PLAYER MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

BL.EN.U4CSE18073	Mirza Kaazima Ifrah
BL.EN.U4CSE18083	Nuthi Sriram
BL.EN.U4CSE18091	Potti Priya

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

AMRITA SCHOOL OF ENGINEERING, BANGALORE

AMRITA VISHWA VIDYAPEETHAM

BANGALORE 560 035

December - 2021

ACKNOWLEDGEMENT

The satisfaction that accompanies successful completion of any task would be incomplete without mention of people whom made it possible, and whose constant encouragement and guidance have been source of inspiration throughout the course of this project work.

We offer our sincere pranams at the lotus feet of **"AMMA", MATA AMRITANANDAMAYI DEVI** who showered her blessings upon us throughout the course of this project work. We owe our gratitude to **Br. Viswamrita Chaitanya Swamiji**, Director, Amrita School of Engineering, Bangalore.

We thank **Dr. Sriram Devanathan**, Principal and Chairperson, Department of Computer Science and Engineering, Amrita School of Engineering, Bangalore for his encouragement, inspiration and support during the course of work.

It is a great pleasure to express our gratitude and indebtedness to our project guides **Ms. Sreevidya B., Mr. Rajesh M.** Assistant Professor, Department of Computer Science and Engineering, Amrita School of Engineering, Bangalore for their valuable guidance, encouragement, moral support and affection throughout the project work. Finally, we are forever grateful to our parents, who have loved, supported and encouraged us in all our endeavors.

ABSTRACT

The purpose of this study is to develop a Distributed Player Management System. Many professional sport organizations are currently in the process of finding or already using sports information systems (SIS) to integrate data from different information and measurement systems. The problem is that they are not designed keeping the stakeholders in mind and provide functionality that is partially implemented and does not cater to the needs of the users of the system in its entirety. Keeping the stake holders - Stats man, Selection Committee and Enthusiasts in mind, an all-in-one distributed platform has been developed keeping the convenience of the end users in mind. Distributed Player Management System is a one stop location for the stake holders to Store, Retrieve, Update and Analyze the Player Stats in real-time.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

- HTML - Hypertext Markup Language
- CSS - Cascading Style Sheets
- JS - JavaScript
- EJS - Embedded JavaScript Templating
- API - Application Programming Interface
- NoSQL - Not Only Structured Query Language
- JSON - JavaScript Object Notation
- BSON - Binary JavaScript Object Notation
- RAM - Random Access Memory
- MB - Mega Bytes
- DB - Data Base
- BF - Balls Faced

CHAPTER 1 – INTRODUCTION

1.1 Introduction

Through this project we intend to digitize and simplify the ways in which cricket players' statistics are analyzed and maintained. The major objective of this application is to minimize the cumbersome process that usually takes place while analyzing and maintaining the stats. This reduces the hard manual paperwork and makes the process trouble free. The project contains a vast database with a few Collections. One Collection holds the information about the players (Name, player id, region), the second table holds the statistics of the player (matches, innings, runs, highest score, average, 50's, 100's, wickets, economy, strike rate), the third table holds the information about players' personal details (date of birth, age, birth place).

1.2 Motivation

To ease the life of Stats Man, Selection Committee, and Enthusiasts by digitizing the menial tasks usually done by Stats Man and Selection Committee.

1.3 Problem Statement

As mentioned all the data being kept by the organizers is on paper and even if data is being updated continuously the data is not integral and consistent. And fans and enthusiasts who are eagerly checking the stats of their favorite cricket player need a player management system which can handle the load of a lot of enthusiasts and it has to be able to keep up with the update rate of these stats throughout the match.

CHAPTER 2 – REQUIREMENTS

2.1 Software Requirements

Browser with latest updates

2.2 Hardware Requirements

RAM - Minimum 128 MB

Processor - At least Pentium 3

Good internet connectivity

2.3 Technology Stack

Frontend - HTML, CSS, JavaScript, EJS

Middleware - Mongoose

Backend - Node.js

Database - MongoDB

CHAPTER 3 – DESIGN

The design of the web app is extremely simple to understand and Fig 3.1 depicts a variety of use cases and user groups for this system.

