FUTSALKTM: Online Futsal Booking Web App Documentation

Developed By: Utshav Gopali
Submitted to: Rohit Shrestha

Acknowledgement

We would like to express our deepest gratitude to all those who contributed to the successful completion of the **FUTSALKTM** web application project. This work would not have been possible without the support, guidance, and encouragement of numerous individuals and institutions.

First and foremost, we are profoundly thankful to our project supervisor and academic mentors for their continuous guidance, constructive feedback, and expert insights. Their mentorship has been instrumental in shaping the direction and quality of this project, from ideation to execution. Their encouragement during critical stages helped us maintain clarity, purpose, and persistence.

We extend our sincere appreciation to our fellow classmates, peers, and collaborators who provided valuable suggestions, exchanged ideas, and helped us troubleshoot technical issues. Their collaborative spirit, mutual support, and shared enthusiasm made this project an enriching and intellectually rewarding experience.

We would also like to acknowledge the broader developer community and open-source contributors whose publicly available tools, libraries, tutorials, and best practices were crucial in our learning journey. Their dedication to fostering open knowledge played a significant role in helping us overcome challenges in both the frontend and backend aspects of the project.

Special thanks go to the creators and maintainers of key technologies used in this project—including **React.js**, **Vite**, **Node.js**, and supporting libraries—whose robust ecosystems and documentation significantly streamlined our development process.

Finally, we would like to thank our families and friends for their unwavering moral support, encouragement, and patience throughout the duration of this project. Their belief in our capabilities kept us motivated during long development hours and tight deadlines.

This project is a reflection of the collective efforts, cooperation, and goodwill of all those mentioned above. We remain sincerely grateful for their role in making FUTSALKTM a successful and meaningful academic achievement.

Abstract

FUTSALKTM is a web-based futsal booking platform developed to streamline and digitize the process of reserving futsal courts in Kathmandu. The application offers a modern, responsive, and user-friendly interface that caters to both casual players and futsal venue operators. Designed with a mobile-first approach and optimized for accessibility, FUTSALKTM bridges the gap between sports enthusiasts and futsal service providers, providing a centralized platform for seamless court discovery and booking.

The platform allows users to explore a curated list of futsal courts, view available time slots, and make bookings without the need for phone calls or physical visits. Users can register and log in through an intuitive modal-based authentication system. Once authenticated, players can manage their bookings, view profile information, and navigate through different sections of the site with ease. Though currently frontend-only, the system is built with extendable architecture that can accommodate backend services like Firebase or Node.js in the future.

FUTSALKTM is built using **Vite + React** (JavaScript) for its rapid development experience, modular component structure, and efficient performance. The codebase is clean, scalable, and follows modern frontend development practices, making it ideal for academic and portfolio purposes. Data such as court names, time slots, and user bookings are handled through a simulated mock data file, emulating real-time interactions without a database.

The goal of this project is to create a practical solution tailored for the Kathmandu region while providing students and developers an opportunity to explore modern web technologies in a real-world context. FUTSALKTM not only facilitates a more efficient way to book futsal games but also serves as a foundation for future enhancements such as user history tracking, payment integration, and administrative tools.

Table of Contents

Introduction

In today's digital age, the integration of technology into daily life has revolutionized how people interact, communicate, and access services. From online shopping and food delivery to digital payments and health consultations, the modern consumer expects instant, seamless access to services at their fingertips. However, certain sectors — particularly local recreational sports — are still catching up in adopting digital solutions. One such example in Nepal is the **futsal booking system**, which is often still managed through manual phone calls, walk-ins, or spreadsheets. This inefficiency creates a disconnect between the demand for futsal facilities and the ease of access required by players, especially in busy cities like **Kathmandu**.

To bridge this gap, the project titled **FUTSALKTM** was conceptualized and developed as a web-based solution for users to explore, select, and book futsal grounds online. This platform specifically targets **futsal venues in Kathmandu**, a city with a growing population of youth and working professionals seeking accessible sports and fitness options. FUTSALKTM eliminates the need for traditional booking methods by offering a responsive, single-page web application that enables users to browse available courts, select preferred time slots, and confirm bookings — all from the comfort of their homes or mobile devices.

FUTSALKTM has been designed with a **user-centric approach**, focusing on accessibility, responsiveness, and ease of navigation. Built with **Vite and React (JavaScript)**, the application emphasizes component reusability, performance optimization, and rapid development. The system uses mock data to simulate real-world bookings and availability, preparing the project for future integration with backend technologies like **Firebase**, **Node.js**, or **MongoDB**. The frontend is carefully structured with modular components such as HomePage, BookingPage, Profile, AuthModal, and Navigation, all of which contribute to a clean and intuitive user experience.

From an academic and development standpoint, this project serves a dual purpose. First, it provides a **real-world software solution** tailored for a specific local market, demonstrating how modern web technologies can solve practical problems. Second, it offers a **learning platform** for understanding essential concepts of web development, such as state management, component-based architecture, routing, user interaction handling, responsive design, and scalability planning.

This documentation outlines every aspect of the project, starting from the folder structure and core technologies to detailed explanations of each React component. It also covers the development process, challenges encountered, user flows, and potential future enhancements,

including payment gateway integration, real-time availability updates, and administrative dashboards. In doing so, the document aims to not only explain how FUTSALKTM was built but also to inspire future iterations and expansions of the idea.

Ultimately, FUTSALKTM stands as a showcase of how web applications can enhance everyday life by providing simple, accessible solutions to routine problems. It reflects the rising demand for digital transformation in local services and represents the skills and creativity of aspiring developers contributing to their communities through innovative technology.

Aim

The primary aim of the **FUTSALKTM** project is to design and develop a user-friendly, efficient, and modern web-based platform that allows users in **Kathmandu** to **book futsal grounds online**. This platform serves as a digital bridge between futsal facility providers and players by simplifying the booking process, reducing manual communication, and ensuring real-time availability tracking.

