Software Requirements Specification (SRS)

SMART PITCH

by Khaled Amr
Ahmed Osama
Ali Emam
Zeyad Mossad

Project Title: Smart Pitch

Course: CSAI 203 – Introduction to Software Engineering

Semester: Fall 2025

Team Members: (Ahmed Osama 202400923 - Khaled Amr Ameen 202400505 - Ali Emam Ali 202401429 – Zeyad Mossad 202401628)

Representative Email: s-ahmed.omohammed@zewailcity.edu.eg

Date: 7/11/25

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to clearly describe what the *Smart Pitch* web application will do and how it should perform. *Smart Pitch* allows users to easily find, book, and manage football pitch reservations in their area. It also gives pitch owners tools to manage their fields and lets administrators oversee the entire system.

This document acts as a guide for everyone involved in the project — helping the team design, develop, and test the system with a shared understanding of its goals and features.

1.2 Scope

Smart Pitch is a web-based system that connects football players with local pitch owners. It allows users to:

1. Register and log in securely.
2. Search and filter available pitches by date, location, and price.
3. Book, reschedule, or cancel reservations.
4. Manage user profiles and booking history.
5. Enable pitch owners to list, update, or remove their pitches.
6. Allow administrators to manage users, pitches, and bookings.

The system will use **Flask (Python)** for the backend and **HTML/CSS** for the frontend.

Out of scope: integration with real payment systems, GPS-based location tracking, or mobile app development.

1.3 Definitions, Acronyms, and Abbreviations

Term	*Definition*
SRS	Software Requirements Specification
UI	User Interface
MVC	Model–View–Controller pattern
DB	Database
HTTP	Hypertext Transfer Protocol

1.4 References

1. CSAI 203 Course Project Guide (Fall 2025).
2. Flask Official Documentation (https://flask.palletsprojects.com).
3. W3Schools and MDN Developer Documentation for HTML/CSS.

1.5 Overview

This document includes:

- Section 2: Overall description and environment.
- Section 3: Specific functional and non-functional requirements.
- Section 4: Appendices, including data dictionary and glossary.

2. Overall Description

2.1 Product Perspective

The *Smart Pitch* application functions as a **client-server system**. Users interact through a web browser, while the backend (Flask) processes data and communicates with a database. The system follows the **MVC architecture**:

- **Model:** Manages data and logic (users, pitches, bookings).
- **View:** Displays information to users via HTML/CSS.
- **Controller:** Handles requests and routes between model and view.

2.2 Product Functions

1. Register and log in securely.
2. View available football pitches.
3. Search and filter by location, date, and price.
4. Book or cancel time slots.
5. Manage user profile and booking history.
6. Pitch owners can add/edit/remove pitches.
7. Admin manages users, pitches, and bookings.
8. Display confirmation and notification messages.

2.3 User Classes and Characteristics

User Class	Description	Skills/Characteristics
Player/User	Regular users looking to book a pitch.	Basic computer skills.

Pitch Owner	Manages pitch listings and availability.	Knowledge of local pitch details.
Administrator	Monitors system data and user activities.	Technical and management skills.

2.4 Operating Environment

- **Frontend:** HTML/CSS
- **Backend:** Python Flask
- **Database:** SQLite or MySQL
- **Browser Support:** Chrome, Edge, Firefox
- **Operating Systems:** Windows, macOS, Linux

2.5 Design and Implementation Constraints

1. Must follow Flask MVC structure.
2. Use only HTML for UI (no external design tools).
3. Must comply with project timeline and submission format.
4. Backend must be implemented in Python 3.x.

2.6 User Documentation

- User manual for registration, searching, and booking.
- Admin and owner guides for system management.

2.7 Assumptions and Dependencies

1. Internet connection is available.
2. Pitch data provided by owners is accurate.
3. Valid user credentials are required for booking.
4. Email notifications depend on SMTP configuration.

