

Software requirement specification document for project Football court booking system.

Mazen Reda 245747, Kerols Abdallah 245175,

Yousef Ashraf 248527, Abdelrahman Ali 248073

Supervised by:

March 5, 2025

# 1 Introduction

## 1.1 Purpose of this document

This document lays out the requirements needed to build up a platform that can make users book football courts online which can be way easier and more efficient than the normal way.

## 1.2 Scope of this document

The system should have multiple options where the users can check court availability, process payments, make bookings and receive notifications. On the other side the admin will have control over bookings and over how they could manage the court. The platform should be accessible through web browsers, and it should be available on all devices.

## 1.3 Overview

This document covers the system’s features, user needs, technical constraints, and other important aspects to ensure a smooth development process. in this document, we should cover the system features, and how it aligns with the user needs, technical constraints and the other aspects that can give a smooth experience in terms of developing the system.

## 1.4 Business Context

The aim of this system is to make finding and booking courts way easier and eliminating the traditional manual booking errors and schedule issues , which can make the user experience way more smoother.

# 2 General Description

## 2.1 Product Functions

2.1 Product Functions

1) Sign in to the account or log in.

2) searching for available courts.

3) booking processing and making payments

4) receiving notifications and confirmation

5) manage bookings through user dashboard

6). Manage users and courts through an admin panel.

7) Ratings and reviews for courts.

8) Cancellation and refund system.

9) appropriate permissions for different roles.

10) Search bar for easy access.

11) history panel for previous and current bookings.

12) integrating with the Calendar app

13) discounts and special offers.

14) Performance tracking for admins.

## 2.2 Similar System Information

The system should run on the standalone platform, and it should have the option to integrate it with the Calendar app as well as make payments in different ways and gateways if needed.

## 2.3 User Characteristics

The targeted user for the system is football players managers and club owners. The system is very simple to use, and it doesn’t require the user to be good with technology.

## 2.4 User Problem Statement

The system saves the users time and makes it simple and smooth to book football courts in real time without it taking too long and without the errors that can occur while booking Court using the traditional way.

## 2.5 User Objectives

1) Find the courts and book quickly.

2) Make payments safely.

3). receive notifications about the upcoming bookings.

4) manage reservations easily.

## 2.6 General Constraints

1) Works on all web browsers.

2) Safe login for user accounts.

3) Easy to use on mobile devices.

Table 1: Functional Requirement XYZ

1. Register user

|  |  |
| --- | --- |
| Function name | Register user |
| Description | New users that are new on the website to create an account through using their emails, phone numbers, passwords name |
| Critically | Very essential as it's the starting point for any system interaction. |
| Technical issues | Requires data validation and integration with the database. |
| Cost and schedule | Relatively low cost and quick to implement. |
| Risks | Weak validation may lead to fake or incomplete registrations. |
| Dependencies | It depends on user management and database integration. |
| Pre-condition | Users are not registered. |
| Post-condition | A new user account is created in the system. |

1. Login user

|  |  |
| --- | --- |
| Function name | Login user |
| Description | When users are already registered, they can use the information they registered with the login into their account at any time. |
| Critically | Crucial to access protected functionalities like booking and payment. |
| Technical issues | Requires session management and password encryption. |
| Cost and schedule | Relatively low and fast to implement. |
| Risks | Weak security can lead to account breaches. |
| dependencies | Depends on user registration and existing credentials. |
| Pre-condition | User has a registered account. |
| Post-condition | User is logged in with an active session. |

1. Logout

|  |  |
| --- | --- |
| Function name | Logout |
| Description | Users should have the option to log out of their account on the website. |
| Critically | Important for user security and session management. |
| Technical issues | Requires proper session termination. |
| Cost and schedule | Very simple and quick to implement. |
| Risks | Incomplete logout could lead to unauthorized access. |
| Dependencies | Depends on login system. |
| Pre-condition | User is logged in. |
| Post-condition | User session is terminated. |