3.1 Use Case Diagram

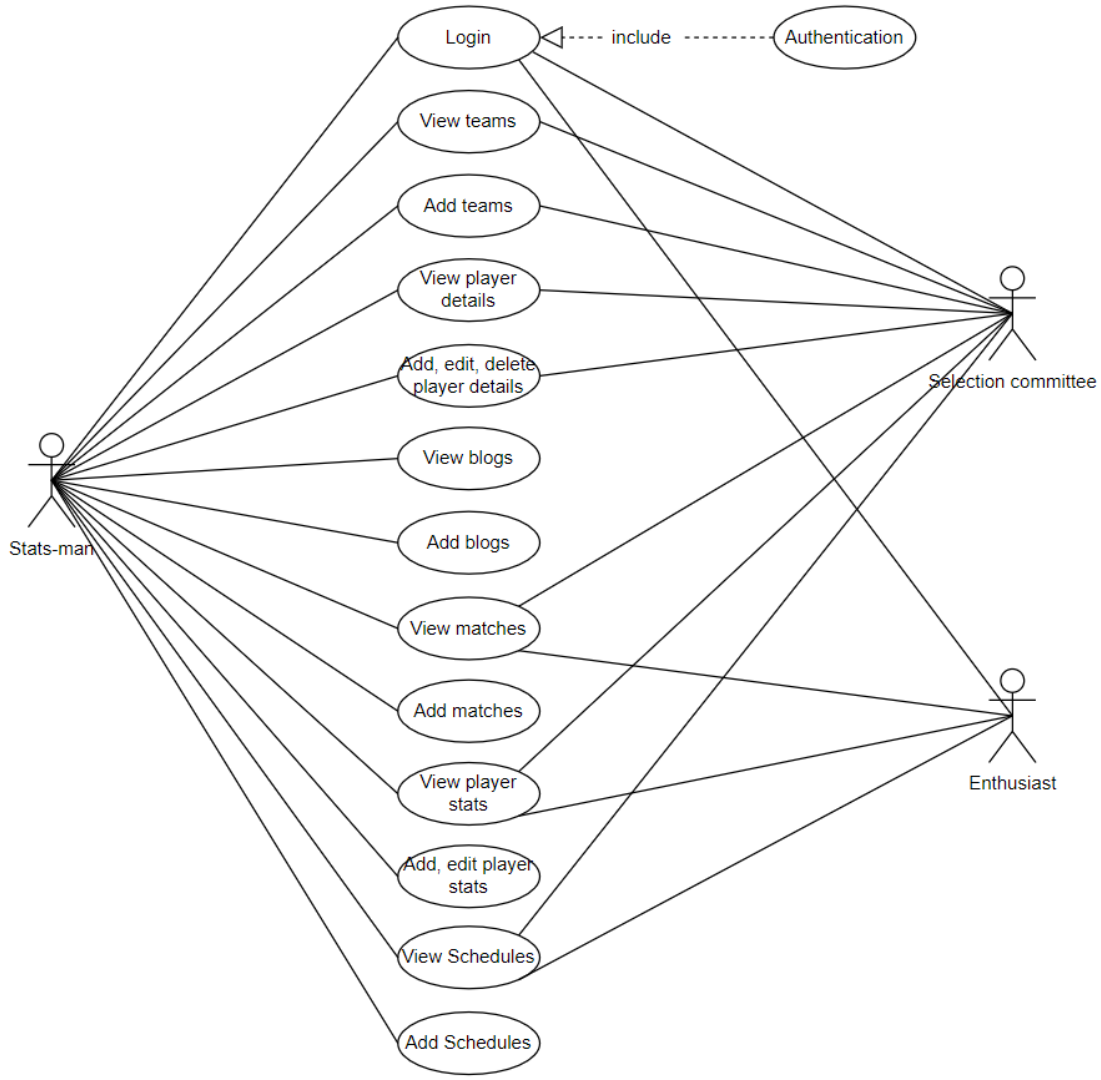

Fig 3.1 Use case diagram

CHAPTER 4 – IMPLEMENTATION

4.1 MongoDB

MongoDB is a document-based, NoSQL database. It stores data in JSON-like documents. MongoDB environment provides a server, which when started can be used to create multiple databases. This server converts the JSON data into a binary form known as BSON in the backend. Thus, it is queried efficiently and is fast. Several MongoDB servers work together to form a MongoDB cluster.

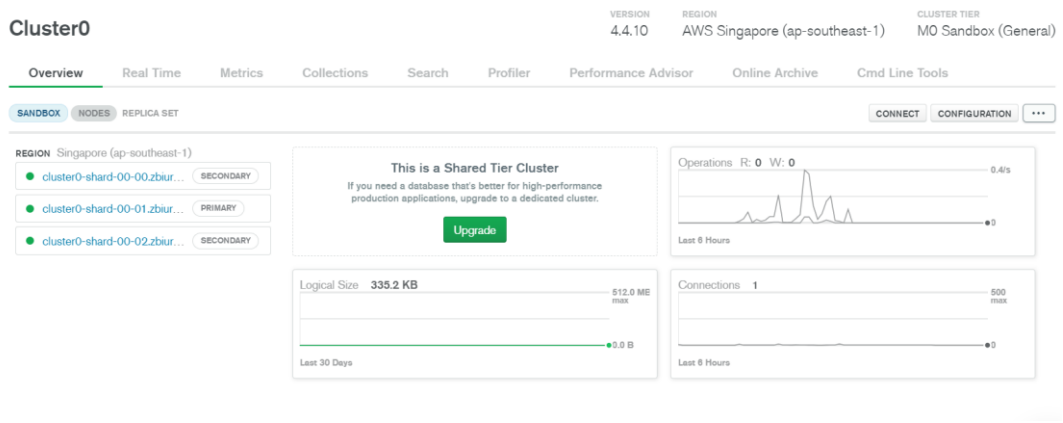

Fig 4.1 Creating a cluster in MongoDB

4.2 Web App Features

Stats man dashboard:

- Take notes in the app in form of mini microblogging
- Continuously keep track of the cricket player & match statistics
- Make schedule of the matches

Selection Committee dashboard:

- Access to players personal information (mobile number, email, etc.)
- View access to cricket player & match stats
- View access of the cricket schedule

Enthusiast dashboard:

- View access to cricket player & match stats
- View access of the cricket schedule

4.3 Schema Design

Match Collection:

```
var matchSchema = new mongoose.Schema({
name:{
type:String
},
image:{
type:String
},
user:{
type:String
}
});
```

Articles Collection:

```
var memorySchema = new mongoose.Schema({
image:{
type:String
},
title:{
type:String
},
description:{
type:String
},
user:{
type:String
},
date:{
```

```
type:Date,
default:Date.now()
}
});
```

Player Collection:

```
var playerSchema = new Schema({
name:{
type:String
},
fatherName:{
type:String
},
dateOfBirth:{
type:String
},
preTeam:{
type:String
},
address:{
type:String
},
mobileNo:{
type:Number
},
email:
{
type:String
},
user:{
```

```
type:String
},
team:{
type:String
},
date:{
type:Date,
default:Date.now()
}
});
```

Player Match Stats Collection:

```
var playerMatchSchema = new Schema({
name:{
type:String
},
nomatch:{
type:Number
},
user:{
type:String
},
runs:{
type:String
},
BF:{
type:String
},
hundreds:{
```

```
type:String
},
fiftys:{
type:Number
},
fours:
{
type:String
},
sixs:{
type:String
},
team:{
type:String
},
match:{
type:String
},
date:{
type:Date,
default:Date.now()
}
});
```

Schedule Collection:

```
var scheduleSchema = new mongoose.Schema({
Teamname:{
type:String
},
```

```
scheduleDate:{
type:Date
},
time:{
type:String
},
user:{
type:String
},
date:{
type:Date,
default:Date.now()
}
});
```

Teams collection:

```
var teamSchema = new mongoose.Schema({
name:{
type:String
},
image:{
type:String
},
user:{
type:String
}
});
```

User collection:

```
var userSchema = new mongoose.Schema({
```

```
username:String,
password:String
});
```

CHAPTER 5 – RESULTS

After running our application, the following results were achieved:

