Howzatt! Cricket Scorekeeper

Project Report

CS104 Course Project

Nihar Chouhan
Student ID: 24B0942
Department of Computer Science

April 29, 2025

Contents

1 Introduction

1.1 Project Overview

The Howzatt! Cricket Scorekeeper provides is a complete digital solution for scoring cricket matches in real time. Developed as part of the CS104 course requirements, the application implements core scoring functionality along with several advanced features including detailed extras handling, multiple dismissal types, and comprehensive match statistics.

1.2 Technical Objectives

The primary technical objectives included the following:

- Development of a multi-page application with consistent data flow
- Implementation of complex cricket scoring rules in `JavaScript`
- Creation of a responsive interface accessible on various devices
- Persistent data storage across multiple pages using `localstorage`
- Comprehensive state management for all match scenarios

2 System Architecture

2.1 Component Overview

The application consists of four interconnected pages:

- `setup.html`: Initial match configuration
- `setup.css`: Styling of `setup.html`
- `live.html`: Primary scoring interface
- `live.css`: Styling of `live.html`
- `scorecard.html`: Detailed match statistics
- `scorecard.css`: Styling of `scorecard.html`
- `summary.html`: Final results display
- `summary.css`: Styling of `summary.html`
- **score.js**: Core logic and state manager for cricket scoring

2.2 Storage Structure

The application utilizes `localstorage` with the following data structure:

Key	Purpose
`matchSettings`	Stores initial match configuration
`matchState`	Tracks complete current match state
`finalMatchState`	Contains completed match data

Table 1: `localstorage` data structure

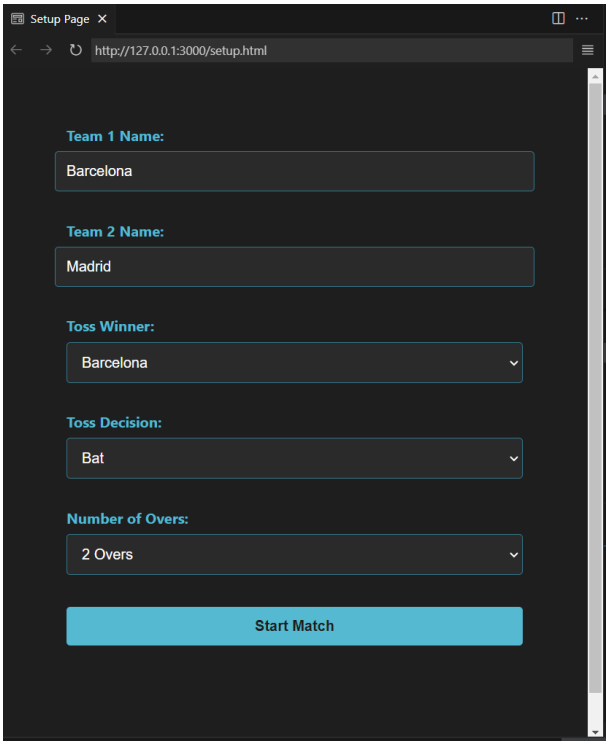

Figure 1: Setup Page

3 Core Implementation

3.1 Setup Page Implementation

The setup interface collects the essential match parameters as shown in 1:

- Team names through text inputs with validation
- Toss outcomes via dropdown selection
- Match length configuration (overs)

Key functions include:

- Input validation before proceeding
- Dynamic dropdown population
- Data serialization for storage

3.2 Live Scoring Page Implementation

3.2.1 Scoring System

The live page implements comprehensive scoring logic:
The live scoring page serves as the central interface for real-time match tracking, featuring:

3.2.2 Setup Initialization

- Requires entry of opening batter names and first bowler before match commencement
- Validates all player names before enabling scoring controls
- Initializes match state with default values

