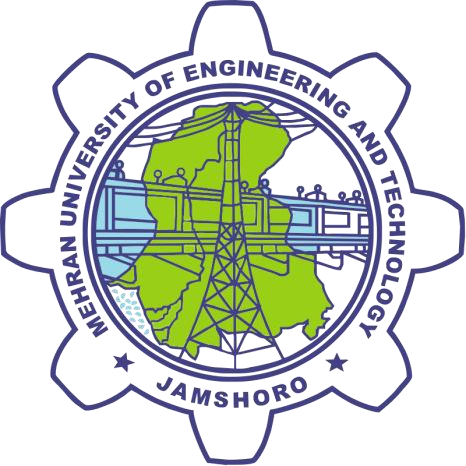
**MEHRAN UNIVERSITY OF ENGINEERING AND TECHNOLOGY, JAMSHORO**



**Assignment**

**Submitted by**

Abdul Rafiy (22SW137)

Abid Ali (22SW022)

# Section: II

**Subject:** Mobile Application Development

**Submitted to: Mam Mariyam**

**Assessment Type: Complex Engineering Problem (CEP)**

**Project Title: Easy Count – Cricket Score Counter App**

## ****1. Problem Identification****

In local and friendly cricket matches, players often face difficulty in keeping proper track of scores, overs, and wickets. Manual scorekeeping on paper is error-prone and does not provide live statistics or an easy way to resume a match later.

**Main problems identified:**

* Manual scoring is time-consuming and inaccurate.
* No digital record of past matches or statistics.
* Hard to generate summaries for innings or players.
* No simple platform for tracking individual and team performance.

Hence, a mobile application was needed to simplify scoring, store match data safely, and present real-time results and summaries.

## ****2. Proposed Solution****

To solve the above problems, we developed a mobile application named **Easy Count** using the **Flutter framework**.  
The app allows users to start, record, and manage live cricket matches digitally with a clean and interactive interface.

**Key functionalities:**

* Start a new match by adding teams, players, and overs.
* Record ball-by-ball updates (runs, wickets, extras).
* View live scoreboard and over summary.
* Save and review previous matches in history.
* View detailed statistics of teams, batsmen, and bowlers.

All data is securely stored **locally** using the **Hive database**, allowing the app to work offline with fast read/write performance.

## ****Main Objectives****

* To provide an easy-to-use digital cricket scoring system.
* To manage match data efficiently using Hive for local storage.
* To display innings summaries, player stats, and match history.
* To enable users to resume, delete, and review matches.
* To ensure a simple, offline-capable, and responsive mobile interface.

## ****3. Project Features****

### ****For Users****

* Start New Match: Add team names, players, overs.
* Live Scoreboard: Record runs, wickets, extras, and track overs.
* Resume Match: Continue a paused or saved game.
* Match History: View previous matches with details.
* Match Summary: Switch between 1st and 2nd innings tabs.
* Player Stats: View batting/bowling performance.

### ****App Information & Support****

* **About Us:** Shows developer info and social links.
* **Contact Us:** Displays phone/email with copy-to-clipboard.
* **Terms and Conditions:** Basic usage policies.

### ****Other Highlights****

* Simple, modern user interface using Material Design.
* Drawer navigation for quick screen access.
* Search bar for finding players in stats.
* Alert dialogs for confirmation and match setup.
* Blue-grey color theme for clean consistency.

## ****5. Data Storage (Hive Database)****

The **Easy Count App** uses the **Hive NoSQL database** for local storage of all data such as match details, player stats, and scores. Hive is a lightweight, high-performance key-value database written in Dart and optimized for Flutter apps.

### ****Why Hive Use:****

1. **Offline Functionality:** Works completely offline without internet.
2. **Speed and Performance:** Extremely fast read/write operations for live scoring.
3. **Simplicity:** No complex SQL queries — data is stored as key-value pairs.
4. **Security:** Data can be encrypted locally to protect user information.
5. **Low Memory Usage:** Ideal for mobile devices and Flutter apps.

### ****Data Stored****

| **Box Name** | **Purpose** | **Data Fields** |
| --- | --- | --- |
| matches | Stores each match record | matchID, team1, team2, overs, winner, date |
| players | Stores batsman & bowler details | name, runs, balls, fours, sixes, wickets |
| history | Maintains past match summaries | teams, scores, result, timestamp |
| stats | Stores aggregated team and player performance | teamName, totalRuns, average, strikeRate |

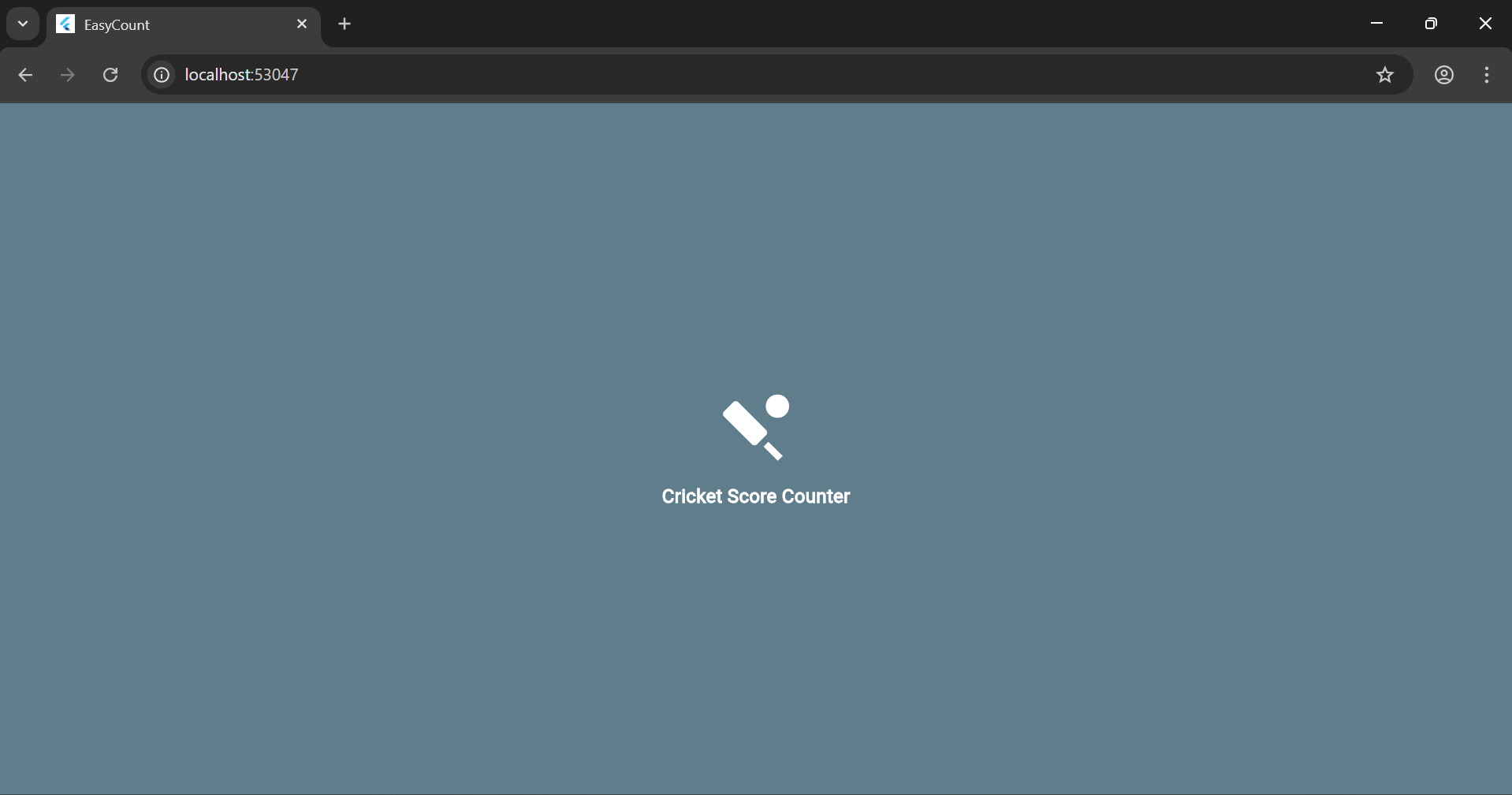
All match data is instantly saved after every update using Hive.box().put() and loaded at app start with Hive.box().get(), ensuring data persistence even after closing the app.

