

INDUSTRIAL TRAINING

*Submitted in partial fulfillment of the
Requirements for the award of the degree*

of

Bachelor of Technology

in

Information Technology

By:

Gursimranjit Singh Tara (69/IT2/2019)



**Department of Information Technology
Guru Tegh Bahadur Institute of Technology**

**Guru Gobind Singh Indraprastha University
Dwarka, New Delhi
Year 2019-2023**

Web Scraper Project (IPL Data Extraction from website EspnCricInfo.com using JavaScript and Node)

Duration

24th March, 2022 – 24th August, 2022



At

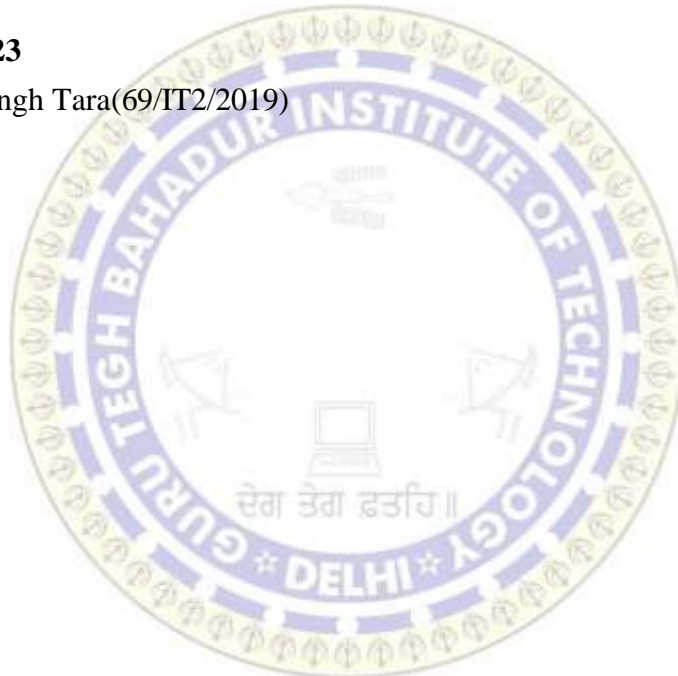
**Pepcoding Education (OPC)
Private LTD.**

DECLARATION

I hereby declare that all the work presented in this Industrial Training Report for the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in **Information Technology**, Guru Tegh Bahadur Institute of Technology, affiliated to Guru Gobind Singh Indraprastha University Delhi is an authentic record of my own work carried out at New Delhi, India from **24th March, 2022 to 24th August, 2022.**

Date: **11-01-2023**

Gursimranjit Singh Tara(69/IT2/2019)



CERTIFICATE

PEPCODING EDUCATION (OPC) PRIVATE LTD.

1st Floor, B-4, Sec-63, Noida, Uttar Pradesh-201301

Website: www.pepcoding.com

Phone: +917048971481



DATE: 25th August 2022

TO WHOM IT MAY CONCERN

This is to certify that Mr. Gursimranjit Singh, a B.Tech (IT) student at Guru Tegh Bahadur Institute Of Technology, has successfully completed his 5 Months of Internship in Data Structures & Algorithms & Web Development from 24th March 2022 to 24th August 2022. We found him sincere in his work and well-coordinated with his colleagues.

We wish him the best of luck for his bright future.

A handwritten signature in black ink that reads 'Sumeet'.

Sumeet Malik

Director

NOTE: The declaration made in the letterhead is valid only after it has been signed by the director.

ACKNOWLEDGEMENT

I would like to express my great gratitude towards **Pepcoding Education (OPC) Pvt. Ltd.** who provided me the help and resources to participate and create this Project. Further I would like to extend my great gratitude towards **Ms. Shipra Raheja** who has given us support and suggestions. Without their help I could not have presented this work up to the present standard. I also take this opportunity to give thanks to all others who gave us support for the project or in other aspects of our study at Guru Tegh Bahadur Institute of Technology.

Gursimranjit Singh Tara(69/IT2/2019)
gursimranjit.2001@gmail.com

Date: **11-01-2023**

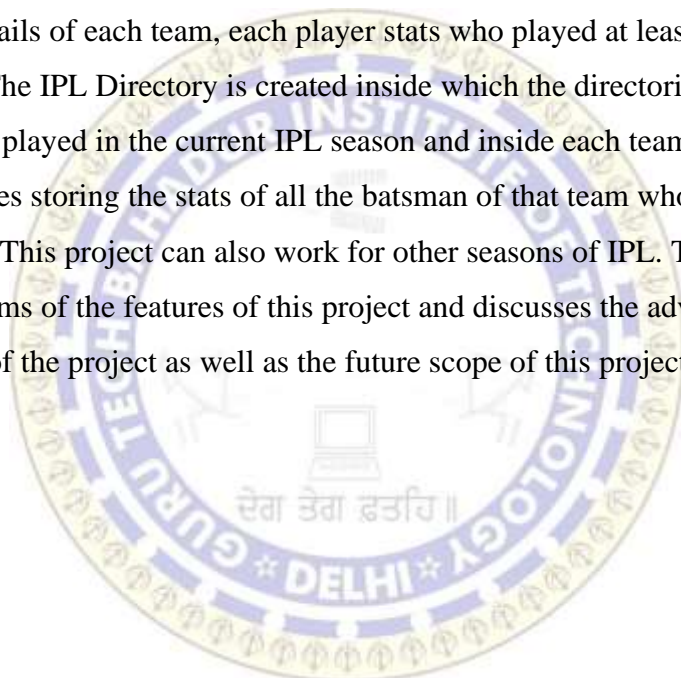


ABSTRACT

Web Scraping is a set of methods, which allows a user to collect information presented on the World Wide Web. In this industrial training project, a brief overview of web scraping or the process of extracting unstructured data from a website and converting it into a structured format is provided. In this project, I have web scrapped ESPNCricInfo.com IPL 2021 page.

(URL: <https://www.espncriinfo.com/series/ipl-2021-1249214>).

The scraped data is converted into structured format. Here, I have used Node's modules to automatically insert the extracted data into the excel spreadsheets which gives us the details of each team, each player stats who played at least one match in the IPL 2021. The IPL Directory is created inside which the directories with names of teams which played in the current IPL season and inside each team directory are excel format files storing the stats of all the batsman of that team who played atleast a single match. This project can also work for other seasons of IPL. This report is presented in terms of the features of this project and discusses the advantages and disadvantages of the project as well as the future scope of this project.



