UE22CS352B - Object Oriented Analysis & Design
Mini Project Report

Title: Sportify: Sports Venue Management System

Submitted by:

Name :
PES1UG22CS380(Neha V Malage)
PES1UG22CS431(Pranav Muppuru)
PES1UG22CS390(Nikita Nair)
PES1UG22CS417(Poorna Pragnya H)

Semester 6
Section G

Faculty Name: Bhargavi Mokashi
January - May 2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under
Karnataka Act No. 16 of
2013)
100ft Ring Road,
Bengaluru –
560085,
Karnataka, India

Problem Statement:

Sports venues today are often managed through disconnected spreadsheets, phone calls and email threads, leading to double-bookings, underutilized courts and high administrative overhead. Customers struggle to find real-time availability or compare pricing, and payments are handled manually or via ad-hoc processes. This project delivers a unified, secure RESTful backend that empowers administrators, venue managers and end-users with role-based access and eliminates manual inefficiencies and delivers a consistent, extensible platform for sports-court management.

Key Features:

This project is a digital platform that helps people book sports venues (like football fields, basketball courts, or tennis courts) online.

How it works?

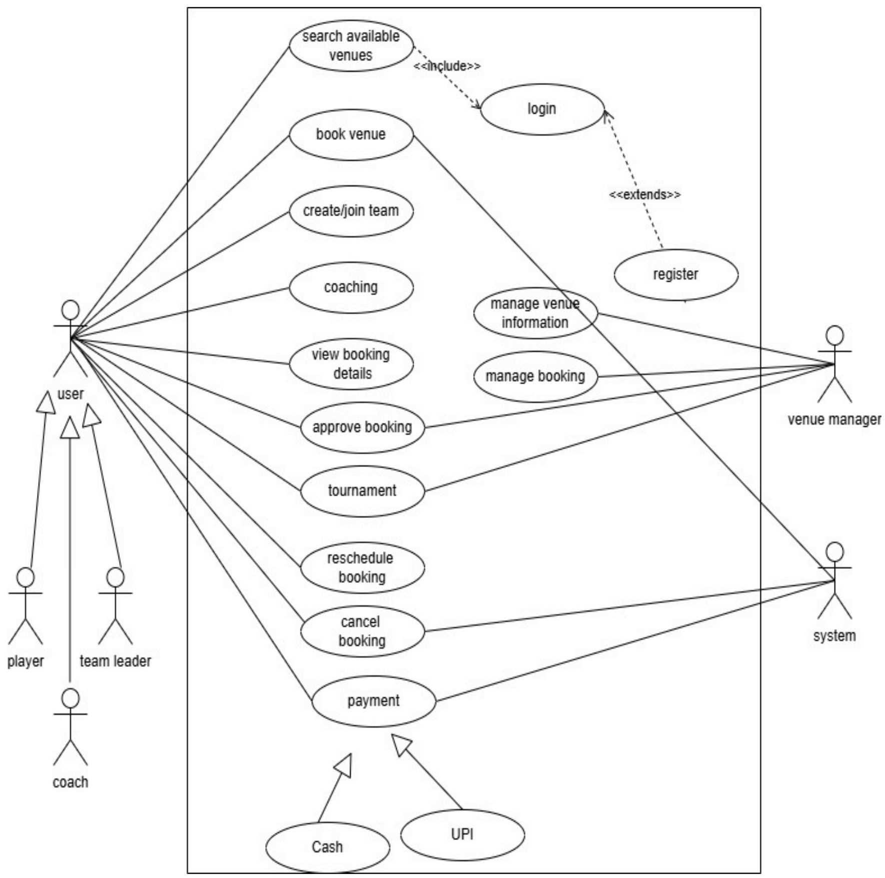
- **For Players**
  - Finding a Venue
  - Users can browse through different sports venues
  - They can see pictures, prices, and available time slots
  - They can filter venues by location, price, or sport type
  - Making a Booking
  - Users select their preferred date and time
  - They choose how long they want to book
  - They can see the total cost
  - They pay online using different payment methods
  - They receive a confirmation with booking details
- **For Venue Managers**
  - Managing Their Venue
  - Venue managers can add their sports facilities to the system
  - They can set prices and available time slots
  - They can upload photos and descriptions

    - They can see all bookings for their venue
    - Handling Bookings
    - They can view all upcoming bookings
    - They can confirm or cancel bookings
    - They can manage their venue's schedule
    - They can see payment status for each booking
- **For Administrators:**
    - System Management
    - Admins can manage all users and venues
    - They can handle any issues or disputes
    - They can generate reports about bookings and payments
    - They can help venue managers with their accounts

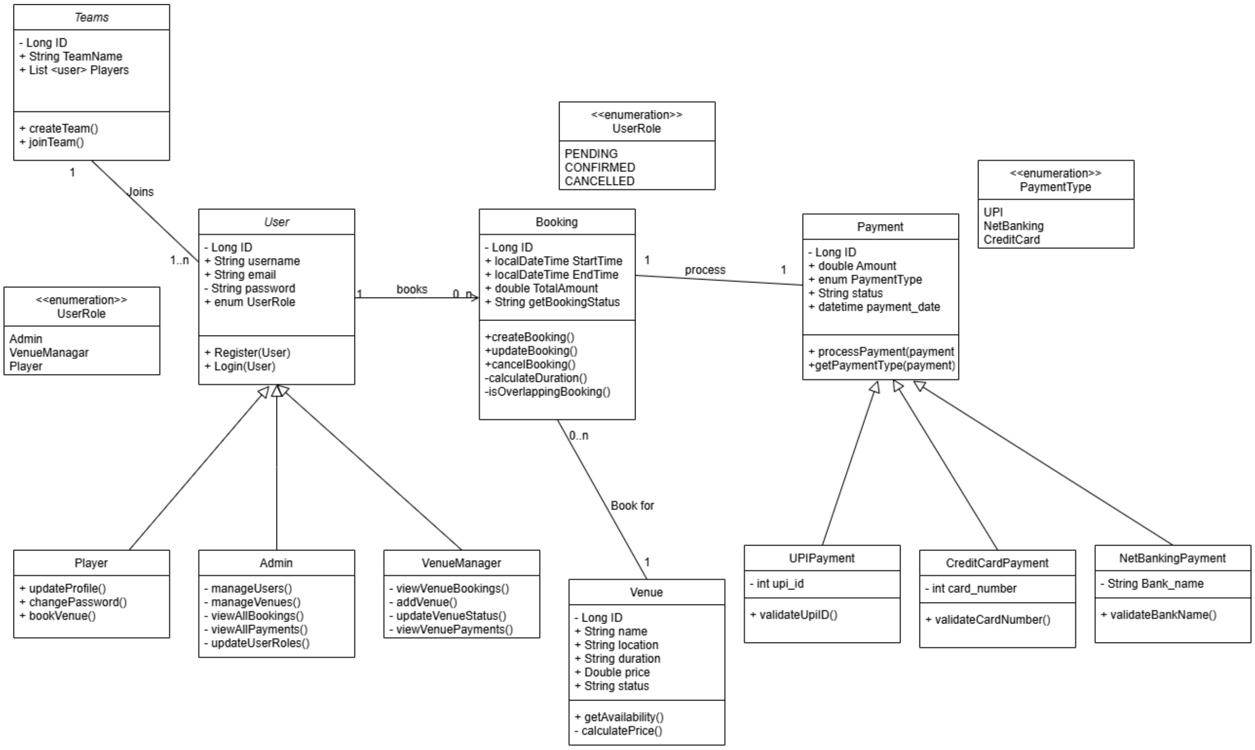
Key Features:

- **Easy Booking**
    - Simple calendar interface
    - Instant availability checking
    - Multiple payment options
- **Venue Management**
    - Easy schedule management
    - Booking tracking
    - Payment tracking
    - Customer management
- **Payment System**
    - Multiple payment methods
- **User Management**
    - User profiles
    - Booking history
    - Payment history
    - Account settings

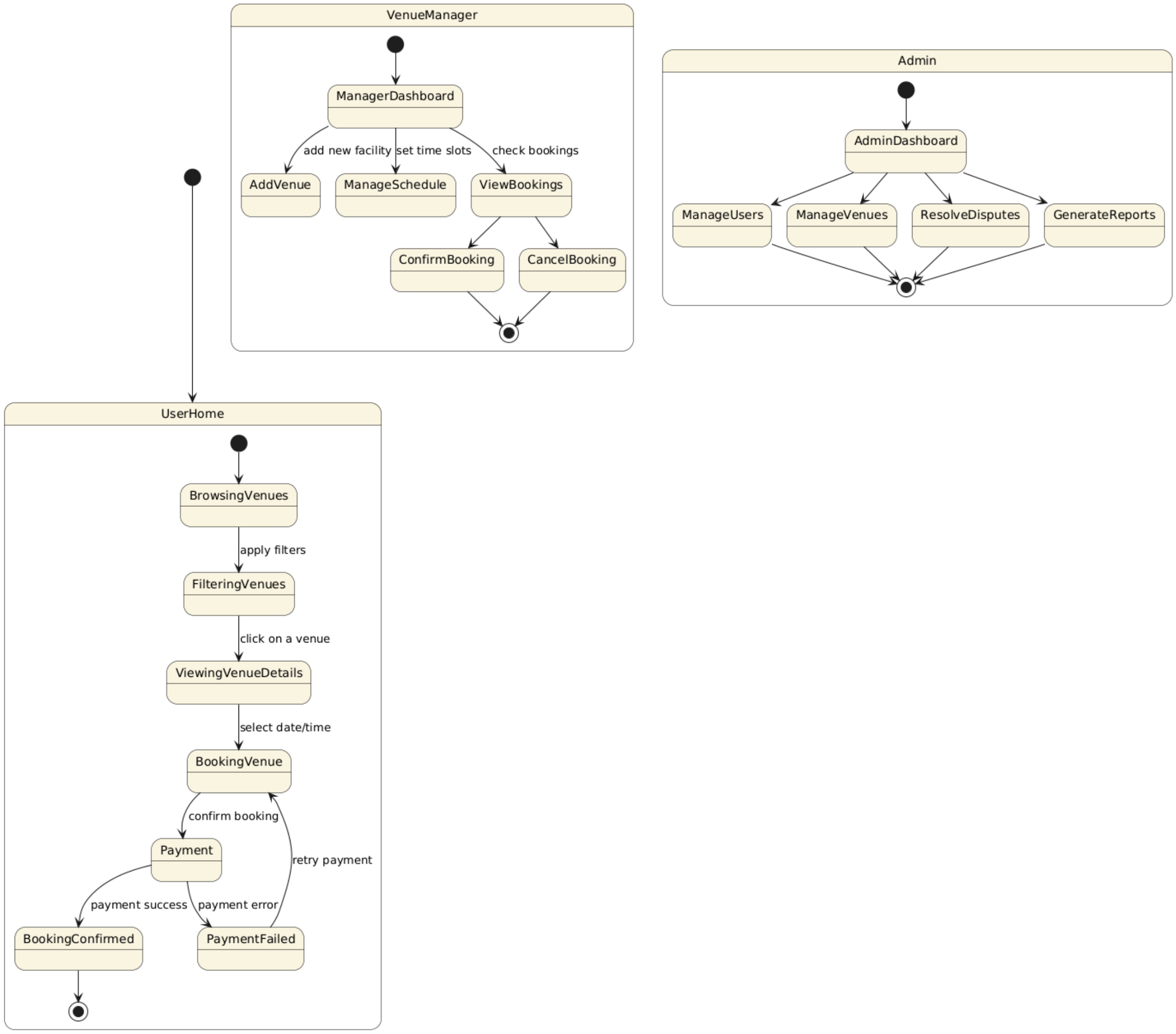
Use Case Diagram:



