

Lec 34

CRICINFO WEBSRAPPER

DATE

7 Oct, 2021, Thursday

Dev
13

Project-1, CRICINFO 2019 WORLDCUP DATA EXTRACTER

We need Courage

- To create, when we do not know
- Courage for the creativity to be done, outside my comfort zone

Activity

- 2.1 Read data from a source : Cricinfo WorldCup 2019 (array)
- 2.2 Process data : get all teams (jsdom) (jsdom)
- 2.3 Change the information extracted
(Array Manipulation)
- 2.4 Write unprocessed data in excel : Match result per team in their own sheet.
- 2.5 Create folders : one for each team.
- 2.6 Write files : Pdf files for scorecard of each match in relevant folder

Purpose of the project :-

- // the purpose of this project is to extract the information of WorldCup 2019 from Cricinfo & present that in the form of excel & pdf scorecards.
- // The real purpose is to learn how to extract the information and get experience with it.
- // A very good reason to ever make a project is to have fun while making it.

CLI

// `npm init -y`

ये नया folder हो जाएगा तो npm init -y करना होगा।

// `npm uninstall minimist`

// `minimist`

parse करके देंगा

yes

input array की

`process.argv` में array की
RE data store होता है।

Minimist help करता है उसे array से data को
extract करके objects करने में।

// `npm uninstall axios` // To download data from web
// `npm uninstall jsdom` // हमारे program के लिए
HTML को load करके DOM बनाएगा JS DOM.

{ DOM is an object tree which is created by
Browser for the programmer. Programmer
can interact with view of page with DOM.
लेकिन हमारे पास Browser ने होने वाले हमारे
लिए ये काम करेगा JS DOM.

→ DOM को देंगा Programmer.

Browser द्वारा 2 काम करके देता है:-

1) View दिया user को।

2) DOM बनाया programmer के लिए और Browser की दो

responsibility

हमारे लिए JS DOM

उसका काम /

Downloaded data की required

info लिए लिए में help करता है JS DOM.

// `ls -t` already installed library होती है node में।

// `npm install Excel4node`

excel workbook बनाया होता है।

→ इसमें एक workbook में काफी ताजी

excel worksheet होते संही हैं।

// `npm install pdf-lib` // pdf-lib pdf बनाता है help
करता है।

तो हमें हमारी सारी dependencies (library to be uninstalled) पड़ाए।

CLI { node SeicinfoExtractor.js --excel = Worldcup • CSV
--dataFolder = data --source = https://www.espncricinfo.com/series/icc-world-cup-2019-114415/matches
Data नाम की folder में तीन डिटेल्स PDF उपलब्ध हैं।
• जीवंत info PDF को Worldcup के द्वारा बनाया गया है।
Filename: SeicinfoExtractor.js

```
→ let minimist = require("minimist");  
→ let axios = require("axios");  
→ let jdom = require("jdom");  
→ let excel = require("excel4node");  
→ let fs = require("fs");  
→ let pdf = require("pdf-lib");  
  
→ let args = minimist(process.argv);  
→ console.log(args.excel); } // check  
→ console.log(args.dataFolder); } // arguments  
→ console.log(args.source); } // arguments  
  
// download using axios }  
// extract information }  
using jdom }  
// manipulate data using functions } // एक अचूक सब करना है।  
// Saving in excel using excel4node }  
// create folder & prepare pdf }
```

→ let response = axios.get('large source');
→ response.then(function(response))

→ we can search for axios in many other libraries to download data from web

→ Lists of libraries to make HTTP requests

Browser HTTP request

Client & Server

download तरीके में data HTTP प्रोटोकॉल

Browser और Server HTTP requests की form में ही होते हैं

→ Ctrl + F12 Search ट्रॉप्पे और Enter press करने पर यहाँ होते हैं
Inspect → Network → डॉक्टर पर हमें दिखेगा कि बहुत सारी

request आएंगे और Server के पास 200 OK

होते हैं जो request को,成功 (Success) वा request पूरी होती है। Status 400

यदि request कर रहा है Browser

Request करने के 2-3 नवीकरण होते हैं, तभी से यह get होता है।

यह अपर्याप्त चेज़में से एक file download होता है।

अब यदि तो Server को request करना चाहिए Browser

होता है, लेकिन हमारे Case में हमें एक request

करके Server से data download करना है without

using Browser, तो हम axios का use करेंगे।

axios एक library है (ऐसी बहुत सारी libraries होती हैं) जो एक https requests प्रोतोकॉल