1. Search courts

|  |  |
| --- | --- |
| Function name | Search courts |
| Description | Users can search for football courts using their location, time of availability, their rating, their reviews and the court type. |
| Critically | Very important for user experience and usability. |
| Technical issues | Requires an optimized database and efficient search algorithms. |
| Cost and schedule | Medium complexity, requires frontend and backend coordination. |
| Risks | Poor performance or irrelevant results may frustrate users. |
| Dependencies | Depends on the availability of court data. |
| Pre-condition | Courts are already available in the database. |
| Post-condition | Search results are shown to the user. |

1. View court details

|  |  |
| --- | --- |
| Functions name | View court details |
| Description | Having full details on every court, including images, location, ratings, reviews, and price. |
| Critically | Essential for users to decide before booking. |
| Technical issues | Requires fetching and displaying detailed court information. |
| Cost and schedule | Low to medium depending on UI design. |
| Risks | Incomplete or wrong data may break trust. |
| Dependencies | Depends on existing court records. |
| Pre-condition | User selects a specific court. |
| Post-condition | Court details are displayed to the user. |

1. View availability

|  |  |
| --- | --- |
| Function name | View availability |
| Description | Users can see the availability of the court in every time slot of the day. |
| Critically | Crucial to know booking slots. |
| Technical issues | Time management and real-time updates needed. |
| Cost and schedule | Medium, needs solid schedule management. |
| Risks | Displaying booked slots as available causes double-booking. |
| Dependencies | Depends on booking records and court schedules. |
| Pre-condition | User has selected a court. |
| Post-condition | Available time slots are displayed. |

1. Make booking

|  |  |
| --- | --- |
| Function name | Make booking |
| Description | Users should have the ability to make a booking for a court of their choice on a selected time slot that they choose. |
| Criticality | One of the core functions of the system. |
| Technical issues | Needs availability checks, user authentication, and payment link. |
| Cost and schedule | High due to logic complexity and multi-step integration. |
| Risks | Duplicate or failed bookings. |
| Dependencies | Depends on login, availability, and payment. |
| Pre-condition | User is logged in and has selected a valid time slot. |
| Post-condition | Booking is successfully stored. |

1. Cancel booking

|  |  |
| --- | --- |
| Function name | Cancel booking |
| Description | Users should have the ability to cancel the bookings that they made. |
| Criticality | Important to offer user flexibility. |
| Technical issues | Needs booking status change and possibly refund handling. |
| Cost and schedule | Medium, less complex than booking. |
| Risks | Refund failure or cancellation of already-used bookings. |
| Dependencies | Depends on existing bookings. |
| Pre-condition | Active booking exists. |
| Post-condition | Booking is canceled and slot freed. |

1. Payment integration

|  |  |
| --- | --- |
| Function name | Payment integration |
| Description | Users should have the ability to integrate Seif payment methods while booking their courts throughout credit cards, debit cards, online payment methods, etc. |
| Criticality | Essential for any paid booking system. |
| Technical issues | Secure integration with payment gateway and transaction verification. |
| Cost and schedule | High due to security and third-party integration. |
| Risks | Failed transactions, insecure data handling. |
| Dependencies | Depends on booking and user information. |
| Pre-condition | User has initiated a booking. |
| Post-condition | Payment is completed and recorded. |

1. Booking confirmation

|  |  |
| --- | --- |
| Function name | Booking confirmation |
| Description | The system should have the ability to send notifications SMS messages to users that have booked a court to confirm their payments and the court being successfully booked. |
| Criticality | Important to formally conclude the booking process. |
| Technical issues | Requires notification (email/onsite) and confirmation record. |
| Cost and schedule | Low to medium depending on notification system. |
| Risks | Missed confirmations could confuse users. |
| Dependencies | Depends on booking and payment completion. |
| Pre-condition | Booking and payment are complete. |
| Post-condition | User receives confirmation message. |

1. View booking history

|  |  |
| --- | --- |
| Function name | View booking history |
| Description | Users should have the ability to see the history of their past and upcoming booking. |
| Criticality | Useful for tracking previous activity. |
| Technical issues | Fetch and display user-linked bookings. |
| Cost and schedule | Low, easy to implement. |
| Risks | Displaying wrong user data. |
| Dependencies | Depends on user and booking records. |
| Pre-condition | User is logged in. |
| Pre-condition | Booking history is shown. |