CONTENTS

Chapter	Page No.
Title Page	I
Declaration	i
Certificate	ii
Acknowledgement	iii
Abstract	iv
Tables and figures	v
1. Introduction	1
1.1 Internship Details	1
1.2 Preliminaries	2
1.3 Technicalities Used	3
2. Requirement Analysis with SRS	5
2.1 Purpose of the project	5
2.2 Languages and Resources used	5
2.3 User Personas	5
2.4 Project Structure	6
2.5 Running the Script	6
2.6 Hosting on GitHub	6
3. Summary and Conclusion	8
4. Advantages	9
5. Limitations	9
6. Future scope	10
References	11
Appendix	12

INTRODUCTION

This industrial project was made while doing a five-month internship at “Pepcoding Education (OPC) Private Ltd.” from March 2022 to August 2022 on the subject of “Web Scraper Project (IPL Data Extraction from website EspnCricInfo.com using JavaScript and Node)”. This industrial project was particularly based on scraping data from the website EspnCricInfo.com IPL 2021 pages using JavaScript and node modules. The team names are stored as directories and in each team directory their players information is stored in spreadsheet format.

1.1 INTERNSHIP DETAILS:

- Internship duration: 5 Months (24th March 2022 – 24th August 2022) (Remote) (Company Name: Pepcoding Education (OPC) Private Ltd.)
- In this industrial training project,
 - Web scraped data from ESPNCricInfo.com IPL 2021 Pages. (URL: <https://www.espncriinfo.com/series/ipl-2021-1249214>)
 - The scraped data was in unstructured format, hence after extracting inserted the data into structured format and stored this information inside the excel spreadsheet files.
 - The team names are stored as directories and in each team directory their players information is stored in spreadsheet format.
 - Used JavaScript and Node modules to accomplish this project.

1.2 PRELIMNARIES: -

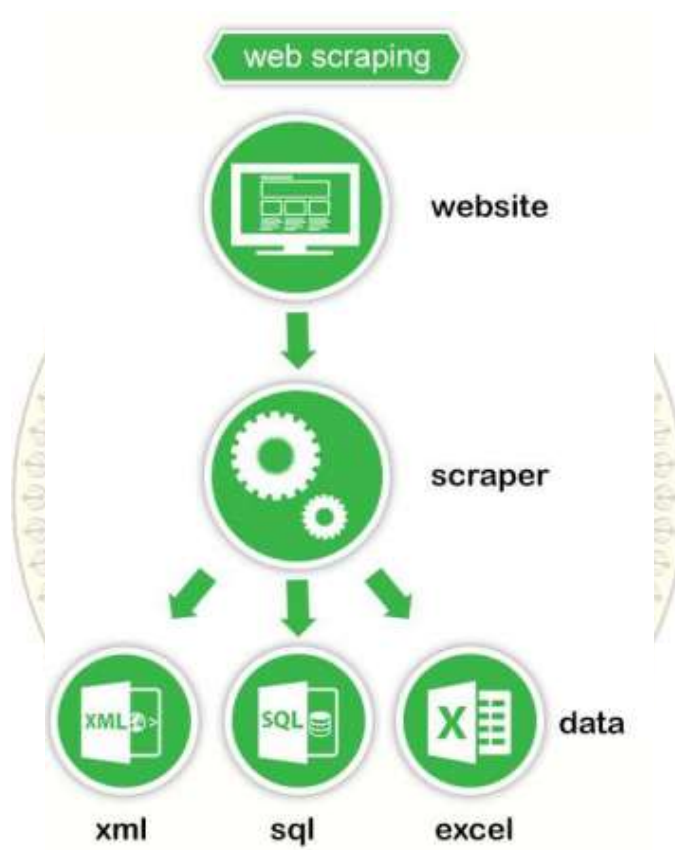
• Directory & Files: -

A directory is a location for storing files on your computer. It is a folder basically which stores different types of files format.

A file a container in a computer system that stores data, information, settings, or commands, which are used with a computer program. In graphical user interface (GUI), such as Microsoft operating systems, represent the files as icons, which associate to the program that opens the file. They can be any format.

- **Web Scraping:**

Web scraping is an automatic method to obtain large amounts of data from websites. Most of this data is unstructured data in an HTML format which is then converted into structured data in a spreadsheet or a database so that it can be used in various applications. There are many different ways to perform web scraping to obtain data from websites. These include using online services, particular API's or even creating your code for web scraping from scratch.



- **Excel Spreadsheets:**

A spreadsheet is a computer application that is designed to add, display, analyze, organize, and manipulate data arranged in rows and columns. It is the most popular application for accounting, analytics, data presentation, etc. Or in other words, spreadsheets are scalable grid-based files that are used to organize data and perform calculations.

1.3 TECHNICALITIES USED: -

JavaScript (JS): -

JavaScript (JS) is the most popular lightweight, interpreted compiled programming language. It can be used for both Client-side as well as Server-side developments. JavaScript also known as a scripting language for web pages. The web scraper is developed using JavaScript in this project.

Node Modules: -

In Node.js, Modules are the blocks of encapsulated code that communicates with an external application on the basis of their related functionality. Modules can be a single file or a collection of multiples files/folders. The reason programmers are heavily reliant on modules is because of their re-usability as well as the ability to break down a complex piece of code into manageable chunks.

In this project we have used three modules mainly: -

1. request module (To make request to server)
2. JSDOM module (To extract data from the website)
3. xlsx module (To store the scraped data as structured format in excel files)

CSS Selectors: -

CSS Stands for "Cascading Style Sheet." Cascading style sheets are used to format the layout of Web pages.

CSS selectors are used to "find" (or select) the HTML elements you want to style or extract. We can divide CSS selectors into five categories:

Simple selectors (select elements based on name, id, class)

Combinator selectors (select elements based on a specific relationship between them)

Pseudo-class selectors (select elements based on a certain state)

Pseudo-elements selectors (select and style a part of an element)

Attribute selectors (select elements based on an attribute or attribute value)

CSS Selectors are used to send request on the element on the webpage by its corresponding selector and then request is made using request module of Node.

GitHub: -

✓ **GitHub Mission & Values:**

GitHub is how people build software.

✓ **GitHub Vision Statement:**

We believe GitHub needs to remain an open platform for all developers. No matter your language, stack, platform, cloud, or license, GitHub will continue to be your home the best place for software creation, collaboration, and discovery.

✓ **About GitHub:**

GitHub is the developer company. We make it easier for developers to be developers: to work together, to solve challenging problems, and to create the world's most important technologies. We foster a collaborative community that can come together—as individuals and in teams—to create the future of software and make a difference in the world.

✓ **Use of GitHub in this project:**

GitHub is used to host the website in the public repository.

