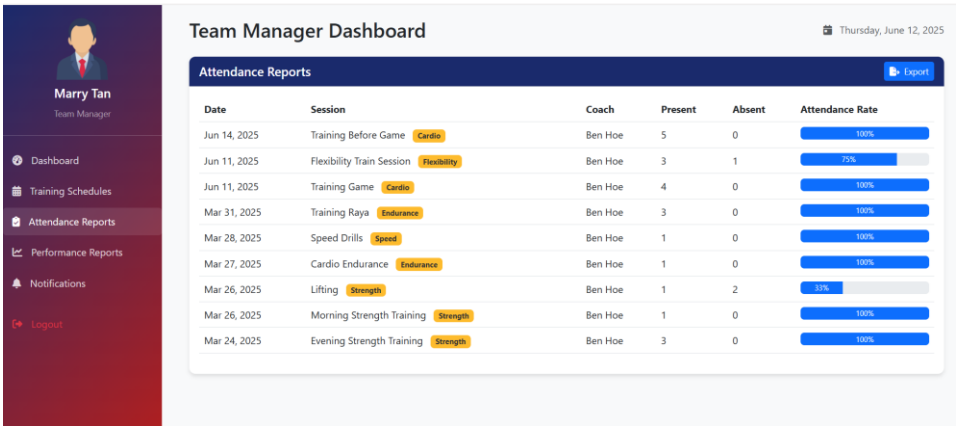
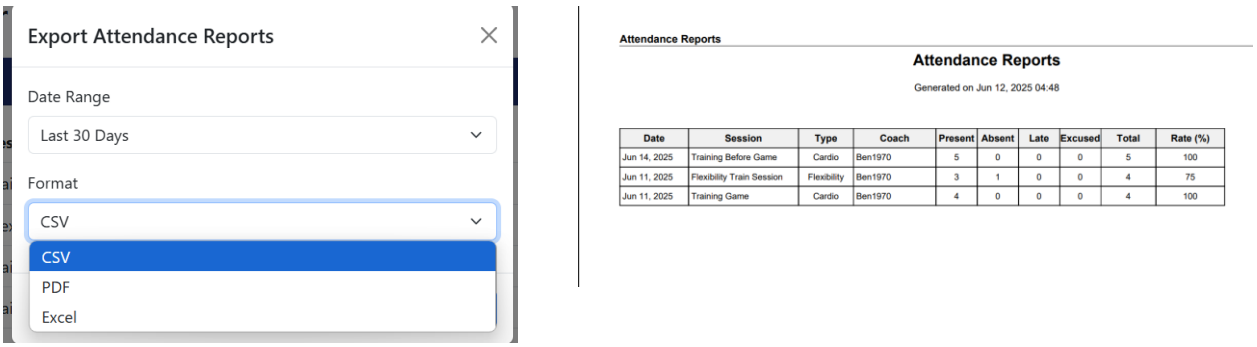


Fig. 20 *Attendance report in Team Manager Dashboard*

Figure 20 shows the interface of the team manager dashboard where they can export files which are the attendance report and performance report. They are first greeted with a list of the report. Figure 21 shows the list of the total attendance based on previous sessions.



(a) (b)

Fig. 21 *Choosing format for Attendance Report(b) Example of Attendance Report*

Figure 21 shows the interface of the attendance report page. User can select what type of format of report do they want. Figure21(b) shows an example of the report that has been generated in pdf format.

6. Discussion

This section discusses the testing results of the proposed system. Two types of testing results are presented: the test plan result and the user acceptance result. The testing phase encompassed the entire system verify its security and adherence to project requirements.

6.1 Test plan result

A test plan outlines the strategy used to verify that a system functions according to its requirements. For this project, the ATMS was tested across various modules, including user registration, login, profile management, attendance tracking, and reporting. Testing ensures that all features work correctly and meet user expectations [10]. The test pan result is displayed as below.