## ****5. Project Design and Interface****

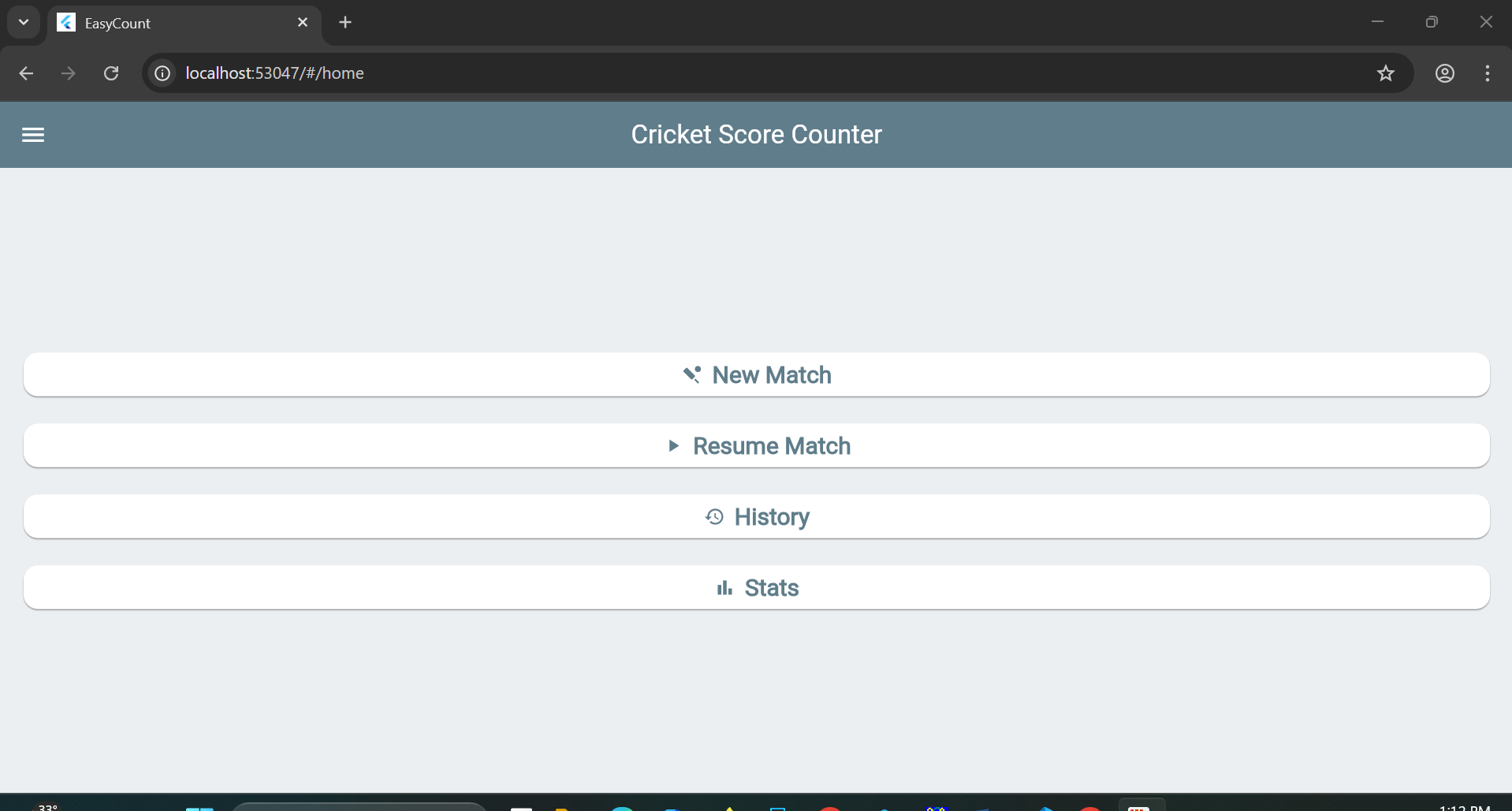
The app’s UI is built using Flutter’s Material Design widgets for a consistent and responsive layout.