Class Diagram



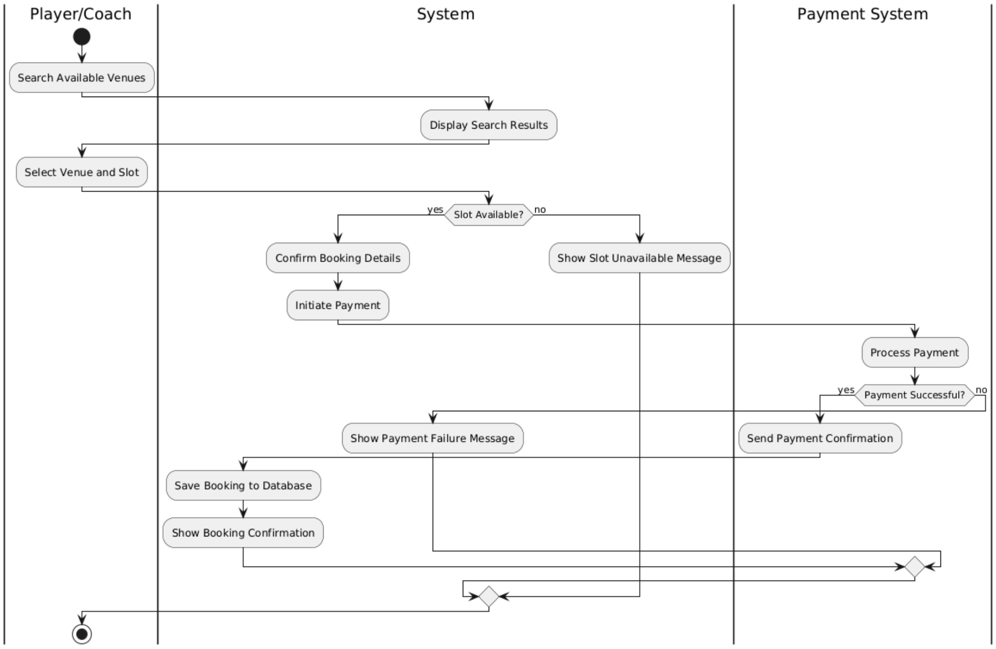
State Diagram



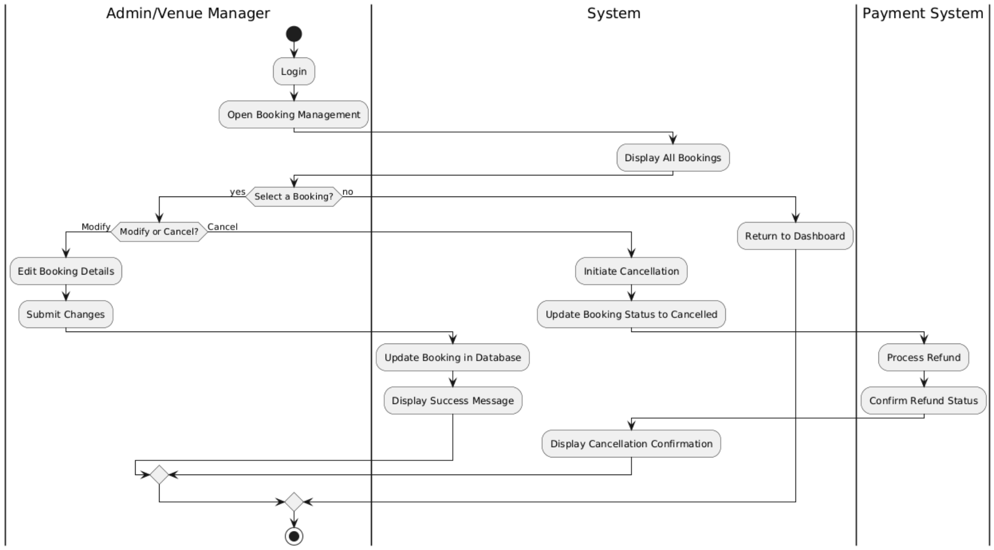
Activity Diagrams

1.Major Use Cases

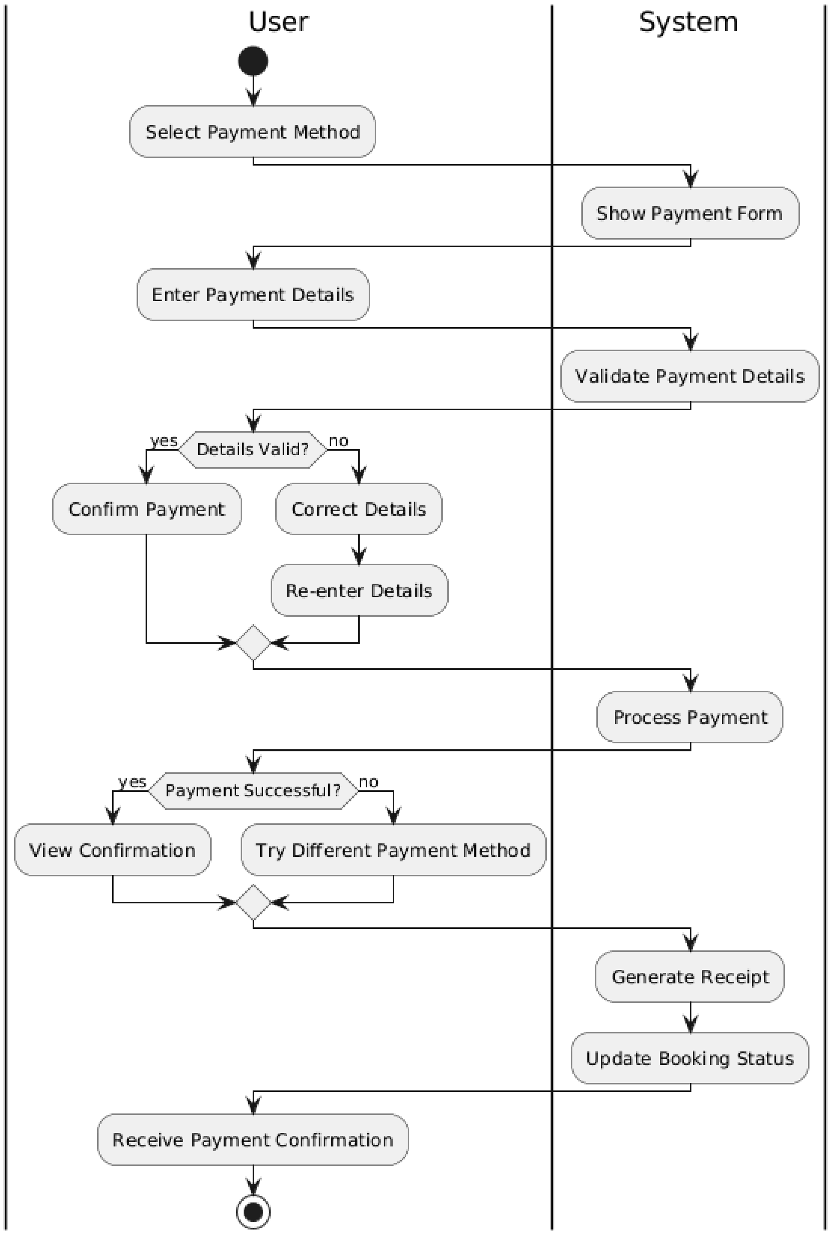
a)Book Venue



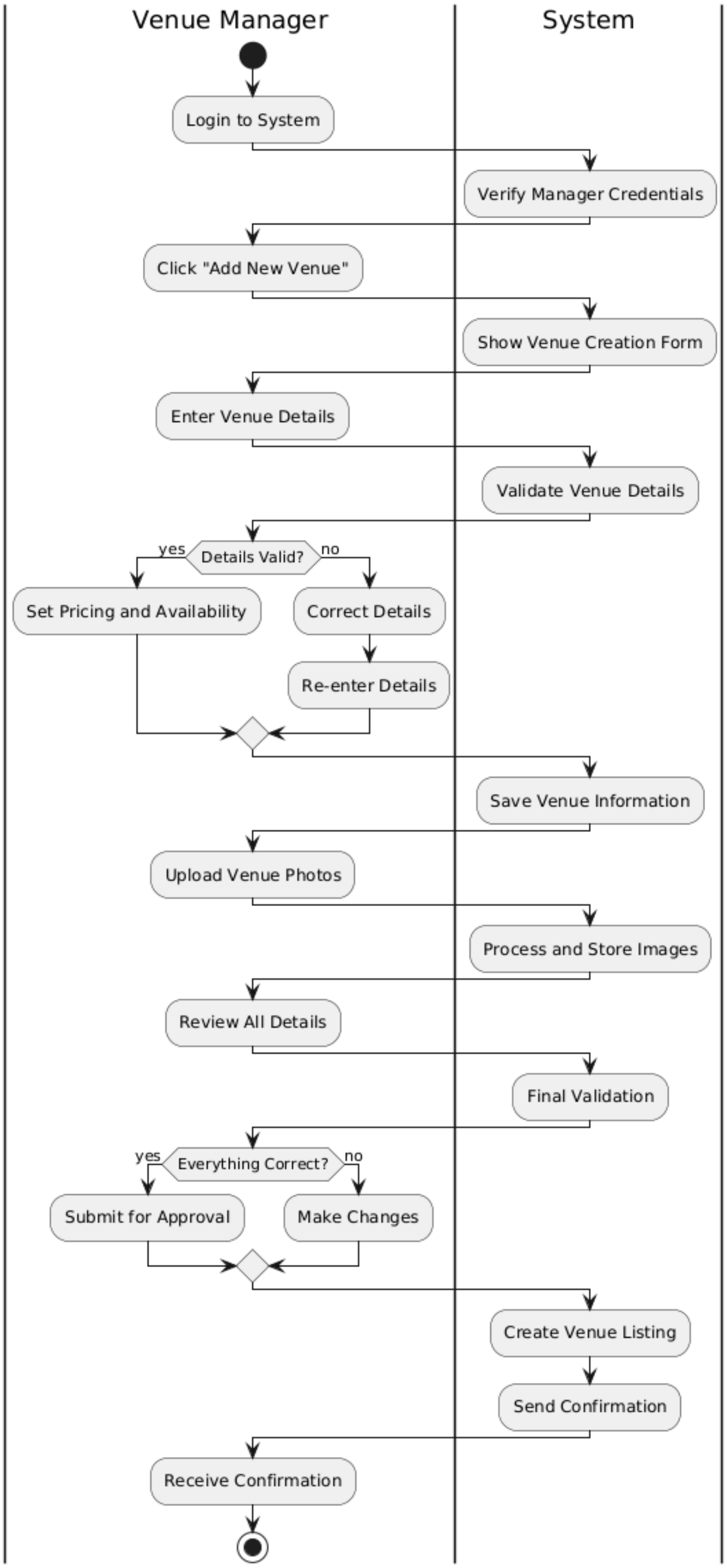
b)Manage Venues



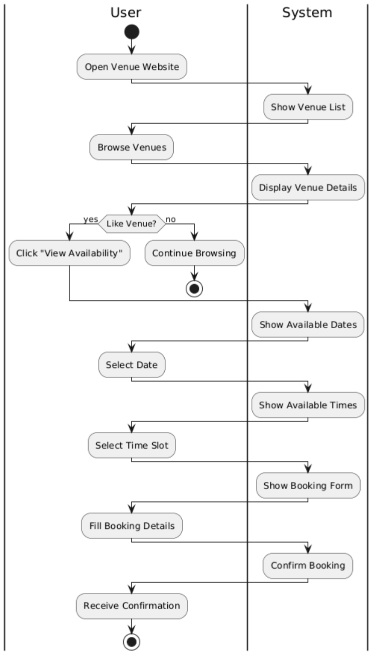
c)Process Payment



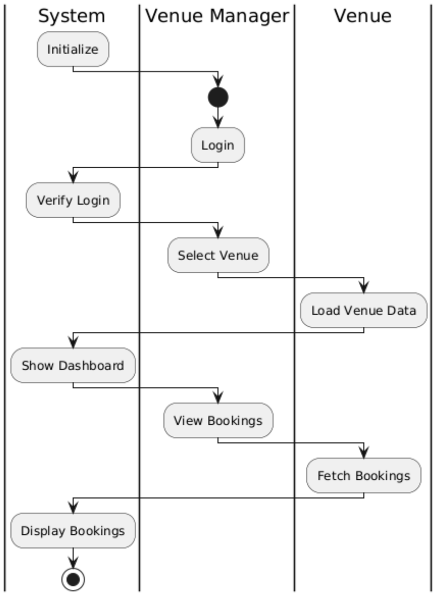
d)Add Venue:


Venue Manager
System
Login to System
Verify Manager Credentials
Click "Add New Venue"
Show Venue Creation Form
Enter Venue Details
Validate Venue Details
yes
Details Valid?
no
Set Pricing and Availability
Correct Details
Re-enter Details
Save Venue Information
Upload Venue Photos
Process and Store Images
Review All Details
Final Validation
yes
Everything Correct?
no
Submit for Approval
Make Changes
Create Venue Listing
Send Confirmation
Receive Confirmation

a)Browse Venues:



b)View Venue Bookings:



Architecture Patterns

Model View Controller Pattern:

1.Model Layer

Purpose: Manages data and business logic

Components:

- Entity classes (“User”, “Venue”, “Booking”, “Payment”)
- Data Transfer Objects (DTOs)
- Enums (“PaymentType”, “BookingStatus”, “UserRole”)

Responsibilities:

- Data representation
- Business rules
- Data validation

2. View Layer

Purpose: Handles user interface and presentation

Components:

- Frontend React components
- API response formatting
- Error message displays

Responsibilities:

- User interface rendering
- Data presentation
- User interaction handling

3. Controller Layer

Purpose: Manages user requests and responses

Components:

- REST controllers (“UserController”, “VenueController”, etc.)

- Request mapping
- Response handling

Responsibilities:

- Request processing
- Service coordination
- Response generation

4. Service Layer

Purpose: Handle business logic and centralize business rules

Components:

- UserService
- AuthService
- VenueService
- BookingService
- PaymentService
- AdminService

5. Repository Layer

Components:

-Repository interfaces

-JPA implementations

Responsibilities:

-Clean data access

-Database abstraction

-Consistent data operations

Design Principles

1.Single Responsibility

Every class has exactly one reason to change—controllers deal only with HTTP, services with business logic, repositories with persistence, builders with object creation, etc

2.Separation of Concerns

Cleanly split the presentation (controllers), application (services), data (repositories) and construction (builders/DTOs) layers so that each layer can evolve independently.

3.Dependency Inversion (Inversion of Control)

