

Wembley Stadium Management



**Introduction**

The Wembley Stadium Management App is designed to simplify and enhance the experience of managing large-scale events, from concerts to football matches. It provides an easy-to-use platform for fans to purchase tickets, receive event updates, and access their seats seamlessly using digital passes.

For stadium administrators and staff, the app helps streamline operations by managing event schedules, validating tickets with QR scanning, coordinating security access, and tracking crowd movement in real time. It also integrates secure payment processing, user authentication, and push notifications to keep everything running smoothly.

This project focuses on building a scalable, efficient, and secure system that can handle high-traffic events while providing a seamless user experience. It’s a great opportunity to work with database management, API integration, real-time data processing, and mobile-first design—key elements in modern software development.

**Purpose:**

the main goal of this Wembley stadium management project is to design, build and apply a functioning , with strong security and efficient software solution that is suitable to replicate the same operations of the current Wembley stadium .The project highlight the complication of the Wembley stadium environments , it achieves that, by automating routing tasks, intensify data, it improves decision-making steps, and it making sure all the communication across the stadium work smoothly. By using the features of C# , a versatile and object oriented programming language, the project can do a high-performance system that will integrates seamlessly with the current technologies standards and it aligns to the industry qualities.

stakeholders to make informed decisions based on accurate and up-to-date information.

**Development Process Overview**

Initially, the team had two different backend implementations developed independently by Iliya Hajimohammadi and Kassra Niroumand. Each of these had its own logic and structure for the project, as both members started working separately to experiment with different architectural approaches.

By **week 9**, during a team meeting, all group members came together and agreed to consolidate the work into a single version to enhance collaboration and avoid redundancy. After reviewing both implementations, the team chose one unified backend as the foundation for continued development. From that point onward, the team began contributing to the same codebase to ensure consistency and smoother progress.

All collaborative code have since been maintained on GitHub repository.

This is the other backend code contributed by iliya Hajimohammadi :

🔗 <https://github.com/iliyaDot/MyEshop.git>

**Testing Section**

1. **Testing Approach**

The testing strategy focuses on validating REST API functionalities across major workflows:

* Unit Tests for controller logic and service classes.
* Integration Tests to simulate real user interactions with authentication and booking flows.
* Manual Testing for edge cases like invalid tickets or expired sessions.

A combination of Postman scripts and automated test runners ensures endpoints behave reliably under different scenarios. Status codes, response payloads, and authorization flows are all validated.

1. **Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Input** | **Expected Output** | **Actual Output** | **Status (Pass/Fail)** |
| TC-001 | User registration | Name, email, password | Account successfully created | Account successfully created | Pass |
| TC-002 | User login | Valid email & password | Successful login | Successful login | Pass |
| TC-003 | Incorrect login attempt | Invalid email/password | Error message displayed | Error message displayed | Pass |
| TC-004 | Ticket purchase | Select event, seat, pay | Ticket successfully booked | Ticket successfully booked | Pass |
| TC-005 | Ticket QR code validation | Scan valid QR code | Access granted | Access granted | Pass |
| TC-006 | Invalid ticket scan | Scan expired/fake QR code | Access denied, error message | Access denied, error message | Pass |
| TC-007 | Event listing | Open event page | Display list of events | Display list of events | Pass |
| TC-008 | Event details display | Select an event | Show event details | Show event details | Pass |
| TC-009 | Payment processing | Enter card details & pay | Payment successful | Payment successful | Pass |
| TC-010 | Payment failure handling | Invalid card details | Payment declined message | Payment declined message | Pass |
| TC-011 | Logout functionality | Click logout | User logged out successfully | User logged out successfully | Pass |
| TC-012 | Notification system | Event update sent | User receives notification | User receives notification | Pass |
| TC-013 | Crowd monitoring (admin) | Check crowd stats | Display real-time crowd info | Display real-time crowd info | Pass |
| TC-014 | Admin dashboard access | Login as admin | Access to admin panel | Access to admin panel | Pass |
| TC-015 | Accessibility features | Use screen reader | App elements properly read | App elements properly read | Pass |

1. **Testing Review**

The Wembley Stadium Management System (WSMS) test plan outlines an API-centric approach to validating core operations. The cases span across authentication, ticketing, analytics, and admin management, ensuring both user-facing and backend systems are reliable.