### ****Main Screens:****

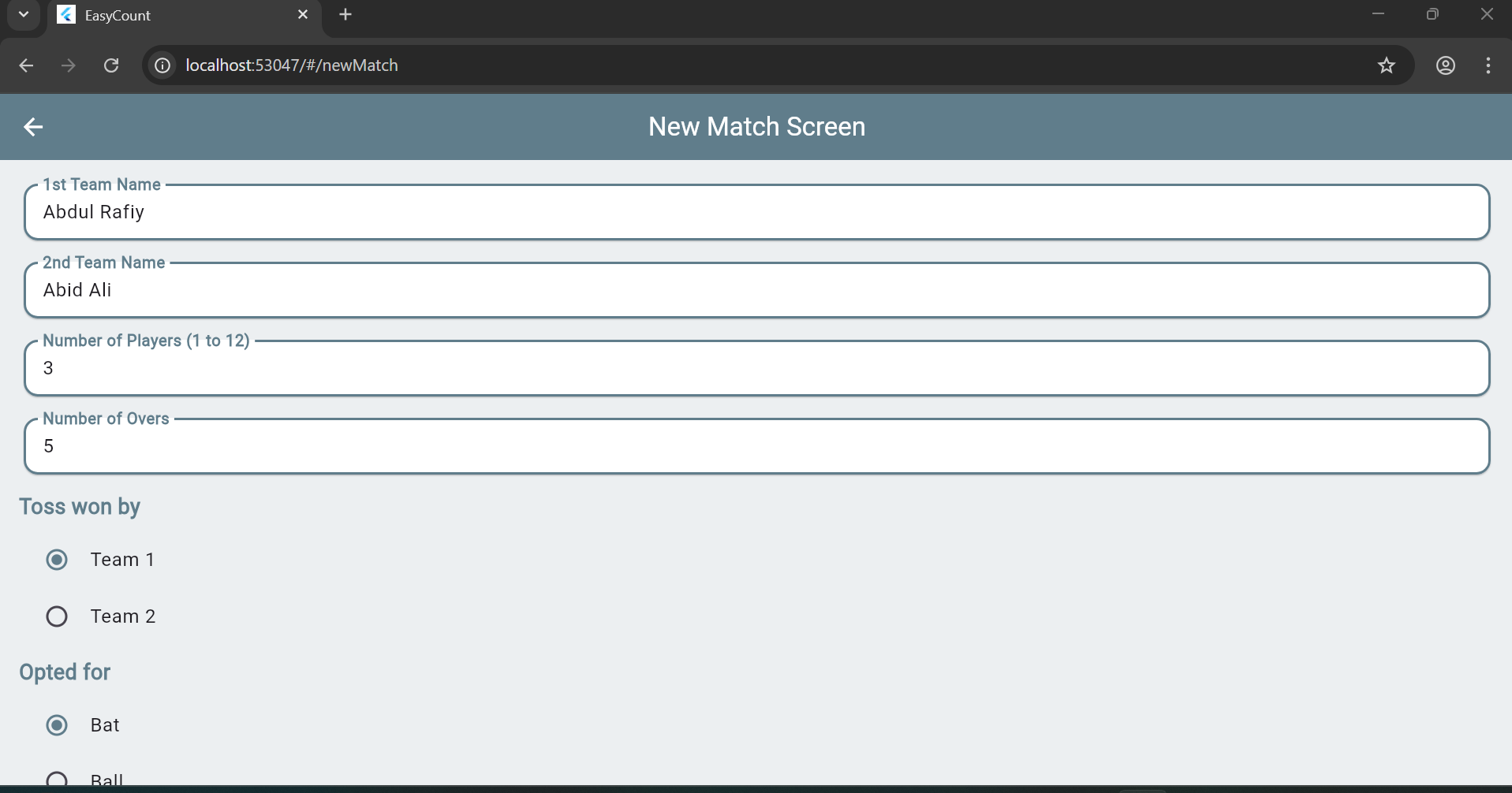
1. **Splash Screen**

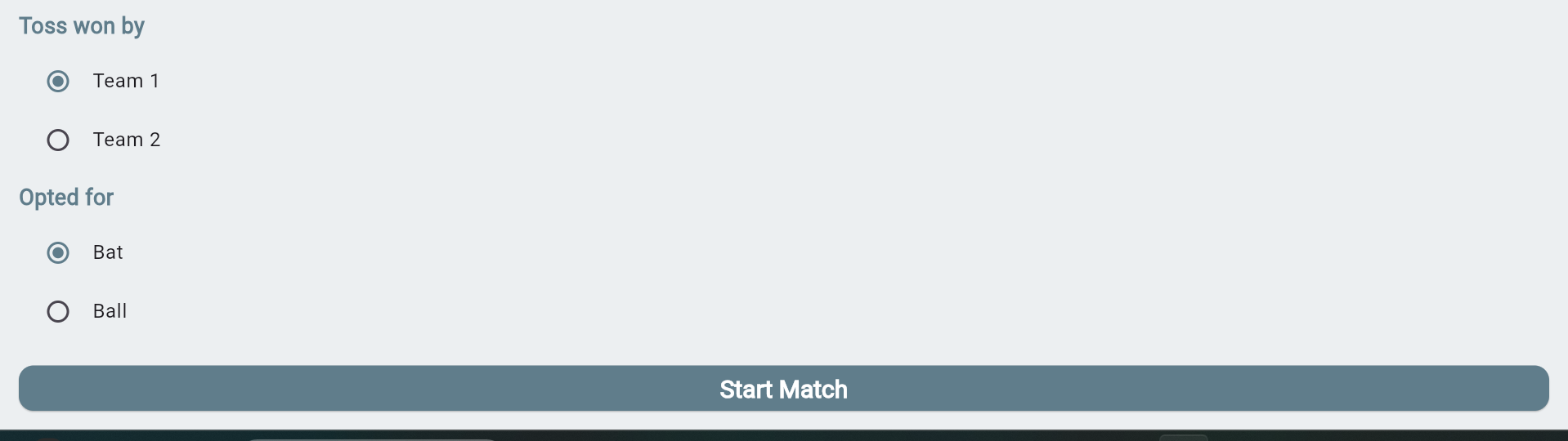


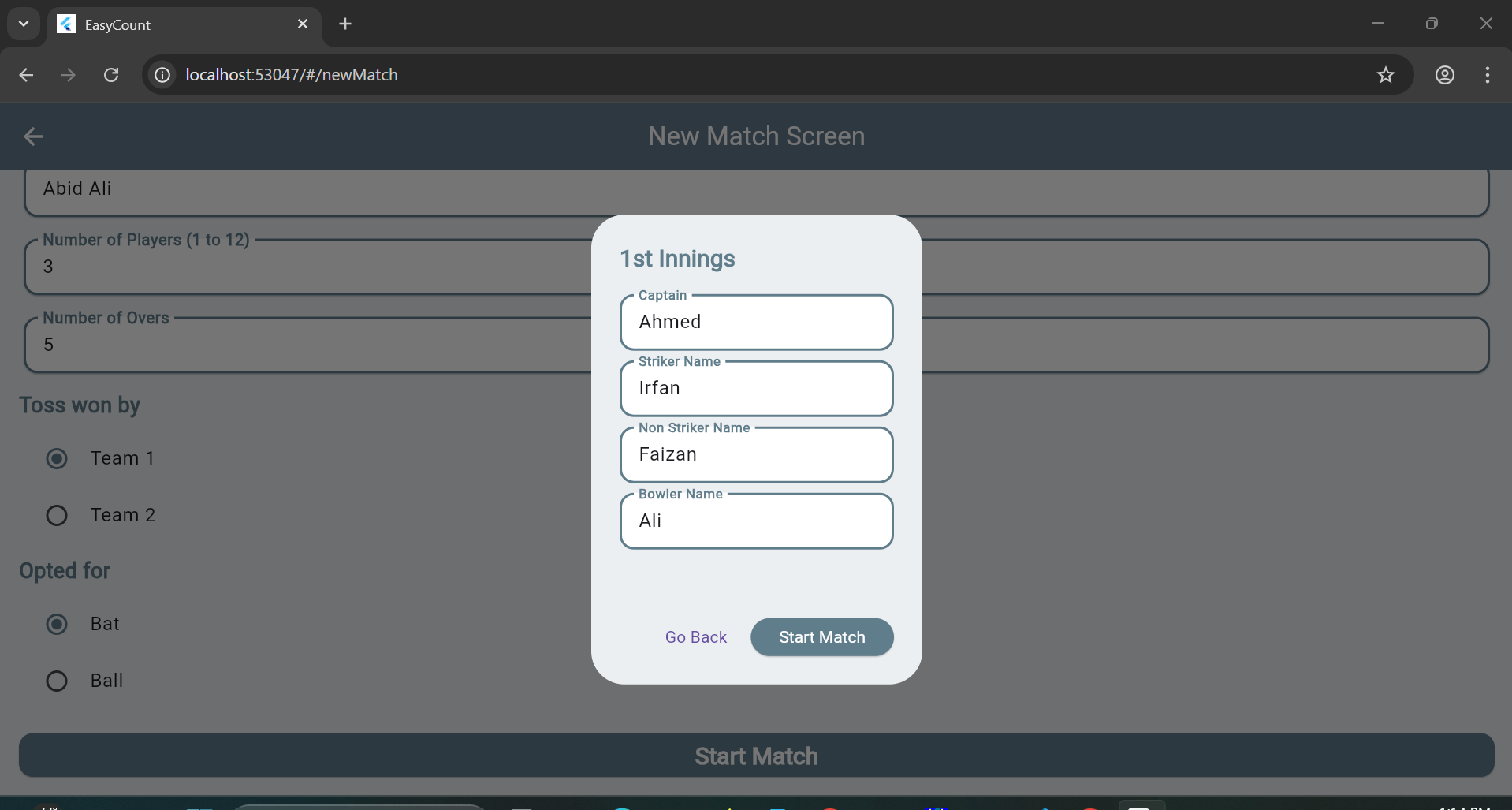