Fig 5.1 Home page

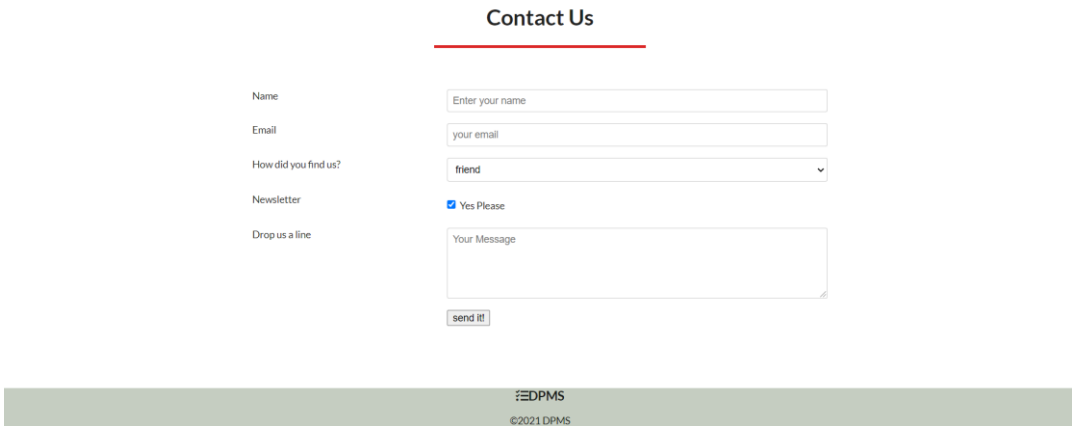

Fig 5.2 Contact us page

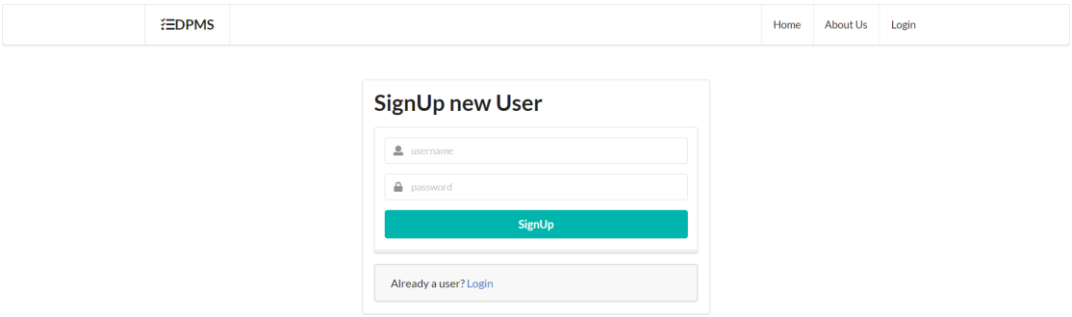

Fig 5.3 Sign up page

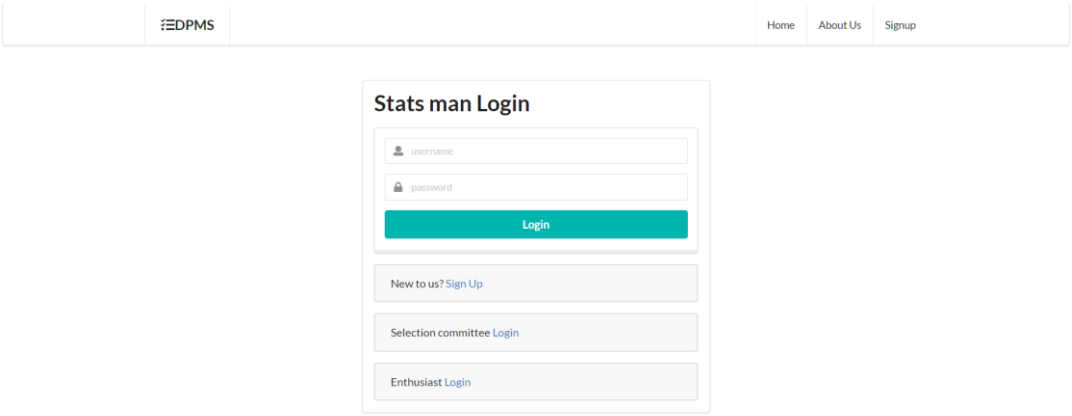

Fig 5.4 Stats-man login page

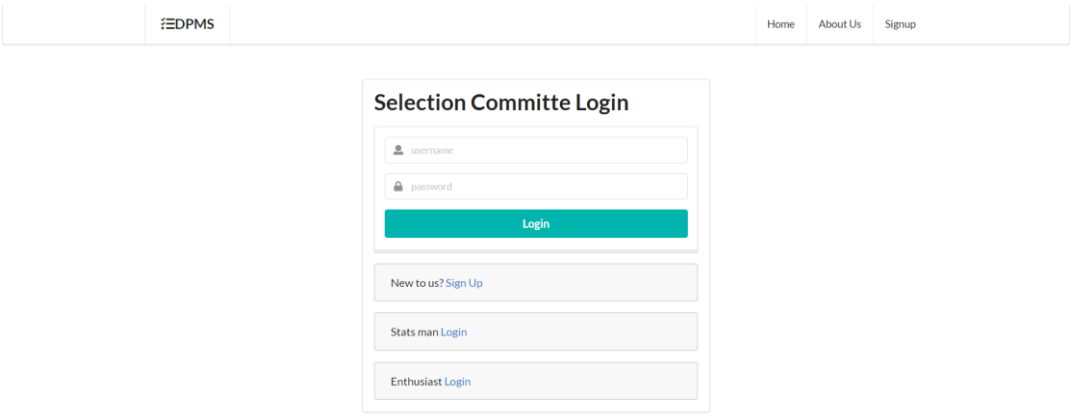

Fig 5.5 Selection Committee login page

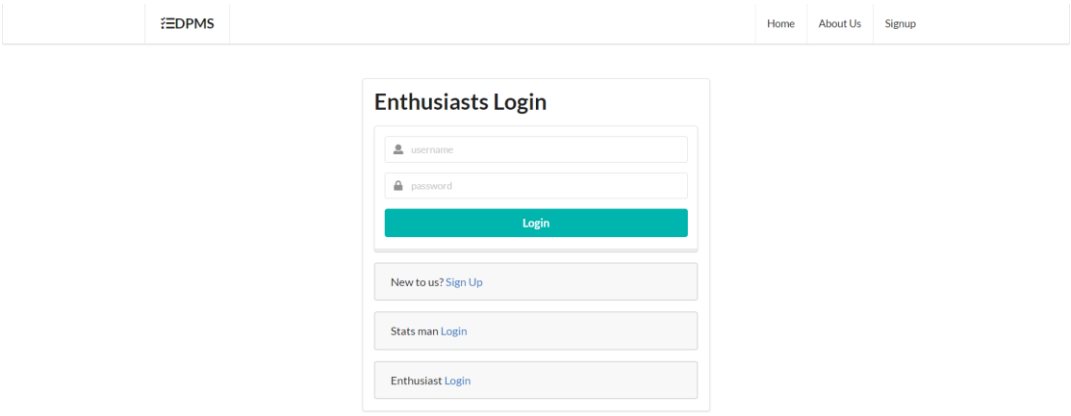

Fig 5.6 Enthusiasts' login page