होता है (Browser का काम है) एक

behalf पर axios का काम कर सकती है।

// let html = response.data; // अब यह

// console.log(html); source url का

data download होता

होता है इसी जैसा

form में है download,

जो इस लिए करने के check करके करता है

है।

→ let responsePromise = varis.get({args, source});
→ responsePromise.then(function(response))

5

let html = response.data;

↳ जो html form के data अंतर्वाले का HTML
नाम के variable में store करा दिया।

let dom = new jsdom.JSDOM(html);

↳ इससे ही dom मिलता है)
) जो capital letters
वाला JSDOM द्वारा है
jsdom द्वारा दिया गया है

JSDOM का एक नया object मिलता है
जिसमें html(data) पास के रूपमें

let document = dom.window.document;

↳ jsdom के अंदर JSDOM होता है;

उसके अंदर window यही screen वर्ता ही
represent करती ही dom में

document जैसे folder यह window में
जिससे ही webpage के अंदर के
data के changes का पात्र होता है।
जो ही document की चाहिए।

let matches = [];

↳ अब हम matches को array बनायें।

48 Match होते हैं तो 48 matches का एक

array होता है और हर Match में 5 information

होती है → Team1, Team2, Team1Score, Team2Score,
Result की हावी वे 5 पैरिये जो अलग - 2
datatype की हैं तो वे object में store करानी
पड़ेगी तो अब Match नाम का object बनोगा।

→ //let match = {
 \downarrow
 \downarrow

PAGE NO.	123
DATE	

→ तो हमें html data में सारे Matches की Info निकालनी पड़ेगी। अभी तक हमें हमारे पास उसे पूरे Web Page की data है, जो वो सब तो चाहिए नहीं हमें तो वह Match के relevant data चाहिए ले कि वो data जिस Match को निकालने के लिए हम querySelectorAll को user करते हैं।

→ WebPage पर जाओ और inspect करें।
अपने पहले इक हरे Match के हवाले पर Blue चाहिए (select)
कई lines पर नहीं, तो पहले
<div class="match-score-block">

पर hover करते ही ना तो ये स्ट्रक्चर पूरे Match की select हो रहा है, जोर हमें तो 48 Matches की info ही तो चाहिए तो हमें वो ही data निकालना चाहिए फिर div के निचली class Match Score Block है।

→ let matchDivs = document.querySelectorAll(
 \downarrow
 ("div.match-score-block"));

→ इस line का meaning है document द्वारा
 div selector करके को ऐसे class लिए हैं
 match score block नाम की।

→ // console.log(matchDivs.length);

→ यह बार matchDivs की length print करता है।
 देखते होंगे ये check करने के लिए कि साफ़ी 48 match का data आ रहा है जो matchDivs के अंदर।

→ 48 ✓

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

{

सभी Matches के लिए का Loop होता है।
यद्यपि 48 match हैं और matchDivs
की length भी 48 है।

let matchDiv = matchDivs[i];

इसके अंदर पढ़ता है।

1st match की data होती है।

अब 2nd till value of
i goes on.