Through this system, FUTSALKTM seeks to:

- Eliminate the traditional hassle of calling or visiting futsal centers to book time slots.
- Provide a centralized online solution where users can view available courts, book preferred time slots, and manage their profiles.
- Enable futsal businesses to manage their schedules, handle customer bookings efficiently, and improve overall service quality.

By integrating **React (Vite)** and modern frontend technologies, the project also aims to demonstrate technical competence in developing a complete, scalable single-page web application suitable for real-world deployment.

Objectives

The primary objective of the **FUTSALKTM** project is to **design and develop an intuitive, responsive, and efficient web application** that facilitates **online futsal booking in Kathmandu**. This application aims to replace the traditional manual booking methods with a seamless, digital-first approach that benefits both futsal players and venue operators.

To achieve this, the project is guided by the following specific objectives:

1. Simplify the Futsal Booking Process

- Provide a **user-friendly interface** where users can browse available futsal courts and book their preferred time slots without needing to call or visit the venues.

- Allow users to **view real-time availability**, ensuring transparent and conflict-free scheduling.

2. Design a Responsive and Accessible User Interface

- Build a web application that is **responsive across various devices** (smartphones, tablets, desktops).
- Ensure accessibility through proper layout, navigation, and color contrast to cater to users of different abilities and technical backgrounds.

3. Implement Secure Authentication Workflow

- Provide an **authentication system** through modal windows (login/register), allowing users to securely sign in and manage their bookings.
- Simulate login flows with the possibility to integrate real-world authentication (e.g., Firebase) in future versions.

4. Build with Modern Development Tools and Practices

- Use **Vite** for fast build times and development performance.
- Develop using **React (JavaScript)** to build a **component-based architecture**, promoting modularity, reusability, and scalability.

5. Maintain a Clean and Scalable Project Structure

- Organize the project into a meaningful folder structure (components, data, src, etc.) that supports further development and maintenance.
- Ensure **clean code** practices, including reusable components, clear state management, and consistent naming conventions.

6. Provide a Smooth User Experience (UX)

- Allow smooth navigation between different sections of the site: Home, Booking Page, Contact Page, Courts List, and Profile Page.
- Simulate booking confirmations and form validations to give users a realistic interaction flow.

7. Prepare for Backend Integration

- Structure the app to **easily integrate backend services** such as:
  - Firebase for authentication and storage
  - MongoDB for booking records
  - Node.js/Express for APIs

8. Address Local Context and Relevance

- Target futsal courts specifically in the **Kathmandu Valley**, understanding local user needs, habits, and expectations.
- Make the solution localized, simple, and relevant to the community's current digital infrastructure.

9. Enable Future Enhancements and Expansion

- Lay the groundwork for advanced features such as:
  - Admin dashboards
  - Booking history
  - Payment gateway integration (eSewa, Khalti)
  - Real-time updates
  - Email/SMS confirmations
  - Calendar view of availability

10. Contribute to Developer Learning and Practice

- Apply and showcase knowledge in:
  - React hooks (useState, useEffect)
  - Routing and navigation
  - Component communication via props
  - Mock data usage and frontend-only simulations
- Serve as a capstone or portfolio project to demonstrate technical skills and problem-solving ability.

Scope

The scope of the **FUTSALKTM** project defines the boundaries, functionalities, and technological implementations involved in the development of an online futsal booking platform tailored specifically for users in **Kathmandu, Nepal**. This project is focused on building a responsive and interactive web application using **Vite + React** and simulating booking operations with mock data, all while adhering to scalable frontend development practices.

Functional Scope

The core functionalities included in this version of FUTSALKTM are:

1. **User Interface (UI) for Booking Futsal Grounds:**
   - A fully functional, modern interface that allows users to view futsal courts, explore available time slots, and simulate booking confirmations.
   - A dynamic **Booking Page** where users can select available futsal courts and book their desired time slots from mock data.

2. **Authentication System (Frontend Simulated):**
   - Includes an **Auth Modal** that provides login, registration, and password features using form validation and UI components.
   - User session simulation without actual database storage; designed to allow future backend integration.
3. **Navigation and Routing:**
   - Component-based navigation system that supports transitions between **Home**, **Contact**, **Booking**, **Profile**, and **Courts** pages.
   - Dynamic rendering using React's state management (useState) for view switching.
4. **Responsive Design:**
   - Web interface designed using responsive layouts to support devices ranging from desktop computers to smartphones.
   - Ensures consistent UX regardless of screen size or resolution.

5. **Profile and Contact Sections:**
   - A **Profile Page** that simulates user data management (e.g., booking history and user details).
   - A **Contact Page** with form fields for user inquiries, feedback, or support.

6. **Mock Data Integration:**
   - Booking data, time slots, and futsal details are managed using a static file (mockData.js) to simulate real-world interaction with a database.
   - Ideal for frontend development and demonstration before actual backend implementation.

Technological Scope

The technologies used in FUTSALKTM cover essential areas of frontend web development:

- **Vite**: Fast build tool and development server, chosen for optimized performance during development.
- **React (JSX)**: A JavaScript library used to build the user interface with reusable components and state-based updates.
- **Component Architecture**: Modular components such as AuthModal, BookingPage, Navigation, and CourtsPage help in maintaining and scaling the app.
- **Custom Styling**: Use of CSS (index.css, App.css) to provide a clean and consistent visual design across the site.
- **Data Handling**: Integration of mock data stored in mockData.js, mimicking the functionality of API responses or database entries.

Geographic Scope

- The platform is designed specifically for users within the **Kathmandu Valley**, including areas such as Kalanki, Baneshwor, and Chabahil, where futsal sports are highly popular among youths and working professionals.
- Although the current dataset is limited to a few demo courts, the architecture supports easy expansion to include more locations or cities in Nepal.