Table 5 *Test Plan Result*

Module	Test Case Description	Expected Result	Result
Login and Registration	• Fill all required fields and submit	• Account created, redirect to login page	Pass
	• Input valid ID and password	• Redirect to correct dashboard	Pass
	• Input invalid credentials	• Show error: "invalid username or password"	Pass
Admin Management	• Create new user	• User account created successfully	Pass
	• Delete existing user	• User removed from database	Pass
	• Edit user role	• Role updated correctly	Pass
	• View notification	• Display notification received	Pass
Coach Dashboard	• Create new training session	• Session added to schedule	Pass
	• Edit existing training session	• Updated session details saved	Pass
	• Mark/edit athlete attendance	• Attendance updated correctly	Pass
	• Input athlete performance report	• Performance data stored and visible to athlete	Pass
	• Send notification	• Athlete receives notification	Pass
Team Manager Dashboard	• View upcoming training sessions	• Display list of scheduled sessions	Pass
	• View recent attendance	• Attendance summary displayed	Pass
	• View training schedules	• Full list shown clearly	Pass
	• Export attendance report	• File downloaded/exported successfully	Pass
	• Export performance report	• File downloaded/exported successfully	Pass
	• Send notification	• Notification sent to target users	Pass
Athlete Dashboard	• Update profile picture and save	• Profile is saved	Pass
	• View Training Schedule	• Display current and upcoming sessions	Pass
	• View attendance statistics	• System displays attendance summary	Pass
	• View latest performance report	• Display report from coach	Pass
Report	• Choose format for report	• chosen format exported	Pass
	• Select timeframe for report	• selected timeframe for report exported	Pass
	• View attendance statistics	• System displays attendance summary	Pass
	• View latest performance report	• Display report from coach	Pass

7. Conclusion

In conclusion, the Athlete Training Management System (ATMS) presents a comprehensive digital solution to the inefficiencies inherent in manual training management, addressing key challenges such as data redundancy, miscommunication, and administrative overhead. By centralizing training schedules, attendance tracking, and performance analytics, ATMS significantly improves operational efficiency and enhances collaboration among stakeholders.

While the system offers substantial benefits, its effectiveness depends on user proficiency, highlighting the need for targeted onboarding, especially for non-technical users. Additionally, the migration of legacy data into the new system may introduce transitional challenges. Future enhancements could include predictive analytics to support performance optimization, as well as mobile-native interfaces to improve accessibility and user engagement.

Beyond its immediate application, ATMS reflects the growing importance of centralized, data-driven systems in sports management. It serves as a potential benchmark for institutions aiming to modernize athlete development workflows and demonstrates the broader value of digital transformation in performance-centric environments.

Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support.

Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *Syasya Farzana Mustafa, Suziyanti Marjudi;* ***data collection:*** *Syasya Farzana Mustafa, Suziyanti Marjudi;* ***analysis and interpretation of results:*** *Syasya Farzana Mustafa, Suziyanti Marjudi;* ***draft manuscript preparation:*** *Syasya Farzana Mustafa. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] J Feng, "Designing an artificial intelligence-based sport management system using big data," *Soft Computing*, 2023. Available: https://doi.org/10.1007/s00500-023-09162-0.
[2] T. Wang and J. Park, "Design and implementation of intelligent sports training system for college students' mental health education," *Frontiers in Psychology*, vol. 12, p. 634978, 2021. Available: https://doi.org/10.3389/fpsyg.2021.634978.
[3] K. C. Laudon and J. P. Laudon, *Management Information Systems: Managing the Digital Firm*, 16th ed. Pearson, 2020.
[4] R. M. Stair and G. W. Reynolds, *Fundamentals of Information Systems*, 10th ed. Cengage Learning, 2021.
[5] T. Felke-Morris, *Web Development and Design Foundations with HTML5*, 9th ed. Pearson, 2018.
[6] J. Robbins, *Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics*, 5th ed. O'Reilly Media, 2018.
[7] OWASP Foundation, "OWASP Top Ten Web Application Security Risks," 2021. [Online]. Available: https://owasp.org/www-project-top-ten/. [Accessed: Nov. 1, 2024].
[8] S. Han, "User-centered design principles for enhancing software usability," *IEEE Explore*, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8285972.
[9] J. Smith and X. Zhang, "Automated database design for scalable systems," *IEEE Transactions on Software Engineering*, 2020.
[10] I. Sommerville, *Software Engineering*, 10th ed. Boston, MA, USA: Pearson, 2020.