CS 666: Information Retrieval and Web Agents

Project Report

"CricScorer"

Aayush Mishra (amishr24@jh.edu)

May 9, 2023

About

CricScorer is a ball-by-ball scorer for Cricket matches. It schedules daily data scrapers from the ESPNCricInfo website. This data is then robustly parsed and stored in a human readable and further processable JSON format. This data is essential for application systems that require highest resolution stats for cricket.

Application

A Fantasy Cricket application (discussed later) which uses parts of this project can be installed and run using this APK (only Android supported right now).

Motivation

CricScorer aims to solve the availability challenge of ball-by-ball data for Cricket matches. Many websites show live coverage of the match but this data is not readily available for public usage. With ball-by-ball data, which is the highest resolution score data for cricket matches, all possible stats for players, teams, etc. can be generated. These stats have numerous downstream applications, one of which: Fantasy Cricket, is mildly explored in this project.

1 Working Details

We used ESPNCricInfo as the data source for our project. The website's *robots.txt* allows all agents to access their data, which conveniently avoids any IP issues.

Each day at midnight Indian Standard Time, a scheduler looks for upcoming matches of the day and filters interesting matches according to a priority generator. This is flexible and can be changed according to the needs of the user. Currently, we only collect data from the T20 matches of the ongoing Indian Premier League.

The scheduler initiates recorders and parsers for that match at its scheduled start time, which work with pre-defined frequency and timeouts. Each live match has lots of links associated with it with different information. We currently use every match's **live-cricket-score**, **full-scorecard**, **match-squads** and **match-playing-xi** links to parse all required information.

The parser is the critical component of this system, which must handle ever-evolving edge cases. Once the data is parsed a final JSON is produced which contains all information about the match.

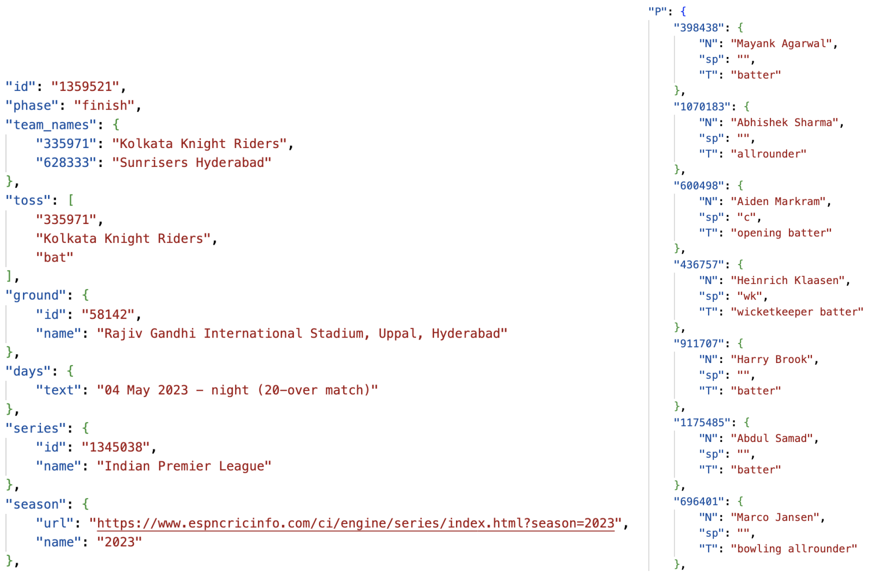
2 Data Format

The final produced JSON conatins information about teams, venue, dates, toss, scorecard, result, player of the match as shown below (Figure 1, along with ball-by-ball data.

The ball-by-ball data is a list of dictionaries of the format shown in figure 2.

3 How to use this project?

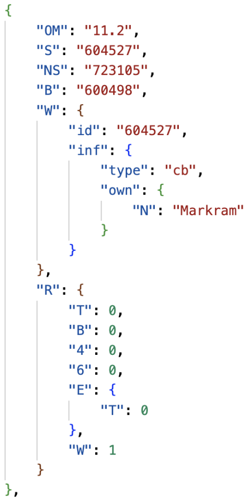
After installing python3 and all required dependencies (included in the zipped code files), one can simply run the program using ***python main.py*** which would start scheduling for matches from the next day. The collected data can later be found in the *data* folder of the root directory.



```
"id": "1359521",
"phase": "finish",
"team_names": {
    "335971": "Kolkata Knight Riders",
    "628333": "Sunrisers Hyderabad"
},
"toss": [
    "335971",
    "Kolkata Knight Riders",
    "bat"
],
"ground": {
    "id": "58142",
    "name": "Rajiv Gandhi International Stadium, Uppal, Hyderabad"
},
"days": {
    "text": "04 May 2023 - night (20-over match)"
},
"series": {
    "id": "1345038",
    "name": "Indian Premier League"
},
"season": {
    "url": "https://www.espncricinfo.com/ci/engine/series/index.html?season=2023",
    "name": "2023"
},
```