High‑level modules (controllers, services) depend on abstractions (interfaces) rather than concrete classes, and a framework‑driven injector wires in the implementations at runtime.

4.Open/Closed

New behavior can be added (new filters in the chain, extra service methods, additional builder steps) without modifying existing tested code—everything is open for extension but closed to alteration.

5.KISS (Keep It Simple, Stupid)

APIs, method signatures and configuration stay as straightforward as possible, minimizing needless complexity and making the code easy to read and maintain.

6.DRY (Don’t Repeat Yourself)

Common logic—object construction, response handling, error handling, JWT processing—is centralized in builders, base controllers, filters and utilities, avoiding duplication across the codebase.

Design Patterns

1.Builder

A fluent construction mechanism gathers all necessary properties for an object before instantiation, keeping creation logic in one place and preventing partially‑initialized entities.
In this project, a dedicated builder assembles venue objects step by step—guaranteeing every required field is set consistently before persistence.

2.Service-Layer

A distinct layer encapsulates business rules, transaction boundaries and security checks, mediating between controllers (or APIs) and data access to maintain a clear separation of concerns.
Here, the service tier orchestrates operations such as venue management, booking validation and payment processing without cluttering request-handling code.

3.Data-Transfer Object (DTO)

Simple, immutable data carriers ferry only the required fields across the network or between layers, insulating the internal domain model from direct exposure and easing validation.
In this codebase, DTOs encapsulate user credentials, signup details and payment information—exchanging only the necessary data at each API boundary.

4. Dependency Injection

Dependencies are declared rather than instantiated, letting a container wire components at runtime; this decouples classes, promotes modularity and makes unit testing straightforward.
All controllers, services, repositories and utilities are injected by the framework—allowing easy swapping of implementations and cleaner test setups.

5.Chain of Responsibility

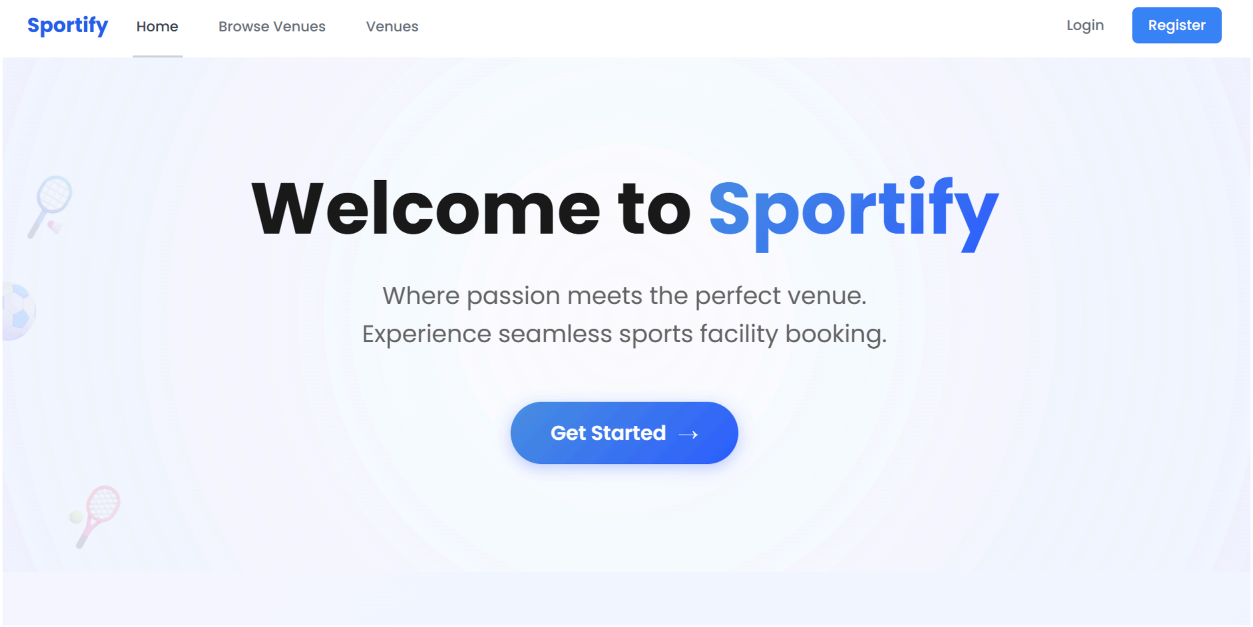
Incoming requests flow through a configurable pipeline of interceptors or filters, where each handler can process, transform, or halt the request, cleanly modularizing cross-cutting concerns such as authentication and logging.

The application applies filters in sequence—handling CORS, JWT validation and authorization checks before any business logic runs.