1. **Home Screen** – Main dashboard with buttons: New Match, Resume Match, History, Stats.



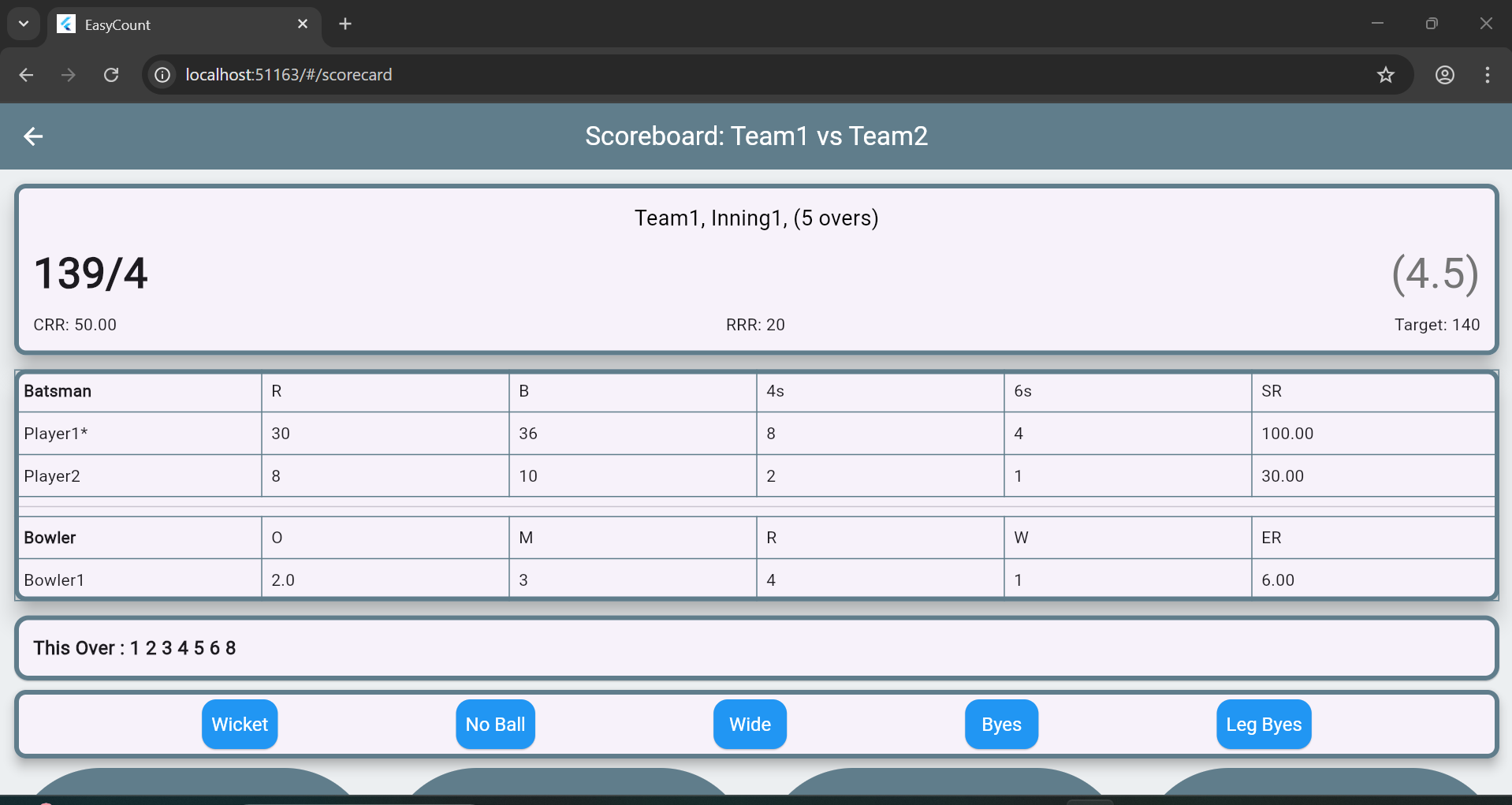
1. **New Match Screen** – Form for team names, players, and overs.