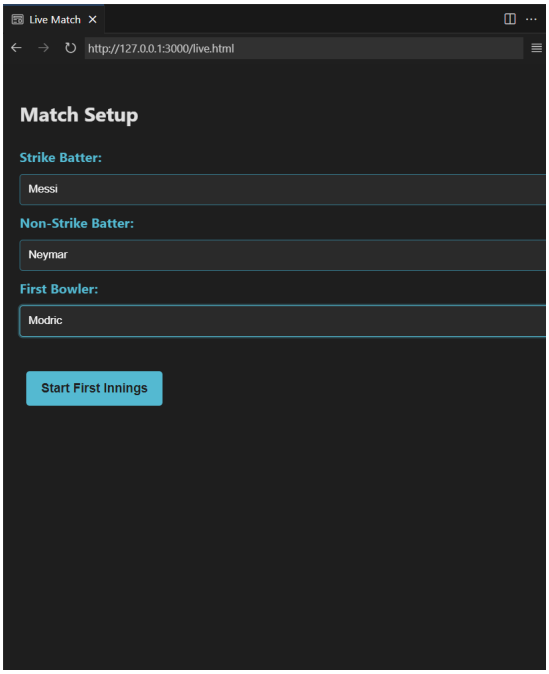

Figure 2: Live Page Setup

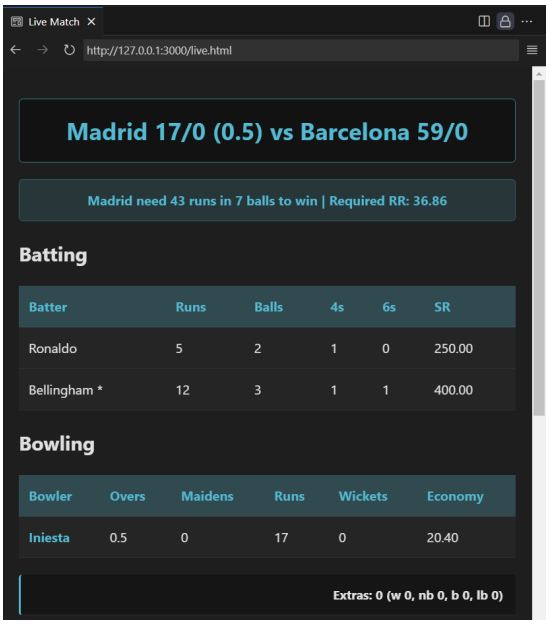

Figure 3: Live Page Score

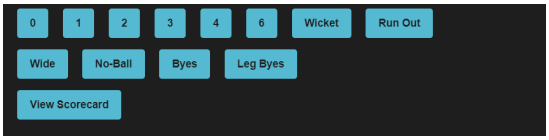

Figure 4: Live Page Buttons

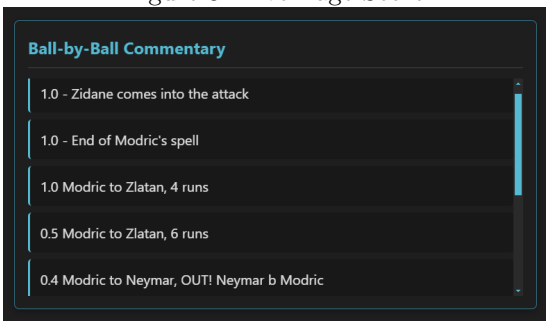

Figure 5: Live Commentary

3.2.3 Score Display

- Shows current innings score in standard cricket format (runs/wickets)
- Displays overs completed in X.Y format (overs.balls)
- Calculates and shows current run rate (CRR)
- For second innings, shows required run rate (RRR) and target

3.2.4 Interactive Buttons

Run Buttons (0-6)

- Record runs scored off each delivery
- Automatically rotate strike for odd-numbered runs
- Update batter statistics (runs, balls, strike rate)
- Track boundary counts (4s, 6s)

Wicket Button

- Opens dismissal modal with type selection
- Records dismissal method (bowled, caught, LBW, etc.)
- Prompts for new batter name
- Updates bowling figures

Extras Buttons

- Wide (+1 run, no ball counted)
- No Ball (+1 run, batter runs count)
- Byes/Leg Byes (runs to extras only)
- Each opens modal for additional runs

3.2.5 Modal Dialogs Implementation

- **Innings Transition**: Swaps batting teams and sets target (See 6)
- **New Bowler**: Rotates bowlers after each over
- **Extras**: Handles special deliveries (wides/no-balls/byes) (See 7
- **Wicket**: Manages dismissals and batter changes (See 8)
- **Run Out**: Processes runs and position changes (See 9)

Common features: Validation, persistent state, and UI consistency.