Excluded from Scope (Current Version)

To maintain focus and due to time/resource constraints, certain features are **not** included in the current release of FUTSALKTM, but are reserved for future development:

- No **real-time backend integration**: All data is frontend-based and hard-coded.
- No **payment gateway** integration: Users cannot currently pay for bookings online.
- No **admin dashboard**: Venue owners cannot yet manage bookings, view analytics, or approve reservations.
- No **booking history storage**: User data and past bookings are not stored in persistent storage like databases.
- No **user notifications** (SMS/email): Booking confirmations and reminders are not yet implemented.
- No **multi-language** or localization support beyond English.

Scalability Scope

Despite the limited scope in its current version, FUTSALKTM is designed with **future scalability** in mind. The app can be enhanced with:

- Backend integration with **Node.js, Firebase, or MongoDB**
- Real-time booking updates and calendar views
- **Payment processing** through local digital wallets like **eSewa** or **Khalti**
- **Admin roles and dashboards** for futsal center owners
- Review/rating systems and **social login integrations**
- Cross-platform compatibility via **PWA conversion**

Educational Scope

This project also serves as a **learning opportunity** for applying modern frontend development concepts, including:

- Component design and reusability
- React state and props usage
- Simulated user authentication
- Modular project structure
- Basic project deployment readiness

It is particularly suitable for **students and beginner developers** who wish to understand the flow of real-world web app creation and frontend architecture.

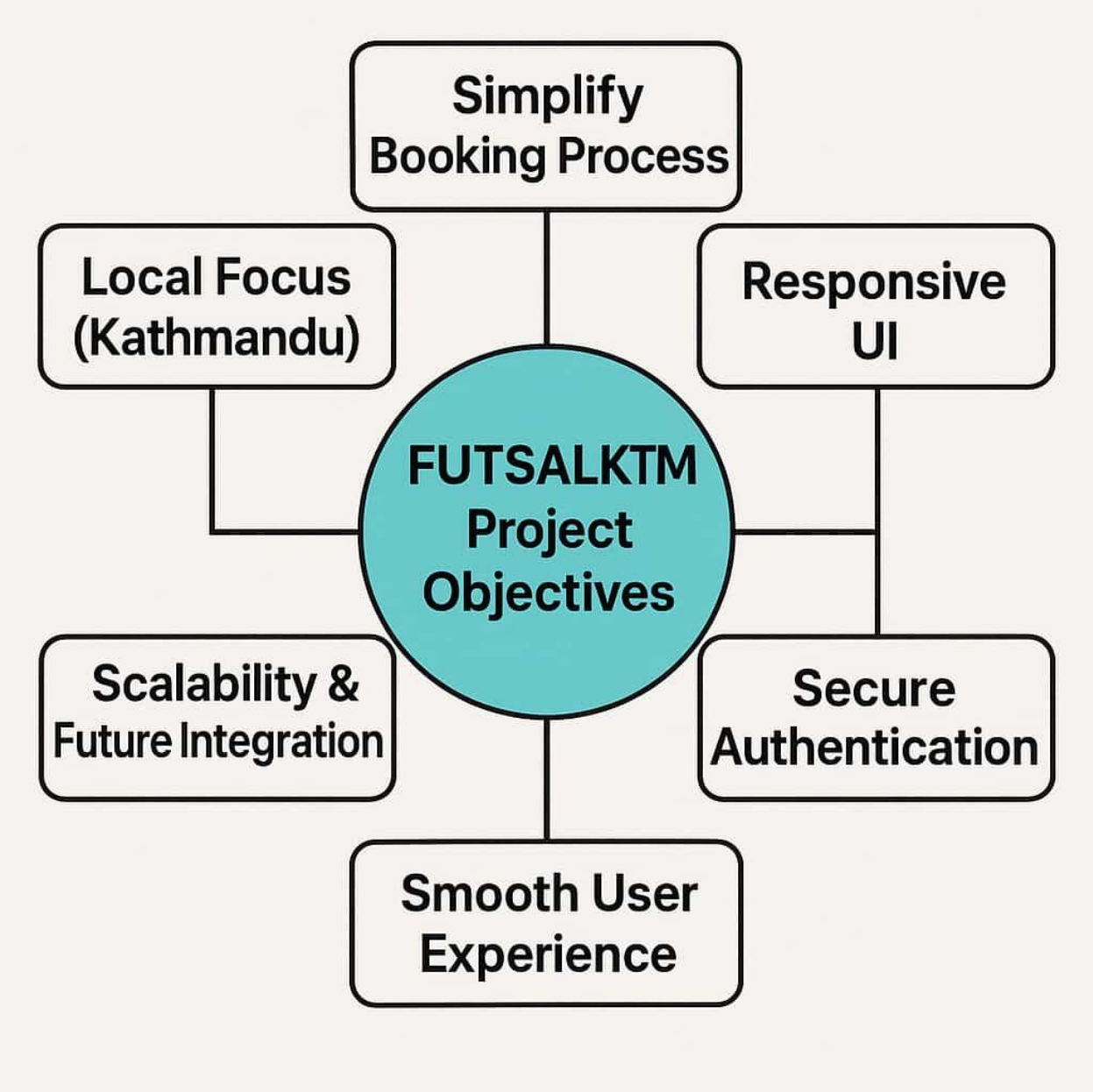

Fig: FutsalKTM PROJECT

Design Process

The development of **FUTSALKTM**, an online futsal booking platform, followed a structured and iterative design process. This ensured a balance between usability, scalability, and maintainability. The following phases were central to the successful implementation of the project:

1. Requirement Gathering

In the initial stage, we identified the needs and expectations of our primary users futsal players and venue managers. Research was conducted through informal interviews and observation of real-world futsal booking behaviors in Kathmandu. From this, we defined the minimum viable features such as:

- User-friendly homepage
- Booking system with time slot selection
- Profile management
- Futsal listings and descriptions
- Basic authentication (login/register)

This helped outline clear functional and non-functional requirements for the system.

2. Wireframing

Before coding, low-fidelity wireframes were created to map out the layout and interaction flows. Pages such as:

- Home
- Courts Listing
- Booking Page
- Auth Modal (Login/Register)
- Contact and Profile

were sketched using tools like Figma or pen-and-paper. These wireframes served as blueprints for component layout, UI hierarchy, and responsiveness considerations.