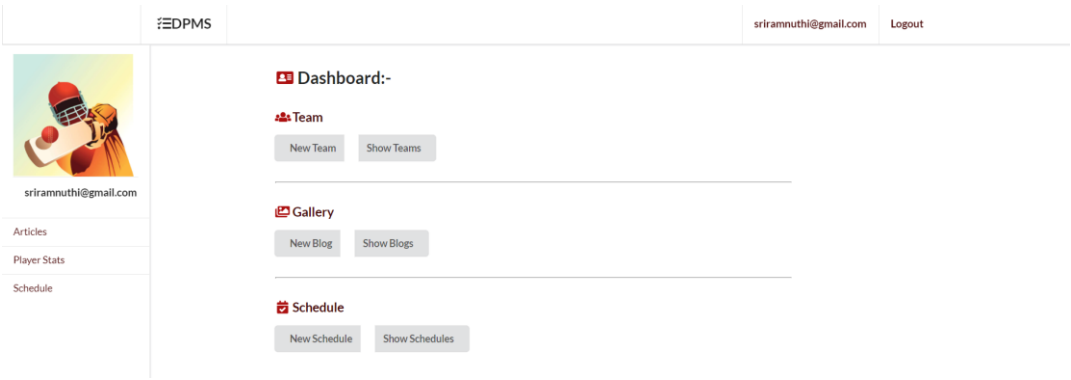

Fig 5.7 Stats-man dashboard

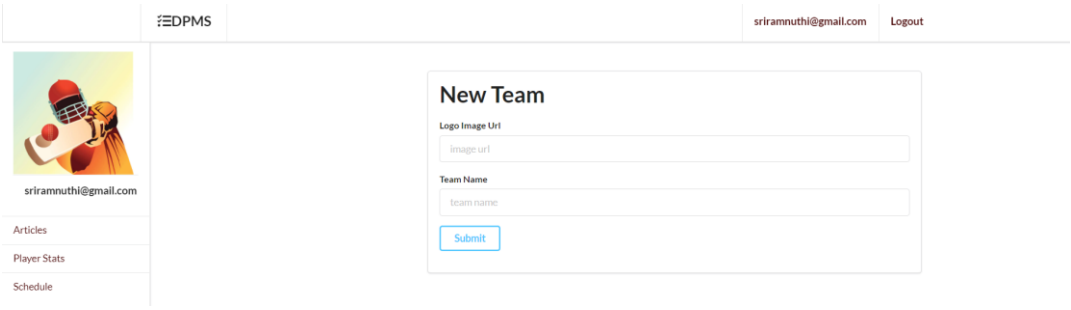

Fig 5.8 Add new team

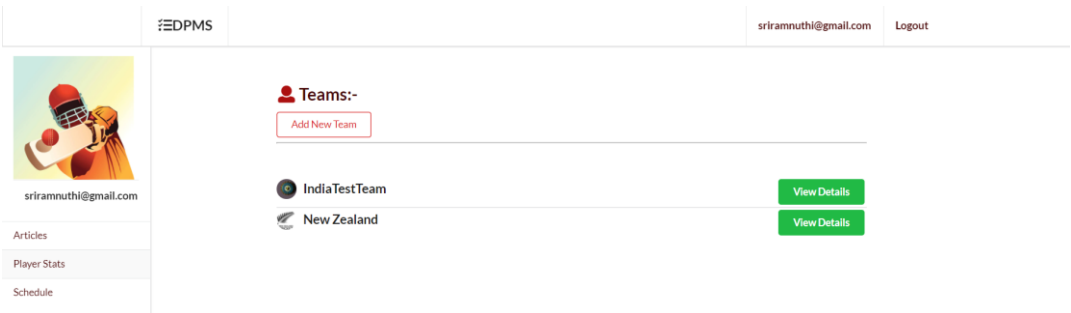

Fig 5.9 Show teams

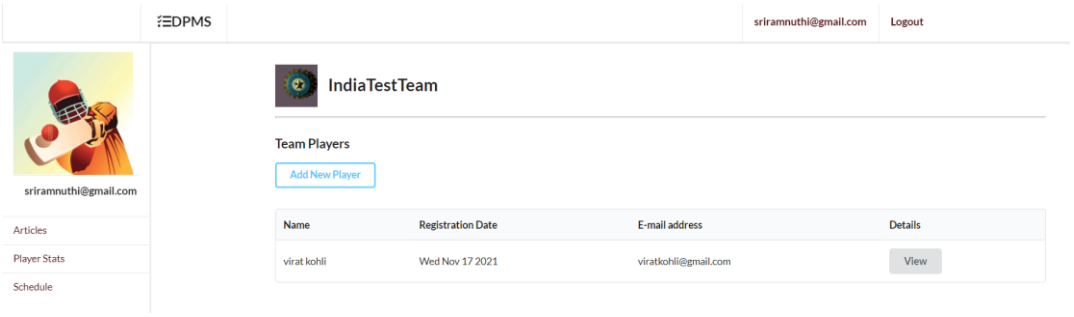

Fig 5.10 View players in a team

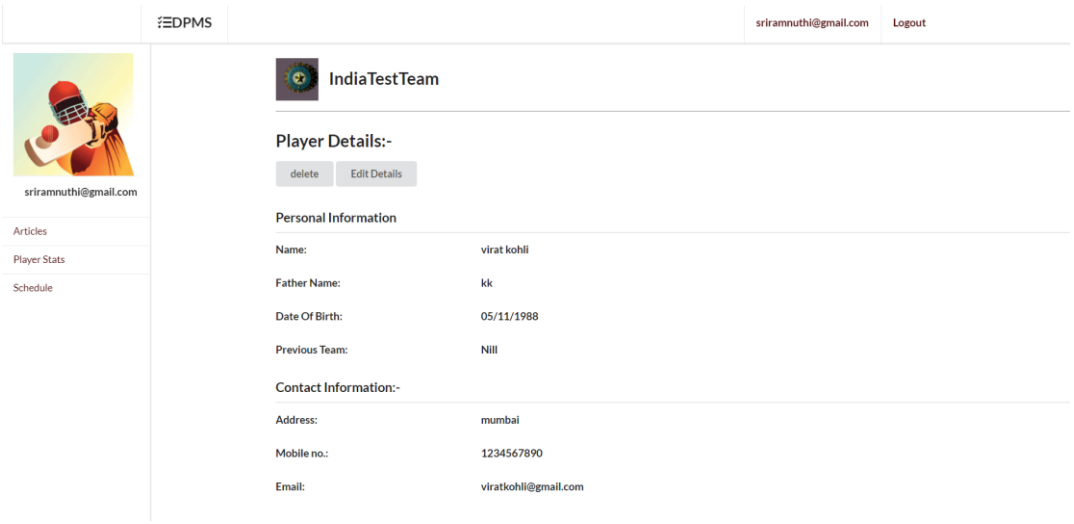

Fig 5.11 View player details

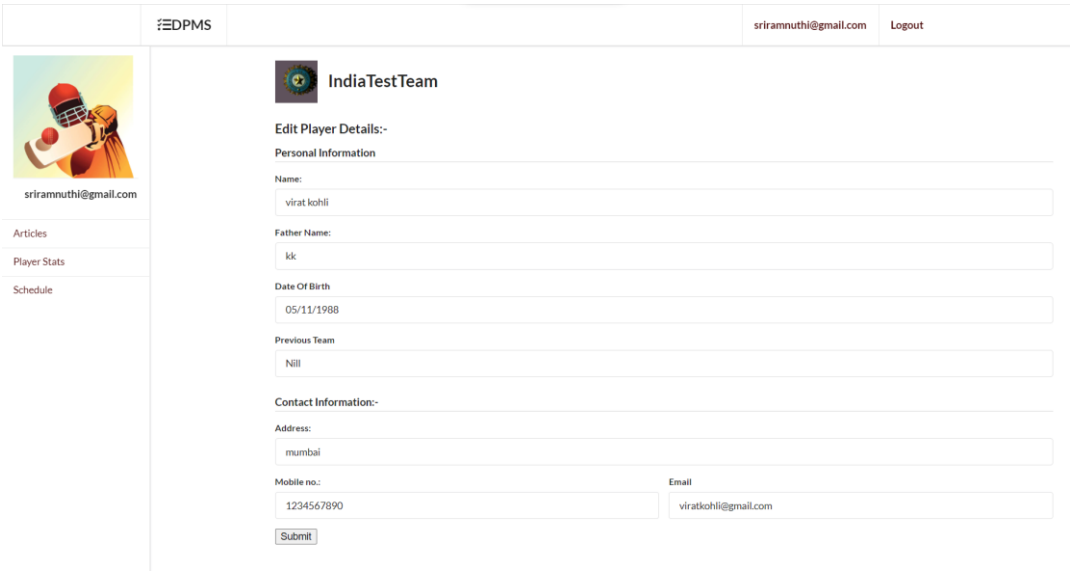