इसके matchDiv होता है।

इसके matchDiv के corresponding
एक Match बनता है।

फिर (रखे) सभी Match object
के बारे में (loop की help से)

सभी objects के matches[] array
में push करते हुए बनता है।

let match = {

match
की टीम
25
information
प्रोटोकॉल
MatchDiv
में 27

team1name ← t1: " ",
team2name ← t2: " ",
team1score ← t1S: " ",
team2score ← t2S: " ",
result ← result: " "

27 5
सभी Match
object
की
properties
27

Let resultSpan = matchdiv.querySelector("div.v.status>span")

दो result वाले span
हों

all की परिस्थि नहीं
इनमें से कौन ही

match में result

प्राप्त हो

("div.v.status>span") ;

website पर match का result पर
जाओ और inspect करो

```
<div class="status-text">  
<span>Match tied </span>  
</div>
```

पहलव matchdiv में से वो
span किमालके वो जिसके parent

है यह dir - और इस dir पर कृषि
class नहीं है, तो class है
status

match.result = resultSpan.textContent ;

दो result दो आगामी match को, उस team1 और team2
के names गी बिल्कुल भी हैं।

Team 1 के नाम पर जाओ और inspect करो website पर तो

```
<div class="name-detail">
```

```

```

```
<p class="name">New Zealand </p>
```

```
</div>
```

Team1 New Zealand

Team2 England

Team 2 के नाम पर जाओ और inspect करो

```
<div class="name-detail">
```

```

```

```
<p class="name">England </p>
```

```
</div>
```

ये एक paragraph

है इसके parent

एक div है जो

इस div का class

नहीं है name-
detail

तो अब हम वो सारे paragraph निकालें हैं
जिनके parent div हैं; उन div के
अपर class लगी हैं name detail की।

PAGE NO.

DATE

let teamParas = matchdiv.querySelectorAll('div.name-detail');

वो सब Para
जिसमें Team के नाम हैं।

matchdiv
इसके
match की
details

इसका match
जिन 2 Team के
बीच में हुआ, उन
दोनों Team के नाम
Select करके इसमें
teamParas में

→ ("div.name-detail > p.name")

और जिनका parent
है matchdiv और उस div
के स्टक class लगी हैं।
जिसका नाम हैं
name-detail

को paragraph
वाले पाप
class name
लगी हैं।

तो अब हमारे पास दोनों Teams के नाम आएंगे हैं।
(teamParas में)

match.t1 = teamParas[0].textContent;
match.t2 = teamParas[1].textContent;

→ Team1 का नाम हमें मिल जाएगा 0th index पर
और Team2 का नाम 1st index पर
अब हम Match की Team1 के Score बतालेंगे और
Team2 के Score।

अब हमें Team 1 का score आविष्करण करें और
Team 2 का score चाहिए

PAGE NO.

DATE

But bro, there is a catch here.

इसे हमें match में 2 score, जिनमें से जो ज़्यादी तो नहीं है,
वहाँ परा 2nd match में दोनों Team रखली हैं या नहीं
जान, किसी 2nd Team के time पर लाइश आगे
तो को Team score तो होगा ही नहीं
तो होसकता है कि 2nd Match में 2 Team Score दिए

2T1 Team Score दिए

2T0 Team Score दिए।

Team 1 के score पर चाहो और inspect रखें

```
<div class = "score-detail">
```

```
    <span class = "score-info">(40/40 ov; target 302) from </span>
```

```
    <span class = "score">212/6 </span>
```

```
</div>
```

तो ये spans score के बिलकुल parent

→ वे div भी किसी class score-detail के

```
let scoreSpans = match.querySelectorAll ("
```

```
    div-score-detail > span-score" );
```

```
scoreIf(scoreSpans.length == 2)
```

```
    t1 = match.querySelector ("
```

```
    div-score-detail > span-score" );
```

```
    t2 = match.querySelector ("
```

```
elseIf(scoreSpans.length == 1)
```

```
    t1 = match.querySelector ("
```

```
    div-score-detail > span-score" );
```

```
else
```

```
    t1 = match.querySelector ("
```

```
    div-score-detail > span-score" );
```

```
y
```

```
matches.push (match);
```

```
let matchesJSON = JSON-
```

```
stringify (matches);
```

```
fs.writeFileSync
```

```
("matches.json",
```

```
matchesJSON,
```

```
"utf-8");
```

```
ISO to
```

```
ISO N.
```

```
इसकी
```

```
में डिटेल
```