“CODERXGursimran/Payment-Gateway-Integration” [Public and Active]

Further, conversion of images has been done on GitHub which are used in the website.

2. REQUIREMENT ANALAYSIS WITH SRS (SOFTWARE REQUIREMENT SPECIFICATION)

2.1 Purpose of the project: -

The purpose of the project is to scrap or extract data from EspnCricInfo.com dedicated IPL 2021 page (URL: <https://www.espncriinfo.com/series/ipl-2021-1249214>) and then convert this extracted data into structured format. The team names are stored as directories and in each team directory their players information is stored in spreadsheet format which can be later used for multiple purposes like data analytics.

2.2 Languages and Resources used: -

The data scraped is done using JavaScript and with help of node modules which have to be installed explicitly. The request module is used to make request to the server, the JSDOM module is used to extract the data using CSS selectors, and lastly thexlsx module is used to store the information into the excel spreadsheet format files.

Project Specifications: -

- Languages used: JavaScript
- JS Environment: Node
- Node Modules used: request, JSDOM, xlsx
- IDE used: Visual Studio Code
- Browser Used: Google Chrome/Microsoft Edge
- Hosted on: GitHub repository
 - Link: “<https://github.com/CODERXGursimran/ESPN-CricInfo-WebScrapper>”
 - Status: Active and Public

2.3 User Personas: -

The unstructured data extracted is modified into structured format by storing the information into the excel spreadsheets using xlsx module of node. This extracted data can be used for multiple purposes like data analysis, data manipulation, data enrichment, simulation and optimization. The project created can also be used to extract data from other EspnCricInfo.com IPL pages like different seasons of IPL played. Only the teams and players who played at least one match during the whole season directories and files are made.

2.4 Project Structure: -

There are four JavaScript files running the whole project: -

1. home.js:

The request module first sends the request to the ESPNcricInfo.com IPL 2021 (URL: <https://www.espncriinfo.com/series/ipl-2021-1249214>) and as a response we get the IPL 2021 page. Then the second request is made to the css selector specified meaning button “view all results”. The page which is loaded acts as the input for the next script file “allMatchPage.js”.

2. allMatchPage.js:

This script accepts the output generated from “home.js” as an input. Then it sends a request to server requesting the scorecard of all the matches ever played during the whole season of IPL. Finally, it produces the output as all the scorecards of all the matches which acts as the input for the next script file “scorecard.js”.

3. scorecard.js:

This script file accepts the output generated from allMatchPage.js i.e., the scorecard, it scraps the data of all the batsman in both the tables and the helper.js file helps in structuring this unstructured data and lastly directories are created using dataOrganizer function. The dataOrgainzer function makes a directory by the name of team name and in each directory stores a file with the name of batsman who played at least one match during the whole season.

4. helper.js:

This file is helping to organize data into the excel spreadsheets by storing batsman name as the name of the excel spreadsheet which contains all the records of the batsman.

2.5 Running the Script: -

1. Install VS Code IDE.

2. Install NodeJS from any browser.

3. Open an integrated Terminal and install the below dependencies.

4. Install all the NodeJs dependences using commands: -

a. `npm install request`

- b. npm install JSOM
 - c. npm install xlsx
4. After installing, type: “node ./home.js” and the script will run.

2.6 Hosting on GitHub: -

1. Create a new Repository “https://github.com/CODERXGursimran/ESPN-CricInfo-WebScraper”
2. Upload all the files of the project that are part of the project through button “Add file” in the repository and commit the changes for every file uploaded.
3. To install the project on local machine click on vertical buttons and download ZIP file. Later extract it and run the script as stated in the “running the script section”.



SUMMARY AND CONCLUSION

This project summarizes everything about web scraping and how we scraped data from the IPL 2021 season of every batsman who played at least one match in the season. This project can also be used to scrap data of other IPL seasons like 2020, 2019, etc. depending on need of the user just we need to change the home page link of the home.js file. This structuring of data into excel files can further be used depending on the tasks like data science, data analysis, training a dataset, developing machine learning algorithms and many more. In data science, to do anything, you need to have data at hand. To get that data, you'll need to research the required sources, and web scraping helps you. Web scraping collects and categorizes all the required data in one accessible location. Researching with a single, convenient location is much more feasible and more comfortable than searching for everything one-by-one.



ADVANTAGES

The following points are the advantages of making the project and writing this report: -

- ✓ Time Efficient
 - The advantage of web Scraping is its time-efficient and low maintenance. For example, downloading big data may take hours, and then analyzing every single row manually at a time is worth spending your entire month.
 - But with web scraping, you can have your computer do all those manual tasks for you in just a few seconds so you have more time to do what you want to do.
- ✓ Complete automation on running the script we get access to humongous amount of information which is presented in a structured manner.
- ✓ Since no human intervention is there hence data redundancy or inaccuracy of data is out of question.
- ✓ Faster, reliable and performance robustness are more advantages.

LIMITATIONS

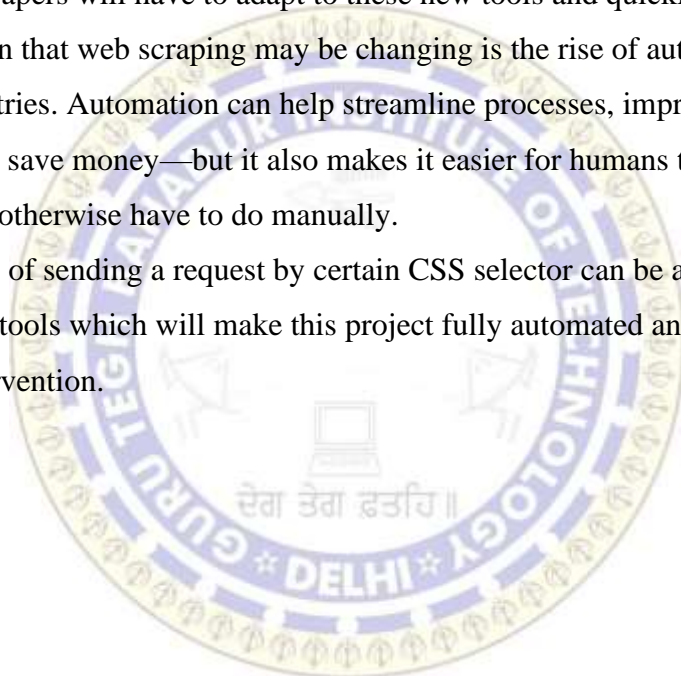
Like any other report the limitations of the study is not out of questions. But the following factors seem to me the main limitations of this study: -

- ✓ Needs perpetual maintenance that is the css selectors used have to be manually updated after a few weeks depending on the dynamic nature of the web page being scrapped.
- ✓ Can get blocked when the scraper accidentally scraps private or unauthorized data from the world wide web.
- ✓ It has a learning curve that is unlike machine learning algorithms it doesn't learn, it has to be taught manually.
- ✓ The web scraper can scrap data from unauthorized web pages or can scrap copyrighted material or content which can cause trouble later.