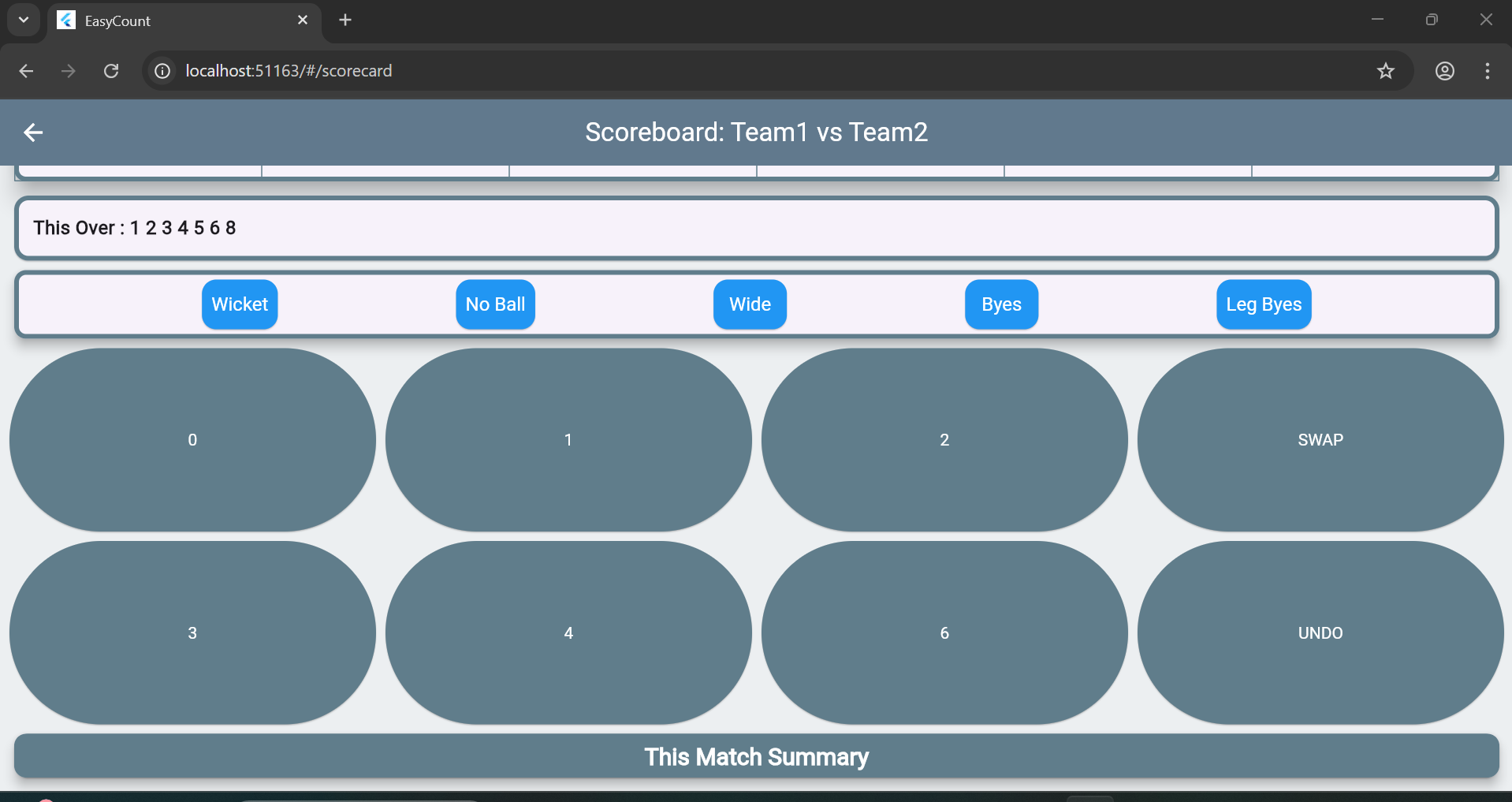




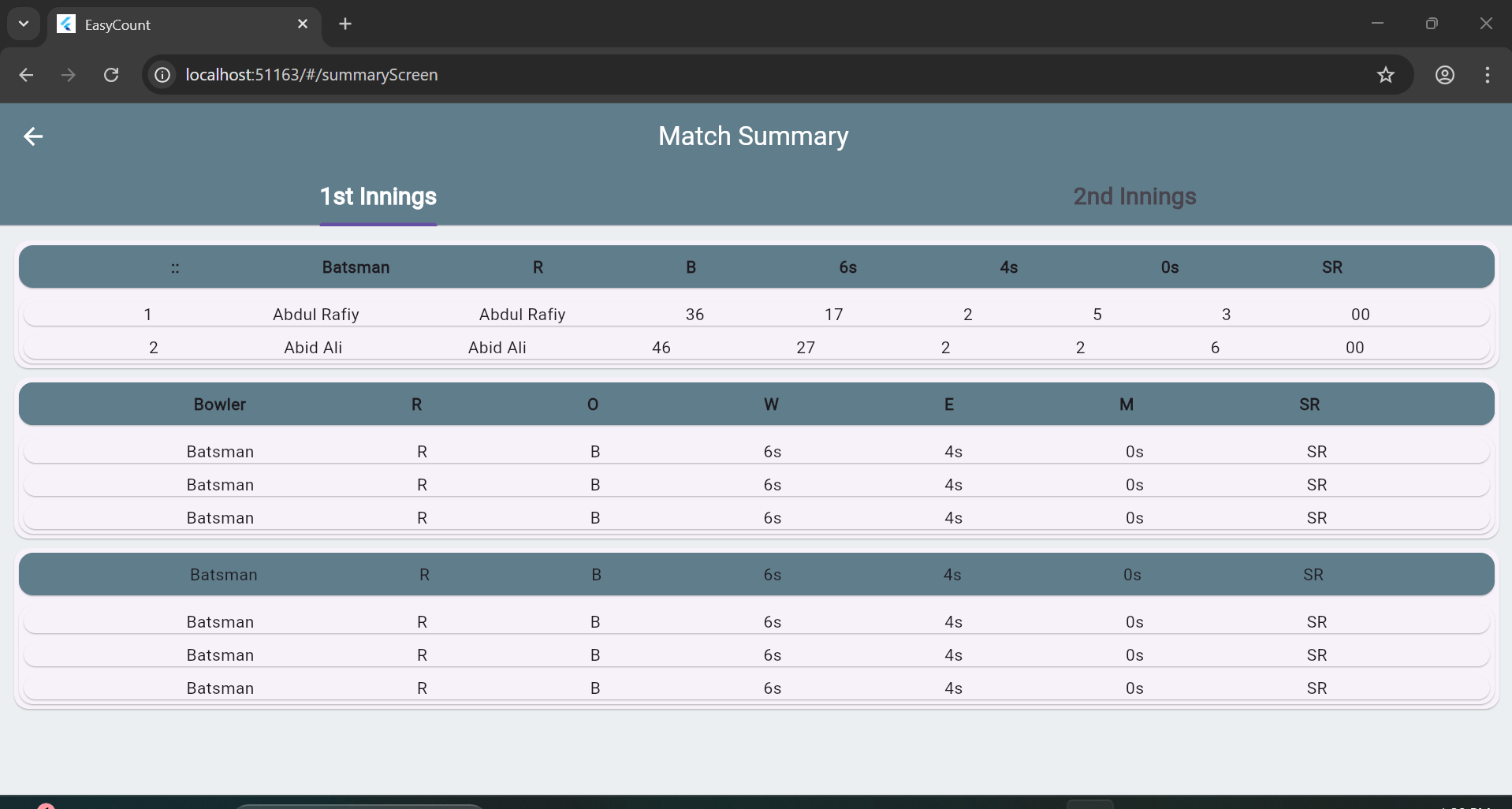


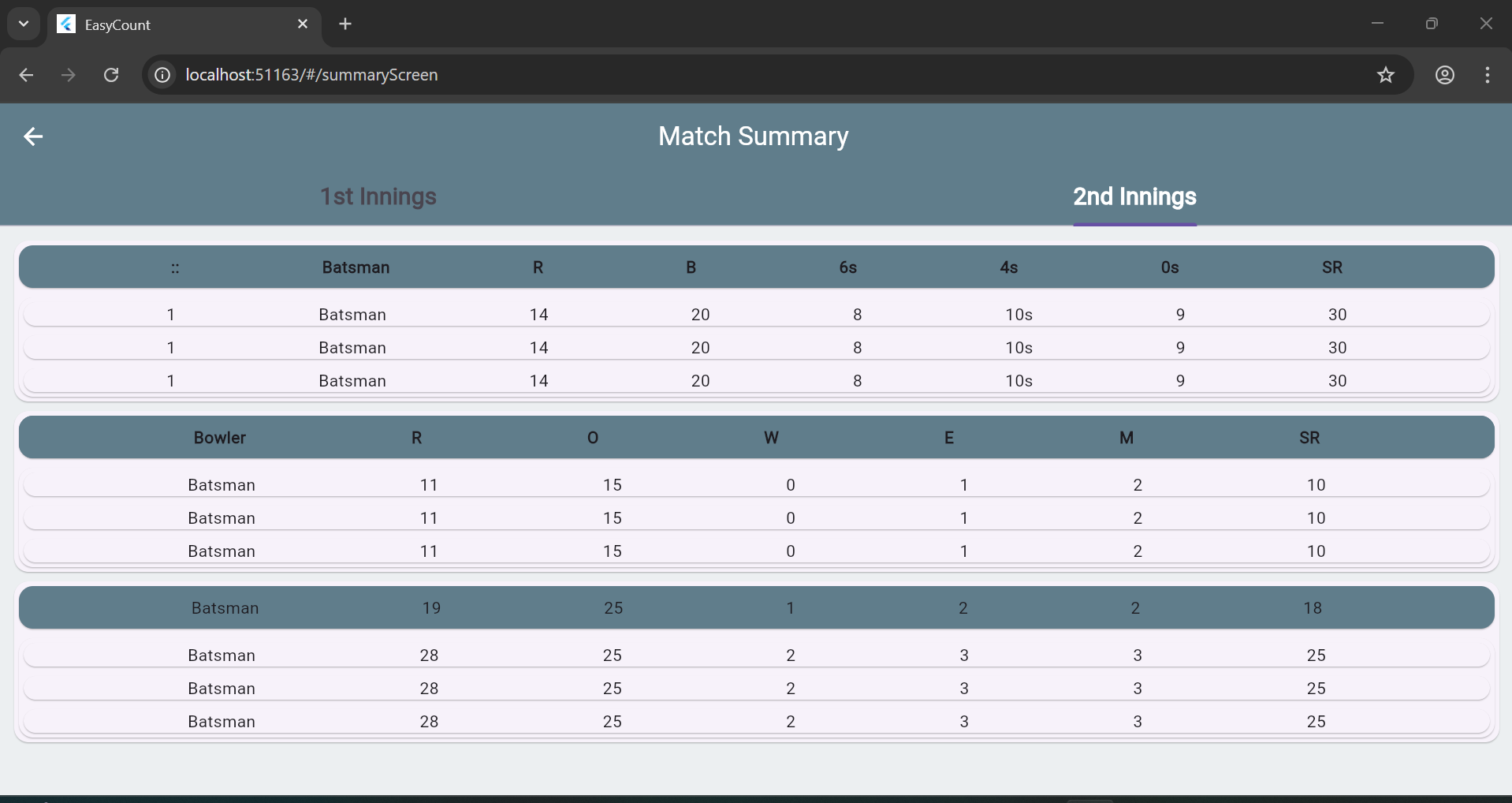
1. **Scoreboard Screen** – Real-time match scoring interface.