PAGE NO. _____
DATE _____

अग्री अग्र इमें ~~for~~ console.log(matches) करेगों तो सब etc Match की data print हो जायेगा लेकिन ये data Match-wise print हो रहा है ऐसे match 1, for match 2 — match 98 में से इन Match की details milते रहे, और उन्हीं टीमों की team-wise data भी होती है।

अग्री
इमें
टेस्ट
data
में
होल
है
ये
इमें

t1 : "Australia",
t2 : "West Indies",
t1.s : "250/3",
t2.s : "241/8",
winner : "Australia Won by x"

इमें data इस form में नहीं दी गई है

let teams = [

इमें किसी भी Match
पर साथ desorganized { name : "India", } first team
नहीं दी गई। matches : [

इमें किसी Team की, }
पर उसके नाम के } Match 1

corresponding उसके }
पर match, the दूसरी } Match 2

team लेना है पर }
3rd के पर Match } ,

4th के पर Match } ,

5th के पर Match } ,

array है } ,

{ name : "England", }

matches : [

& so on
पर एक Match
India के लिए

Second

Team & so on

let teams = [];
for (let i = 0; i < matches.length; i++)

{ Put Team In Teams Array If Missing }

function 2

teams,
parameter matches[i]
return) ;

return
matches
in this loop

console.log(teams); → यहां पर

teams array के लिए

for (let i = 0; i < matches.length; i++) { print }
let teamsJSON = JSON.stringify(teams); ISO to JSON
fs.writeFileSync("teams.json", teamsJSON, "utf-8"); file में
console.log(JSON.stringify(teams)); print की करते देखते

function putTeamInTeamsArrayIfMissing(teams, match)

function {
let t1idx = -1; team1 index को -1 मानता है परन्तु
for (let i = 0; i < teams.length; i++) { for loop
if (teams[i].name == match.t1) {

if (t1idx == -1) { team1 index को -1 मानता है परन्तु
break; if name तो

t1idx = i; team1 index को i मानता है परन्तु

break;

team1 index
index QR
if name तो

teamarray match t1 के equal

में match.t1 दोगोले हो तो इसका

if (t1idx == -1) { अब शीर्षी t1 index
-1 होगा. team1 index
} else { team1 index
t1idx = i; team1 index को i मानता है परन्तु

teams.push({

name: match.t1, तो फिर teams

matches: []); array पर match.t1

}); push करता है

परन्तु matches पर push करता है

शाखा ही उसके matches

अर्थात् Blank space

object में जालके : object को push करता है

दोस्रे

let it2idx = -1; index को मानके चल -1
 for (let i=0; i<teams.length; i++) {

 if (teams[i].name == match.t2) {

 it2idx = i;
 break;
 }
 }

if (it2idx == -1) {
 teams.push({
 name: match.t2,
 matches: []
 });
 }

तो अब जब हमें Team Array print कराया तो

अगर जही भी तो वह Index हमा बन्दी हो

3TR of
 Team name
 it2idx
 # of team
 index
 store कराये

(1) सभी ही ER
 Match की

Team 2 की
 Team के नाम
 भी check किया जाए

Team array में ?

अगर नहीं है
 तो सबके object
 में (वो) name किसके

+ matches आम
 की रख empty
 array)

push करदे

Team array में

तो अब जब हमें Team Array print कराया तो Output ↴

```

[ {name: 'New Zealand', matches: []},
  {name: 'England', matches: []},
  {name: 'Australia', matches: []},
  {name: 'India', matches: []},
  {name: 'South Africa', matches: []},
  {name: 'Sri Lanka', matches: []},
  {name: 'Pakistan', matches: []},
  {name: 'Bangladesh', matches: []},
  {name: 'West Indies', matches: []},
  {name: 'Afghanistan', matches: []}
]
  