FUTURE SCOPE

The future scope of the project is discussed as follows: -

- ✓ This project can also be made using JavaScript automation tools like puppeteer which can automate the task of updating the CSS selector at certain intervals by itself and hence this project can become fully automated.
- ✓ The first sign that web scraping may be changing is the growth of artificial intelligence (AI) and machine learning (ML). These technologies allow for more accurate and real-time processing of information than ever before. This means that web scrapers will have to adapt to these new tools and quickly.
- ✓ Another sign that web scraping may be changing is the rise of automation within many industries. Automation can help streamline processes, improve quality control, and save money—but it also makes it easier for humans to perform tasks they would otherwise have to do manually.
- ✓ The process of sending a request by certain CSS selector can be automated using automation tools which will make this project fully automated and without any human intervention.

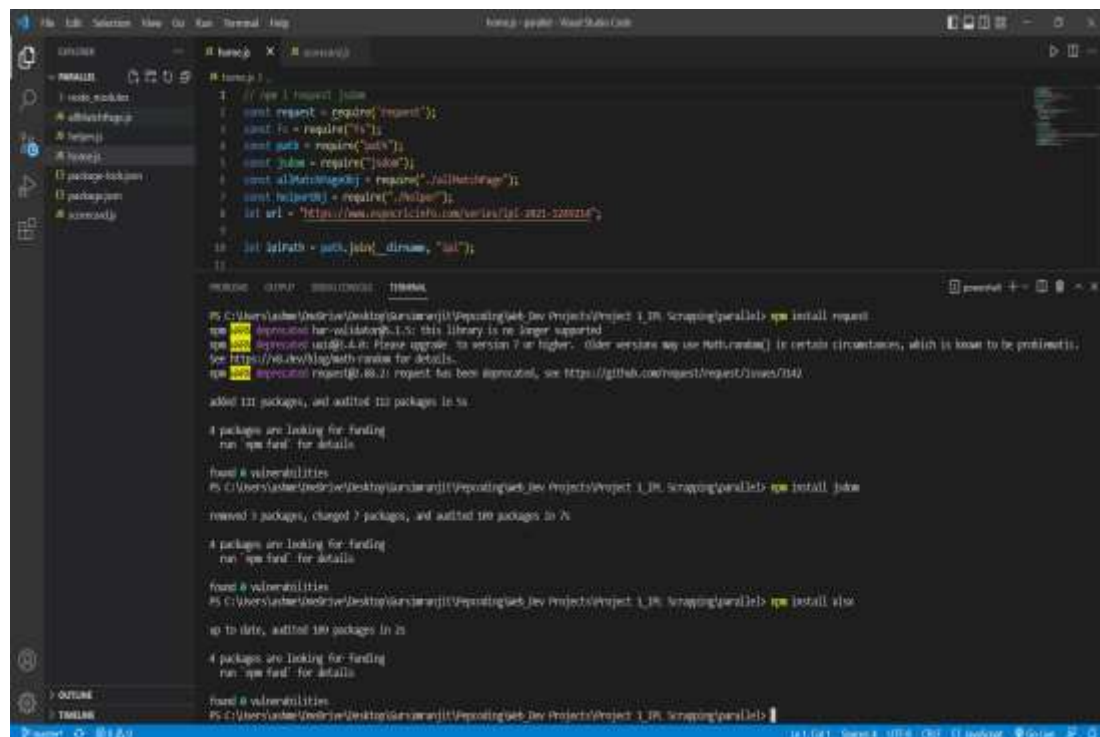


REFERENCES

- [1] <https://www.espncriinfo.com/series/ipl-2021-1249214> page on which the first request is made by request node package.
- [2] <https://www.espncriinfo.com/series/ipl-2021-1249214/match-schedule-fixtures-and-results> to make request for scorecards of all teams.
- [3] <https://www.espncriinfo.com/series/ipl-2021-1249214/mumbai-indians-vs-royal-challengers-bangalore-1st-match-1254058/full-scorecard> sample to access scorecard of all teams like this one.
- [4] <https://www.espncriinfo.com/player/rohit-sharma-34102> sample to access all information of player.
- [5] <https://nodejs.org/en/> to read and download node.
- [6] <https://www.npmjs.com/package/request> to read about request package which makes request to server to initiate scraping process.
- [7] <https://www.npmjs.com/package/jsdom> to read about JSDOM package which extracts data from the web page.
- [8] <https://www.npmjs.com/package/xlsx> to read about xlsx package which stores all data into excel spreadsheets in structured format.
- [9] <https://github.com/> Accessed during the hosting phase and downloading process.
- [10] <https://code.visualstudio.com/docs> for getting insights into some practices of visual studio code, the IDE used to create this project.
- [11] <https://www.comparably.com/companies/github> to get mission, vision and about us statements of the company GitHub.
- [12] <https://www.geeksforgeeks.org/web-development/> accessed throughout the project to get help on various syntax.



1. Installing Dependencies (Node Packages)



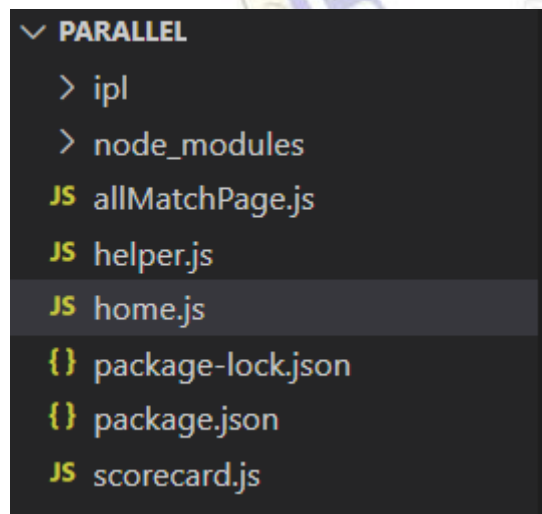
```
1 // Step 1: request module
2 const request = require('request');
3 const fs = require('fs');
4 const path = require('path');
5 const fsPromises = require('fs/promises');
6 const allMatchPageObj = require('./allMatchPage');
7 const helperObj = require('./helper');
8 let url = 'https://www.espncricinfo.com/series/ipl-2021-220216';
9
10 let filepath = path.join(__dirname, 'url');
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
PS C:\Users\ashme\OneDrive\Desktop\Gursimranjit\Peppcoding\Web_Dev Projects\Project 1 IPL Scrapping\parallel> npm install request
npm deprecated har-validator@4.2.1: this library is no longer supported
npm deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances, which is known to be problematic.
See https://v8.dev/blog/math-random for details.
npm deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
added 131 packages, and audited 133 packages in 5s
4 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
PS C:\Users\ashme\OneDrive\Desktop\Gursimranjit\Peppcoding\Web_Dev Projects\Project 1 IPL Scrapping\parallel> npm install jquery
removed 3 packages, changed 2 packages, and audited 140 packages in 7s
4 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
PS C:\Users\ashme\OneDrive\Desktop\Gursimranjit\Peppcoding\Web_Dev Projects\Project 1 IPL Scrapping\parallel> npm install axios
up to date, audited 140 packages in 2s
4 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
PS C:\Users\ashme\OneDrive\Desktop\Gursimranjit\Peppcoding\Web_Dev Projects\Project 1 IPL Scrapping\parallel>
```