1. Rate court

|  |  |
| --- | --- |
| Function name | Rate court |
| Description | The users should be able to leave a rating and a review on the courts that they’ve booked and played in. |
| Criticality | Helpful for quality control and user feedback. |
| Technical issues | Must ensure only valid users rate, and link rating to booking. |
| Cost and schedule | Low to medium. |
| Risks | Abuse or spam ratings. |
| Dependencies | Depends on completed bookings. |
| Pre-condition | User has completed a booking. |
| Post-condition | Rating is submitted and stored. |

1. View rating

|  |  |
| --- | --- |
| Function name | View rating |
| Description | The system should display the ratings for every court, and the reviews for this court as well for all users to see. |
| Criticality | Helps users in decision-making. |
| Technical issues | Calculate average, total reviews, and present clearly. |
| Cost and schedule |  |
| Risks | Outdated or fake reviews. |
| Dependencies | Depends on rating submissions. |
| Pre-condition | User views a court. |
| Post-condition | Ratings are shown. |

1. Admin dashboard

|  |  |
| --- | --- |
| Function name | Admin dashboard |
| Description | The administration should have a full view of the system, including the cord status current bookings, users activities and earnings. |
| Criticality | Crucial for system control and monitoring. |
| Technical issues | Complex UI, access roles, and performance optimization. |
| Cost and schedule | Medium to high depending on features. |
| Risks | Poor dashboard may limit control. |
| Dependencies | Tied to all system data. |
| Pre-condition | Admin is logged in. |
| Post-condition | Dashboard is accessible with management features. |

1. Manage users

|  |  |
| --- | --- |
| Function name | Manage users |
| Description | Admin should have the ability to view ,ban ,unban ,edit or delete users and their accounts. |
| Criticality | Vital for moderation and user management. |
| Technical issues | Editing, banning, and managing data securely. |
| Cost and schedule |  |
| Risks | Accidental deletion or unauthorized changes. |
| Dependencies | Depends on user database. |
| Pre-condition | Admin access granted. |
| Post-condition | User data is updated or modified. |

1. Add new court

|  |  |
| --- | --- |
| Function name | Add new court |
| Description | Admin should have the ability to add new football courts and all their details to the platform. |
| Criticality | Important to grow the system and serve more users. |
| Technical issues | Validate and store court info like name, location, photos. |
| Cost and schedule |  |
| Risks | Duplicate or incorrect entries. |
| Dependencies | Requires admin access. |
| Pre-condition | Admin is logged in. |
| Post-condition | New court is added to the system. |

1. Set pricing

|  |  |
| --- | --- |
| Function name | Set pricing |
| Description | Admins and court owners should have the ability to edit the price of the courts. |
| Criticality | Essential for managing booking fees. |
| Technical issues | Logic for dynamic pricing by time/day/type. |
| Cost and schedule |  |
| Risks | Incorrect prices may affect revenue or fairness. |
| Dependencies | Linked to court data. |
| Pre-condition | Admin selects a court. |
| Post-condition | Price is set for the court. |

1. Manage court availability

|  |  |
| --- | --- |
| Function name | Manage court availability |
| Description | The owners and the admins should have the ability to edit when the court is available (not booked) or unavailable(booked). |
| Criticality | Controls what time slots are bookable. |
| Technical issues | Avoid overlapping or deleting booked times. |
| Cost and schedule |  |
| Risks | Disabling a time with active bookings. |
| Dependencies | Depends on court schedules and bookings. |
| Pre-condition | Admin selects court. |
| Post-condition | Updated availability. |

1. Booking limits

|  |  |
| --- | --- |
| Function name | Boking limits |
| Description | The admin should have the ability to restrict users from booking more courts when they reach a certain limit. |
| Criticality | Prevents abuse of the system by overbooking. |
| Technical issues | Logic to enforce limits per user/day/week. |
| Cost and schedule |  |
| Risks | Unfair limits may block genuine users. |
| Dependencies | Depends on booking history. |
| Pre-condition | User tries to book. |
| Post-condition | Booking is allowed or denied based on limits. |