Fig 5.12 Edit player details

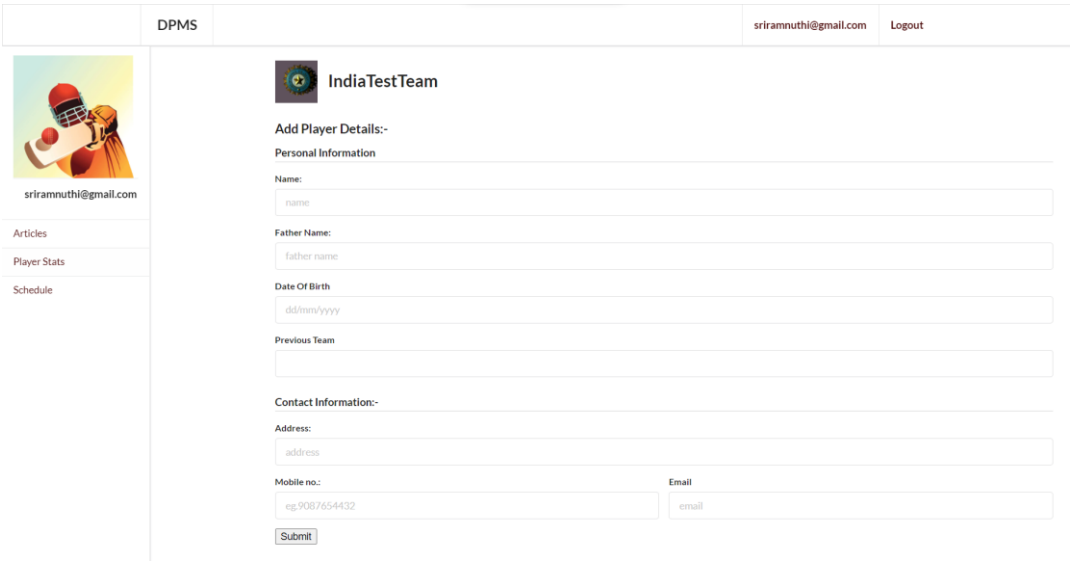

Fig 5.13 Add player details

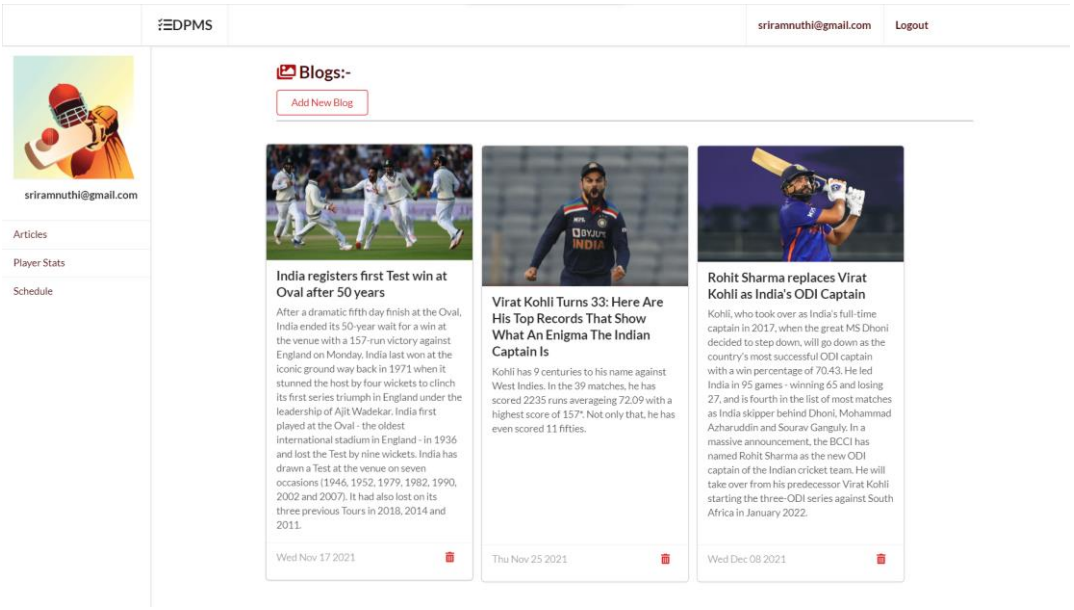

Fig 5.14 Blogs page

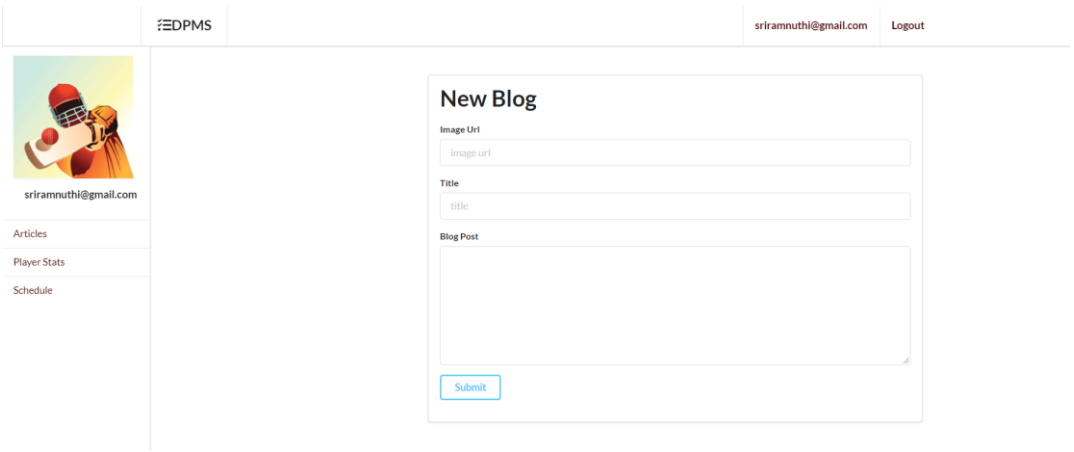

Fig 5.15 Add new blog

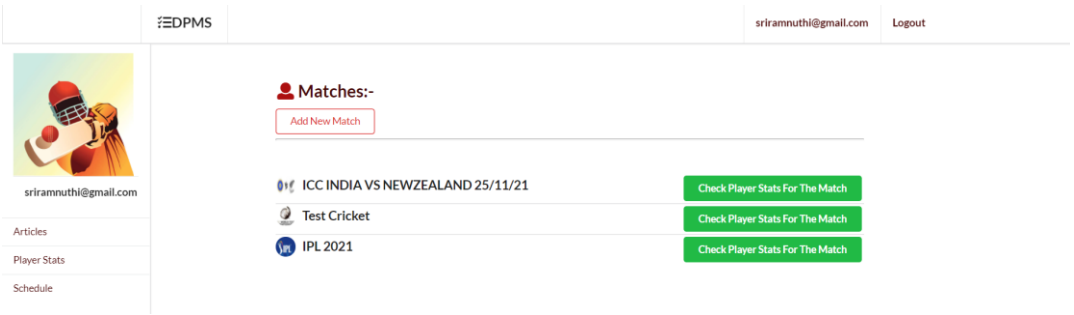

Fig 5.16 View matches

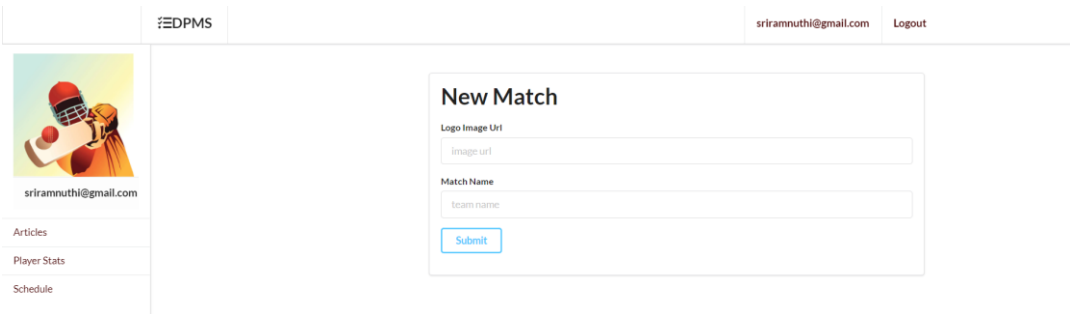

Fig 5.17 Add new match