2. Running script using command “node ./home.js”

```
PS C:\Users\ashme\OneDrive\Desktop\Gursimranjit\Peppcoding\Web_Dev Projects\Project 1 IPL Scrapping\parallel> node ./home.js
content recieved
```

(IPL Directory gets created)

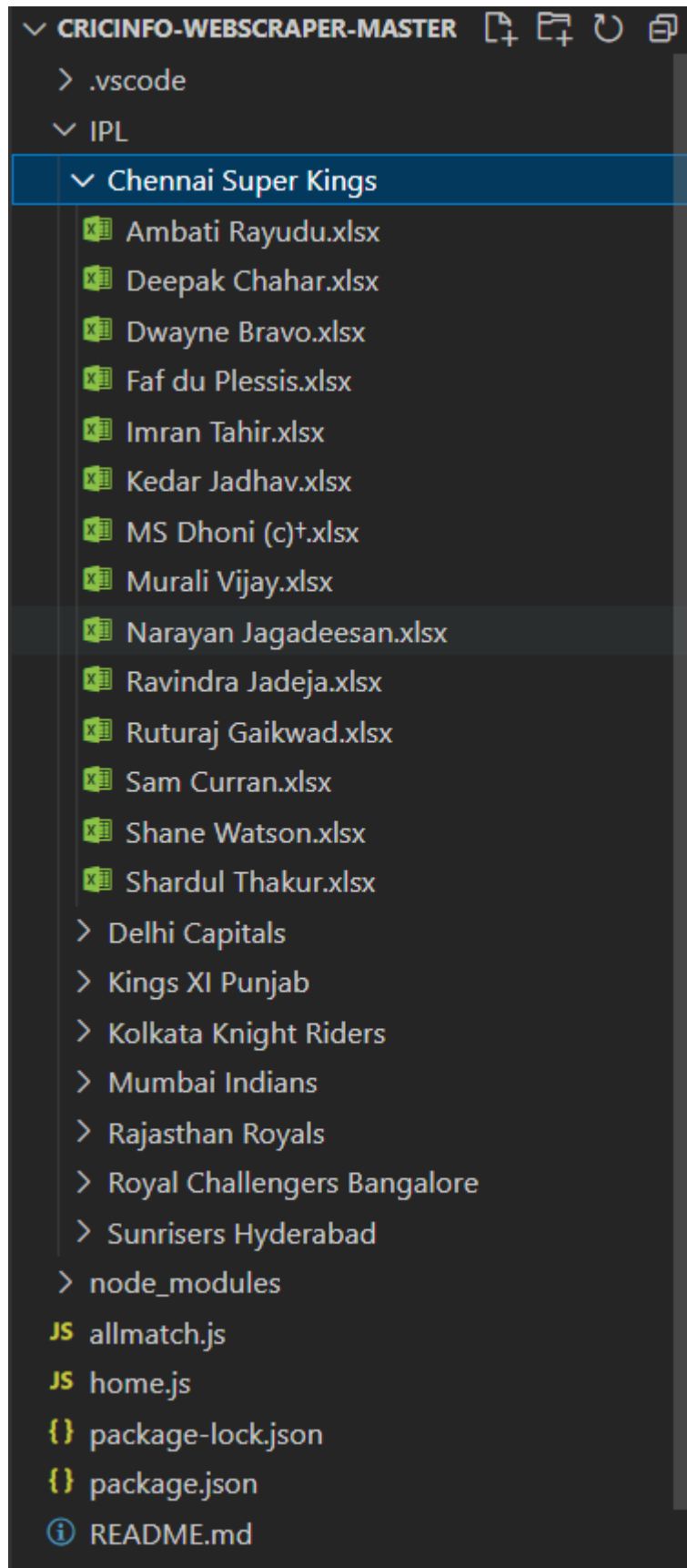


3. Script running terminal status

```
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kolkata-knight-riders-vs-delhi-capitals-42nd-match-1216497/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/chennai-super-kings-vs-mumbai-indians-41st-match-1216521/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/rajasthan-royals-vs-sunrisers-hyderabad-40th-match-1216518/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kolkata-knight-riders-vs-royal-challengers-bangalore-39th-match-1216494/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kings-xi-punjab-vs-delhi-capitals-38th-match-1216546/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/chennai-super-kings-vs-rajasthan-royals-37th-match-1216533/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/mumbai-indians-vs-kings-xi-punjab-36th-match-1216517/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/sunrisers-hyderabad-vs-kolkata-knight-riders-35th-match-1216512/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/rajasthan-royals-vs-royal-challengers-bangalore-33rd-match-1216522/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/mumbai-indians-vs-kolkata-knight-riders-32nd-match-1216526/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/royal-challengers-bangalore-vs-kings-xi-punjab-31st-match-1216531/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/delhi-capitals-vs-rajasthan-royals-30th-match-1216543/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/sunrisers-hyderabad-vs-chennai-super-kings-29th-match-1216528/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/royal-challengers-bangalore-vs-kolkata-knight-riders-28th-match-1216540/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/mumbai-indians-vs-delhi-capitals-27th-match-1216529/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/sunrisers-hyderabad-vs-rajasthan-royals-26th-match-1216507/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/chennai-super-kings-vs-royal-challengers-bangalore-25th-match-1216525/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kings-xi-punjab-vs-kolkata-knight-riders-24th-match-1216523/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/rajasthan-royals-vs-delhi-capitals-23rd-match-1216500/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/sunrisers-hyderabad-vs-kings-xi-punjab-22nd-match-1216542/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kolkata-knight-riders-vs-chennai-super-kings-21st-match-1216501/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/mumbai-indians-vs-rajasthan-royals-20th-match-1216511/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/royal-challengers-bangalore-vs-delhi-capitals-19th-match-1216519/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kings-xi-punjab-vs-chennai-super-kings-18th-match-1216513/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/mumbai-indians-vs-sunrisers-hyderabad-17th-match-1216538/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/delhi-capitals-vs-kolkata-knight-riders-16th-match-1216515/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/royal-challengers-bangalore-vs-rajasthan-royals-15th-match