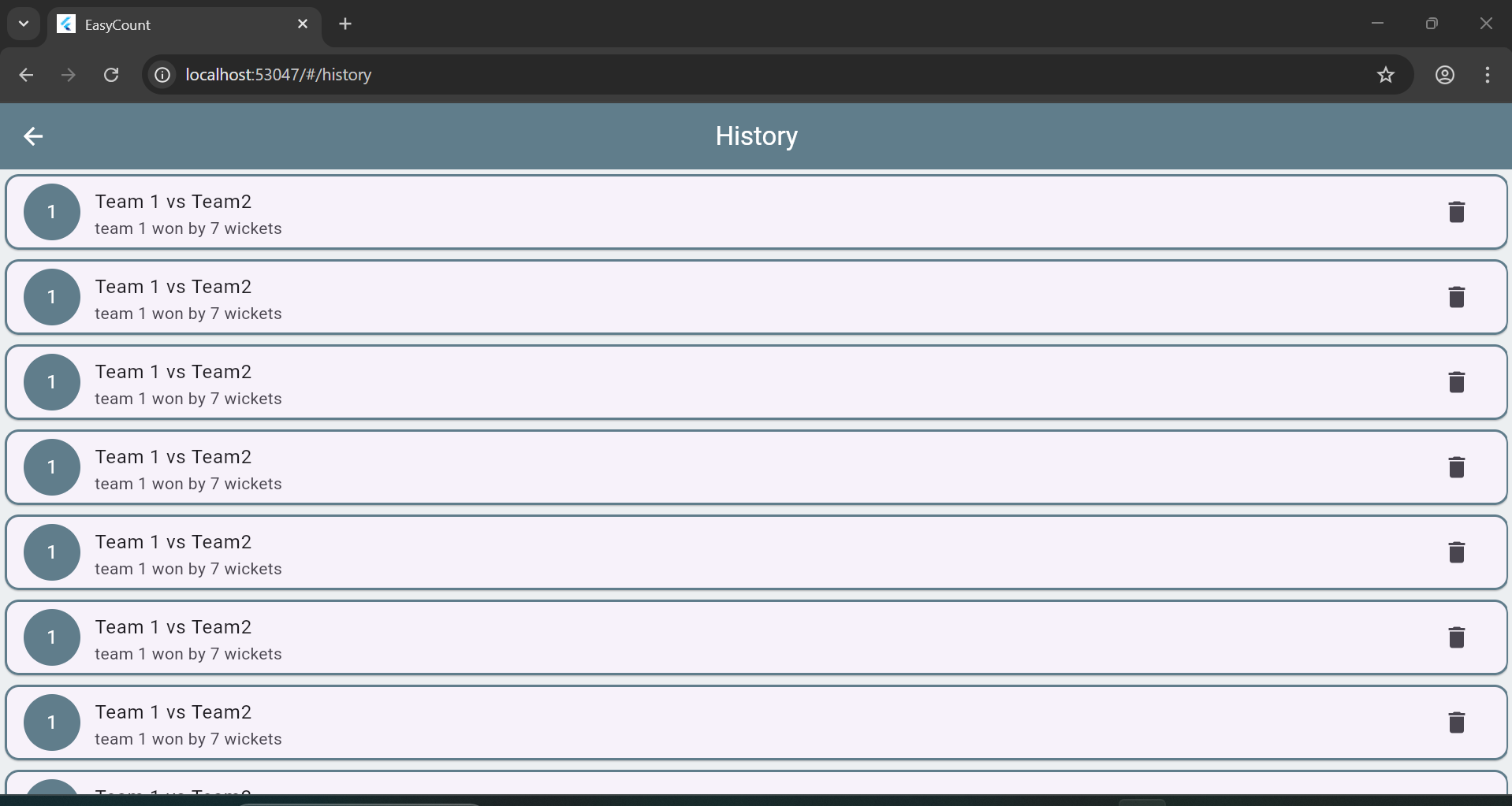


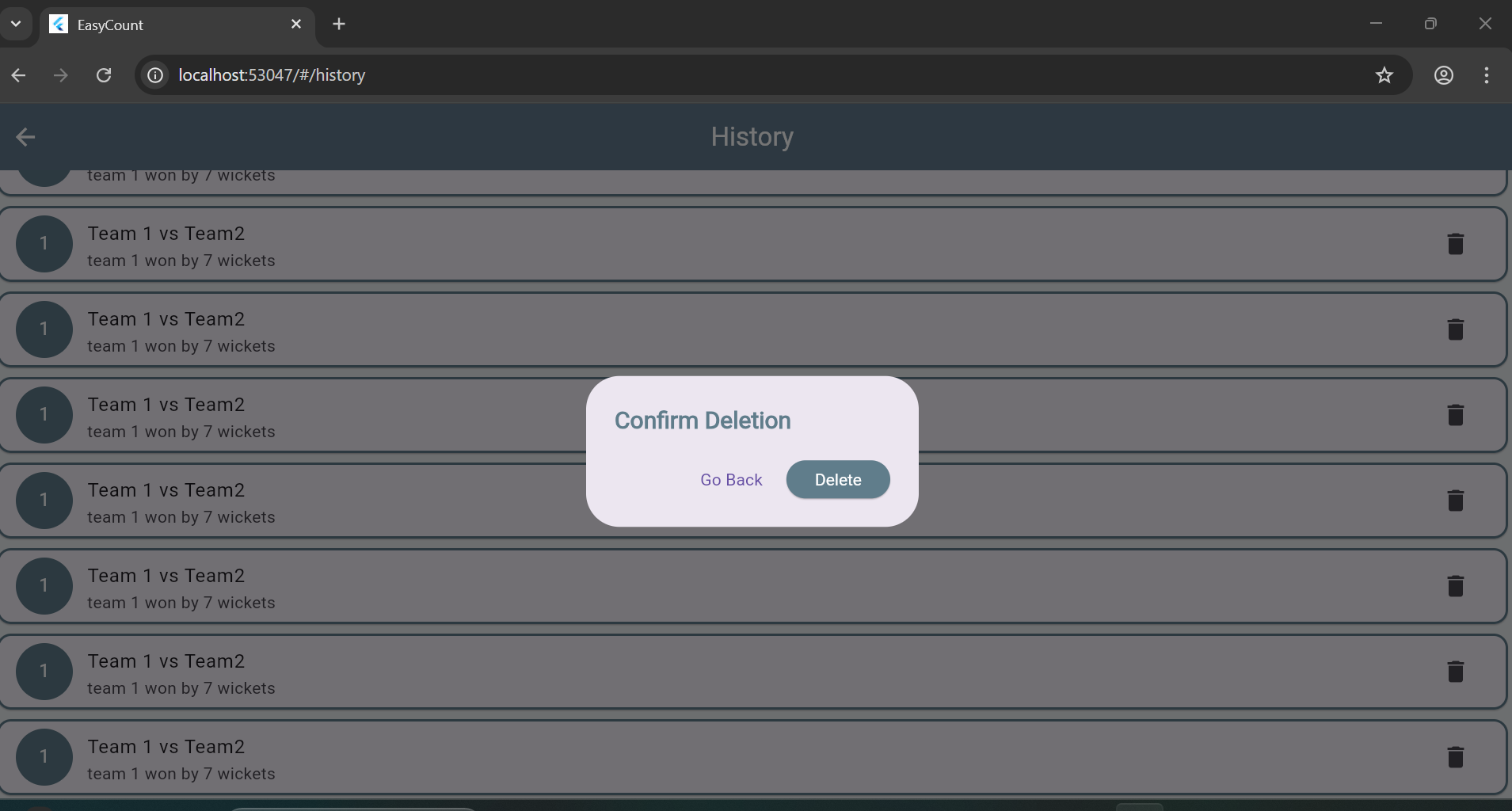
1. **Summary Screen** – Displays 1st and 2nd innings data in tab view.



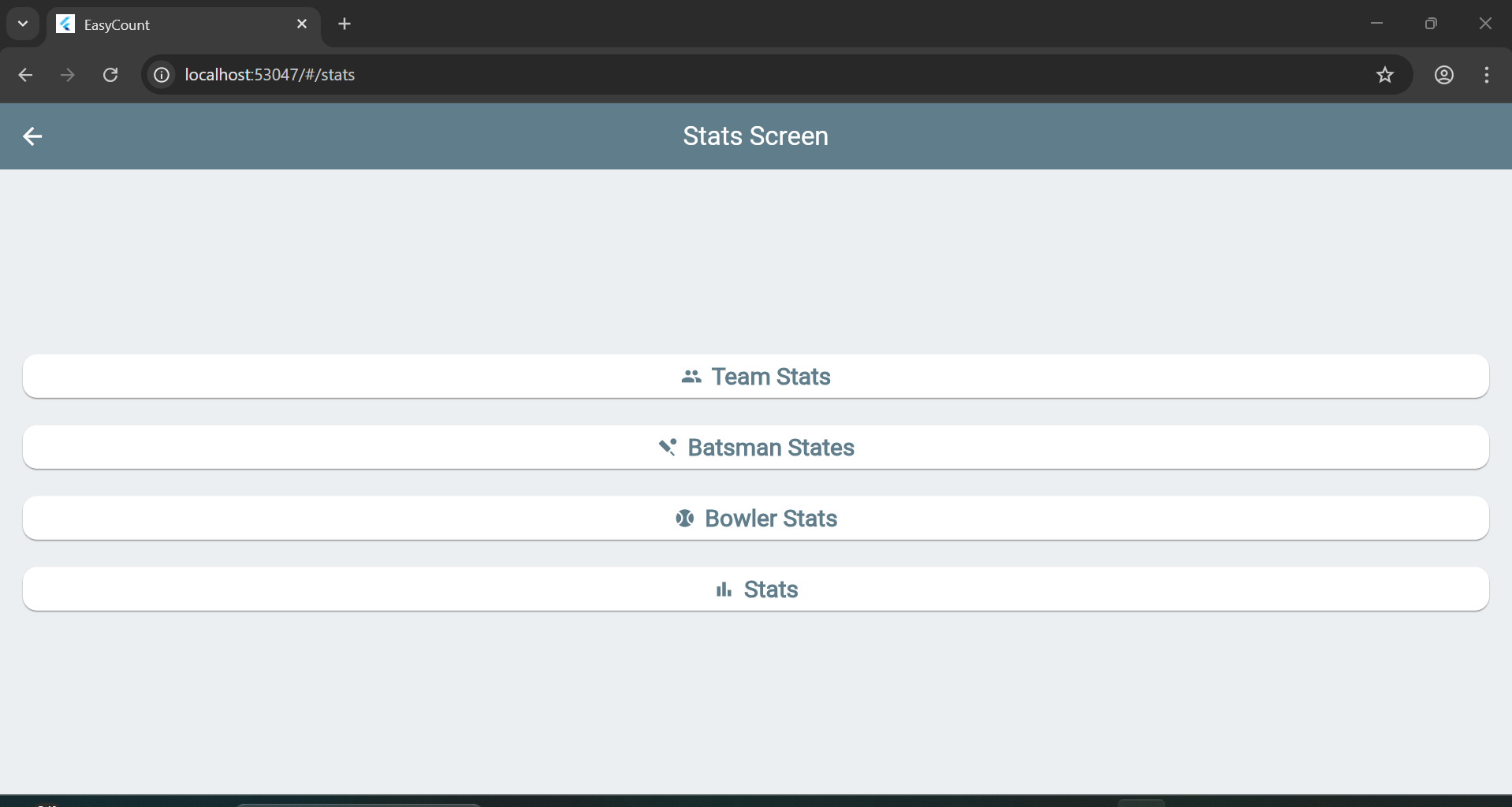


1. **History Screen** – List of previous matches with delete option.