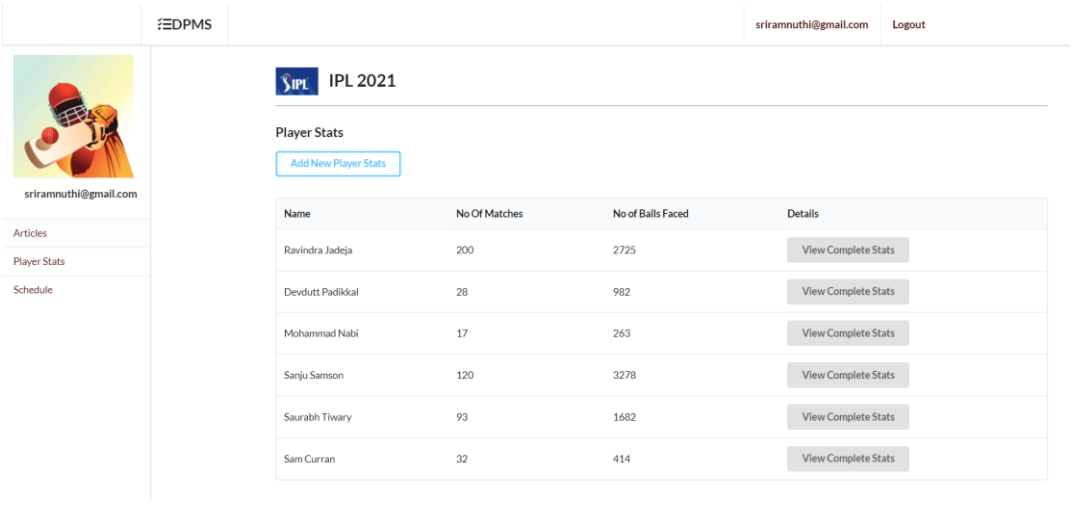

Fig 5.18 View player stats for a match

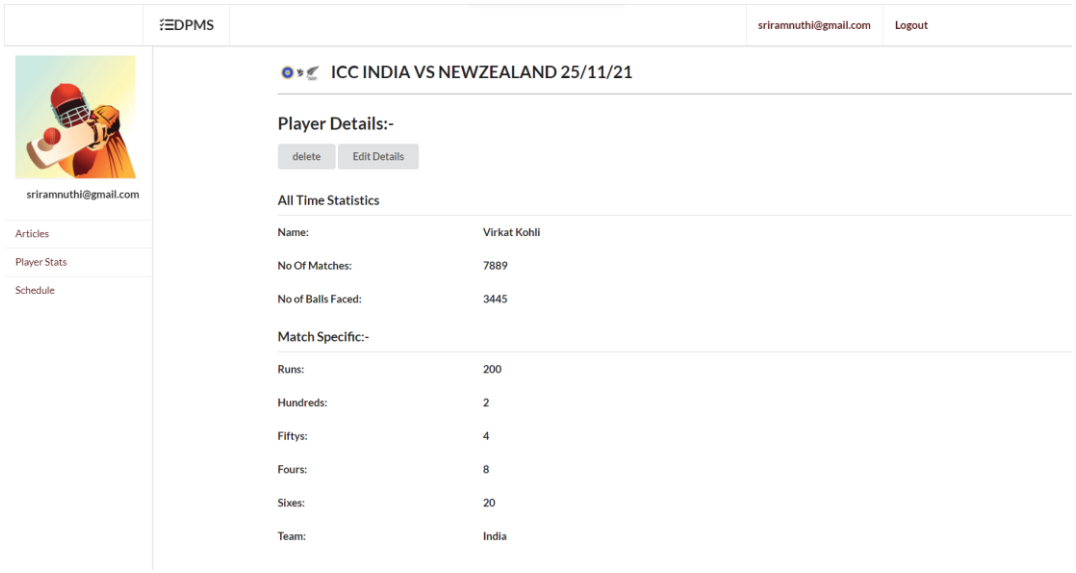

Fig 5.19 Complete player stats

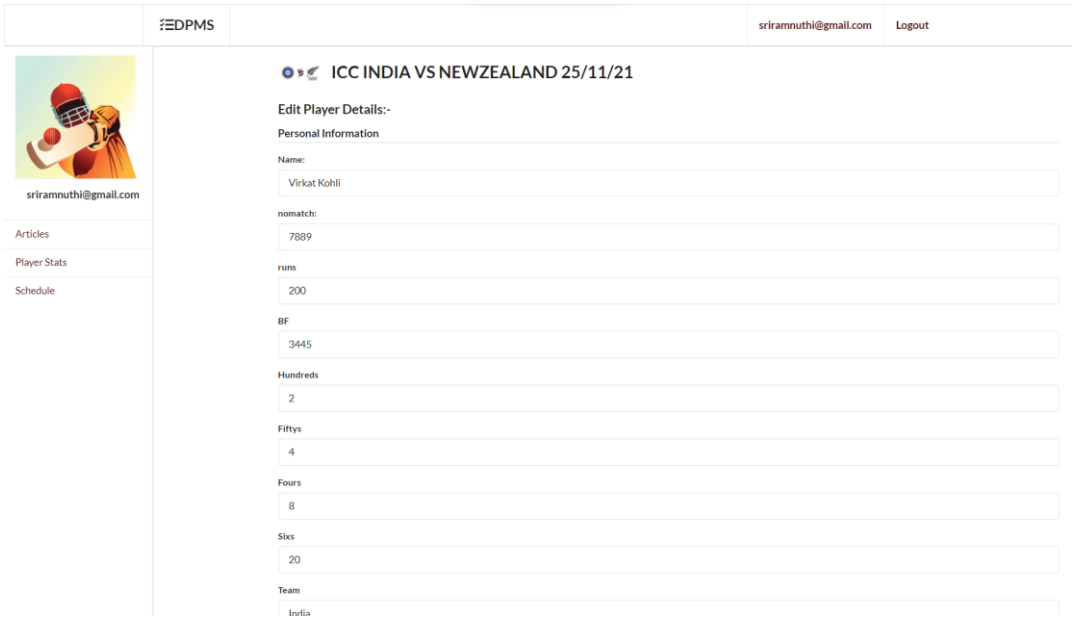

Fig 5.20 Edit player stats

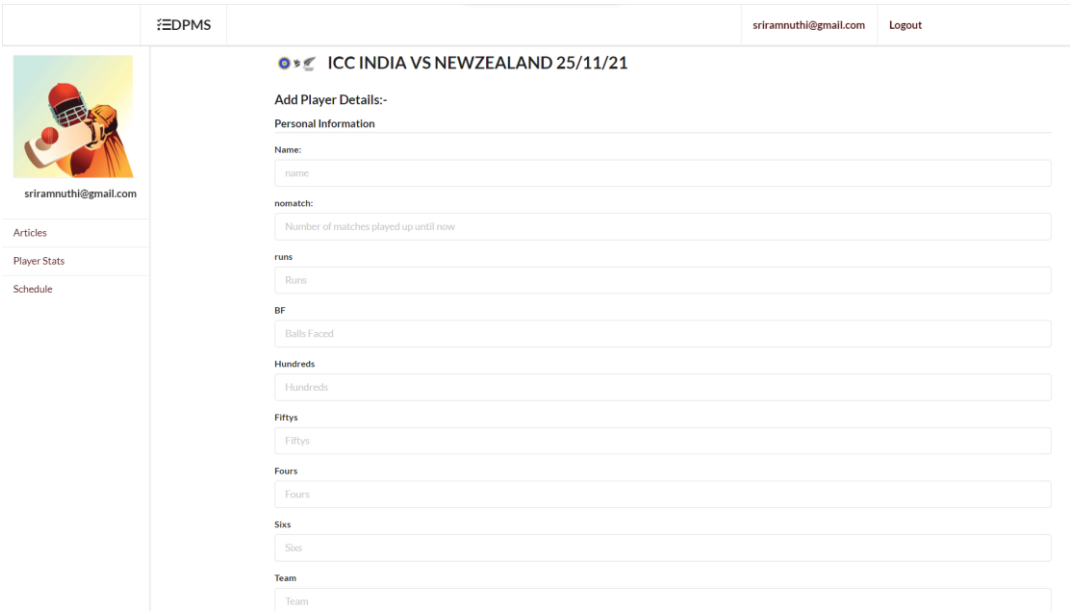

Fig 5.21 Add player stats

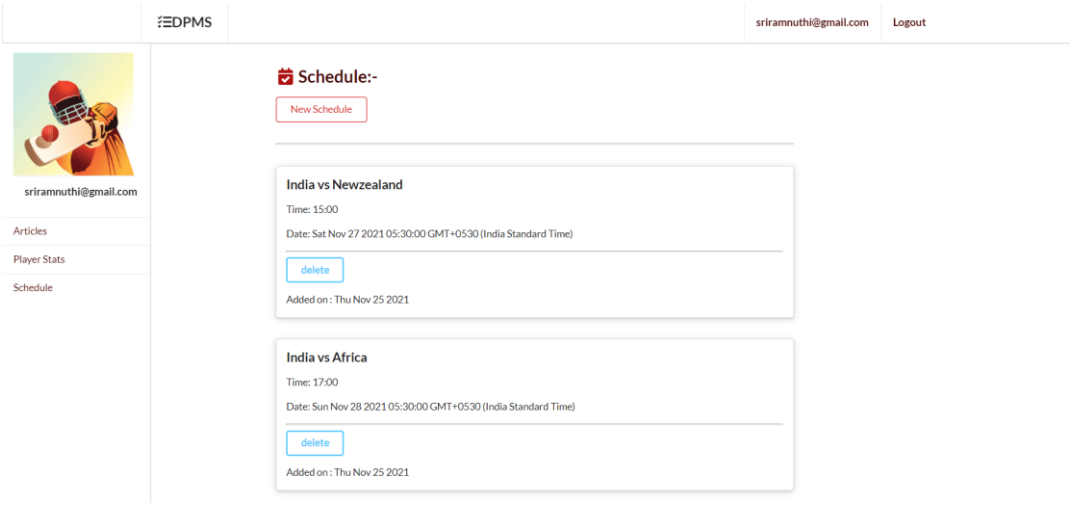

Fig 5.22 View schedules

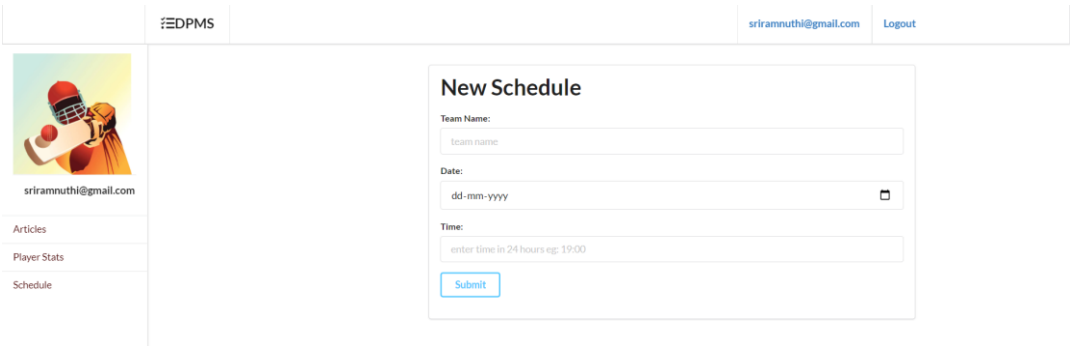