3.2.6 Live Commentary

- Generates ball-by-ball updates in natural language
- Tracks sequence: "Bowler to Batter, outcome"
- Includes special notations for:
  - Wickets (dismissal type)
  - Boundaries (4 or 6)
  - Extras (type and runs)
- Maintains scrollable history of last 50 events
- Updates in real-time after each ball

3.2.7 Match Progression

- Automatically handles over completion
- Tracks bowler changes with over limits
- Manages innings transitions
- Updates all displays simultaneously
- Preserves complete match state after each action

The interface combines these elements in a responsive layout that mirrors professional cricket scoring systems while maintaining intuitive usability. All data persists through page refreshes through `localstorage` integration.

3.3 Scorecard Page Implementation

3.3.1 Data Presentation

The scorecard displays (see 10, 11):

- Complete batting statistics for all players
- Detailed bowling figures with economy rates
- Extras breakdown by type
- Match progression status

3.3.2 Rendering Logic

Key aspects include:

- Dynamic table generation from match state
- Highlighting of current batters
- Accurate formatting of statistical data
- Handling of inning transitions

3.4 Match Summary Page Implementation

3.4.1 Result Calculation

The summary page determines:

- Match outcome (win/loss/tie)
- Victory margin (runs or wickets)
- Score comparison between innings

3.4.2 User Interface

Features include (see 12):

- Clear result declaration
- Key statistics highlights
- Match reset functionality

4 Advanced Features

4.1 Comprehensive Extras Handling

The application implements all standard cricket extras:

Extra Type	Scoring Rules	Implementation
Wide	+1 run, no ball counted	Special case in scoring logic
No Ball	+1 run, batter runs count	Additional run handling
Byes	Runs to extras only	Counts as ball faced
Leg Byes	Runs to extras only	Counts as ball faced

4.2 Dismissal Types Implemented

The system handles multiple dismissal scenarios:

- **Bowled**: Batter is bowled
- **Caught**: Batter is caught out
- **LBW**: Leg before wicket
- **Run Out**: Batter is run out
- **Stumped**: Wicketkeeper stumps batter
- **Hit Wicket**: Batter hits own wicket

4.3 Run Out Scenarios

Special handling for run outs includes:

- Runs scored before dismissal
- Selection of which batter was out
- New batter positioning
- Accurate ball counting
- Complex strike rotation handling

4.4 Ball-by-Ball Commentary

The live commentary feature provides the following as shown in 5:

4.4.1 Basic Events

- **Runs**: `2.3 Bumrah to Kohli, 4`
- **Dot Ball**: `3.1 Shami to Smith, no run`

4.4.2 Wickets

- **Bowled**: `5.2 Jadeja to Root, OUT! Bowled`
- **Caught**: `7.4 Cummins to Kohli, OUT! c Head b Cummins`

4.4.3 Extras

- **Wide**: `4.5 Archer, WIDE + 1`
- **No Ball**: `6.3 Starc, NO BALL + 6`

4.4.4 Match Transitions

- **Over End**: `Bumrah comes into the attack`
- **Innings Break**: `Team A 35/1 (2 overs)`

5 Technical Challenges

5.1 Data Management Across Pages

Significant challenges were encountered in:

- Maintaining data consistency across multiple HTML files
- Synchronizing state during page transitions
- Handling inning transitions in `scorecard.html`
- Preserving complex state in `localstorage`

5.2 Inning Transition Implementation

Key difficulties included:

- Proper target score calculation
- Complete state reset between innings
- Batter and bowler reinitialization
- Accurate data preservation from first innings

5.3 Strike Rotation Complexity

Particular challenges arose with:

- Run out scenarios and subsequent strike management
- Odd-run strike rotation
- Wicket fall and new batter positioning
- Over completion and automatic rotation

6 Conclusion

6.1 Key Achievements

The application successfully implements:

- Complete cricket scoring system
- Responsive cross-device interface
- Persistent data storage
- Comprehensive match analysis
- All specified custom features

6.2 Technical Learnings

Key technical insights gained:

- Complex state management techniques
- Web storage API applications
- Cricket rules implementation challenges
- Cross-page data synchronization