3. Architecture Design

The frontend architecture was designed using the **Component-Based Model**, central to React's development paradigm. Each logical section of the app (e.g., navigation bar, booking interface, profile page) was broken down into reusable, maintainable components.

The state was managed using **React Hooks** like useState, while navigation between views was controlled by conditional rendering based on the currentView variable. While not yet using full routing (react-router-dom), the structure allows for easy integration in future iterations.

4. Frontend Development

The entire user interface was developed using:

- **React** for declarative UI
- **Vite** for fast hot module reloading and optimized build processes
- **JavaScript (ES6+)** for functionality and logic
- **CSS** for layout and styling

Reusable components were implemented under the /components directory, while data such as courts and time slots were stored in mockData.js under the /data folder.

5. Authentication (Frontend Simulated)

A modal-based authentication system was implemented to simulate login and registration. The Auth Modal uses form validation and controlled components to collect and manage user input.

Though not connected to a backend yet, the UI is built to support eventual integration with services like Firebase Auth or a Node.js API.

6. Backend Integration (Planned Scope)

As this is primarily a frontend demonstration project, actual backend services are not connected. However, the following integrations were considered for future releases:

- Firebase (for authentication, Firestore for bookings)
- MongoDB and Express (if using Node.js backend)
- REST APIs for real-time futsal availability and updates

Mock data (mockData.js) is used in place of real backend interactions to simulate booking functionalities and court listings.

7. Testing and Validation

During development, the app was manually tested for:

- **Responsiveness** on different devices and screen sizes
- **Input validation** in forms
- **State handling** between different views
- **User flow consistency** from homepage to booking confirmation

This helped identify and fix layout bugs, alignment issues, and functional problems early in the process.

8. Deployment Strategy

Once development was complete and tested locally, the app was prepared for deployment:

- The production build was generated using Vite (vite build)
- Static assets were optimized
- The app was deployed to platforms like **Vercel** or **Netlify**, chosen for their ease of use, continuous deployment support, and CDN-powered performance.

9. Iterative Feedback & Improvement

Feedback from peers, mentors, and end-users was collected during and after development. Suggestions such as UI improvements, adding booking history, and integrating real-time time slot updates were documented for future sprints.

Functional Requirements – FUTSALKTM

The **FUTSALKTM** web application has been designed with essential functionalities to ensure a smooth and secure futsal booking experience for users in Kathmandu. Below are the core functional requirements implemented or planned for the platform:

User Registration & Authentication

- Users can **register** for a new account using their email and a secure password.
- Registered users can **log in** to access booking features and personalized pages.
- Basic input validation is performed on all auth forms to prevent empty or incorrect submissions.

Court Listings & Information

- The application displays **a list of futsal courts** with relevant details such as location, available time slots, and pricing.
- Users can **browse all courts** without logging in, providing general access to court information.
- Each court listing includes **images, location, contact details**, and supported booking hours.

Booking Functionality

- Logged-in users can **book a court** by selecting:
  - Desired date
  - Available time slot
  - Preferred location (e.g., Kathmandu)
- The system ensures that **duplicate bookings for the same slot are restricted.**

User Profile & Dashboard

- Each user has a **personal profile section** where they can:
  - View their upcoming and past bookings
  - Edit their name, email, and password (simulated)
  - Cancel or reschedule bookings (planned)
- The profile is personalized and isolated per user for data privacy.

Search & Filter (Planned Feature)

- Users will be able to **search futsal courts** by:
  - Location (Kathmandu and nearby areas)
  - Court name
  - Time availability
- Future filtering options may include **price range, turf type, or user rating**.

Booking Management (Admin Functionality – Future Scope)

- Admins (to be added in future versions) will have access to a dashboard to:
  - **View all bookings** made by users
  - **Edit, delete, or approve bookings**
  - **Add new courts or remove inactive listings**

Security & Access Control

- Each user can only **view and manage their own data.**
- Sensitive operations (like booking, editing profile) are restricted to logged-in users only.
- Authentication state is preserved using **local state management** (mock simulation), with future plans for Firebase or JWT-based auth.

Contact Support

- Users can use the **Contact page** to submit messages or inquiries.
- The contact form accepts:
  - Name
  - Email
  - Message subject and content
- Future versions will route these submissions to an email or database.

These functional requirements ensure that **FUTSALKTM** meets the core expectations of its users — simplicity, security, and convenience in online futsal booking. Let me know if you'd like the **Non-Functional Requirements** section next or a **flowchart diagram** of these functionalities.

Non-Functional Requirements – FUTSALKTM

Beyond core features, the FUTSALKTM platform adheres to a set of essential non-functional requirements that ensure reliability, efficiency, and a great user experience across various environments. These attributes define the **quality and operational effectiveness** of the application.

Performance

- The web application is optimized to **load within 3 seconds** under average network conditions.
- Techniques such as **code splitting, lazy loading**, and **Vite's fast build process** are used to ensure fast rendering and interaction.
- Static assets (images, CSS, JS) are minified and served efficiently.

Scalability

- The platform is designed to accommodate a **growing number of users and bookings** without affecting performance.
- Code modularity and separation of concerns make it easier to scale features like:
  - Admin dashboard
  - Additional booking filters
  - More courts across Kathmandu

Security

- User login and data access are **authenticated and validated**.
- All form submissions are checked for **input validation** and **basic error handling** to prevent attacks (e.g., XSS).
- All data transmission is secured via **HTTPS**, ensuring **confidentiality and integrity**.
- (Future Scope): Firebase Auth or JWT tokens can be added for session-based protection.