1. Notification system

|  |  |
| --- | --- |
| Function name | Notification system |
| Description | Send notifications to the users about their payment confirmation offers or sales on the courts and their booking status. |
| Criticality | Enhances user interaction and engagement. |
| Technical issues | Email/SMS notifications for key events. |
| Cost and schedule |  |
| Risks | failed delivery. |
| Dependencies | Tied to various triggers like bookings, payments. |
| Pre-condition | Trigger event happens. |
| Post-condition | Notification is sent. |

1. Profile management

|  |  |
| --- | --- |
| Function name | Profile management |
| Description | Users can update their information like change, passwords, or name and update other details. |
| Criticality | Allows users to maintain their information. |
| Technical issues | Updating sensitive data with proper validation. |
| Cost and schedule |  |
| Risks | Data loss or errors in editing. |
| Dependencies | Requires logged-in session. |
| Pre-condition | User is logged in. |
| Post-condition | Profile data is updated. |

1. Forgot password

|  |  |
| --- | --- |
| Function name | Forgot password |
| Description | The system should provide a way for users that forgot their password to recover it and make a new password via email or Messages for verification of their identity. |
| Criticality | Important for account recovery. |
| Technical issues | Secure token generation, reset link, and expiration handling. |
| Cost and schedule |  |
| Risks | Unauthorized password resets. |
| Dependencies | Depends on email or phone record. |
| Pre-condition | User forgets password. |
| Post-condition | Password reset link is sent. |

1. Admin manage reviews

|  |  |
| --- | --- |
| Function name | Admin manage reviews |
| Description | Admin and moderators should have the ability to edit or remove inappropriate reviews or content on the website. |
| Criticality | Maintains review quality and community standards. |
| Technical issues | Filtering, editing, or removing reviews. |
| Cost and schedule |  |
| Risks | Censoring legitimate feedback or missing abusive reviews. |
| Dependencies | Depends on review system. |
| Pre-condition | Reviews exist. |
| Post-condition | Review is moderated. |

1. Transaction history

|  |  |
| --- | --- |
| Function name | Transaction history |
| Description | The system should give the users history of the payments that they made, and the admin should have it for the record, keeping in the database. |
| Criticality | Adds transparency and helps with user accounting. |
| Technical issues | Pull and format past payments correctly. |
| Cost and schedule |  |
| Risks | Displaying incorrect financial data. |
| Dependencies | Depends on payment system. |
| Pre-condition | User is logged in. |
| Post-condition | Past transactions shown. |

1. Booking receipt download

|  |  |
| --- | --- |
| Function name | Booking receipt download |
| Description | The system should offer a downloadable receipt in a PDF file format for the users that have booked a court. |
| Criticality | Gives users a formal proof of booking. |
| Technical issues | Generate downloadable PDF or receipt format. |
| Cost and schedule |  |
| Risks | Incomplete or incorrect data on receipt. |
| Dependencies | Depends on confirmed booking and payment. |
| Pre-condition | Booking is complete. |
| Post-condition | User downloads the receipt. |

1. Map integration

|  |  |
| --- | --- |
| Function name | Map integration |
| Description | The system should be integrated with the Maps application on devices so the users can track the courts that they booked. |
| Criticality | Improves ease of finding court location. |
| Technical issues | Integration with Google Maps API or similar. |
| Cost and schedule |  |
| Risks | API failures or incorrect geolocation. |
| Dependencies | Requires court coordinates. |
| Pre-condition | User selects a court. |
| Post-condition | Map is displayed. |

1. Filter courts

|  |  |
| --- | --- |
| Function name | Filter courts |
| Description | The system should provide a filtering option so users can filter courts based on location, rating, availability range ,reviews, etc. |
| Criticality | Enhances usability and search precision. |
| Technical issues | Dynamic filters tied to court data. |
| Cost and schedule |  |
| Risks | Poor filter logic, slow response. |
| Dependencies | Relies on court attributes. |
| Pre-condition | User initiates a search. |
| Post-condition | Filtered results displayed. |

1. Save favorites

|  |  |
| --- | --- |
| Function name | Save favorites |
| Description | The system should have the option to let users, save certain courts that they like into their favorite category. |
| Criticality | Helps users quickly access preferred courts. |
| Technical issues | User-linked saved items, stored in DB. |
| Cost and schedule |  |
| Risks | Lost or deleted favorites. |
| Dependencies | Requires user account. |
| Pre-condition | User selects a favorite. |
| Post-condition | Favorite is saved. |