Github Link to CodeBase:

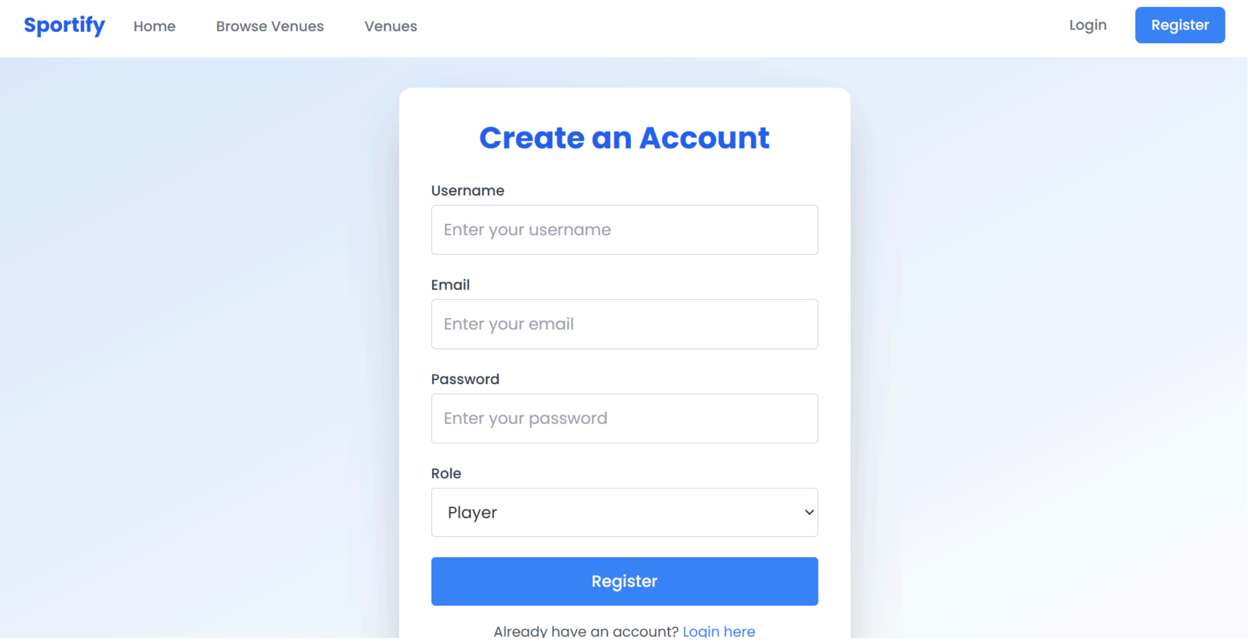
https://github.com/NehaMalage/sportify.git

Screenshots



Welcome to Sportify

Where passion meets the perfect venue.
Experience seamless sports facility booking.



Create an Account

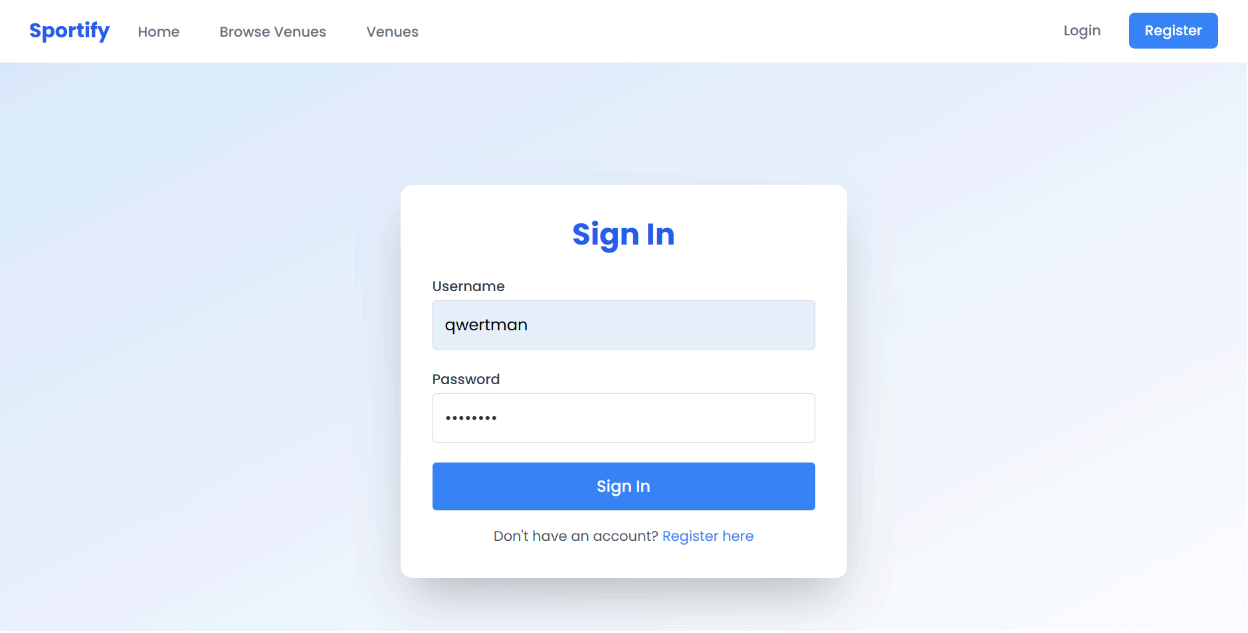
Username: Enter your username

Email: Enter your email

Password: Enter your password

Role: Player

Already have an account? Login here

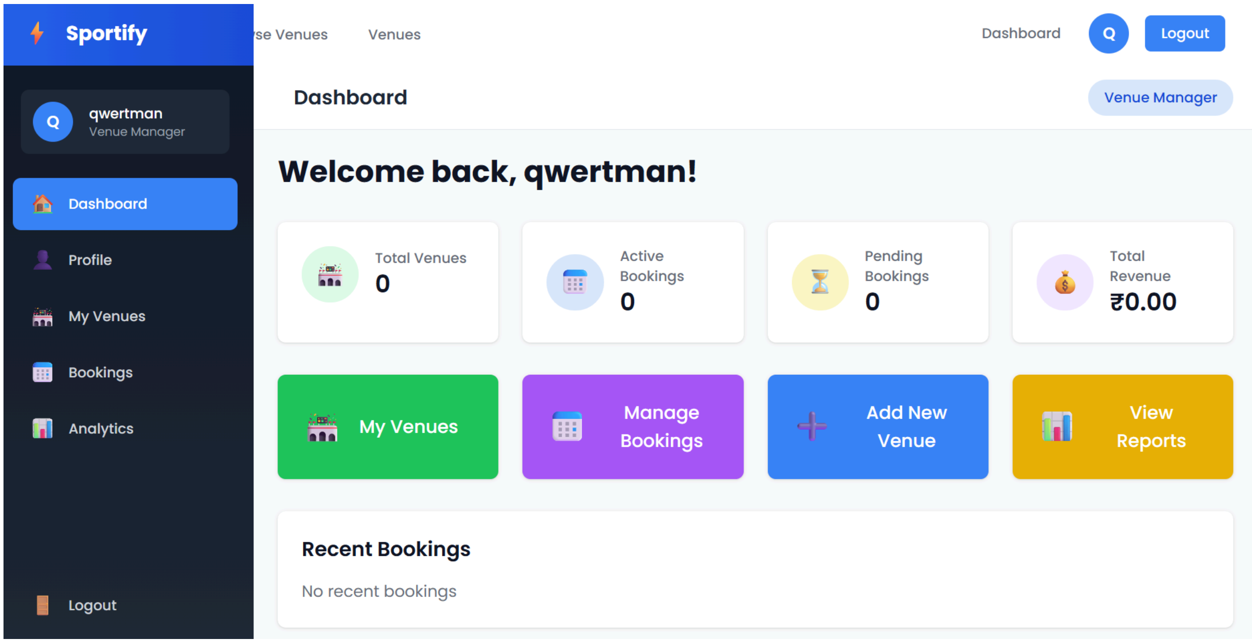


Sign In

Username: qwertman

Password: ••••••••

Don't have an account? Register here



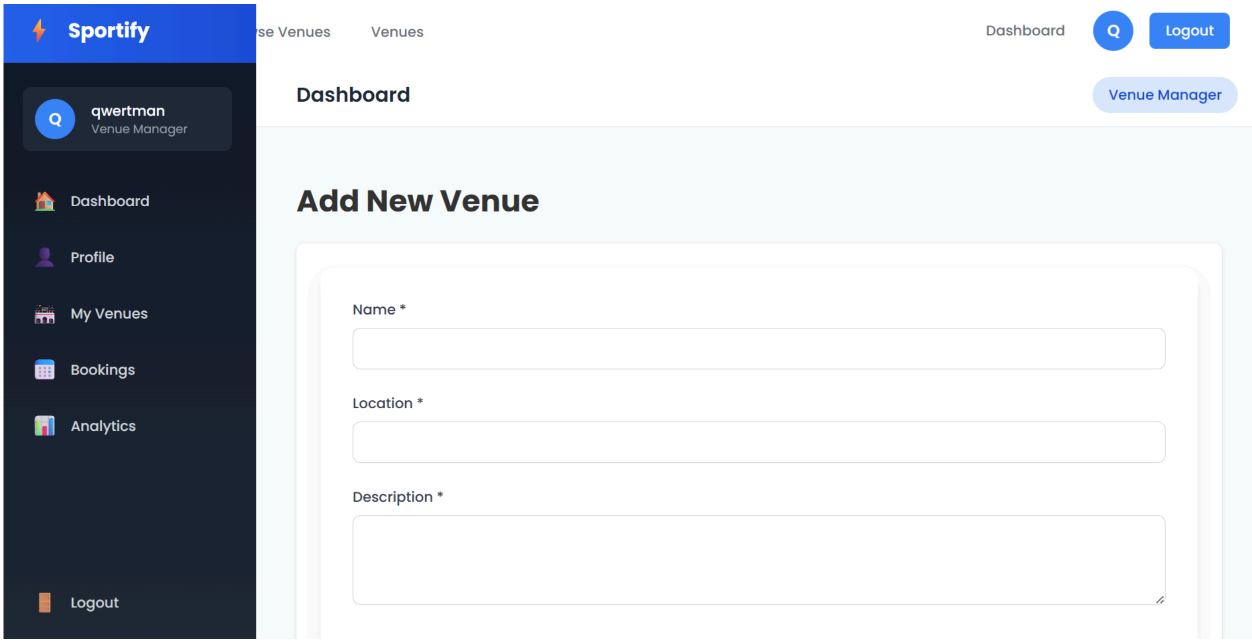
Dashboard

Welcome back, qwertman!