6.3 Future Enhancements

Potential improvements include:

- Multi-match statistics tracking
- Player performance analytics
- Advanced visualization features

References

[1] ESPNcricinfo, *Official Cricket Scoring Rules*

[2] W3Schools, *HTML, CSS, and JavaScript Tutorials*, `https://www.w3schools.com`

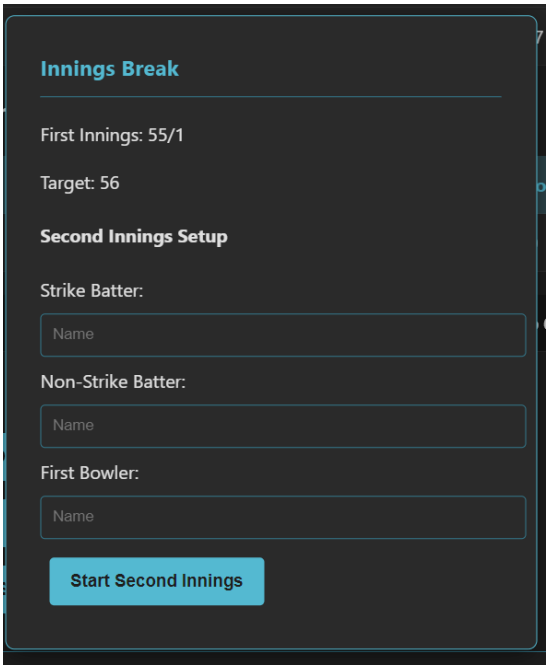

Figure 6: Innings Transition Modal