-1216514/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/chennai-super-kings-vs-sunrisers-hyderabad-14th-match-1216516/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kings-xi-punjab-vs-mumbai-indians-13th-match-1216503/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/rajasthan-royals-vs-kolkata-knight-riders-12th-match-1216504/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/delhi-capitals-vs-sunrisers-hyderabad-11th-match-1216532/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/royal-challengers-bangalore-vs-mumbai-indians-10th-match-1216547/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/rajasthan-royals-vs-kings-xi-punjab-9th-match-1216527/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kolkata-knight-riders-vs-sunrisers-hyderabad-8th-match-1216545/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/chennai-super-kings-vs-delhi-capitals-7th-match-1216539/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kings-xi-punjab-vs-royal-challengers-bangalore-6th-match-1216510/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/kolkata-knight-riders-vs-mumbai-indians-5th-match-1216508/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/rajasthan-royals-vs-chennai-super-kings-4th-match-1216496/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/sunrisers-hyderabad-vs-royal-challengers-bangalore-3rd-match-1216534/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/delhi-capitals-vs-kings-xi-punjab-2nd-match-1216493/full-scorecard
https://www.espnricinfo.com/series/ipl-2020-21-1210595/mumbai-indians-vs-chennai-super-kings-1st-match-1216492/full-scorecard

Chris Morris 2 5 0 0 40.00
Gurkeerat Singh Mann 2 2 0 0 100.00
Washington Sundar 5 2 1 0 250.00
Dubai (DSC) | Oct 25 2020 | Chennai Super Kings | Royal Challengers Bangalore | Super Kings won by 8 wickets (with 8 balls remaining)
Ruturaj Gaikwad 65 51 4 3 127.45
Faf du Plessis 25 13 2 2 192.30
Ambati Rayudu 39 27 3 2 144.44
MS Dhoni (c) † 19 21 3 0 90.47
Abu Dhabi | Oct 25 2020 | Mumbai Indians | Rajasthan Royals | Royals won by 8 wickets (with 10 balls remaining)
Ishan Kishan 37 36 4 1 102.77
Quinton de Kock † 6 4 0 1 150.00
Suryakumar Yadav 40 26 4 1 153.84
Saurabh Tiwary 34 25 4 1 136.00
Kieron Pollard (c) 6 4 0 1 150.00
Hardik Pandya 60 21 2 7 285.71
Krunal Pandya 3 4 0 0 75.00
Abu Dhabi | Oct 25 2020 | Rajasthan Royals | Mumbai Indians | Royals won by 8 wickets (with 10 balls remaining)
Robin Uthappa 13 11 2 0 118.18
Ben Stokes 107 60 14 3 178.33
Steven Smith (c) 11 8 1 1 137.50
Sanju Samson † 54 31 4 3 174.19
Dubai (DSC) | Oct 13 2020 | Chennai Super Kings | Sunrisers Hyderabad | Super Kings won by 20 runs
Sam Curran 31 21 3 2 147.61
Faf du Plessis 0 1 0 0 0.00
Shane Watson 42 38 1 3 110.52
Ambati Rayudu 41 34 3 2 120.58
MS Dhoni (c) † 21 13 2 1 161.53
Ravindra Jadeja 25 10 3 1 250.00
Dwayne Bravo 0 1 0 0 0.00
Deepak Chahar 2 2 0 0 100.00
Dubai (DSC) | Oct 13 2020 | Sunrisers Hyderabad | Chennai Super Kings | Super Kings won by 20 runs
David Warner (c) 9 13 0 0 69.23
Jonny Bairstow † 23 24 2 0 95.83
Manish Pandey 4 3 1 0 133.33
Kane Williamson 57 39 7 0 146.15
Priyam Garg 16 18 1 0 88.88
Vijay Shankar 12 7 0 1 171.42
Rashid Khan 14 8 1 1 175.00
Shahbaz Nadeem 5 5 1 0 100.00
Sandeep Sharma 1 2 0 0 50.00
T Natarajan 0 1 0 0 0.00
```

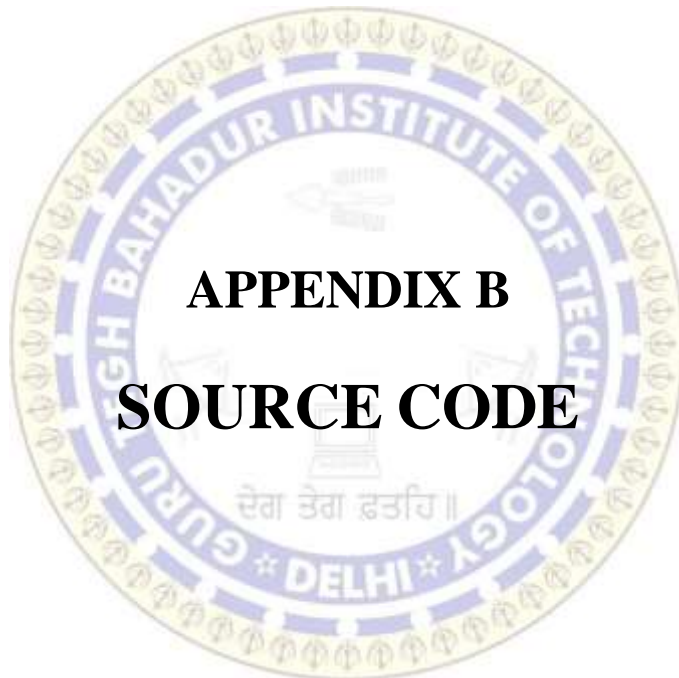
4. After all data is received IPL Directory status



