1. Introduction

1.1 Purpose of the Test Plan

The purpose of this Test Plan is to define the scope, approach, resources and deliverables for testing the **SportSphere** platform — a web application that connects players, venues, and coaches through booking, hosting, and discovery functionalities.

1.2 Objective

- To ensure the SportSphere platform meets functional and non-functional requirements.
- To verify that all modules work reliably across browsers and devices.
- To detect defects early and ensure they are resolved before deployment.

2. Scope of Testing

2.1 In-Scope

Testing will be conducted for the following modules:

1. **User Authentication**
    - Registration (Email Verification)
    - Login
    - Forgot Password
    - Profile Management
2. **Home Page**
    - Navigation menu
    - CTA buttons
    - Footer and social media links
    - Responsive UI
3. **Venue Module**
    - View Venues
    - Venue details page

   - Availability calendar
   - Booking flow

4. **Coach Module**
   - View coaches
   - Coach details
   - Booking sessions

5. **Games Module**
   - Host game
   - Join game
   - Game details

6. **Dashboard**
   - User bookings
   - Payment history
   - Notifications

2.2 Out-of-Scope

- Payment gateway integration testing with real transactions
- Load testing beyond 1,000 users
- Localization or multiple languages
- Native mobile app testing

3. Test Strategy

3.1 Levels of Testing

Level	Description
Unit Testing	Performed by developers on controllers, services, and components
Integration Testing	API and UI integration (React ↔ Node.js ↔ MongoDB)
System Testing	End-to-end functionality testing
Mutation Testing	Evaluates the strength of test cases by introducing small code changes (mutants) and checking whether tests detect them.
GUI Testing	Tests the visual interface and user interactions to ensure all UI elements (buttons, forms, navigation) work and appear correctly.

---

4. Test Types

4.1 Functional Testing

- UI validation
- Form validation
- Navigation
- Booking flow
- Authentication

4.2 Non-Functional Testing

- **Usability**: intuitive design, readability
- **Compatibility**: Chrome, Safari

---

5. Test Environment

5.1 Hardware

- Desktop/Laptop (Windows / macOS)
- Minimum 8GB RAM

5.2 Software

- Browsers: Chrome,Safari
- Backend: Node.js + Express
- Database: MongoDB Atlas
- Frontend:
- Deployment: Vercel / Render

5.3 Tools

- Jest (Unit Testing, if used)
- GitHub
- Selenium (GUI Testing)
- Non Functional Testing (Apache - Jmeter)

---

6. Test Data

Example test datasets:

- Valid/Invalid user emails
- Mock OTP codes
- Venue booking time slots
- Dummy coach profiles

---

7. Test Deliverables

- Test Plan (this document)
- Test Scenarios
- Test Cases
- Bug Report

---

8. Roles & Responsibilities

Role	Responsibility
Tester	Create & execute test cases, report bugs
Developer	Fix defects, support debugging
UI/UX Designer	Validate layout issues

---

9. Exit Criteria

Testing will end when:

- All critical and major defects are fixed
- 90%+ test cases passed
- All modules tested end-to-end

---

10. Entry Criteria

Testing begins when:

- All modules are deployed on testing URL
- API endpoints are functional
- UI components integrated