3. Specific Requirements

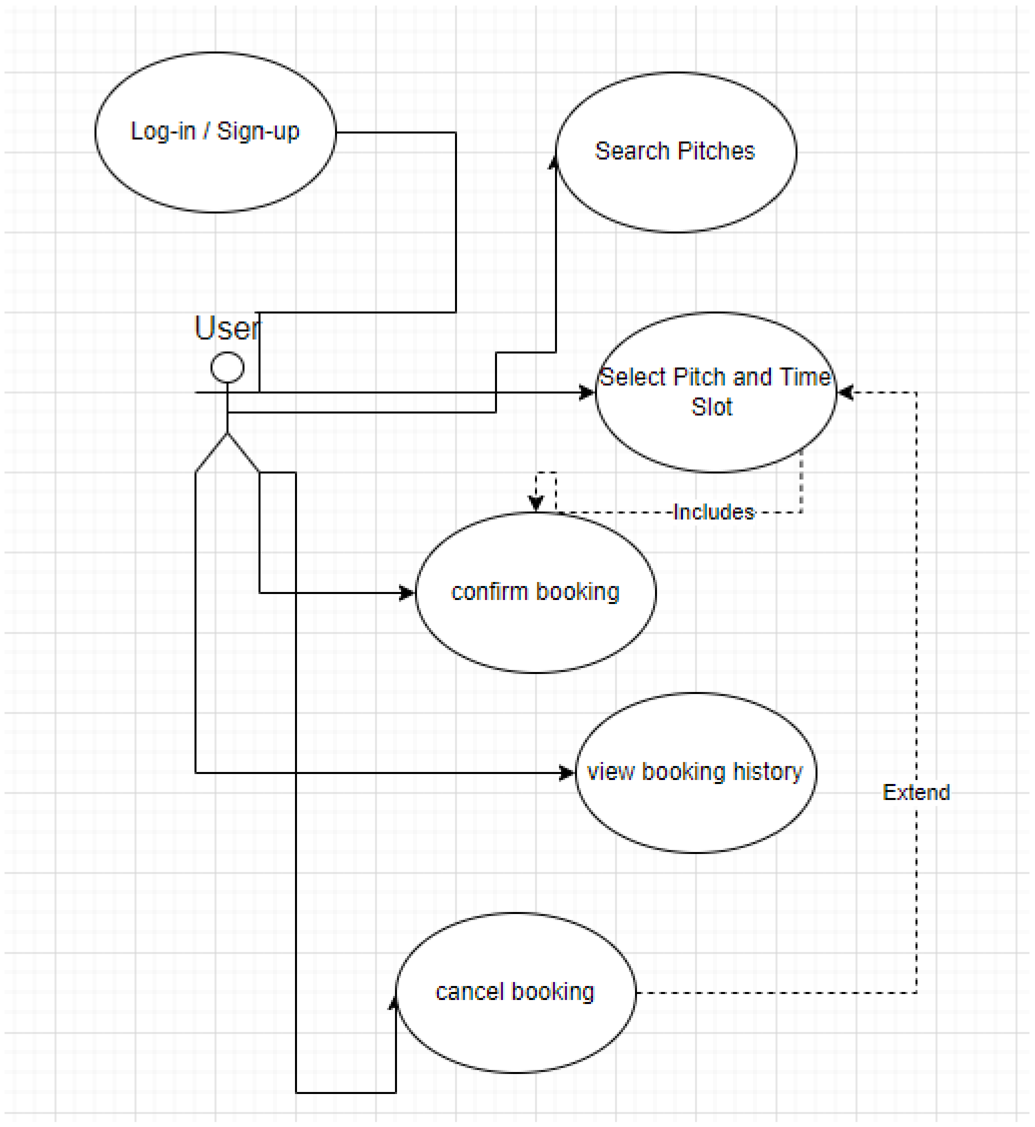
3.1 Functional Requirements

N	Functional Requirement	Description
1	User Registration	Register with unique email and password.
2	User Login/Logout	Secure authentication process.
3	View Pitches	Display list of available pitches.
4	Search/Filter	Filter by location, price, or date.
5	Book Pitch	Reserve available pitch slot.
6	Manage Bookings	View, cancel, or reschedule bookings.
7	Pitch Management	Owners manage their pitch listings.
8	Reviews & Ratings	Users can rate pitches.
9	Admin Management	Admin manages users, pitches, and reports.
10	Notifications	Confirmation and update messages to users.