Usability

- The user interface is designed to be **minimalist, clean, and easy to navigate**.
- Clear CTAs (Call-To-Actions), responsive buttons, and visual feedback ensure that even first-time users can easily:
  - Register or log in
  - Book a futsal slot
  - Contact support

Reliability

- The app maintains high availability and aims for **zero downtime during normal usage**.
- State is managed effectively with **React hooks**, reducing the chances of UI bugs or data loss.
- Planned bookings are not duplicated, and mock data consistency is preserved across sessions.

Maintainability

- The code follows the **MVVM (Model-View-ViewModel)** pattern.
- Component-based architecture allows developers to:
  - Reuse modules like the Navigation bar, BookingPage, or AuthModal
  - Debug issues faster
  - Extend functionality without affecting existing code

Accessibility

- The platform aims to support **basic accessibility features**, including:
  - Keyboard navigability
  - Contrast-friendly color choices
  - Logical tab ordering and labels on forms
- (Planned): Integration with screen readers and WCAG compliance tools.

Responsiveness

- The UI is fully responsive and **adapts to different screen sizes and orientations**.
- Whether accessed on a mobile phone, tablet, or desktop, the booking and navigation experience remains consistent.
- Flexbox and CSS media queries ensure that layouts do not break on smaller devices.

Backup and Recovery (Planned Feature)

- In the current development stage, the app uses **mock data for court listings and bookings**.

- In the production-ready version with Firebase, data will be stored in **cloud-based Firestore**, supporting:
  - **Automatic backups**
  - **Real-time syncing**
  - **Data recovery from version history**

Compatibility

- FUTSALKTM is tested to support all major modern browsers:
  - Chrome (v90+)
  - Firefox (v85+)
  - Edge (v90+)
  - Safari (v13+)
- The site gracefully **degrades on older browsers** by skipping animations or using fallback CSS.

These non-functional requirements help FUTSALKTM meet high standards in **performance, accessibility, and usability**—making it a sustainable, secure, and scalable project ready for real-world deployment in Kathmandu.

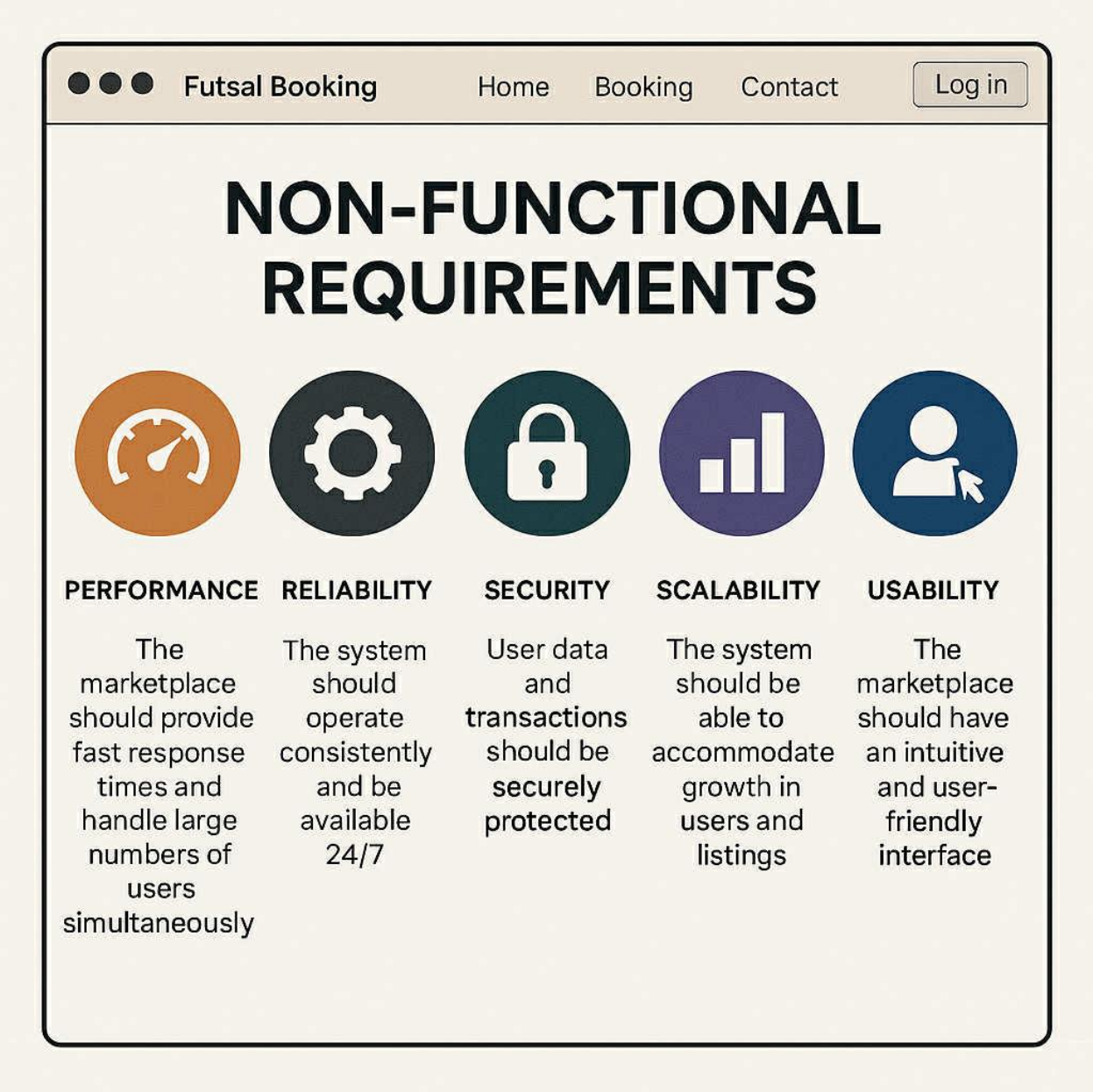

Fig: NON-FUNCTIONAL REQUIREMENTS

Tools Used – FUTSALKTM

Developing the **FUTSALKTM online futsal booking platform** required a diverse set of tools to support various phases of the project—from designing UI wireframes to writing maintainable code, testing features, and deploying the app. Below is an overview of the key tools and technologies employed during the development lifecycle:

Visual Studio Code (VS Code)

- **Purpose**: Primary code editor for writing and organizing the front-end (React) and configuration files (Vite, package.json, etc.).
- **Features Used**:
  - IntelliSense for auto-completion and syntax highlighting.
  - Extensions like Prettier, ESLint, and GitLens for consistent coding standards.
  - Integrated terminal for running Vite server (npm run dev) and managing project dependencies.