Total Venues	0
Active Bookings	0
Pending Bookings	0
Total Revenue	₹0.00

Recent Bookings

No recent bookings



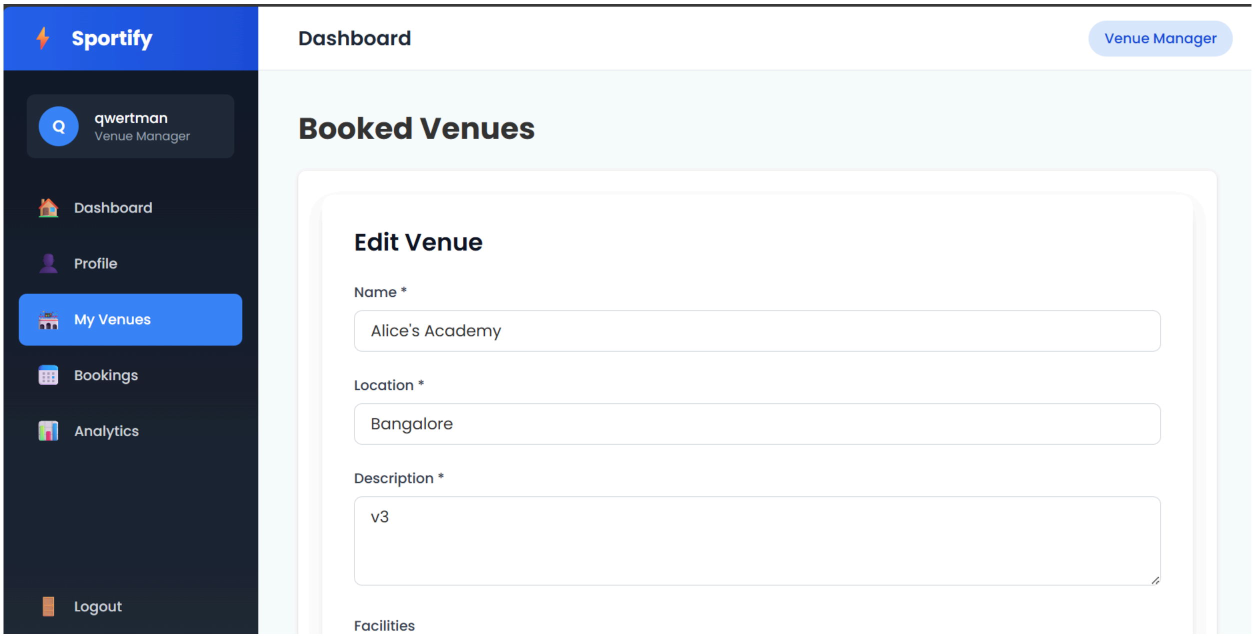
Dashboard

Add New Venue

Name *

Location *

Description *



Booked Venues

Edit Venue

Name *

Alice's Academy

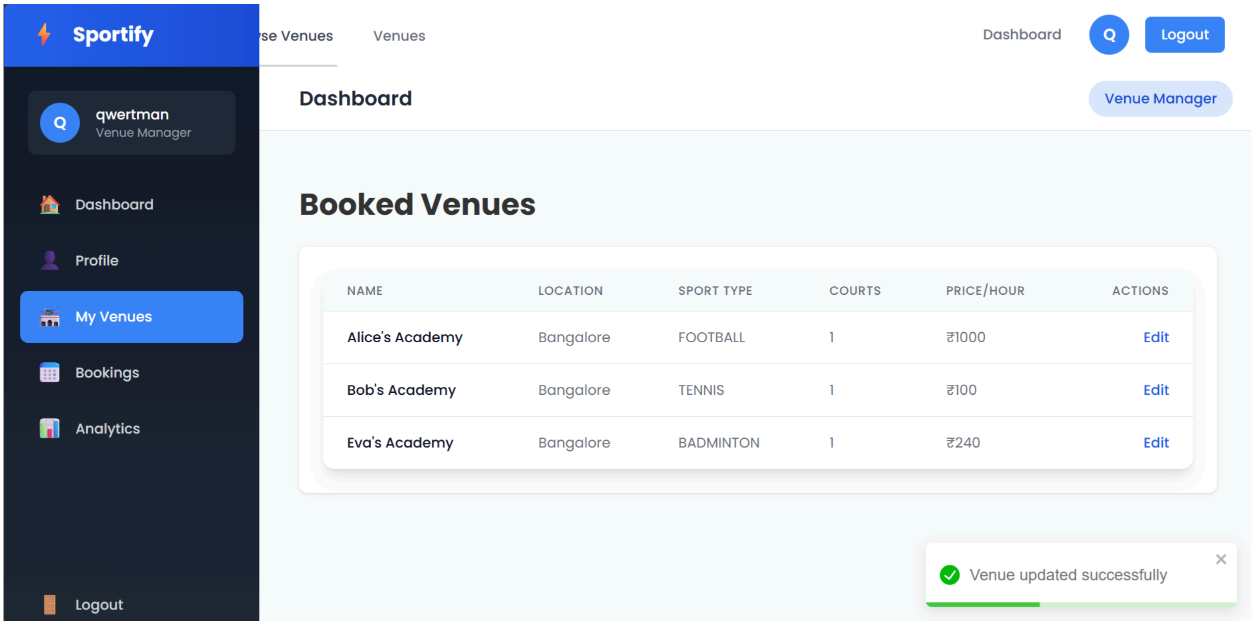
Location *

Bangalore

Description *

v3

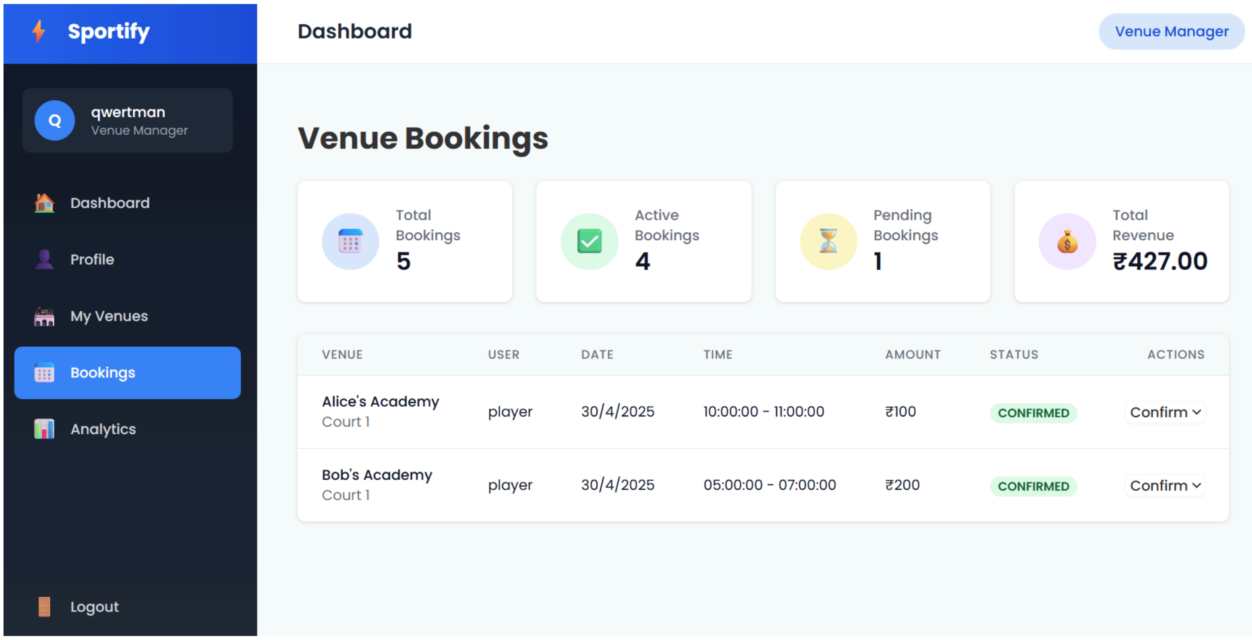
Facilities



Booked Venues

NAME	LOCATION	SPORT TYPE	COURTS	PRICE/HOUR	ACTIONS
Alice's Academy	Bangalore	FOOTBALL	1	₹1000	Edit
Bob's Academy	Bangalore	TENNIS	1	₹100	Edit
Eva's Academy	Bangalore	BADMINTON	1	₹240	Edit

Venue updated successfully



Venue Bookings

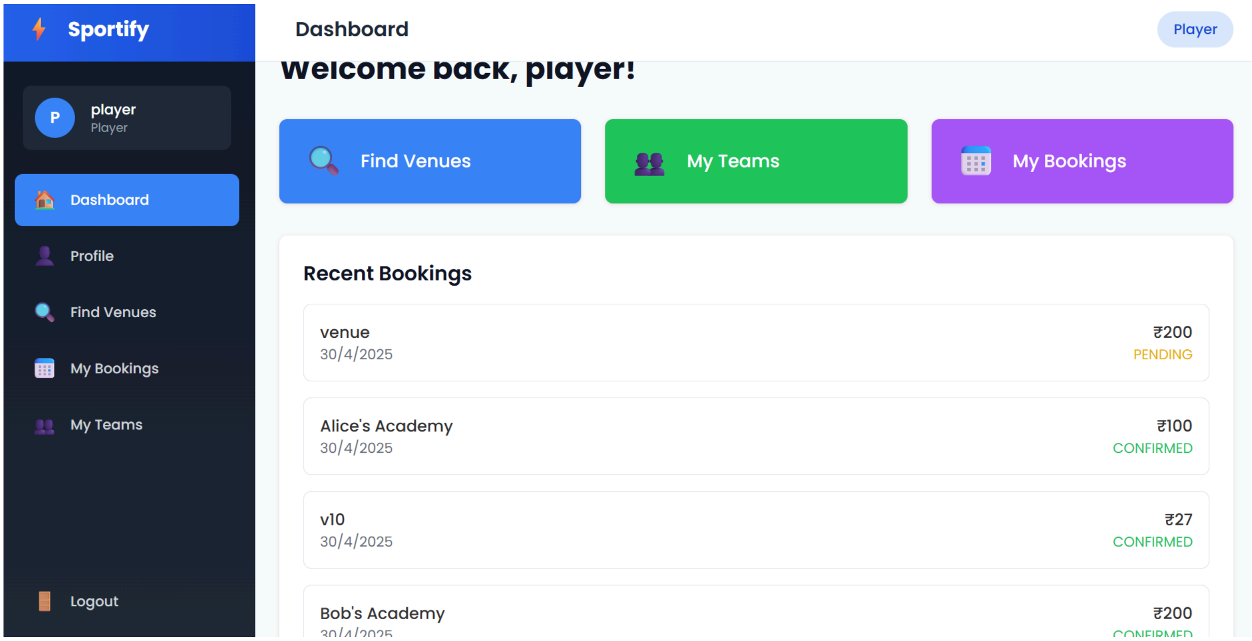
Total Bookings 5

Active Bookings 4

Pending Bookings 1

Total Revenue ₹427.00

VENUE	USER	DATE	TIME	AMOUNT	STATUS	ACTIONS
Alice's Academy Court 1	player	30/4/2025	10:00:00 - 11:00:00	₹100	CONFIRMED	Confirm
Bob's Academy Court 1	player	30/4/2025	05:00:00 - 07:00:00	₹200	CONFIRMED	Confirm



Welcome back, player!