Fig 5.23 Add new schedule

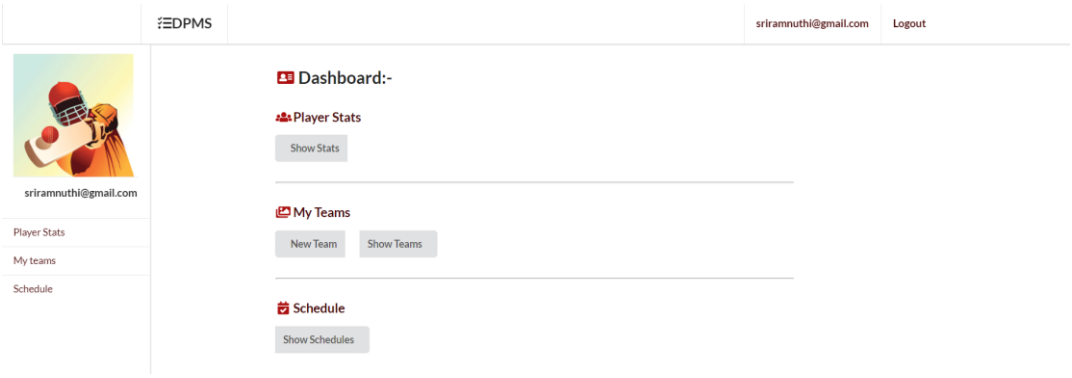

Fig 5.24 Selection committee dashboard

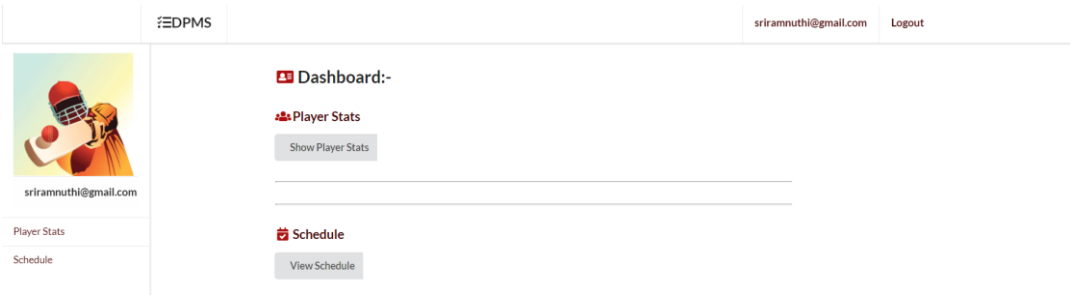

Fig 5.25 Enthusiasts' dashboard

CHAPTER 6 – CONCLUSION

We have made a fully functional web application, which eliminates the time-consuming process that usually takes place while analyzing and maintaining the statistics. This in turn reduces the hard manual paperwork and makes the entire process trouble-free. Thus, we have developed an end-to-end solution for a real-life scenario using MongoDB, a document-based, NoSQL database.

REFERENCES

[1] https://en.wikipedia.org/wiki/Cricket_statistics

[2] https://docs.mongodb.com/manual/crud/

[3] https://www.mongodb.com/atlas/database