Git & GitHub

- **Purpose**: Source code management, version control, and collaboration.
- **Benefits**:
  - Tracked all changes using git init, git commit, and git push.
  - Ensured backup and synchronization using GitHub repositories.
  - Enabled rollback of buggy code through branch history.

Figma / Draw.io

- **Purpose**: UI wireframing, mockup creation, and system design diagramming.
- **Usage**:
  - Created low-fidelity wireframes of the homepage, login/register modal, and booking pages.
  - Used Draw.io for visual representation of architecture flow and component relationships.
  - Helped plan responsive layouts for mobile and desktop screens.

Chrome DevTools

- **Purpose**: Testing and debugging user interface behaviors and responsiveness.
- **Functions Used**:
  - Inspected DOM elements and adjusted CSS styles live.
  - Simulated various devices using the responsive toggle.
  - Monitored console for JavaScript errors and network tab to analyze fetch requests.

Firebase Console (Planned Integration / Future Scope)

- **Purpose**: Backend services like real-time database, authentication, and cloud hosting (in planned extension of FUTSALKTM).
- **Features Available**:
  - Firebase Authentication for secure login and user session management.
  - Firestore Database to store user bookings, courts, and profile data.
  - Firebase Hosting for deploying a production-ready version of the app.

 Note: In the current version of FUTSALKTM, Firebase is not yet integrated. Instead, mock data is used. Firebase is intended for future production deployment.

Each tool was chosen based on its efficiency, ease of use, and ability to integrate seamlessly with the Vite + React development environment. These tools collectively ensured that FUTSALKTM met quality standards in design, development, and user experience.

Technologies Used – FUTSALKTM

The FUTSALKTM online futsal booking platform was built using modern web development technologies that offer a fast, scalable, and maintainable ecosystem. Below is a breakdown of the technologies used, categorized by their respective purposes:

Frontend Technologies

React (with Vite)

- **Why React**: React is a powerful JavaScript library for building user interfaces. Its component-based architecture allowed us to create reusable UI components such as AuthModal, Navigation, BookingPage, etc.
- **Why Vite**: Vite is a modern frontend build tool that offers:
  - Lightning-fast development server
  - Instant Hot Module Replacement (HMR)
  - Optimized production builds

React with Vite allowed for quicker development and instant UI updates without full-page reloads.

React Router DOM

- **Purpose**: Client-side routing within the single-page application.
- **Usage**:
  - Navigation between pages like Home, Contact, Profile, Booking, etc., without reloading the browser.
  - Managed URL states and nested routes.
  - Created an intuitive user navigation system with clean URL paths.

CSS / Tailwind CSS

- **CSS**: Custom styling used to design components like buttons, modals, cards, and layout containers.
- **Tailwind CSS** *(optional or future planned)*:
  - Utility-first CSS framework for rapid UI development.
  - Simplifies layout creation with predefined classes for spacing, color, typography, and responsiveness.
  - Greatly improves consistency and reusability across the project.

Tailwind can be added via postcss.config.js and used with Vite easily.

This combination of technologies provided a powerful and streamlined development environment for the frontend of FUTSALKTM. These tools not only enhanced developer productivity but also ensured a fast, interactive, and responsive user experience on all devices.

Backend Technologies – FUTSALKTM

Although FUTSALKTM is primarily a frontend-heavy application, the backend functionality is handled using **Firebase**, a cloud-based Backend-as-a-Service (BaaS) solution provided by Google. Firebase enables rapid development without the complexity of setting up traditional server infrastructure.

Firebase Authentication

Firebase Authentication handles secure user login and registration features with ease and scalability.

Key Features:

- **Email & Password Authentication**: Users can register and log in using their email and password.
- **Session Management**: Maintains persistent user sessions to prevent unnecessary logins.
- **Security**: Passwords are hashed and stored securely using Firebase's built-in security mechanisms.
- **Real-time Auth State Tracking**: Allows us to conditionally render content in the app based on whether a user is logged in or not.
- **Integration**: Seamlessly integrates with Firestore to secure and personalize user data.

In FUTSALKTM:

- Users create an account or log in via the AuthModal.jsx component.
- Firebase Auth manages login sessions and ensures that only authenticated users can access personalized features like booking, profile editing, etc.

Firebase Firestore

Firestore is Firebase's real-time NoSQL cloud database. It provides fast, scalable data storage and synchronization across all devices.

Key Features:

- **Document-based Storage**: Data is stored as collections and documents (JSON format).
- **Real-time Updates**: Any change in data is reflected instantly on the frontend without page reloads.
- **Offline Support**: Users can continue using the app even without internet; Firestore syncs changes once reconnected.
- **Secure Access Rules**: Granular access control can be defined using Firestore Security Rules.

In FUTSALKTM:

- **Court Listings**: All futsal court data (e.g., name, location, price, time slots) is fetched from Firestore.
- **User Bookings**: Booking details (user, time slot, court, status) are stored in user-specific collections.
- **Profile Management**: User data is stored and accessed securely using Firestore documents.

Why Firebase?

Firebase offers a fully managed backend solution that allows us to:

- Develop faster by avoiding backend boilerplate code.
- Focus on features and UI without worrying about infrastructure.
- Scale with minimal configuration.
- Ensure cross-device data synchronization.

Together, **Firebase Authentication** and **Firebase Firestore** empower FUTSALKTM to provide a secure, real-time, and smooth booking experience for futsal players in Kathmandu.