3.2 Use Case Model

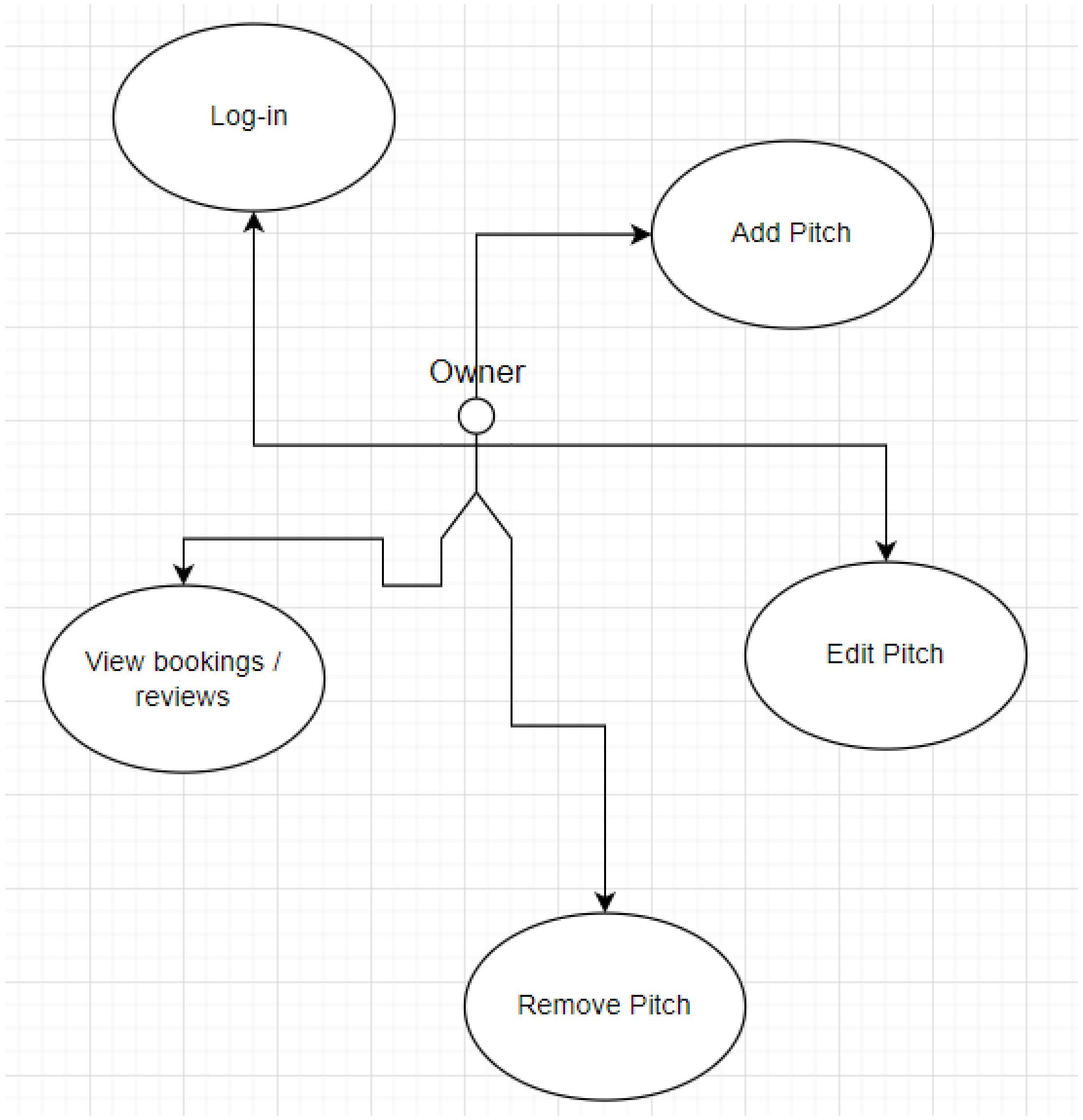
3.2.1 Use Case Diagrams

- **Figure 1:**



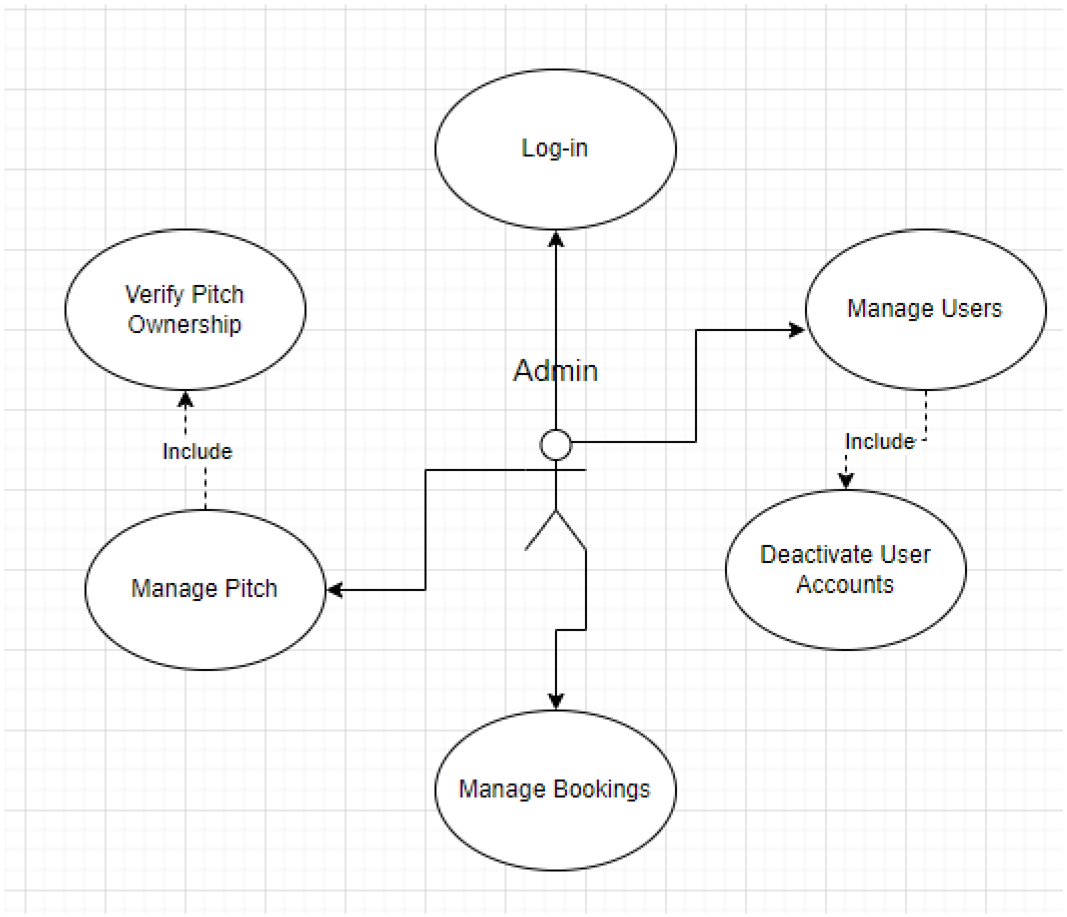
User Booking Process

- **Figure 2:**



Pitch Owner Management

- **Figure 3:**



Admin Operations

3.2.2 Use Case Descriptions

Use Case 1: Book a Pitch

- **Actors:** User
- **Preconditions:** User logged in; pitch available.
- **Main Flow:**
    1. Log-in / Sign-up
    2. Search available pitches.
    3. Select pitch and time slot.
    4. Confirm booking.

5. System records booking and sends confirmation.

- **Postcondition:** Booking record created.