```

```

function putMatchInAppropriateTeam(teams, match) {
    let team1 = teams[t1idx];
    team1.matches.push(
        {
            vs: match.t2,
            selfscore: match.t1s,
            oppscore: match.t2s,
            result: match.result
        }
    );
    let t2idx = -1;
    for(let i = 0; i < teams.length; i++) {
        if(match.t2 == teams[i].t1) {
            t2idx = i;
            break;
        }
    }
}

```

let team2 = teams[t2idx];
team2.matches.push

```

    {
        vs: match.t1,
        selfscore: match.t1s,
        oppscore: match.t2s,
        result: match.result
    }
}

```

```

let t1idx = 0;
for(let i = 0; i < teams.length; i++) {
    if(match.t1 == teams[i].t1) {
        t1idx = i;
        break;
    }
}

```

```

if(match.t1 == teams[i].t1) {
    t1idx = i;
}

```

2nd Team

2nd Score

Match

Match

Final Team Match

Final

Opponent

Team Match

Matches.json

```

{
    "t1": "New Zealand",
    "t2": "England",
    "t1s": "241/8",
    "t2s": "241/9",
    "result": "Match tied"
}

```

```

{
    "t1": "Australia",
    "t2": "England",
    "t1s": "223",
    "t2s": "226/2",
    "result": "England won"
}

```

Teams.json

```

{
    "name": "New Zealand",
    "matches": [
        {
            "vs": "England",
            "selfscore": "241/8",
            "oppscore": "241",
            "result": "Match tied"
        },
        {
            "vs": "India",
            "selfscore": "239/8",
            "oppscore": "221",
            "result": "New Zealand won"
        }
    ]
}

```

```

{
    "name": "India",
    "matches": [
        {
            "vs": "New Zealand",
            "selfscore": "239/8",
            "oppscore": "221",
            "result": "New Zealand won"
        }
    ]
}

```

& so on.

& so on.

In the same loop, where we update args · excel = Workbook
teams array, library to write excel file in node → google ~~txt~~ excellib
args · CSV

we need to use createExcelFile function library
createExcelFile(teams, args · excel);
createFolders(teams, args · dataFolder)
function createExcelFile(teams)

let web = new Excel.Workbook();

for (var i = 0; i < teams.length; i++)

{ let sheet = web.addWorksheet(teams[i].name);

sheet.cell(2, 1).string("VS");

sheet.cell(2, 2).string("Self Score");

sheet.cell(2, 3).string("Opp Score");

sheet.cell(2, 4).string("Result");

for (let j = 0; j < teams[i].matches.length;

{ sheet.cell(2 + j, 1).string(teams[i].matches[j].VS);

sheet.cell(2 + j, 2).string(teams[i].matches[j].selfScore);

sheet.cell(2 + j, 3).string(teams[i].matches[j].oppScore);

sheet.cell(2 + j, 4).string(teams[i].matches[j].result);

sheet.cell(2 + j, 2).string(teams[i].matches[j].selfScore);

sheet.cell(2 + j, 3).string(teams[i].matches[j].oppScore);

sheet.cell(2 + j, 4).string(teams[i].matches[j].result);

web.write(args · excel);

पहले रूप से Template file लगाओ, उसको load करें, blank space में data सेट

function createscorecard (teamName,
match,
matchFileName)

```
{ let t1 = teamName ;  
  let t2 = match.vs ;  
  let t1S = match.selfScore ;  
  let t2S = match oppScore ;  
  let result = match.result ;
```

let bytesOfPDFTemplate = fs.readFileSync
("Template.pdf")

Template PDF को read देता है → RAM में स्टेटमेंट

let PDFDocPromise = pdf.PDFDocument.load
(bytesOfPDFTemplate);

bytes RAM में आएंगे तो वह pdf.PDFDocPromise बोल देगा यह PDF doc

pdfDocPromise.then(function(pdfdoc)

```
  let page = pdfdoc.getPage(0);  
  page.drawText(t2, {  
    x: 320,  
    y: 715,  
    size: 8  
  });
```

page.drawText(t1S, {

```
  x: 320,  
  y: 701,  
  size: 8  
});
```

Page.drawText
(t2, {
 x: 320,
 y: 715,
 size: 8
});

page.drawText(~~text~~, {x: 320,

y:

size: 8});

