CS104 Project Report: Cricket Scorekeeper Web Application

Sanchana (22B1034)

April 29, 2025

Introduction

This report documents the development of a comprehensive cricket scorekeeper web application built using HTML5, CSS3, and JavaScript (ES6). The application provides real-time match tracking with automatic statistics calculation and result determination.

Project Overview

The application features a four-page workflow with the following components:

- **Setup Page**: Initial configuration with team details, toss results, and match format
- **Live Scoring Page**: Interactive interface for real-time match tracking with commentary
- **Scorecard Page**: Detailed statistical breakdown with innings-wise data
- **Summary Page**: Final match result presentation with key metrics

Technical Implementation

Core Architecture

The application follows a client-side MVC pattern with:

- **Model**: Match state stored in localStorage
- **View**: HTML/CSS presentation layer
- **Controller**: JavaScript event handlers and business logic

State Management

The match state object maintains all critical match information:

```
const matchDetails = {
    team1Name: '',          // Original team names
    team2Name: '',
    firstInningsTeam: '',   // Team batting first
    secondInningsTeam: '',  // Team batting second
    battingTeam: '',        // Current batting team
    bowlingTeam: '',        // Current bowling team
    tossWinner: '',         // Team that won toss
    tossDecision: '',       // 'bat' or 'field'
    overs: 2,               // Match length
    currentInnings: 1,      // 1 or 2
    innings1: {             // First innings data
        runs: 0,
        wickets: 0,
        balls: 0,
        extras: 0,
        batsmen: [],        // Array of batter objects
        bowlers: [],        // Array of bowler objects
        commentary: []      // Custom commentary feature
    },
    innings2: { /* ... */ } // Second innings data
};
```

Key Features

Innings Management

The application handles innings transitions through:

```
function endInnings() {
    if (isFirstInnings) {
        // Track batting order
        matchDetails.firstInningsTeam = matchDetails.battingTeam;
        matchDetails.secondInningsTeam = matchDetails.bowlingTeam;

        // Swap teams for second innings
        matchDetails.battingTeam = matchDetails.secondInningsTeam;
        matchDetails.bowlingTeam = matchDetails.firstInningsTeam;

        // Update state
        isFirstInnings = false;
        matchDetails.currentInnings = 2;
    }
    // ... rest of function
}
```

Real-time Scoring

The scoring system features:

- Run recording (0-6)

- Wicket tracking
- Extra management (wides, no-balls)
- Automatic strike rotation
- Over completion detection

Customization: Commentary System

As a special customization, I implemented a real-time commentary feature that:

- Generates context-aware messages for each event
- Stores commentary with timestamp and score context
- Displays in a scrollable feed with latest events first

```
function addCommentary(message) {
    const innings = isFirstInnings ? 'innings1' : 'innings2';
    const over = Math.floor(matchDetails[innings].balls / 6);
    const ball = (matchDetails[innings].balls % 6) + 1;

    matchDetails[innings].commentary.unshift({
        displayOver: over,
        displayBall: ball,
        message: message,
        timestamp: new Date().toLocaleTimeString(),
        score: `${matchDetails[innings].runs}/${matchDetails[innings].
    wickets}`
    });
    updateCommentaryUI();
}
```

Data Flow

The application manages data through:

Initialization

1. User inputs match details on setup page
2. System initializes match state object
3. Data persists to localStorage

Live Updates

1. User records match events via scoring buttons
2. System updates match state
3. UI refreshes to reflect changes
4. Commentary generated for significant events

Results Calculation

The summary page automatically determines the winner:

```
if (matchDetails.innings2.runs >= target) {
    // Chasing team won
    resultText = `${secondBattingTeam} wins by ${10-wicketsLost}
   wickets`;
} else {
    // Defending team won
    resultText = `${firstBattingTeam} wins by ${target-runsScored-1}
   runs`;
}
```

Complete File Structure

```
cricket-scorekeeper/
 setup.html        # Match setup page
 setup.css         # Setup page styles
 live.html         # Live scoring page
 live.css          # Live page styles
 scorecard.html    # Scorecard page
 scorecard.css     # Scorecard styles
 summary.html      # Summary page
 summary.css       # Summary styles
 commentary.css    # Commentary section styles (customization)
 score.js          # Core application logic
```