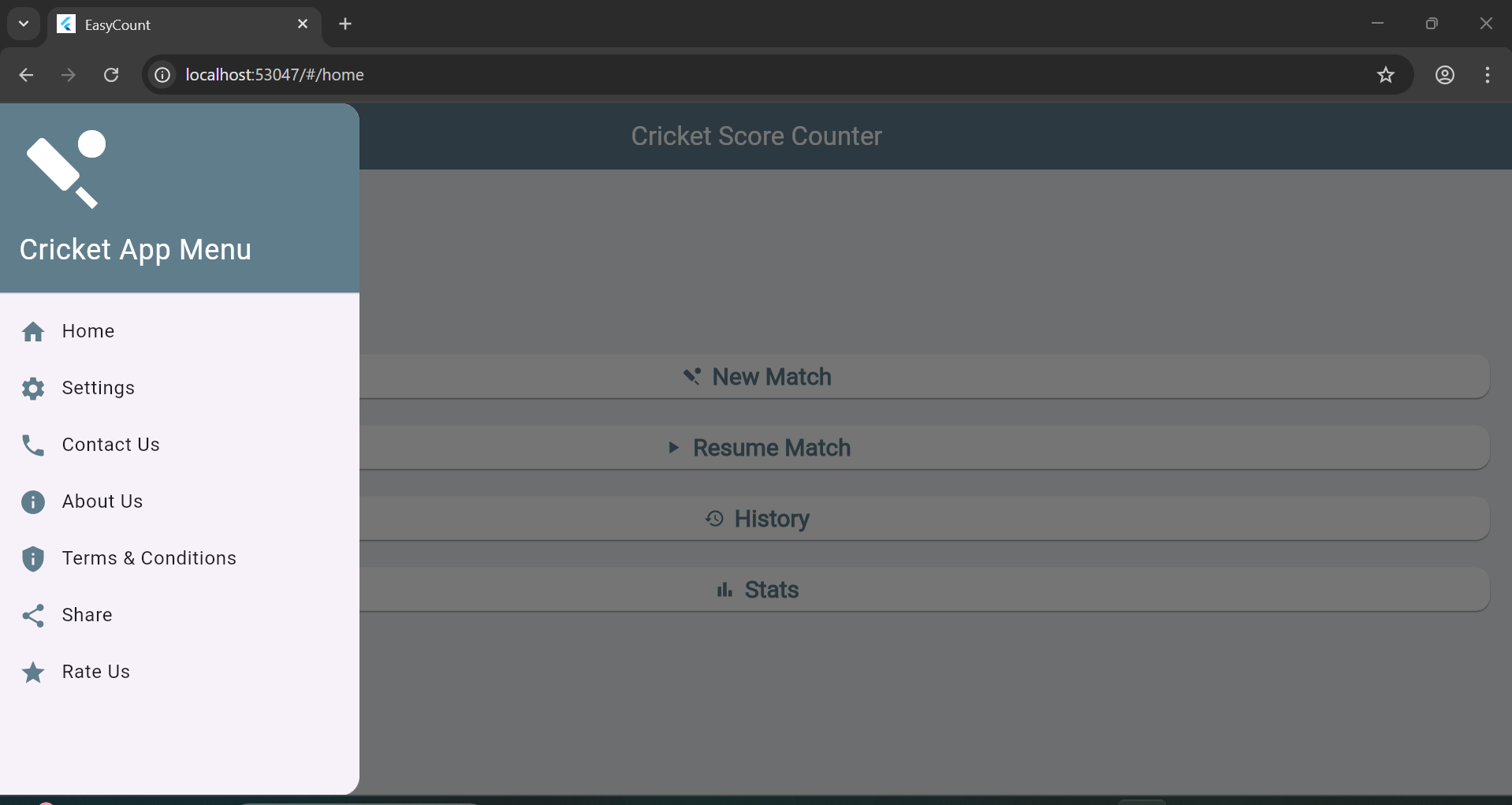


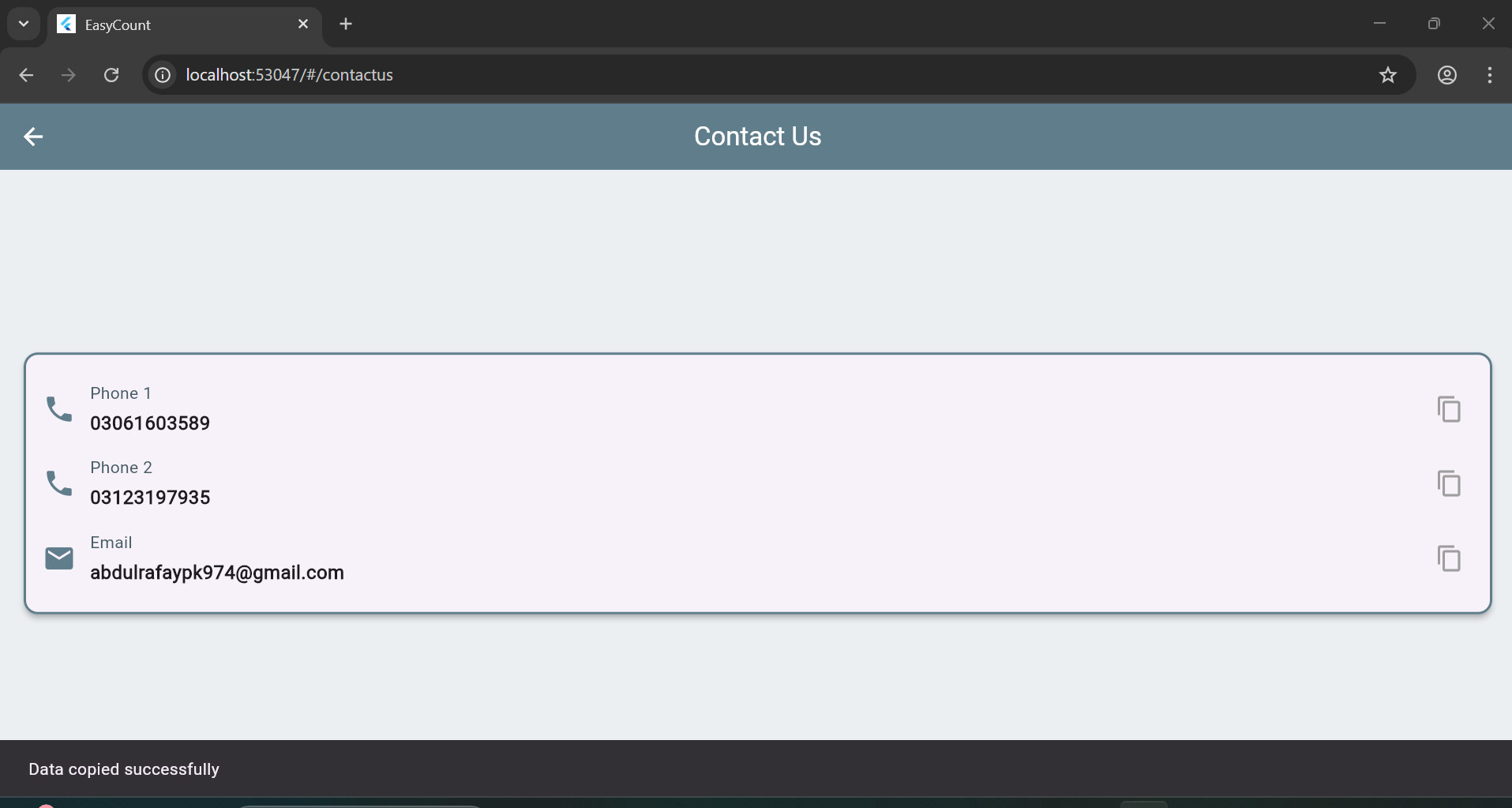


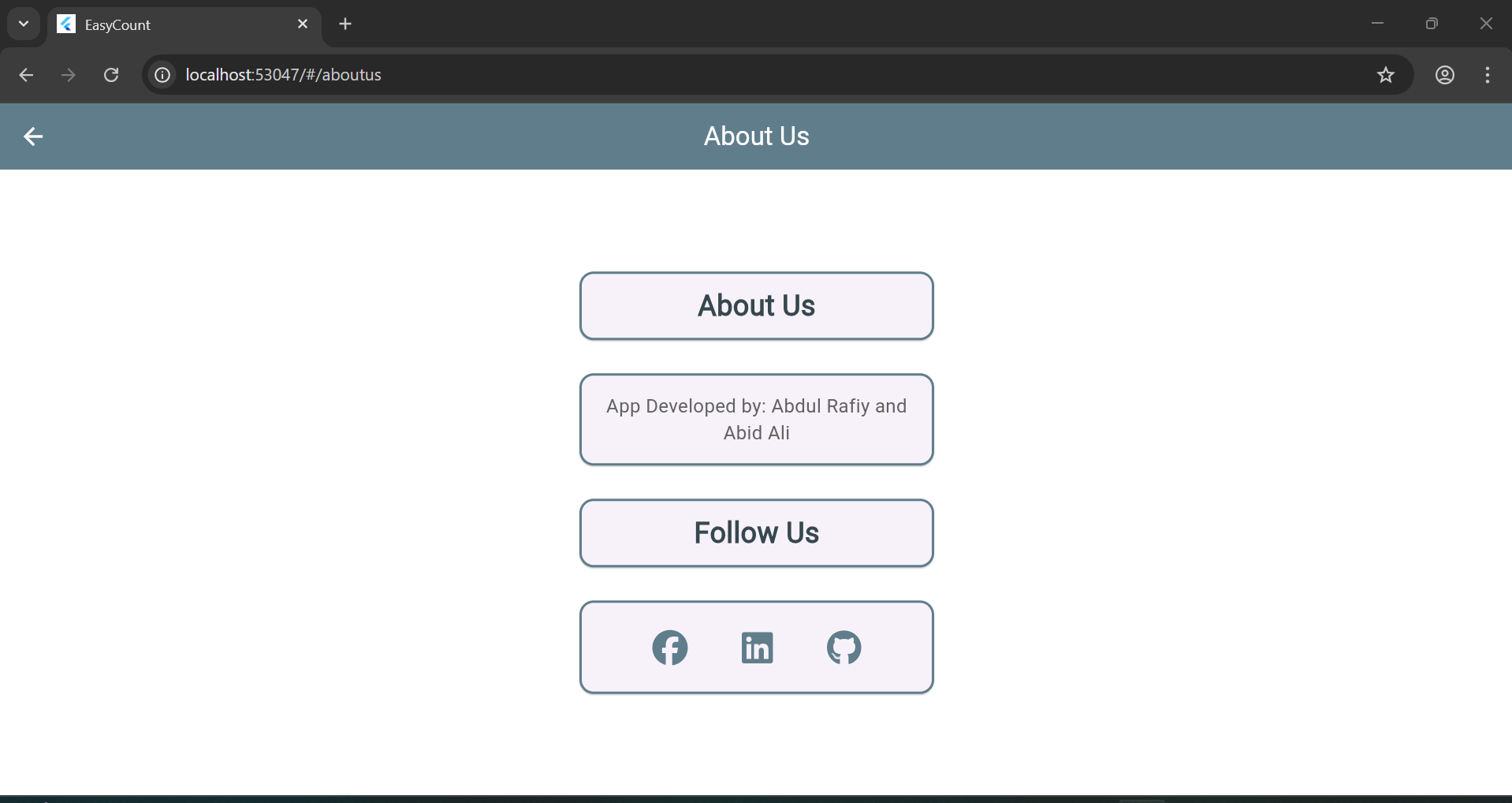
1. **Stats Screen** – Buttons for Team, Batsman, and Bowler stats.