Strengths:

* Role-based API security.
* Comprehensive coverage of real-world scenarios.
* Focus on QR code-based ticket validation enhances operational efficiency.

Recommendations:

* Extend test coverage to include rate-limiting and error handling.
* Add performance/load testing for high-volume event simulations.
* Test third-party API integration for analytics or external payment gateways.

Conclusion: The WSMS test plan lays a strong foundation for validating critical operations and user experiences. Execution of pending tests and deeper non-functional validation will be key to verifying system readiness.

**Design:**

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a login form

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

**Pseudo code :**

A screenshot of a computer code

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

A computer code with black text

AI-generated content may be incorrect.A black text on a white background

AI-generated content may be incorrect.

**Overall Conclusion**

The WSMA revolutionizes large-scale event management by integrating modern software principles with operational efficiency and security. Built using C#, the system is designed to handle high-traffic events while ensuring real-time updates, secure transactions, and seamless communication between attendees and administrators.

A key achievement of this project is the automation of essential stadium operations, reducing manual workload and increasing accuracy. The app centralizes event scheduling, ticket validation, security coordination, and crowd monitoring, enabling efficient event management. Features like QR-based ticket scanning, push notifications, and secure payments enhance user experience, while robust authentication ensures security and data protection.

Technically, the app follows a modular, scalable design, allowing easy integration with third-party services via APIs. Its mobile-first approach ensures accessibility across platforms, improving usability. The rigorous testing phase covers core functionalities, including authentication, ticketing, payments, and accessibility, reinforcing system reliability. However, additional stress testing and security audits could further enhance resilience in peak demand scenarios.

The project aligns with industry standards, emphasizing security, accessibility, and efficiency. The admin dashboard supports real-time event monitoring, improving oversight and decision-making. Accessibility features ensure inclusivity, making the system user-friendly for all.

**Summary of work:**

In conclusion, the Wembley Stadium Management App is a powerful, scalable solution that enhances efficiency, security, and user experience. While refinements in performance testing and security could further strengthen the system, it establishes a solid foundation for future advancements, meeting the growing demands of modern stadium management.

**Limitations and reasons:**

In this project, we explored two distinct code structures, each based on a different design approach, to determine which would offer a more user-friendly experience. Our primary objective was to develop a final version that was not only efficient but also intuitive and easy to use to navigate. To achieve this, we created two comprehensive design plans, each with unique functionalities and structural layouts. W e conducted a through analysis of both options, evaluating their strengths and weaknesses in terms of usability, maintainability, scalability and overall performance. After a detailed comparison, we chose the structure that proved to be more practical and aligned with our specific project requirements .This choice was guided not only by the technical factor but also by how effectively the design met user expectations and supported our long-term development goals.

**Change of approach :**

Looking forward we can improve our approach by streaming the initial phase and establishing clear evaluation criteria from the beginning instead of fully developing two separate structures, we could start with prototypes or wireframes to assess feasibility and user experience earlier in the process. Additionally , incorporating user feedback at earlier stages would help decision more effectively. Adopting an agile methodology with iterative testing and refinement could also reduce redundant effort while maintaining flexibility .This adjustments would allow us to allocate resources more efficiently ,make faster decision and enhance the overall quality and responsiveness of the final product.

**Team Contributions Summary:**

Throughout the course of the project, each team member contributed to various aspects of development and documentation. Below is a summary of the roles and contributions made by each team member:

* **Iliya Hajimohammadi (Team Leader):**  
  Initially developed a full version of the backend logic independently. Led team meetings and coordinated the overall project direction. Also reviewed and finalized the report content for consistency and clarity.
* **Kassra Niroumand (Lead Developer):**  
  Took the lead in developing the final backend logic and structure. His codebase was selected as the main project base after week 9. Responsible for the core application functionality and retrieval logic.
* **Arda Janbek Ozturk (Developer):**  
  Contributed to front-end and minor back-end enhancements after the team settled on the unified codebase.
* **Ralph Ndip (Secretary):**  
  Authored the **pseudocode** section of the report and contributed to outlining the development process and weekly progress documentation.
* **Omar Hend (Tester):**  
  Contributed to writing the **report** alongside Ralph. Also responsible for testing the application and verifying the correctness and stability of various features throughout development.