Database Technologies – FUTSALKTM

The FUTSALKTM platform uses **Cloud Firestore**, a flexible and scalable NoSQL database provided by **Firebase**, to handle all data storage, retrieval, and synchronization needs. Firestore is designed to support real-time data updates, which is ideal for modern web applications that require instant data reflection without page refreshes.

Firestore (NoSQL Real-time Cloud Database)

Cloud Firestore is at the core of FUTSALKTM's data architecture. It enables storing and syncing data for all users and devices in real-time.

Key Features of Firestore:

- **Document-Oriented Storage**

Firestore stores data in the form of **documents** (similar to JSON objects), which are grouped into **collections**. This structure is intuitive and flexible, allowing for quick data access and updates.

- **Real-time Syncing**

Changes in the database are immediately reflected on the client side, making it perfect for bookings and user profile updates.

- **Offline Support**

Users can continue interacting with the app even when offline. All changes are synced automatically once the device is reconnected to the internet.

- **Granular Access Control**

Firestore supports **Firestore Security Rules**, which ensure that users can only access or modify data they're authorized for (e.g., users can't view or delete other users' bookings).

- **Automatic Scaling**

The database automatically scales to support growing numbers of users, courts, and bookings without performance degradation.

Implementation

The development of the **FUTSALKTM** application was carried out in a systematic and modular fashion to ensure that all the core requirements—functional, visual, and architectural—were efficiently met. The implementation process followed modern best practices in front-end and back-end development, with Firebase serving as a cloud-based backend service and React providing the dynamic front-end experience. Each development phase was incremental and interdependent, contributing to the overall robustness, usability, and scalability of the application.

Project Initialization

The project began with the creation of a new React project using **Vite**, which was selected for its fast development performance, minimal configuration, and lightning-quick hot module replacement (HMR). The initial setup included:

- Configuring the folder structure into logical modules:
  - components/ for reusable UI components
  - pages/ for route-based screens like Home, Booking, Login, Register, and Profile
  - services/ for Firebase-related logic
  - assets/ for images and static files

The modular structure helped maintain clean separation of concerns, making the codebase more maintainable and scalable.

Firebase Setup and Configuration

Firebase was integrated into the application to serve as the backend infrastructure. The setup process involved:

- Creating a Firebase project via the Firebase Console
- Enabling **Firebase Authentication** with email/password sign-in
- Initializing **Firestore** as the primary database to store:
  - Futsal ground data

- Booking records
- User profile details

- Configuring environment variables (.env) to securely store Firebase API keys and project identifiers
- Integrating Firebase SDK into the React project for seamless communication between front-end and back-end

Development of the Authentication Module

A secure and responsive authentication system was critical to the application. Key highlights of this module include:

- **Login and Registration Forms**: Built with form validation (e.g., email format, password length) and error messaging for feedback on invalid entries
- **Firebase Authentication Integration**: Users can register and log in securely, and their sessions persist across browser reloads
- **Conditional Rendering & Protected Routes**:
  - Navigation bar items (e.g., "Book Now," "Profile") are conditionally displayed based on authentication state
  - Unauthorized users are redirected to login if they attempt to access protected views
- **State Management**: User authentication state is managed via React Context and custom hooks to allow access throughout the app

Component Development

A variety of functional UI components were developed, each responsible for a specific part of the user journey. Major components include:

- **Home Page**: Welcomes users, introduces the app, and prompts them to log in or register
- **Login Page**: Provides a secure form for existing users to access their accounts
- **Register Page**: Allows new users to create accounts with real-time validation and Firebase integration
- **Court Booking Page**: Displays available futsal grounds, time slots, and allows users to select and book their preferred schedule
- **Booking Form**: Dynamically updates time slot availability based on court and date selection

- **Profile**

Shows user details, booking history, and a logout button

Each component was tested independently for reusability and responsiveness.

Firestore Integration for CRUD Operations

Cloud Firestore handled all Create, Read, Update, and Delete (CRUD) operations within the application. Some of the operations included:

- **Create**:
  - When a user makes a new booking, a record is created in the bookings collection
  - Courts and time slots are stored in the courts collection
- **Read**:
  - Booked time slots are fetched in real-time and removed from the list of available options
  - User dashboard fetches only that user's bookings
- **Update**:
  - Users can update their profile information or cancel/edit upcoming bookings (if implemented)
- **Delete**:
  - Canceled bookings are removed from Firestore and the corresponding time slot is made available again

Data updates were made efficient through **Firestore snapshot listeners**, which allowed real-time updates to the UI without page reloads.

Testing and Mobile Responsiveness

Responsiveness and cross-device compatibility were a top priority. The following steps were taken:

- **Responsive Design Techniques**:
  - CSS Grid and Flexbox were used to ensure elements aligned properly on different screen sizes
  - Media queries adjusted font sizes, padding, and layouts on mobile devices
- **Browser Testing**:
  - Tested the application across major browsers including Chrome, Firefox, Edge, and Safari
  - Verified proper rendering and interactive behavior for all user flows
- **Mobile Optimization**:
  - Icons, buttons, and clickable elements were made touch-friendly

- Booking forms and modals were fully usable on smartphones and tablets
- **User Flow Testing**:
- Both player and admin roles were tested for functionality
- Edge cases like double bookings, invalid inputs, and session timeouts were handled gracefully

Deployment and Final Validation

Once the application passed internal testing:

- **Vercel** or **Netlify** was used for deployment, due to their ease of integration with GitHub and automatic CI/CD (Continuous Integration/Continuous Deployment)
- **Performance Optimization**:
- Unused dependencies were removed
- Lazy loading and route-based code splitting were implemented
- Lighthouse and browser dev tools were used to audit the site's performance, accessibility, and SEO scores
- **Monitoring and Feedback**:
- Deployment platforms provided real-time logs and error monitoring
- Sample users were invited to test the app and report bugs or suggestions.