5. After opening any excel file (eg: M.S. Dhoni)

MS Dhoni (Protected View) - Excel									
File Home Insert Page Layout Formulas Data Review View Developer Help Tell me what you want to do									
	A	B	C	D	E	F	G	H	I
1	teamname	playername	run	balls	four	six	sr	opponent	Venue
2	Chennai Super Kings	MS Dhoni (c)	11	12	1	0	91.66	Kolkata Knight Riders	Oct 7 2020 Abu Dhabi
3	Chennai Super Kings	MS Dhoni (c)	1	4	0	0	25.00	Kolkata Knight Riders	Oct 29 2020 Dubai (DSC)
4	Chennai Super Kings	MS Dhoni (c)	28	28	2	0	100.00	Rajasthan Royals	Oct 19 2020 Abu Dhabi
5	Chennai Super Kings	MS Dhoni (c)	47	36	4	1	130.55	Sunrisers Hyderabad	Oct 2 2020 Dubai (DSC)
6	Chennai Super Kings	MS Dhoni (c)	19	21	3	0	90.47	Royal Challengers Bangalore	Oct 25 2020 Dubai (DSC)
7	Chennai Super Kings	MS Dhoni (c)	21	13	2	1	161.53	Sunrisers Hyderabad	Oct 13 2020 Dubai (DSC)
8	Chennai Super Kings	MS Dhoni (c)	16	18	2	1	100.00	Mumbai Indians	Oct 23 2020 Sharjah
9	Chennai Super Kings	MS Dhoni (c)	3	5	0	0	60.00	Delhi Capitals	Oct 17 2020 Sharjah
10	Chennai Super Kings	MS Dhoni (c)	0	2	0	0	0.00	Mumbai Indians	Sep 29 2020 Abu Dhabi
11	Chennai Super Kings	MS Dhoni (c)	10	6	0	1	166.66	Royal Challengers Bangalore	Oct 10 2020 Dubai (DSC)
12	Chennai Super Kings	MS Dhoni (c)	15	12	2	0	125.00	Delhi Capitals	Sep 25 2020 Dubai (DSC)
13	Chennai Super Kings	MS Dhoni (c)	19	17	0	3	170.58	Rajasthan Royals	Sep 22 2020 Sharjah
14									





APPENDIX B

SOURCE CODE

FILE 1 (File Name: "home.js")

```
// npm i request jsdom

const request = require('request');

const fs = require("fs");

const path = require("path");

const jsdom = require("jsdom");

const allMatchPageObj = require("./allMatchPage");

const helperObj = require("./helper");

let url = "https://www.espncriinfo.com/series/ipl-2021-1249214";

let iplPath = path.join(__dirname, "ipl");

helperObj.dirCreator(iplPath);

// first request

request(url, cb);

function cb(error, response, body) {

  if (error) {

    console.log('error:', error.message); // Print the error message

  } else if (response && response.statusCode == 404) {

    console.log("Page not found");

  } else {

    console.log("content recieved");

    // console.log(body);

    extractData(body);

  }

}

function extractData(body) {

  const JSDOM = jsdom.JSDOM;

  // pass to newJSDOM

  let dom = new JSDOM(body);

  // 2. // no meaning

  // document represent the whole html page

  let document = dom.window.document;

  let element = document.querySelector(".ds-block.ds-text-center.ds-

uppercase.ds-text-ui-typo-primary.ds-underline-offset-4")

  // let element = document.querySelector("span.ds-text-tight-m.ds-font-

bold.ds-text-ui-typo-primary.hover\:ds-underline.hover\:ds-decoration-ui-
```

```

stroke-primary.ds-block.ds-block.ds-text-center")

    let link = element.getAttribute("href");

    // console.log("link", link);

    let AllMatchPageKaLink = "https://www.espncriinfo.com" + link;

    console.log(AllMatchPageKaLink);

    // allmatch page

    allMatchPageObj.AllmatchFn(AllMatchPageKaLink)

}

```

FILE 2: (Name: “AllMatchPage.js”)

```

// npm i request jsdom
const request = require('request');
const fs = require("fs");
const jsdom = require("jsdom");
const scoreCardObj = require("./scorecard");
function AllMatchPageExecutor(url) {
    request(url, cb);
}
function cb(error, response, body) {
    if (error) {
        console.log('error:', error.message); // Print the error message
    } else if (response && response.statusCode == 404) {
        console.log("Page not found");
    } else {
        console.log("content recieved");
        // console.log(body);
        extractData(body);
    }
}
function extractData(body) {
    const JSDOM = jsdom.JSDOM;
    // pass to newJSDOM
    let dom = new JSDOM(body);
    // document represent the whole html page

```

```

let document = dom.window.document;

let matchBoxes = document.querySelectorAll(".ds-flex.ds-mx-4.ds-pt-2.ds-
pb-3.ds-space-x-4.ds-border-t.ds-border-line-default-translucent")

for (let i = 0; i < matchBoxes.length; i++) {

    let curMatch = matchBoxes[i];

    let allAnchors = curMatch.querySelectorAll("a");

    let scoreCardAnchor = allAnchors[2];

    let link = scoreCardAnchor.getAttribute("href");

    let scoreCardLink = "https://www.espn.com" + link;

    console.log(scoreCardLink);

    scoreCardObj.scoreCardFn(scoreCardLink);

}

console.log(".....");
}

module.exports = {
    AllmatchFn: AllMatchPageExecutor
}

```

FILE 3: (Name: “Scorecard.js”)

```

// Q1 print the result
// npm i request jsdom
const request = require('request');
const fs = require("fs");
const path = require("path");
const jsdom = require("jsdom");
const helperObj = require("./helper");

function scoreCardExecutor(url) {
    request(url, cb);
}

function cb(error, response, body) {
    if (error) {
        console.log('error:', error.message); // Print the error message
    } else if (response && response.statusCode == 404) {
        console.log("Page not found");
    } else {
        console.log("content recieved");
    }
}

```

```

    // console.log(body);
    extractData(body);
  }
}

function extractData(body) {
  const JSDOM = jsdom.JSDOM;

  // pass to newJSDOM
  let dom = new JSDOM(body);

  // 2. // no meaning
  // document represent the whole html page
  let document = dom.window.document;

  // result
  let output = document.querySelectorAll(".ds-text-compact-xxs.ds-p-2.ds-px-4 p>span");
  let resultElem = output[0];
  let res = resultElem.textContent;

  // console.log("result :", res);

  let otherContentElem = document.querySelector(".ds-text-tight-m.ds-font-regular.ds-text-ui-typo-mid");
  let otherContent = otherContentElem.textContent;

  // 0 -> team 1 name , 1-> team 2 name
  let teamNamesElement = document.querySelectorAll(".ds-flex.ds-items-center.ds-cursor-pointer.ds-px-4");
  let firstTeamName = teamNamesElement[0].textContent;

  // team INNINGS (20 overs)
  let firstTeamNameArr = firstTeamName.split("INNINGS");
  firstTeamName = firstTeamNameArr[0].trim();

  let secondTeamName = teamNamesElement[1].textContent;
  let secondTeamNameArr = secondTeamName.split("INNINGS");
  secondTeamName = secondTeamNameArr[0].trim();

  // tables -> from a match -> 4 tables -> 2 batting , 2 bowling
  // -> 0 -> team 1 batting , 1-> team 2 bowling,
  // 2 -> team 2 batting 3 -> team 1 bowling

  let teamStatsElements = document.querySelectorAll(".ReactCollapse--content table");

  // let htmlString = "<table>" + teamStatsElements[0].innerHTML +