PAGE NO

DATE

page.drawText (

result

x: 320

y: 673

size: 8

});

→ ~~final PDF Bytes~~ changed bytes → ~~final PDF Bytes~~

→ let final PDF BytesKaPromise = pdfDoc.save();
→ finalPDFBytesKaPromise.then(function

~~bytes~~ ^{changed} ~~final PDF Bytes~~)

{ fs.writeFileSync(fileName, finalPDFBytes); }

});

y)

z)

file
~~at~~
to

changed bytes

Dec 35

8 Oct, 2021

Dev
14

Web Development

Done Revision of Dec - 13

Lec 36

Dev
15

9 Oct, 2021

Web Development

Lecture 14

Done Revision of Lecture 13

Lec 37

10 Oct, 2021

PAGE NO

DATE

Dev
16

Web Development

Project 1 → Complete Code

COMPLETE CODE

<code> → CricinfoExtractor.js
↳ File name

The explanation
of each line
of code
is
covered in
notes of
Lecture 13

CLI

1. npm init -y.
2. npm install minimist
3. npm install axios
4. npm install jsdom
5. npm install excel4node
6. npm install pdf-lib



```
node CricinfoExtractor.js --excel="worldcup.csv"  
--dataDir="data" --source=https://www.espncricinfo.com/  
series/icc-worldcup-2019-11415/matchresult
```

CODE

```
→ let minimist = require ("minimist");  
→ let axios = require ("axios");  
→ let jsdom = require ("jsdom");  
→ let excel = require ("excel");  
→ let fs = require ("fs");  
→ let pdf = require ("pdf-lib");  
  
→ let args = minimist (process.argv);
```

→ let responsePromise = axios.get(args, source);
 → responsePromise.then(function(response){

let html = response.data;
 let dom = new JSDOM(html);
 let document = dom.window.document;

→ let matches = [];
 → let matchDivs = document.querySelectorAll("div.match-score-block");
 → for(let i=0; i<matchDivs.length; i++)
 {
 let matchDiv = matchDivs[i];

let match = {

t1 = "",
 t2 = "",
 t1S = "",
 t2S = "",
 result = ""

}

→ let resultSpan = matchDiv.querySelector("div.status>span");

→ match.result = resultSpan.textContent;
 → let teamParas = matchDiv.querySelectorAll("div.name-detail>p.name");
 → match.t1 = teamParas[0].textContent;
 → match.t2 = teamParas[1].textContent;

```

→ let scoreSpans = matchDiv.querySelectorAll
   ("div.Score-detail > span");
→ if(scoreSpans == 2)
  {
    match.t1S = scoreSpans[0].textContent;
    match.t2S = scoreSpans[1].textContent;
  }
→ else if(scoreSpans == 1)
  {
    match.t1S = scoreSpans[0].textContent;
    match.t2S = " ";
  }
→ else
  {
    match.t1S = " ";
    match.t2S = " ";
  }
→ matches.push(match);
→ let matchesJSON = JSON.stringify(matches);
→ fs.writeFileSync("matches.json", matchesJSON,
   "utf-8");
→ let teams = [];
→ for(let i=0; i<matches.length; i++)
  {
    addTeamToTeamArrayIfMissing(teams, matches[i].t1);
    addTeamToTeamArrayIfMissing(teams, matches[i].t2);
  }

```

```
→ for(let i=0; i< matches.length; i++)  
  →   }  
  →   addMatchToSpecificTeam  
  →     (teams, matches[i].t1, matches[i].t2,  
  →      , matches[i].t1s, matches[i].t2s,  
  →      matches[i].result);  
  →   addMatchToSpecificTeam  
  →     (teams, matches[i].t2, matches[i].t1,  
  →      matches[i].t2s, matches[i].t1s,  
  →      matches[i].result);  
  → }  
→ let teamsAsJSON = JSON.stringify(teams);  
→ fs.writeFileSync ("teams.json", teamsAsJSON, "utf-8");  
→ prepareExcel(teams, args.excel);  
→ prepareFoldersAndPDFs(teams, args, dataDir);  
  → }
```

function prepareExcel(teams, excelFileName)