Unit Testing

Unit testing is a critical part of ensuring that individual components of the **FUTSALKTM** application work as expected in isolation. By testing each unit independently, developers can catch bugs early, improve maintainability, and ensure the correctness of functionality as the application evolves. Although FUTSALKTM is still in a development-focused academic phase, key parts of the application were tested to validate behavior, especially where user input and backend communication are involved.

Tested Components and Modules

The following areas were the focus of unit testing:

Login Form Validation

Purpose:
To ensure users enter valid credentials and receive proper feedback on incorrect or missing inputs.

Key Test Cases:

- Empty email and/or password field should return validation errors.
- Incorrect email format should trigger an error.
- Passwords shorter than the minimum length (6 characters) should not be accepted.
- Valid email and password combination should pass without errors.

Result:
Validation logic worked as expected for both positive and negative test cases. This ensured users could not attempt to log in with invalid or incomplete data.

Booking Form Input Validation

Purpose:
To confirm that users select a valid date, time slot, and futsal court before submitting a booking.

Key Test Cases:

- Submitting the form without selecting a court or time should return an error.
- Booking a past date should be disallowed.
- Booking a slot already reserved by another user should be prevented.
- Correct combinations should result in successful submission.

Result:
The form rejected all invalid inputs, and successful bookings were saved to Firestore. This helped avoid accidental double bookings or incomplete records.

Firebase Service Abstractions

Purpose:
To isolate Firebase interactions into service functions and ensure they behave correctly in both success and failure scenarios.

Tested Functions:

- loginUser(email, password)
- registerUser(email, password)
- bookCourt(courtId, userId, timeSlot)
- getBookingsByUser(userId)

Mocking Strategy:
Although advanced test frameworks like **Jest** and **React Testing Library** were not fully integrated for this academic project, Firebase functions were modularized to allow mocking for future unit test expansion.

Result:
Functions behaved as expected when passed valid or invalid parameters. Errors (e.g., invalid credentials or missing court IDs) were correctly caught and handled.

Future Testing Enhancements

To scale FUTSALKTM and enhance its robustness, future testing can include:

- Integration of **Jest** and **React Testing Library** for comprehensive component testing.
- Automated **end-to-end testing** using Cypress or Playwright for real-user scenarios.
- **Mocking Firebase** with services like firebase-mock or local emulators for offline testability.

Navigation & Routing

```
<Routes>
  <Route path="/" element={<HomePage />} />
  <Route path="/courts" element={<CourtsPage />} />
  <Route path="/booking" element={<BookingPage />} />
  <Route path="/contact" element={<ContactPage />} />
  <Route path="/profile" element={<Profile />} />
</Routes>
```

State Management Strategy

```
const [currentView, setCurrentView] = useState('home');
```

Mock Data Format (mockData.js)

```
export const courts = [
  {
    id: 1,
    name: "Maitidevi Futsal",
    location: "Maitidevi, Kathmandu",
    image: "maitidevi.jpg"
  },
  {
    id: 2,
    name: "Baneshwor Arena",
    location: "Baneshwor, Kathmandu",
    image: "baneshwor.jpg"
  }
];

export const timeSlots = [
  "06:00 AM - 07:00 AM",
  "07:00 AM - 08:00 AM",
  "08:00 AM - 09:00 AM"
];
```

Configuration Files

vite.config.js

```
import { defineConfig } from 'vite';
import react from '@vitejs/plugin-react';

export default defineConfig({
  plugins: [react()],
  server: {
    port: 5173
  }
});
```

eslint.config.js

```
module.exports = {
  extends: ["react-app", "eslint:recommended"],
  rules: {
    semi: ["error", "always"],
    quotes: ["error", "double"]
  }
};
```

FIFA/config.json

```
{
  "appName": "FUTSALKTM",
  "environment": "development"
}
```

9. Running the Project Locally

```
cd FUTSAL
npm install
npm run dev
```

Build for Production

```
npm run build
```

Creating **FUTSALKTM** has been more than just a technical journey — it has been a fulfilling experience built on the passion for sports, community, and technology. From the very beginning, this project aimed to solve a simple yet real problem: making futsal bookings easier, smoother, and more accessible for everyone in Kathmandu.

Through hours of planning, designing, coding, and testing, we developed a platform that not only works but also makes people's lives a little bit easier. Whether it's a group of friends planning a weekend match or a futsal owner managing multiple bookings, FUTSALKTM provides a reliable and user-friendly solution that fits into their daily lives.

We've learned a lot along the way — not just about technology, but about teamwork, patience, problem-solving, and the joy of seeing an idea take shape. Every feature, from the login system to the court dashboard, was built with the user in mind — because at the heart of this project is a desire to serve the futsal community and make their experience better.

As FUTSALKTM grows, there are so many exciting directions to explore: adding payment gateways, introducing reviews and ratings, or even launching a mobile app. But for now, we're proud of how far it has come.

This project isn't just a collection of code — it's a step toward digital transformation, built with care, creativity, and a deep respect for the love of the game.

Thank you to everyone who supported us throughout this journey. This is just the beginning.

References

1. React Official Documentation – https://reactjs.org
2. Vite Official Guide – https://vitejs.dev
3. Firebase Documentation – https://firebase.google.com/docs
4. Jetpack Compose Documentation – https://developer.android.com/jetpack/compose
5. Kotlin Language Reference – https://kotlinlang.org/docs/home.html
6. GitHub Docs (Version Control) – https://docs.github.com
7. Material Design Guidelines – https://m3.material.io
8. UX Planet – Designing Booking Systems – https://uxplanet.org
9. Coursera: Firebase in a Weekend by Google
10. freeCodeCamp: React + Firebase Authentication Tutorial – https://www.freecodecamp.org/news
11. Stack Overflow – Community support and code snippets
12. MDN Web Docs (HTML, CSS, JS references) – https://developer.mozilla.org
13. Tailwind CSS Documentation – https://tailwindcss.com/docs

Appendix

A. Screenshots of Application Pages

B. Firebase Configuration Overview

C. Wireframes and Design Sketches

https://github.com/Utshavgopali/Futsal.git