TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

A Proposal On

IPL Archive System

Submitted By:
Aditya Timalsina (077BEI009)
Ashutosh Bohara (077BEI012)
Pranav Joshi (077BEI029)

Submitted To:
Bibha Sthapit
Department of Electronics
and Computer Engineering

1. Introduction

The game of cricket is a complex sport involving many variables to be considered by both players and the management team. Each inning of a cricket game generates a lot of data that, for the purpose of archiving, must be stored properly. Currently, the most popular cricket league is the Indian Premier League (IPL) with over 170 players joining in from across the world. The IPL is played in seasons occurring in the interval of 1 year, with each season having up to 80 games. With each new season of IPL, new players are added, and some players may retire.

IPL Archival System is a comprehensive cricket database, modelled on the Indian Premier League. It can work as a resource for cricket enthusiasts with a repository of one of the most prestigious cricket leagues globally. The system will provide access to an expansive collection of match records, player statistics, team performances, and a myriad of additional resources. Furthermore, the system will be accessible through a web-based user interface making it convenient and easy-to-use for the end user. The system will make the process of data retrieval, and performing complex queries effortless.

Although we will be designing our system around the Indian Premier League, it can be used with any other cricket leagues such as DPL, PPL, and Nepal T20 League. We decided to go with IPL, as the data for IPL games are more easily available online. However, the system can be exported to be used with other leagues with minimal redesign.

We plan on providing the following information from the database:

- Information about each season (winner, awards, etc)
- Information about each game (lineup, runs, best player, etc)
- Player Stats (for each game or overall stats)
- Team Stats (past and present players, trophies, performance analysis)
- Auction (the transfer of players across teams)

2. Problem Statement

A cricket game generates a lot of data, and storing and managing such data with the existing one is a daunting task. With every new season of IPL, new players are added, and upwards of 80 games are played. A record of each such game must be kept for reference or as a resource for enthusiasts. With that said, we present the following problem statement:

To address the need for archiving for IPL, and possibly other cricket leagues, we must present a robust and efficient system capable of storing large amounts of data associated with cricket games and players and performing complex queries on the available data to retrieve the required information. The system needs to be scalable and versatile enough to be used with other leagues without much redesign.

3. Technologies Used

A web app has a three-tier architecture: front-end, back-end, and database. Depending on the practicality of the project, different organizations/developers utilize different programming languages. Considering that our project is primarily concerned with databases, we will prioritize database design above front-end and back-end design. We will be designing the front-end with React, the back-end with GoLang, and the database with PostreSQL.

3.1. PostgreSQL

As we were instructed to choose SQL database for our project, we picked PostgreSQL as the best option. PostgreSQL is a robust and feature-rich relational database management system that offers exceptional benefits for any web application. With PostgreSQL, we can ensure the reliability, scalability, and performance of our web application's data storage layer. Its ACID-compliant nature guarantees data integrity, providing a solid foundation for any application's critical information. PostgreSQL's advanced indexing techniques, query optimization capabilities, and support for parallel query execution empowers an application to handle large datasets and high traffic loads with ease. The flexibility of PostgreSQL allows to model complex data structures and leverage advanced features like JSON support, full-text.

3.2. Go

We will be using Go (also known as Golang) for backend design. Go is a modern, statically typed programming language that was created primarily for creating efficient and scalable back-end systems. Go has various advantages that make it an appealing choice for backend development. It has a basic and concise syntax, making it easier to read and maintain the code. Concurrency capabilities inherent into Go, such as goroutines and channels, allow for effective parallelism and management of high-concurrency workloads, resulting in good performance. In addition, the language has a robust standard library that covers a wide range of capabilities, decreasing the requirement for external dependencies. The compilation process in

Go generates standalone executables, making deployment and distribution simple. Furthermore, Go's emphasis on simplicity and ease of use results in shorter development cycles, making it a perfect choice for projects with short development cycles. Overall, Go's combination of efficiency, simplicity, scalability, and built-in concurrency features makes it an appropriate choice for our project's backend language.

3.3. React

We will be using React, a JavaScript library, for frontend design. React is a well-known JavaScript toolkit for creating user interfaces, distinguished by its component-based architecture and virtual DOM rendering. It has various advantages that make it an excellent choice for our project. We can design reusable UI components with React, allowing for rapid development and simple maintenance of large-scale apps. React's virtual DOM renders more effectively and improves performance. Its declarative approach makes designing interactive UIs easier by expressing how the UI should look dependent on the application's state. Additionally, React has a big ecosystem of community-driven libraries and tools, making it highly versatile and capable of satisfying a wide range of development demands, making it an appropriate tool for our project.

4. Expected Result

With this project, we aim to build a comprehensive system for the storage and retrieval of data in a cricket league-based competition. There will be different assignments for the league stage and the playoffs stage of the competition. Team statistics and player statistics will be generated on the basis of entered game data. A league table based on the match data for each match will be generated, which will evaluate the teams to qualify for the playoffs. A standard playoff configuration will be implemented from which the tournament will proceed towards its denouement. Individual statistics for all the players for different metrics on behalf of their contribution in different departments of the game will be made. Individual awards for standout players in a metric will be evaluated as well. On the frontend department, we aim to implement a simple user interface for all the data entry and retrieval from our archival system.