1. Multi-language support

|  |  |
| --- | --- |
| Function name | Multi-language support |
| Description | The website should have the ability should be translated to multiple languages for wiser accessibility. |
| Criticality | Expands user base and accessibility. |
| Technical issues | Translate static and dynamic content. |
| Cost and schedule |  |
| Risks | Poor translation or layout issues. |
| Dependencies | All UI elements and labels. |
| Pre-condition | User selects language. |
| Post-condition | UI displays selected language. |

1. Responsive design

|  |  |
| --- | --- |
| Function name | Responsive design |
| Description | The system should be accessible on different devices across-the-board like desktops ,tablets ,and smart phones, and it should be simple and easy to use. |
| Criticality | Critical for good experience across devices. |
| Technical issues | Design must adapt fluidly to all screen sizes. |
| Cost and schedule |  |
| Risks | Poor responsiveness leads to frustrated users. |
| Dependencies | Applies to all UI pages |
| Pre-condition | User visits site from any device. |
| Post-condition | Proper layout is shown for the device. |

# 4. Interface Requirements

## 4.1 User Interfaces

There should be a website for the system. The website should be easy to use flexible with a simple interface that works on various devices.

**4.2 Hardware Interfaces**

The hardware interfaces can require web servers like Azure or AWS to support database to save users data like messages, schedules, and names users.

**4.3 Communications Interfaces**

The system shall be integrated with emails/ SMS APIs for notifications.

examples:

1-when a user sign up, they should receive a confirmation email or a SMS message to confirm.

2. User for gas their password they should be able to get a message with a code to reset the password

**4.4 Software Interfaces**

Payment gateway API (e.g., Stripe/Fawry), Google Maps API.

**5 Performance Requirements**

• . Within two seconds the system must respond

• Must keep up with 100 concurrent users at a time.

• Booking conflict detection latency ≤ 1 second.

# 6 Design Constraints

• Follow OWASP guidelines for security

• Must support responsive design (W3C standards).

• Should work on devices with low specs.

**6.1 Standards Compliance**  
-1 Must adhere to local and international pesticide regulations to ensure legal operation and safety.

2- Secure data handling must be implemented for compliance with data protection laws and to support regulatory auditing.

-3 System must comply with web accessibility standards (WCAG) to be usable by people with disabilities.

Secure data handling for regulatory auditing.

* **6.2 Hardware Limitations**  
  The system is designed to run on standard desktop and mobile browsers without the need for specialized hardware.
* Only basic peripheral devices like printers may be required for report printing.
* The system must operate efficiently on devices with moderate hardware specifications.

.

**6.3 others as appropriate**  
1- Development should rely on open-source or widely supported technologies to minimize cost and increase compatibility.

2- The system must be accessible via any modern web browser without requiring additional installations.

3- Code should be modular and follow standard software design principles to support future scalability and maintenance.

7 **Other non-functional attributes**

1.performance

• loading a page should take under three seconds

• ⁠ the system shall support at least 100 users at a time

2.security

• The system will encrypt users’ data

• ⁠ the system will secure the payment processing

• the system will give abilities based on the roles of the users that are accessing the system

3. Maintainability and scalability.

• The design should be simple so it can be easily updated

• ⁠ The system should be scalable, so it can support more users and courts and the future

4. Reliability and availability

• The system should have at least 99.9% up time guarantee

• ⁠ data backup, and recovery system shall be provided

5. compatibility.

• the system should work on major web browsers, and it should be designed to be mobile friendly

6. Usability

• The interface should be user-friendly and simple to navigate through for non-technical users

# 9. Operational Scenarios This section describes a set of scenarios that illustrate, from the user’s perspective, the experience of utilizing the system in various situations. Examples: - Scenario 1: Booking a Court The user logs in, searches for available courts, selects a preferred time, completes the payment, and receives confirmation. - Scenario 2: Forgot Password The user selects the “Forgot Password” option, enters their email, receives a verification code, and resets their password securely.