{

 let web = new excel4Node.Workbook();

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

{

 let tsheet = web.addWorksheet(teams[i].name);

 tsheet.cell(1, 1).string("VS");

 tsheet.cell(1, 2).string("Self Score");

 tsheet.cell(1, 3).string("Opp Score");

 tsheet.cell(1, 4).string("Result");

 for(let j=0; j< teams[i].length; j++)

{

 tsheet.cell(2+j, 1).string(teams[i].matches[j].vs);

 tsheet.cell(2+j, 2).string(teams[i].matches[j].selfscore);

 tsheet.cell(2+j, 3).string(teams[i].matches[j].oppScore);

 tsheet.cell(2+j, 4).string(teams[i].matches[j].result);

VS	Self Score	Opp Score	Result
Pakistan	221	315/9	Pakistan won by 94 runs.
India	286	314/9	India won by 28 runs.
Afghanistan	269/7	200	New Zealand won by 29 runs.
Australia	333/8	381/5	Australia won by 65 runs.
West Indies	332/3	321/5	West Indies won by 42 runs.
Sri Lanka			Abandoned with a ball bowled.
England	280/6	386/6	England won by 106 runs.
Bangladesh	240/1	248/8	Bangladesh won by 6 runs.
South Africa	330/6	309/8	New Zealand won by 21 runs.
New Zealand			

function prepareFoldersAndPDFs(teams, dataDir)

{

if(fs.existsSync(dataDir) == true)

{

fs.rmdirSync(dir, recursive: true);

}

fs.mkdirSync(dataDir);

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

{

let teamFolderName =

path.join(dataDir, teams[i].name);

if(fs.existsSync(teamFolderName) == false)

{

fs.mkdirSync(teamFolderName);

}

for(let j=0; j<teams[i].matches.length; j++)

{

let match = teams[i].matches[j];

createMatchScoreCardPdf(teamFolderName, homeTeam, match);

}

}

}

Worldcup 2019

Team 1	
Team 2	
Team 1's Score	
Team 2's Score	



Save as : Template.pdf

Save in the same folder as of Code

```
→ function createMatchScorecardPDF (teamFolderName,  
          homeTeam, match)  
{  
    → let matchFileName = path.join(  
        teamFolderName, match.vst + "pdf");  
    → let templateFileBytes = fs.readFileSync ("Template.pdf");  
    → let pdfdocPromise = pdf.PDFDocument.load  
        (templateFileBytes);  
    → pdfdocPromise .then (function (pdfdoc)  
    {  
        → let page = pdfdoc .getPage (0);  
        → page .drawText (homeTeam, {  
            x: 320,  
            y: 8700,  
            size: 8,  
            y});  
        → page .drawText (match .res, {  
            x: 320,  
            y: 685,  
            size: 8,  
            y});  
        → page .drawText (match .selfScore, {  
            x: 320,  
            y: 658,  
            size: 8,  
            y});  
    })  
}
```

Printed on	
Date	

→ page.drawText(matchResult)

x: 320

y: 643

size: 8.

y);

→ let unchangedBytes = pdfDoc.save();

→ unchangedBytes.then(function(changedBytes){

}

if(fs.existsSync(matchFileName + ".pdf") == true){

{ fs.writeFileSync(matchFileName + ".pdf", changedBytes); }

}

else{

{

fs.writeFileSync(matchFileName + ".pdf", unchangedBytes);

}

}

}

}

रखे दिये गये pdf
से अलग
हो सकता है

उत्तर

फ़िलेमें
की file

exist करती है

तो उसी
name
के आगे

1 add

करके

new

name

दोस्रे

pdf

बनादी

ऐसा करने

से नहीं

दूसरे pdf

हो सकता

data

delete

करके नहीं

pdf हो जाएगा

जाएगा।