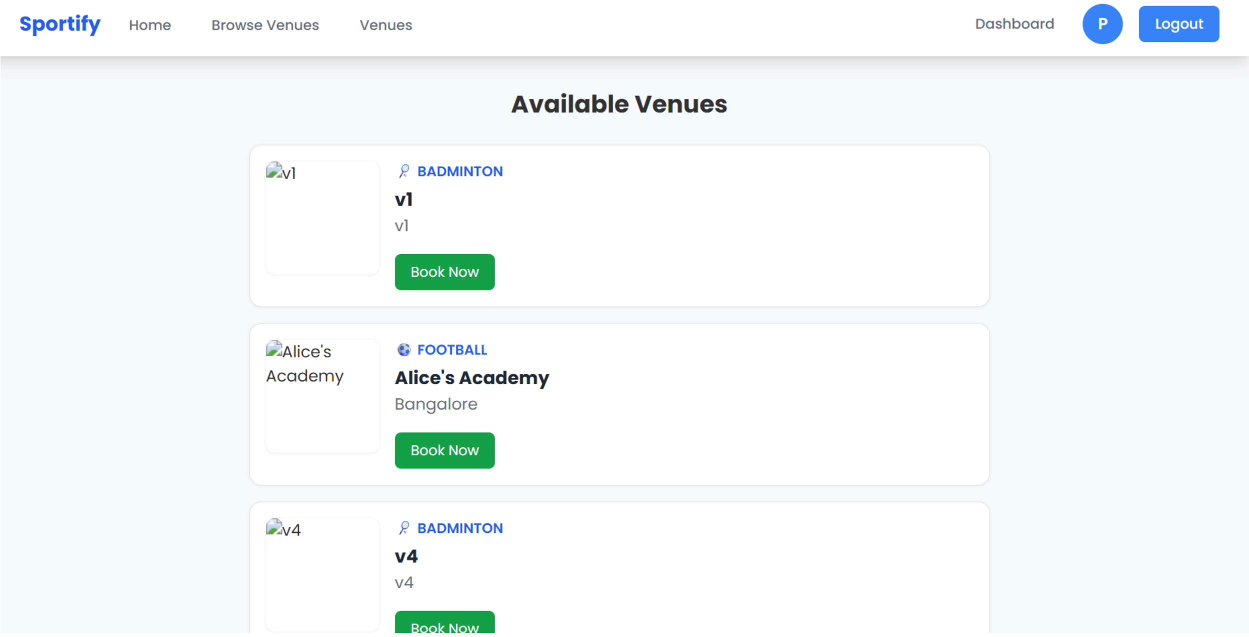
Recent Bookings

venue
30/4/2025
₹200
PENDING

Alice's Academy
30/4/2025
₹100
CONFIRMED

v10
30/4/2025
₹27
CONFIRMED

Bob's Academy
30/4/2025
₹200
CONFIRMED



Available Venues

BADMINTON

v1

v1

FOOTBALL

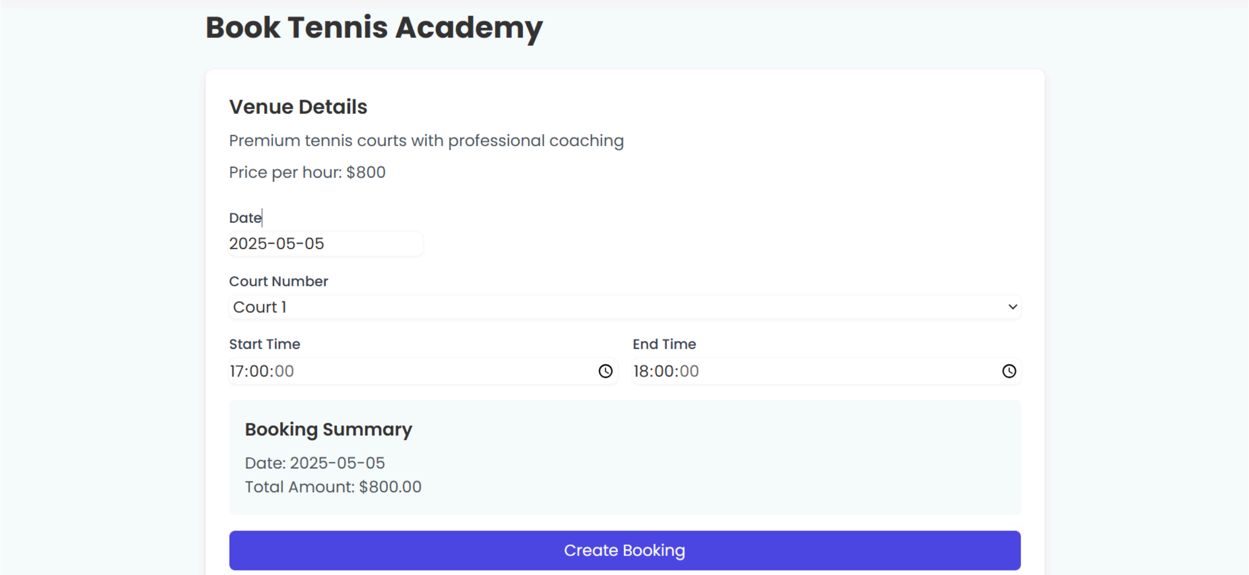
Alice's Academy

Bangalore

BADMINTON

v4

v4



Book Tennis Academy

Venue Details

Premium tennis courts with professional coaching

Price per hour: $800

Date
2025-05-05

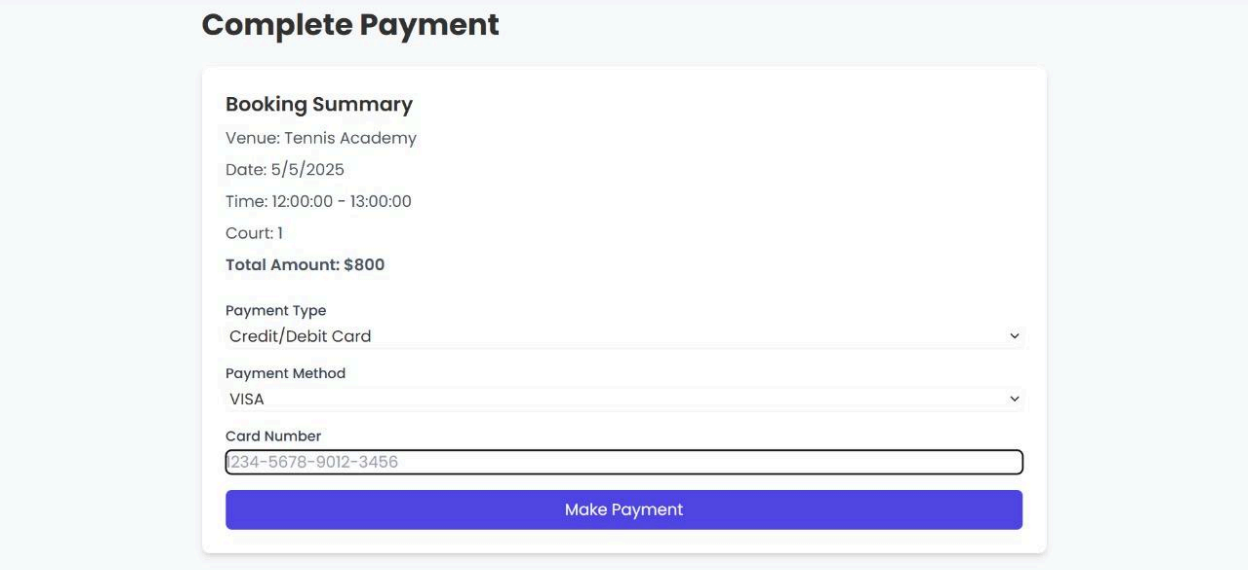
Court Number
Court 1

Start Time
17:00:00

End Time
18:00:00

Booking Summary

Date: 2025-05-05
Total Amount: $800.00



Complete Payment

Booking Summary

Venue: Tennis Academy

Date: 5/5/2025

Time: 12:00:00 - 13:00:00

Court: 1

Total Amount: $800

Payment Type

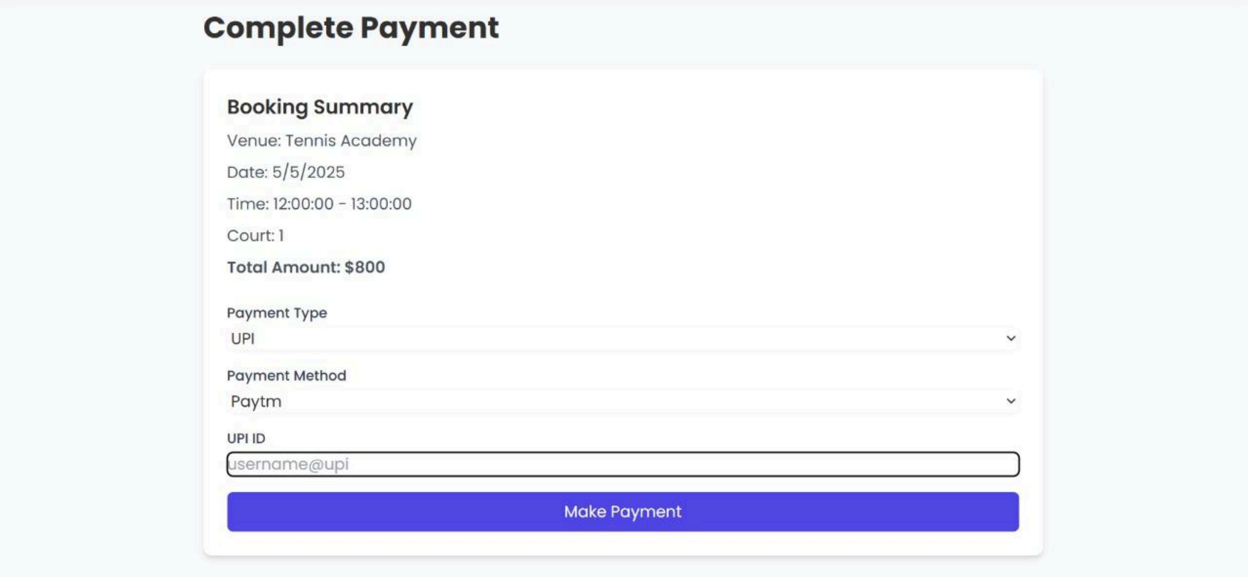
Credit/Debit Card

Payment Method

VISA

Card Number

1234-5678-9012-3456