```

```

"</table>";

// // console.log(htmlString);

// fs.writeFileSync("firstTeam.html", htmlString);

// console.log("File created");

let firstBattingTeam = teamStatsElements[0];

let secondBattingTeam = teamStatsElements[2];

processTeam(firstBattingTeam, firstTeamName, secondTeamName, res,
otherContent);

processTeam(secondBattingTeam, secondTeamName, firstTeamName, res,
otherContent);
console.log("~~~~~")
~~~~~")
}

function processTeam(teamElement, currTeam, opponentTeam, result,
otherDetails) {

// it will print all the stats of it's player

let allRowswithextras = teamElement.querySelectorAll("tbody tr.ds-border-
b.ds-border-line.ds-text-tight-s");

// console.log(allRowswithextras.length);

for (let i = 0; i < allRowswithextras.length; i++) {

// required rows -> remove extra rows

let cRow = allRowswithextras[i];

let cols = cRow.querySelectorAll("td");

// console.log(cols.length)

if (cols.length == 8) {

let name = cols[0].textContent.trim();

let runs = cols[2].textContent;

let balls = cols[3].textContent;

let fours = cols[5].textContent;

let sixes = cols[6].textContent;

let sr = cols[7].textContent;

console.log("Name " + name + " plays for " + currTeam + " against " +
opponentTeam + " Runs " + runs + " balls " + balls +

" fours " + fours + " sixes " + sixes + " sr " + sr +

" result " + result + " other details " + otherDetails

);

let dataObj = {

```



```

        name, runs, balls, fours, sixes, sr, opponentTeam, result, otherDetails
    }
    dataOrganizer(currTeam, name, dataObj);
}
}
console.log(".....");
}

function dataOrganizer(teamName, playerName, dataObj) {
    // folder will not be present
    // folder will be present
    const teamPath = path.join(__dirname, "ipl", teamName);
    helperObj.dirCreator(teamPath);
    // file will not be present
    const playerPath = path.join(teamPath, playerName + ".xlsx");

    helperObj.fileHandler(playerPath, dataObj);
}

module.exports = {
    scoreCardFn: scoreCardExecutor
}

```

FILE 4: (Name: “helper.js”)

```

const fs = require("fs");
const path = require("path");
// npm i xlsx
const xlsx = require("xlsx");
function dirCreator(inputPath) {
    let isPresent = fs.existsSync(inputPath);
    if (isPresent == false) {
        fs.mkdirSync(inputPath);
    } else {
        // console.log(inputPath, " already present ")
    }
}

function fileHandler(inputPath, dataObj) {

```

```

let isFilePresent = fs.existsSync(inputPath);

let arr = [];

if (isFilePresent == false) {
    // fileCreator(inputPath, dataObj)
    arr.push(dataObj);
    excelWriter(inputPath, arr);
} else {
    arr = excelReader(inputPath);
    arr.push(dataObj);
    excelWriter(inputPath, arr);
    // fileUpdater(inputPath, dataObj);
}
}

// JSON.stringify -> write
// JSON.parse -> reading
function fileCreator(playerPath, dataObj) {
    let arr = [dataObj];
    fs.writeFileSync(playerPath, JSON.stringify(arr));
}

function fileUpdater(playerPath, dataObj) {
    // file content read
    let dataBuffer = fs.readFileSync(playerPath);
    // buffer -> JSON
    let arr = JSON.parse(dataBuffer);
    // JSON entry add
    arr.push(dataObj);
    //
    fs.writeFileSync(playerPath, JSON.stringify(arr));
    // console.log("entry updated ", path.basename(playerPath));
}

function excelReader(filePath) {
    // file -> read -> workbook
    let wb = xlsx.readFile(filePath);
    // get a worksheet from a workbook

```

```

let excelData = wb.Sheets["sheet1"];

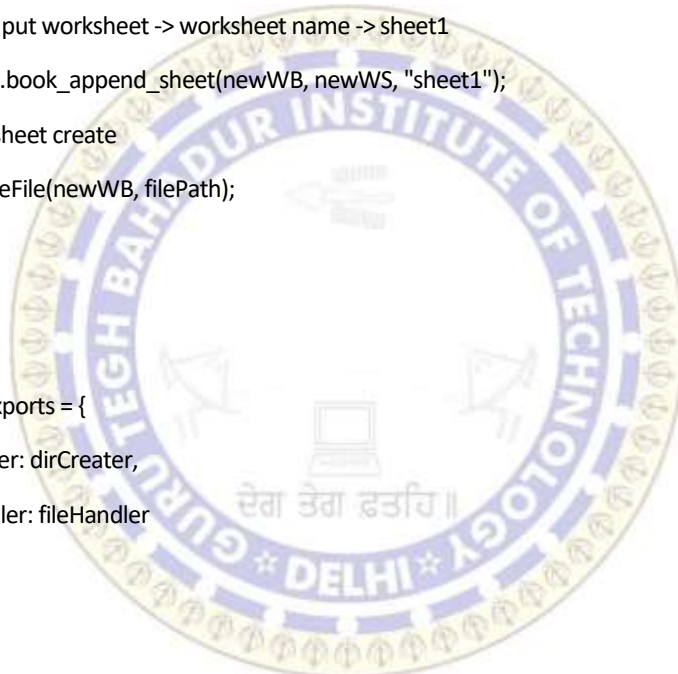
// sheet -> json
let ans = xls.utils.sheet_to_json(excelData);

return ans;
}

function excelWriter(filePath, json) {
  // console.log(xls.readFile(filePath));
  // empty workbook
  let newWB = xls.utils.book_new();
  // worksheet
  let newWS = xls.utils.json_to_sheet(json);
  // wb -> put worksheet -> worksheet name -> sheet1
  xls.utils.book_append_sheet(newWB, newWS, "sheet1");
  // worksheet create
  xls.writeFile(newWB, filePath);
}

module.exports = {
  dirCreator: dirCreator,
  fileHandler: fileHandler
}

```





x---END OF REPORT---x