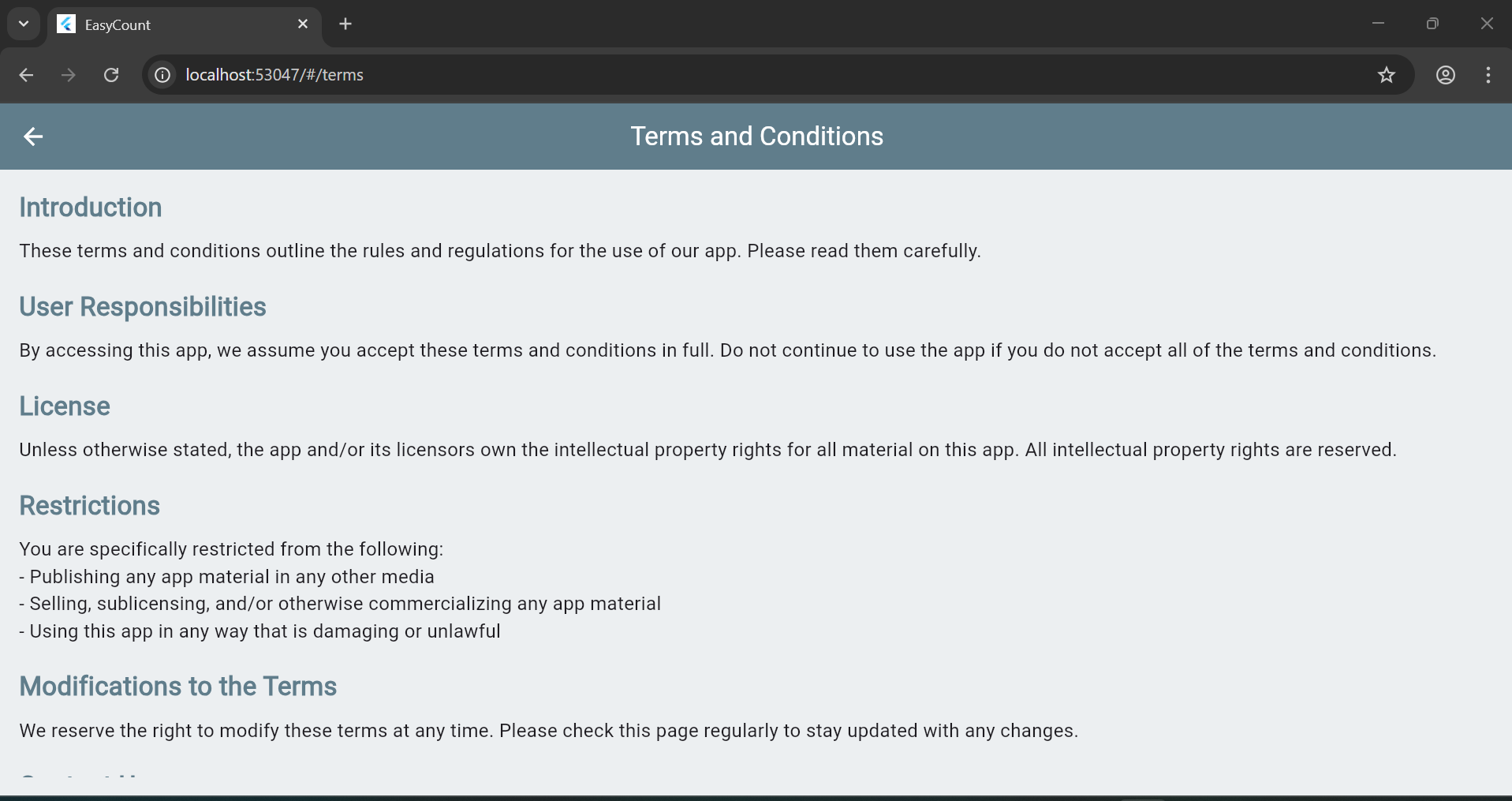


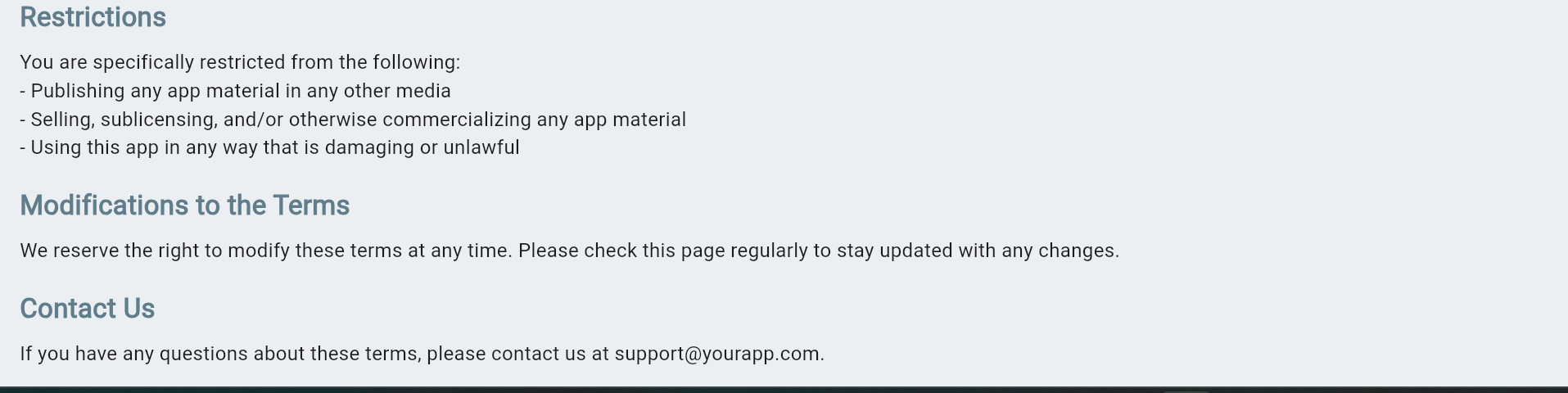
1. **Drawer Screens** – About Us, Contact Us, Terms & Conditions.











## ****6. Technologies and Tools Used****

| **Tool / Technology** | **Purpose** |
| --- | --- |
| Flutter | Cross-platform app development framework |
| Dart | Programming language for Flutter |
| Hive Database | Local NoSQL storage for match data |
| Android Studio | IDE for designing and testing |
| font\_awesome\_flutter | For social media icons |
| url\_launcher | For external social link handling |
| Material Design Widgets | For consistent UI layout |

## ****7. Issues and Bugs Encountered****

| **Issue** | **Description** | **Solution** |
| --- | --- | --- |
| Duplicate State Class Names | All screens used AppState | Renamed to unique \_ScreenState |
| Data not saving | Hive boxes not opened before use | Initialized Hive in main() |
| UI overflow on small devices | Widgets overflowed | Wrapped with SingleChildScrollView |
| Navigation errors | Incorrect route names | Mapped routes properly in main.dart |
| Search bar issues | TextField rebuild delay | Used setState() to refresh results |