Complete Payment

Booking Summary

Venue: Tennis Academy

Date: 5/5/2025

Time: 12:00:00 - 13:00:00

Court: 1

Total Amount: $800

Payment Type

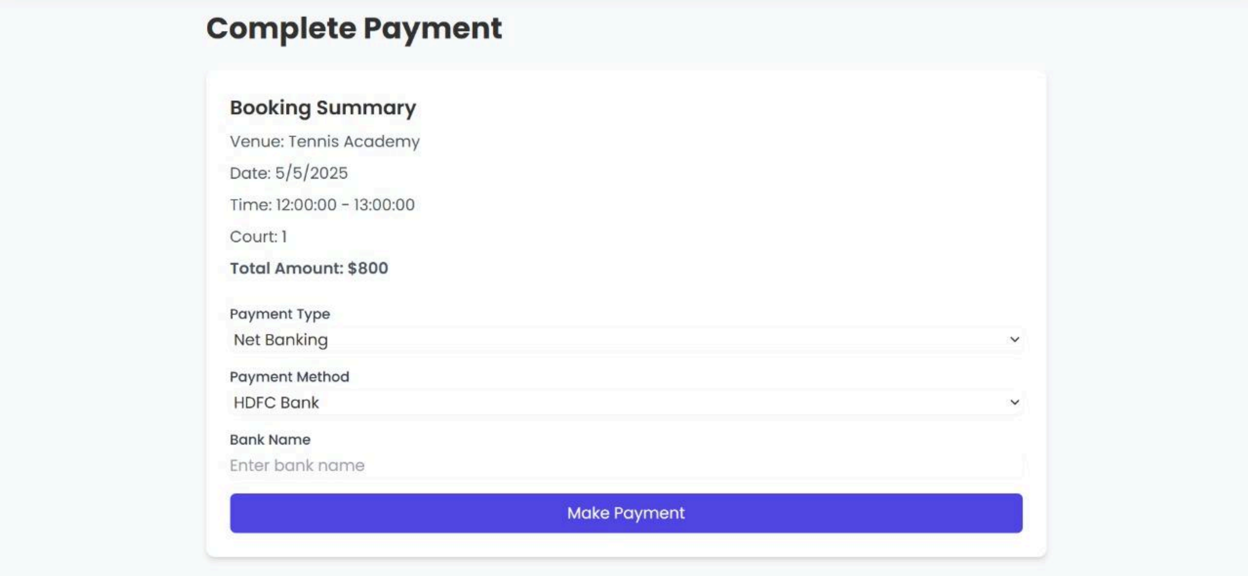
UPI

Payment Method

Paytm

UPI ID

username@upi



Complete Payment

Booking Summary

Venue: Tennis Academy

Date: 5/5/2025

Time: 12:00:00 - 13:00:00

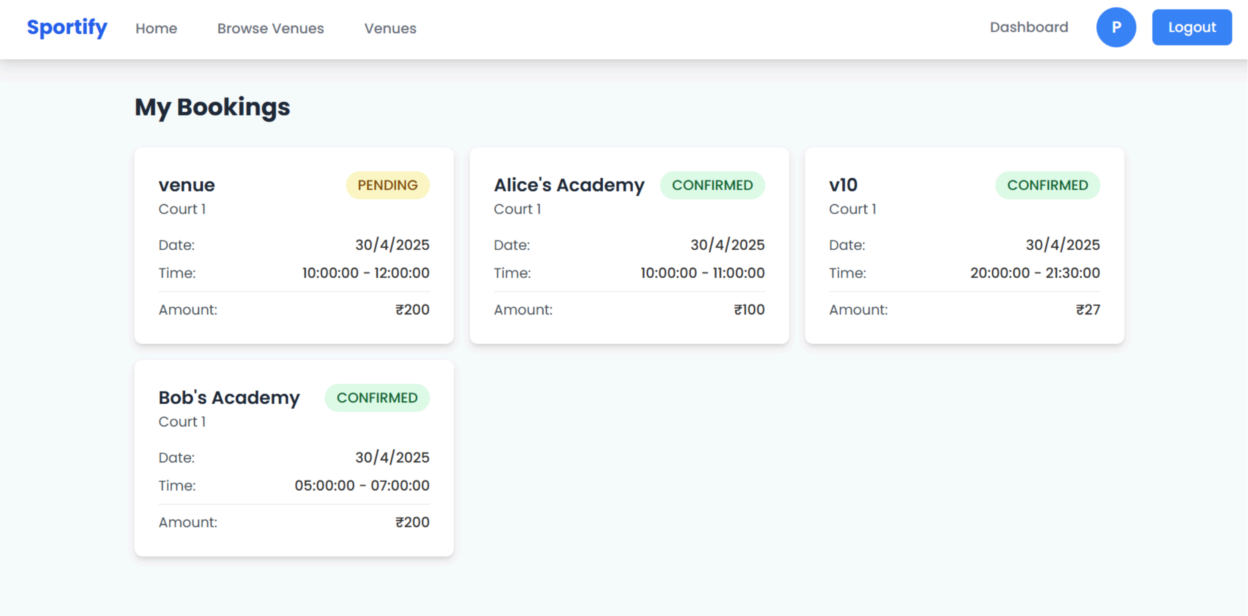
Court: 1

Total Amount: $800

Payment Type: Net Banking

Payment Method: HDFC Bank

Bank Name: Enter bank name



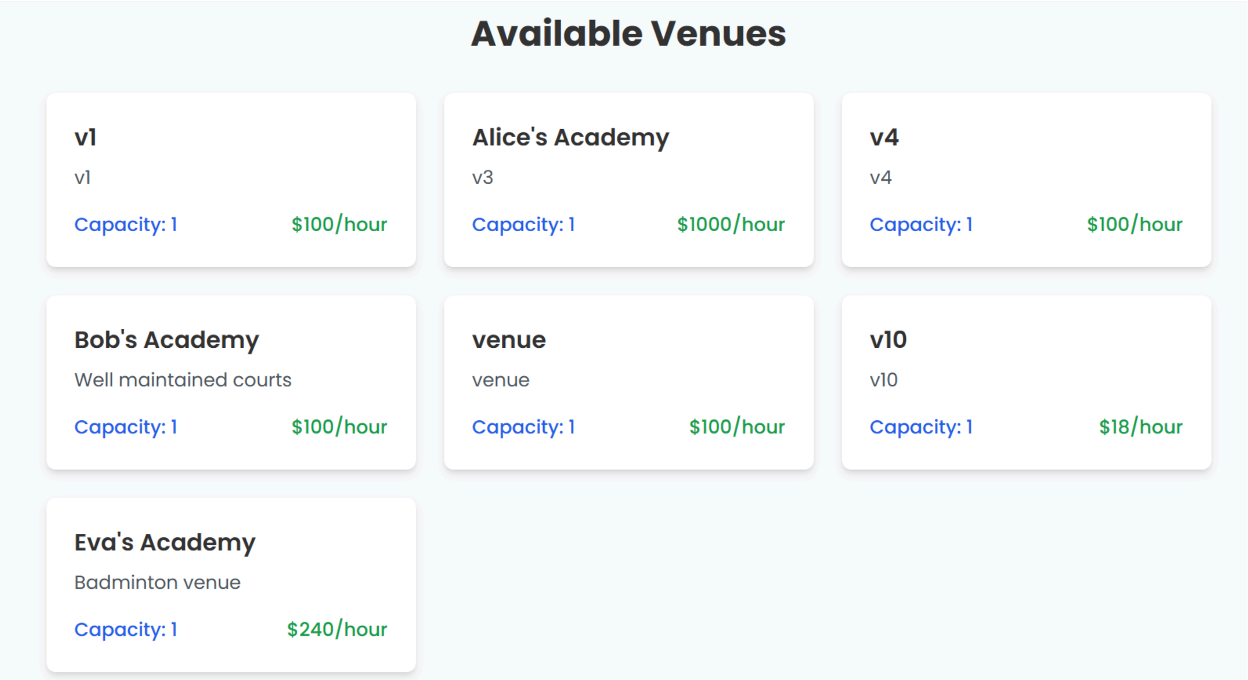
My Bookings

venue PENDING
Court 1
Date: 30/4/2025
Time: 10:00:00 - 12:00:00
Amount: ₹200

Alice's Academy CONFIRMED
Court 1
Date: 30/4/2025
Time: 10:00:00 - 11:00:00
Amount: ₹100

v10 CONFIRMED
Court 1
Date: 30/4/2025
Time: 20:00:00 - 21:30:00
Amount: ₹27

Bob's Academy CONFIRMED
Court 1
Date: 30/4/2025
Time: 05:00:00 - 07:00:00
Amount: ₹200