This section should describe a set of scenarios that illustrate, from the user’s perspective, what will be experienced when utilizing the system under various situations. In the article Inquiry-Based Requirements Analysis (IEEE Software, March 1994), scenarios are defined as follows: In the broad sense, a scenario is simply a proposed specific use of the system. More specifically, a scenario is a description of one or more end-to-end transactions involving the required system and its environment. Scenarios can be documented in different ways, depending up on the level of detail needed. The simplest form is a use case, which consists merely of a short description with a number attached. More detailed forms are called scripts. These are usually represented as tables or diagrams and involved identifying an action and the agent (doer) of the action. For this reason, a script can also be called an action table. Although scenarios are useful in acquiring and validating requirements, they are not themselves requirements, because the describe the system’s behavior only in specific situations; a specification, on the other hand, describes what the system should do in general.

# 10. Preliminary Schedule Adjusted This section outlines an initial project plan including major tasks, dependencies, and tentative start/end dates. - Tasks: Requirement gathering, design, development, testing, deployment. - Tools: Gantt/PERT charts. - Dependencies: UI depends on backend logic; testing depends on development. - Resources: Web developer, UI designer, server, and database.

This section provides an initial version of the project plan, including the major tasks to be accomplished, their interdependence’s, and their tentative start/stop dates. The plan also includes information on hardware, software, and resource requirements. The project plan should be accompanied by one or more PERT or GANTT charts.

11. Preliminary Budget Adjusted  
 Item: Estimated Cost:   
  
1)Server Hosting $50/month   
2) UI/UX Design $200   
3) Development $1000   
4)Payment Gateway Setup Based on usage   
5)External APIs | Free/Paid   
6)Documentation & Support $300   
 Total Estimated Budget: $1550

This section provides an initial budget for the project, itemized by cost factor.

# 12. Appendices 12.1 Definitions, Acronyms, and Abbreviations - SRS: Software Requirements Specification - UI: User Interface - API: Application Programming Interface - DB: Database - OWASP: Open Web Application Security Project 12.2 Collected Material - Wireframes of user interface. - ER Diagrams for database. - Notification and verification message samples. - Interview and survey findings (if any).

Specifies other useful information for understanding the requirements. All SRS documents should include at least the following two appendices:

**12.1 Definitions, Acronyms, Abbreviations**

Provides definitions of unfamiliar definitions, terms, and acronyms.

**12.2 Collected material**

# 12. Appendices

## 12.1 Definitions, Acronyms, and Abbreviations

-SRS: Software Requirements Specification  
- UI: User Interface  
- API: Application Programming Interface  
- DB: Database  
- OWASP: Open Web Application Security Project  
- WCAG: Web Content Accessibility Guidelines  
- PDF: Portable Document Format  
- UX: User Experience

## 12.2 Collected Material

- Wireframes for core UI pages including home, booking, and admin dashboard.  
- Entity-Relationship (ER) diagrams showing relationships among users, courts, and bookings.  
- Example email/SMS messages for booking confirmation and password recovery.  
- Survey feedback from local football club managers regarding their current booking process.  
- Interviews with five end users (players and admins) to validate core system expectations.  
- User flow diagrams and interaction mockups.

# 13. References

1. OWASP Foundation. \*OWASP Top Ten Security Risks\*. Retrieved from https://owasp.org/www-project-top-ten/  
2. Stripe, Inc. \*Stripe API Reference\*. Retrieved from https://stripe.com/docs/api  
3. Google. \*Google Maps Platform Documentation\*. Retrieved from https://developers.google.com/maps/documentation  
4. Fawry. \*Fawry Developer Portal\*. Retrieved from https://developer.fawry.com  
5. IEEE. \*Recommended Practice for Software Requirements Specifications (IEEE 830-1998)\*.  
6. World Wide Web Consortium (W3C). \*Web Content Accessibility Guidelines (WCAG) 2.1\*. Retrieved from https://www.w3.org/WAI/standards-guidelines/wcag/  
7. ISO/IEC/IEEE 29148:2018. \*Systems and Software Engineering — Life Cycle Processes — Requirements Engineering\*.  
8. Jakob Nielsen. \*Usability Engineering\*. Academic Press, 1994.