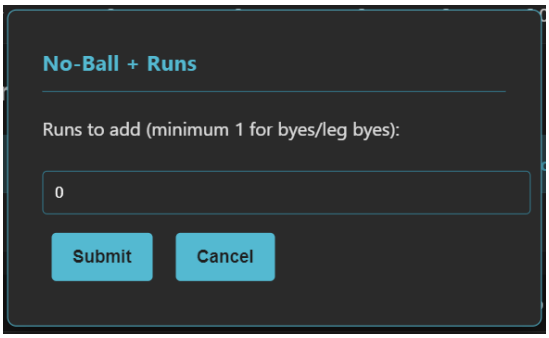

Figure 7: Extras Modal

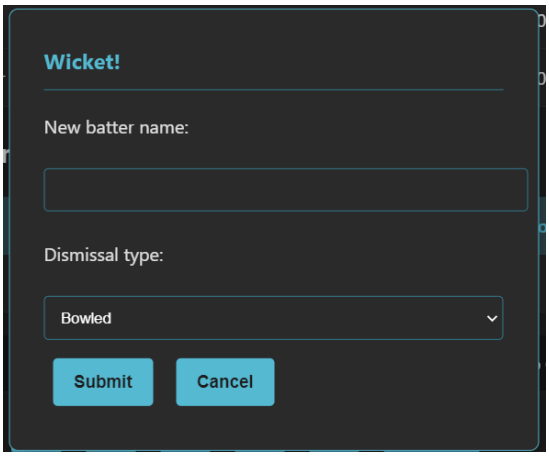

Figure 8: Wicket Modal

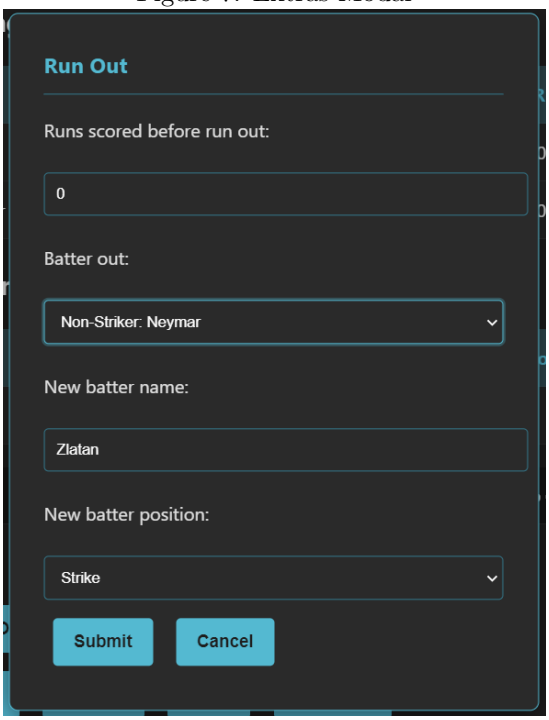

Figure 9: Run Out Modal

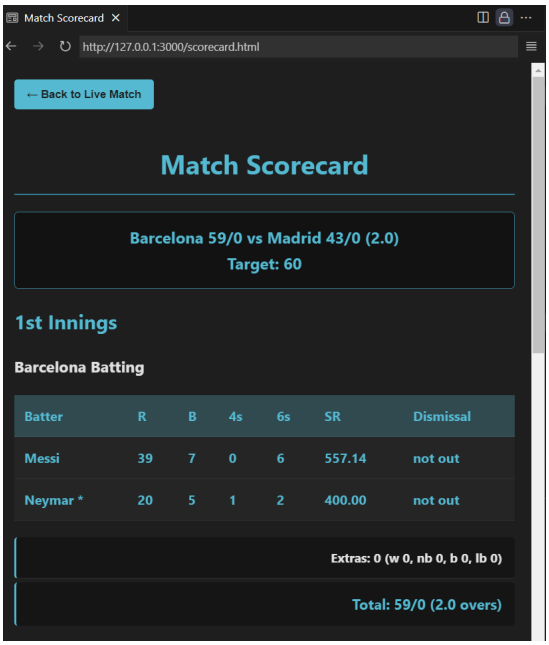

Figure 10: Scorecard 1

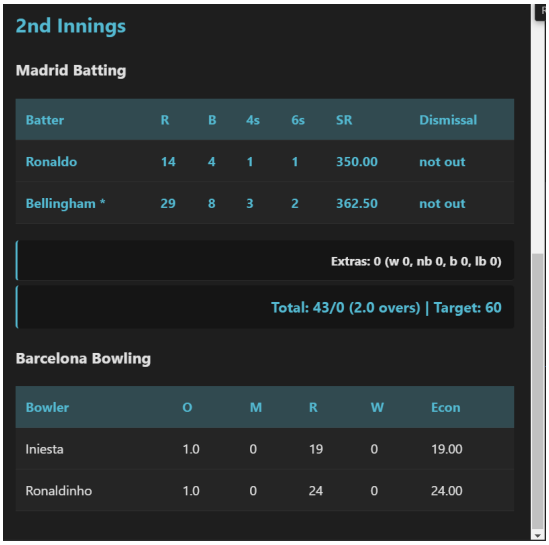

Figure 11: Scorecard 2

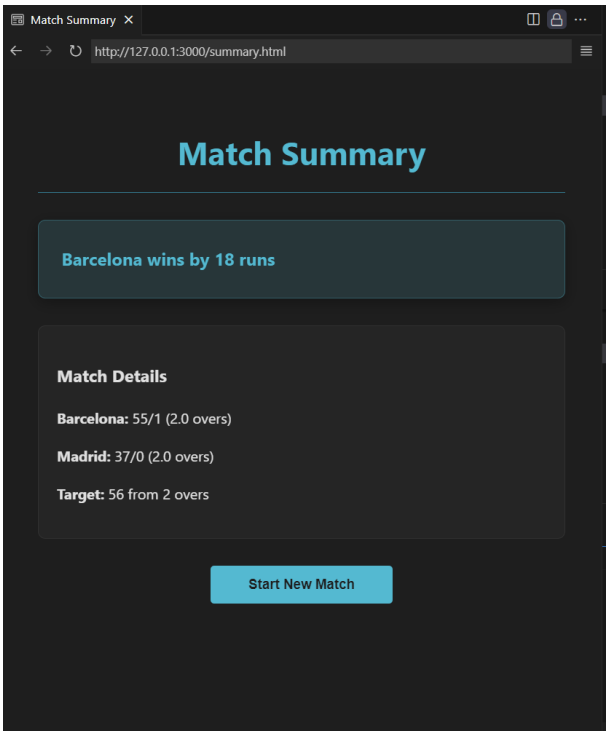

Figure 12: Match Summary Page