Available Venues

v1
v1
Capacity: 1 $100/hour

Alice's Academy
v3
Capacity: 1 $1000/hour

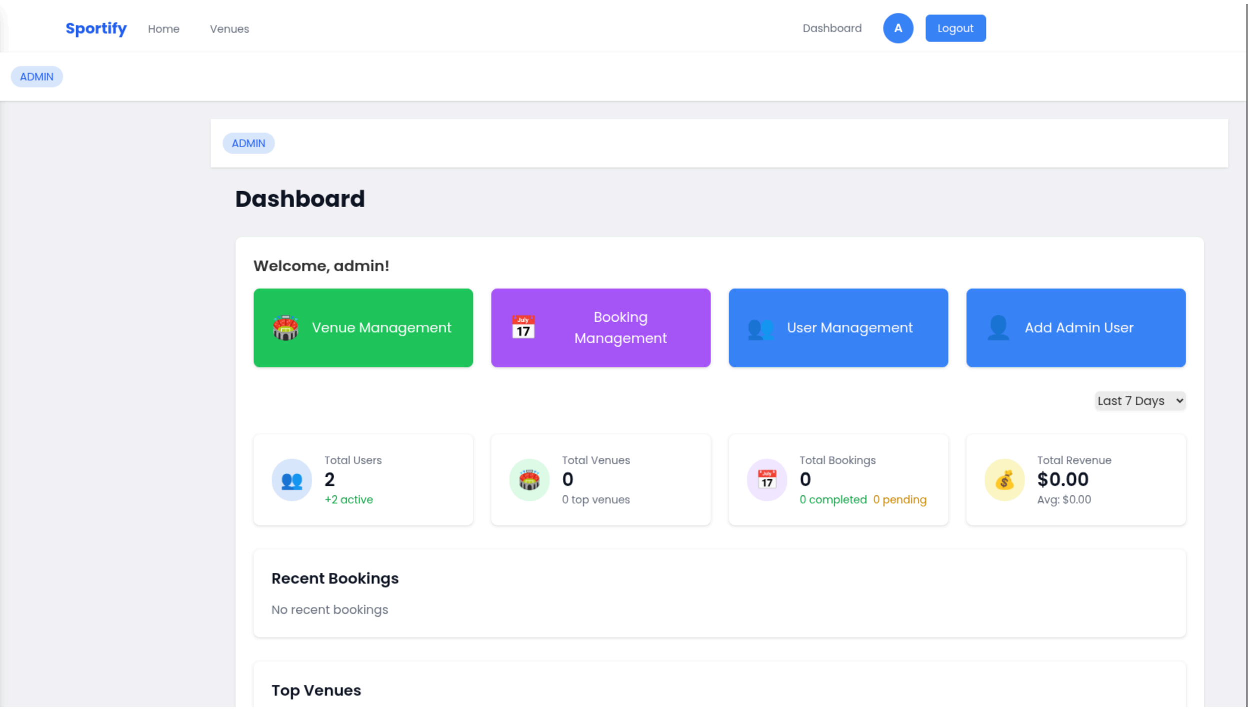
v4
v4
Capacity: 1 $100/hour

Bob's Academy
Well maintained courts
Capacity: 1 $100/hour

venue
venue
Capacity: 1 $100/hour

v10
v10
Capacity: 1 $18/hour

Eva's Academy
Badminton venue
Capacity: 1 $240/hour



Dashboard

Welcome, admin!

Total Users
2
+2 active

Total Venues
0
0 top venues

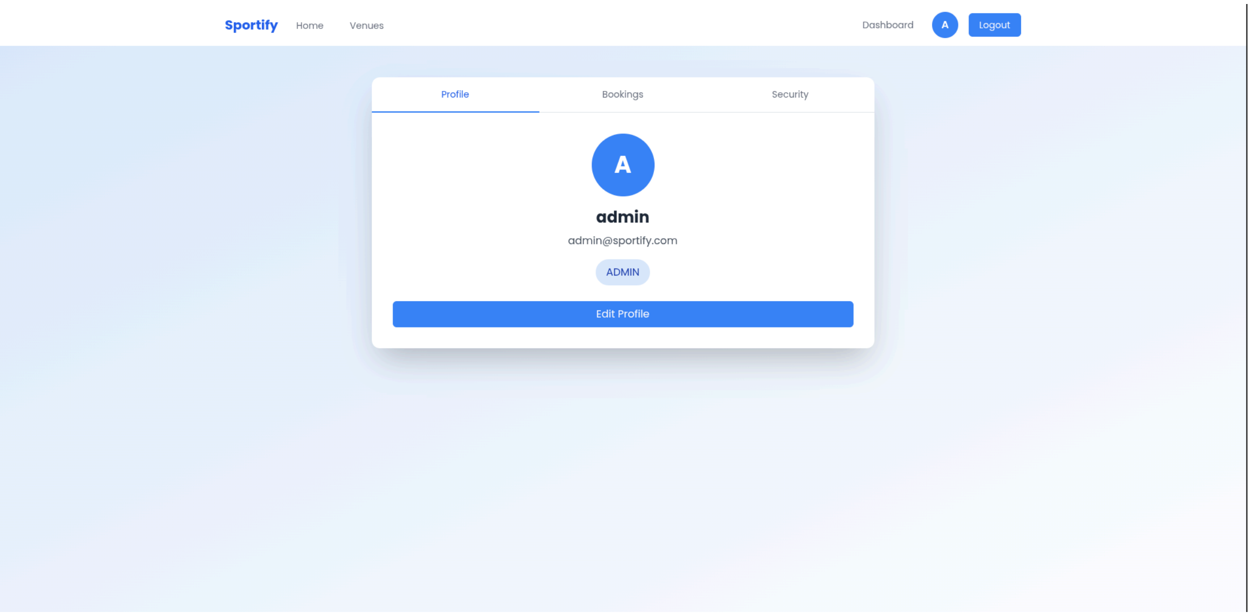
Total Bookings
0
0 completed 0 pending

Total Revenue
$0.00
Avg: $0.00

Recent Bookings

No recent bookings

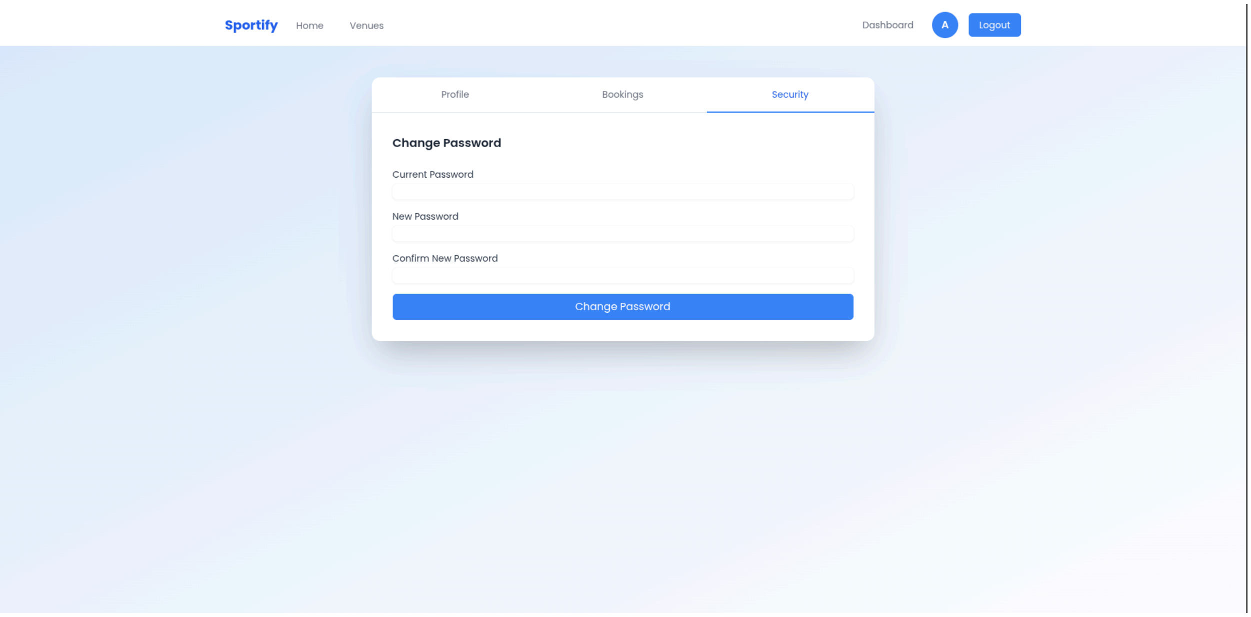
Top Venues



admin

admin@sportify.com

ADMIN

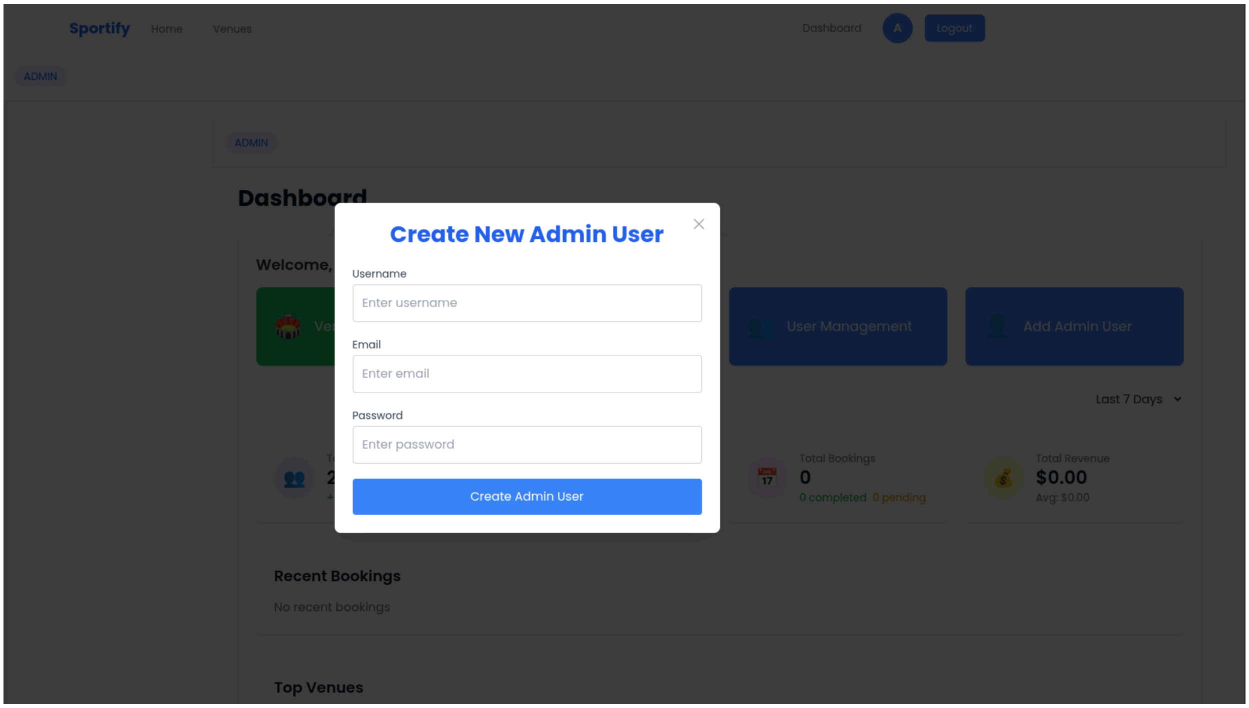


Change Password

Current Password

New Password

Confirm New Password



Dashboard

Welcome,

Create New Admin User

Username: Enter username

Email: Enter email

Password: Enter password

Total Bookings 0
0 completed 0 pending

Total Revenue $0.00
Avg: $0.00

Recent Bookings

No recent bookings

Top Venues

Individual contributions of the team members:

Team Member	Module Worked On
Pranav Muppuru	Venue manager functionalities, Booking analytics, Team formation
Nikita Nair	Booking management and Payment system
Neha V Malage	Venue manager functionalities, Updating booked venue status, frontend
Poorna Pragnya.H	Authentication, User Management, Admin Functionality