Sports Match Ticketing System

Requirements Documentation (SMART)

---

1 Project Overview

The Sports Match Ticketing System is a console-based C++ application that allows administrators to manage match inventory and fans to book tickets.
The system follows Agile methodology and is delivered incrementally through sprints.

---

2 Stakeholders

- **Admin**: Manages matches and monitors bookings
- **Fan**: Views matches and books tickets

---

3 Functional Requirements (SMART)

FR-01: Add Match

Description:
The system shall allow an Admin to add a new match with a unique ID, match code, and capacity.

SMART Analysis:

- **Specific:** Add match with code and capacity
- **Measurable:** Match appears in match list
- **Achievable:** Implemented using C++ classes
- **Relevant:** Core system functionality
- **Time-bound:** Available in Sprint 2

FR-02: View Matches

Description:
The system shall allow a Fan to view all available matches along with their remaining seats.

SMART Analysis:

- **Specific:** Display match ID, code, and available seats
- **Measurable:** Matches are listed correctly
- **Achievable:** Console output
- **Relevant:** Required before booking
- **Time-bound:** Available in Sprint 1

FR-03: Book Ticket

Description:
The system shall allow a Fan to book a ticket for a selected match if seats are available.

SMART Analysis:

- **Specific:** Book one seat per request
- **Measurable:** Booked count increases by one
- **Achievable:** Implemented in Match class
- **Relevant:** Main user goal
- **Time-bound:** Available in Sprint 1

FR-04: Prevent Overbooking

Description:
The system shall prevent ticket booking when the match capacity is fully booked.

SMART Analysis:

- **Specific:** Block booking if booked = capacity
- **Measurable:** Booking request rejected
- **Achievable:** Capacity check logic
- **Relevant:** Data integrity
- **Time-bound:** Available in Sprint 1

FR-05: Generate Ticket

Description:
The system shall generate a ticket with a unique ticket ID and associate it with the selected match and fan.

SMART Analysis:

- **Specific:** Create ticket object per booking
- **Measurable:** Ticket stored in match ticket list
- **Achievable:** Ticket class
- **Relevant:** Confirms reservation
- **Time-bound:** Available in Sprint 1

FR-06: View Booked Tickets (Fan)

Description:
The system shall allow a Fan to view all tickets they have booked.

SMART Analysis:

- **Specific:** Filter tickets by fan name
- **Measurable:** Correct ticket list shown
- **Achievable:** Iteration over tickets
- **Relevant:** User transparency
- **Time-bound:** Available in Sprint 2

FR-07: View Match Bookings (Admin)

Description:
The system shall allow an Admin to view all booked tickets for a selected match.

SMART Analysis:

- **Specific:** Display ticket ID and owner
- **Measurable:** Tickets listed per match
- **Achievable:** Match aggregation
- **Relevant:** Admin monitoring
- **Time-bound:** Available in Sprint 2

FR-08: Delete Match

Description:
The system shall allow an Admin to delete a match and all associated tickets.

SMART Analysis:

- **Specific:** Remove match and tickets
- **Measurable:** Match no longer listed
- **Achievable:** Vector erase operation
- **Relevant:** Match management
- **Time-bound:** Available in Sprint 2

---

4 Non-Functional Requirements (SMART)

NFR-01: Usability

The system shall use a console-based menu with clear options and single-word inputs.

- **Specific:** CLI interface
- **Measurable:** Users can complete booking without errors
- **Achievable:** Console I/O
- **Relevant:** Ease of use
- **Time-bound:** Sprint 2

NFR-02: Reliability

The system shall not crash on invalid user input.

- **Specific:** Input validation
- **Measurable:** No runtime errors
- **Achievable:** Conditional checks
- **Relevant:** System stability
- **Time-bound:** Sprint 2

NFR-03: Maintainability

The system shall use a multi-file OOP structure.

- **Specific:** Header/source separation
- **Measurable:** Classes in separate files

- **Achievable:** C++ multi-file project
- **Relevant:** Future enhancements
- **Time-bound:** Sprint 1

---

5 Assumptions & Constraints

- All inputs are single words (no spaces)
- The system is console-based only
- No database; in-memory storage (CSV in future sprint)

---

6 Acceptance Criteria Summary

- Admin can add and delete matches
- Fan can view and book tickets
- Overbooking is prevented
- Tickets are correctly associated with matches
- System handles invalid input gracefully