Use Case 2: Manage Pitches (Owner)

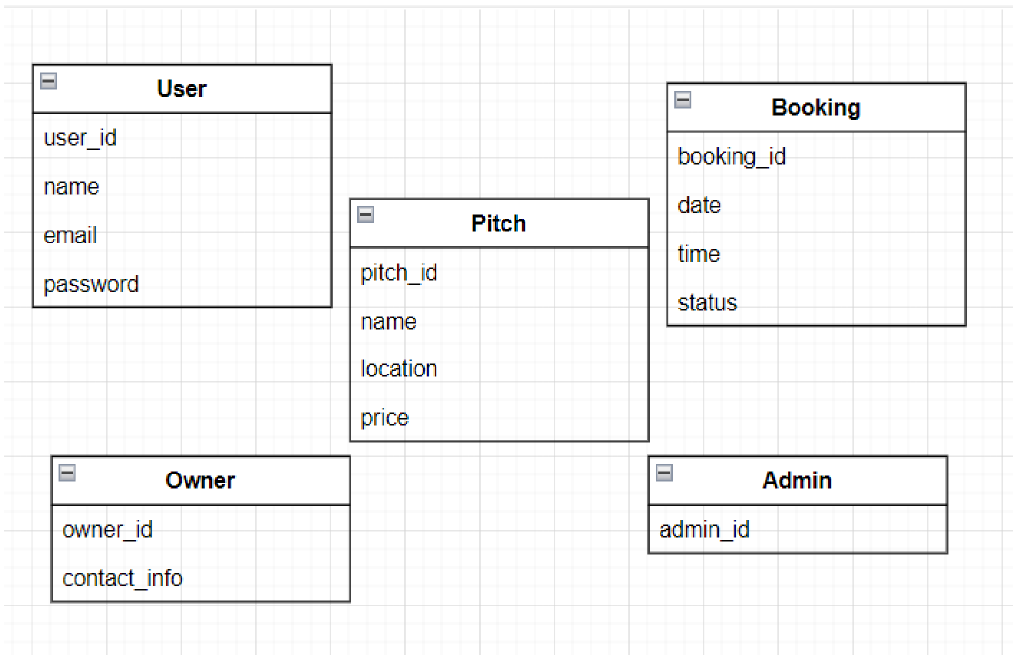
- **Actors:** Pitch Owner
- **Preconditions:** Owner logged in.
- **Main Flow:** Add, edit, or delete pitch details.
- **Postcondition:** Pitch data updated in the database.

Use Case 3: Manage Users/Bookings (Admin)

- **Actors:** Admin
- **Main Flow:** View and manage users and bookings.
- **Postcondition:** Database updated after operations.

3.3 Domain Model

3.3.1 Conceptual Class Diagram



3.3.2 Class Descriptions

Class	Description	Attributes	Relationships
User	Represents registered players.	User_id, name, email, password	1–M with Booking
Pitch	Represents football fields.	pitch_id, name, location, price	1–M with Booking
Booking	Reservation record.	booking_id, date, time, status	M–1 with User and Pitch
Owner	Manages pitches.	owner_id, contact	1–M with Pitch
Admin	Manages users and data.	admin_id, privileges	Access to all records

3.4 Non-Functional Requirements

1- Notifications / Reminders: Notify users of upcoming bookings or cancellations.
2- Customizable Booking Themes: Users can pick colors or layouts for their dashboard

3.5 External Interface Requirements

3.5.1 User Interface

Pages: Login, Register, Search, Booking, Dashboard, Admin Panel.

3.5.2 Hardware Interface

Accessible via any device with internet connection (PC, laptop).

3.5.3 Software Interface

Backend: Flask | Database: SQLite/MySQL | Frontend: HTML/CSS.

3.5.4 Communication Interface

HTTP/HTTPS communication | Optional email notifications via SMTP.

4. Appendices

4.1 Appendix A: Data Dictionary

Entity	Field	Type	Description
User	user_id	Integer	Unique identifier
Pitch	pitch_id	Integer	Unique pitch ID
Booking	booking_id	Integer	Booking record
Owner	owner_id	Integer	Pitch owner ID

4.2 Appendix B: Glossary

Term	Definition
Booking	A confirmed reservation for a pitch.
Pitch Owner	Individual managing a football field.
User	Person using the system to find and book pitches.
Admin	User with full system access and control.