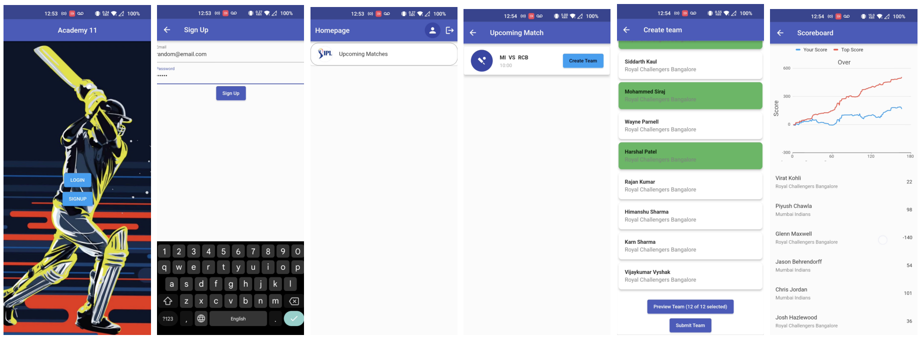
```
"P": {
    "398438": {
        "N": "Mayank Agarwal",
        "sp": "",
        "T": "batter"
    },
    "1070183": {
        "N": "Abhishek Sharma",
        "sp": "",
        "T": "allrounder"
    },
    "600498": {
        "N": "Aiden Markram",
        "sp": "c",
        "T": "opening batter"
    },
    "436757": {
        "N": "Heinrich Klaasen",
        "sp": "wk",
        "T": "wicketkeeper batter"
    },
    "911707": {
        "N": "Harry Brook",
        "sp": "",
        "T": "batter"
    },
    "1175485": {
        "N": "Abdul Samad",
        "sp": "",
        "T": "batter"
    },
    "696401": {
        "N": "Marco Jansen",
        "sp": "",
        "T": "bowling allrounder"
    },
```

Figure 1: Meta data and player info generated about a sample match.



```
{
    "OM": "11.2",
    "S": "604527",
    "NS": "723105",
    "B": "600498",
    "W": {
        "id": "604527",
        "inf": {
            "type": "cb",
            "own": {
                "N": "Markram"
            }
        }
    },
    "R": {
        "T": 0,
        "B": 0,
        "4": 0,
        "6": 0,
        "E": {
            "T": 0
        },
        "W": 1
    }
},
```

Figure 2: Sample data about one ball. Contains over marker, striker, non-striker, bowler, wicket info (if any) and run info (including extras).



Email
random@email.com
Password
••••••

MI VS RCB
10:00

Siddarth Kaul
Royal Challengers Bangalore

Mohammed Siraj
Royal Challengers Bangalore

Wayne Parnell
Royal Challengers Bangalore

Harshal Patel
Royal Challengers Bangalore

Rajan Kumar
Royal Challengers Bangalore

Himanshu Sharma
Royal Challengers Bangalore

Karn Sharma
Royal Challengers Bangalore

Vijaykumar Vyshak
Royal Challengers Bangalore

Virat Kohli Royal Challengers Bangalore	22
Piyush Chawla Mumbai Indians	98
Glenn Maxwell Royal Challengers Bangalore	-140
Jason Behrendorff Mumbai Indians	54
Chris Jordan Mumbai Indians	101
Josh Hazlewood Royal Challengers Bangalore	36

Figure 3: The app experience

4 Academy 11

Inspired from the memes that celebrate players who perform poorly, we developed a new fantasy Cricket application that lets users pick a team which is supposed to perform the **worst** in a given match. We call it *Academy 11*. For this application, ***CricScorer*** also schedules fantasy score calculators for live matches. This data is then accessed through APIs which gives a real time user experience. To enjoy this game, use the android application linked on the first page. Sample screenshot of the app experience can be seen in figure 3. Users can also access the APIs directly, to get upcoming (filtered) matches of the day by sending GET requests at **139.59.1.172:8000/upcoming**, or to get team squads by sending POST requests at **139.59.1.172:8000/squads?match_id=?**. As this server is currently small-scale and can't handle DOS attacks, we would appreciate if the API is used sparingly.

5 Project Strengths and Achievements

- The Project uses more than 5000 lines of code to systematically do its job. The logic of ball parsing, to get players involved in each ball, capture all the events of the ball, handle subsequent updates to already parsed balls and thorough logging for potential future fixes/updates is fairly complex. In the initial days of development, the code would break frequently. Since then, it has been thoroughly tested for edge cases which still appear sometimes but the current version is fairly robust and has been running for a few days without any issues.
- ***CricScorer*** is built with scale in mind and is fairly modular. Even if the website structure changes, parts of the code can be easily updated while keeping the same functionality. The currently prototypical file system handling can also be easily replaced with a robust database.
- The fantasy application is a fun game to enjoy with friends, which can be later converted to a business.

6 Limitations

- As with all complex systems, the development is an evolving process. There are already some known issues like continuous extra balls, late updates and potential innings switch bugs which need to be fixed. In given time constraints, we developed the best possile version that we could.
- The project currently only works for (or has been tested on) IPL T20 matches. In the future, we aim to broaden this horizon for One-day Internationals and Test matches (which typically have four innings and go on for several days).
- With slightly more effort, an arbitrary **stats generator** can be added to the project which can generate stats based on all possible filters given the information available in the ball-by-ball dictionaries.

7 Evaluation

The reason ***CricScorer*** was built is to remove human effort (which can give a 100% accuracy) in cricket scoring. Therefore, evaluating the correctness of ***CricScorer*** requires human effort. We already evaluated the final produced outputs against real data and found that the system only produced a couple of inaccuracies around 500 balls recorded, which have already been logged as known issues soon to be fixed. This puts the accuracy of our system easily at more than 99% (given it doesn't break). To further evaulate the correctness, one can look at the final output produced by ***CricScorer*** and compare it with the real data available on ESPNCricinfo.

8 Disclaimer about project contribution

Everything provided in the zipped code files (which is essentially ***CricScorer*** in its entirety) has been written solely by myself. A friend of mine (not at Hopkins) helped create the Android Application for *Academy 11*. In the app development, my contribution is limited to the design and interface ideas.