

# **Database and Management System**

## **Project Report**

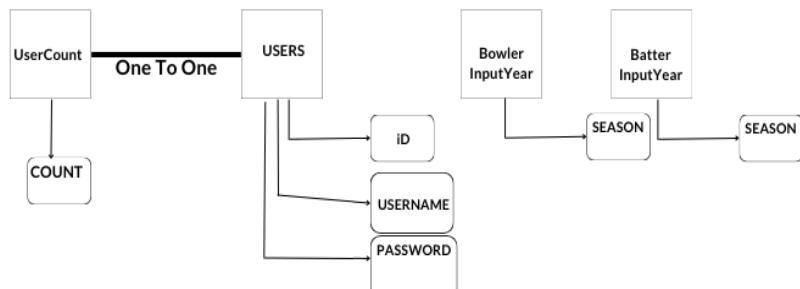
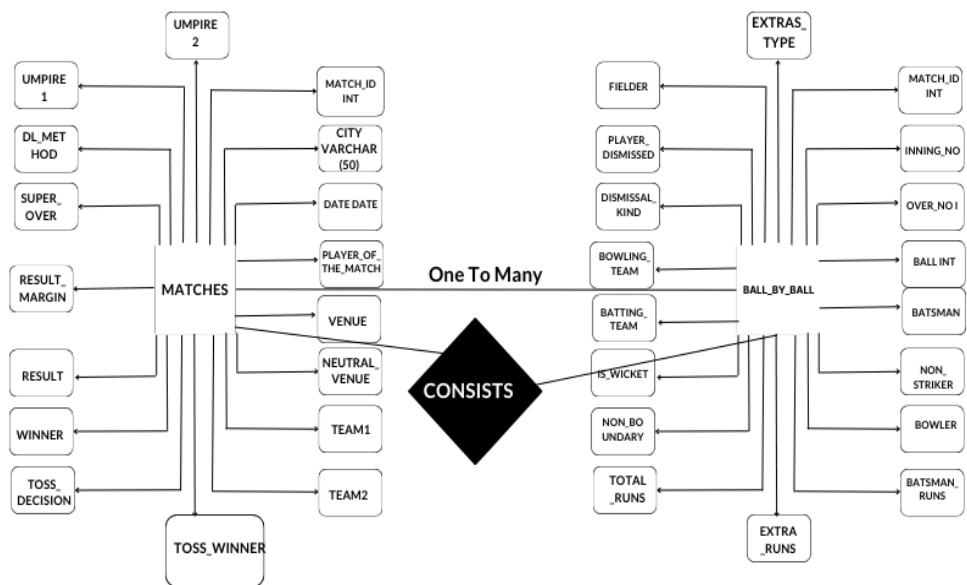
### **Team Members**

- Kaushik J - PES1UG21CS272
- Korey Varun - PES1UG21CS283

## Short Abstract

The IPL Analytics Dashboard, developed for the Database and Management System course, is a Flask-based web application offering insights into the Indian Premier League (IPL). This project utilizes a MySQL database, implementing stored procedures for efficient data retrieval and visualization. Users, categorized as admin or normal users, can register, log in, and explore detailed analytics on general match statistics, individual player performances, and team metrics. Admin users enjoy additional functionalities, such as user management operations (create, update, delete). The project emphasizes database triggers for maintaining user counts and underscores the integration of database management principles within a dynamic analytics framework.

## ER Diagram



## **Relational Schema**

MATCHES Table:

MATCHES (MATCH\_ID, CITY, DATE, PLAYER\_OF\_THE\_MATCH, VENUE, NEUTRAL\_VENUE, TEAM1, TEAM2, TOSS\_WINNER, TOSS\_DECISION, WINNER, RESULT, RESULT\_MARGIN, SUPER\_OVER, DL\_METHOD, UMPIRE1, UMPIRE2)

Where: MATCH\_ID is the primary key.

BALL\_BY\_BALL Table:

BALL\_BY\_BALL (MATCH\_ID, INNING\_NO, OVER\_NO, BALL, BATSMAN, NON\_STRIKER, BOWLER, BATSMAN\_RUNS, EXTRA\_RUNS, TOTAL\_RUNS, NON\_BOUNDARY, IS\_WICKET, DISMISSAL\_KIND, PLAYER\_DISMISSED, FIELDER, EXTRAS\_TYPE, BATTING\_TEAM, BOWLING\_TEAM)

Where: (MATCH\_ID, INNING\_NO, OVER\_NO, BALL) is the primary key.  
MATCH\_ID is a foreign key referencing MATCHES(MATCH\_ID).

USERS Table:

USERS (ID, USERNAME, PASSWORD) Where: ID is the primary key.

USERCOUNT Table:

USERCOUNT (COUNT)

This table is used to record the count of the number of users in combination with a trigger.

BATTERINPUTYEAR Table:

BATTERINPUTYEAR (SEASON)

This table is used for storing the input of the dynamic stored procedure.

BOWLERINPUTYEAR Table:

BOWLERINPUTYEAR (SEASON)

This table is used for storing the input of the dynamic stored procedure.

## DDL SQL Commands

```
CREATE TABLE MATCHES(
MATCH_ID INT NOT NULL,
CITY VARCHAR(50),
DATE DATE,
PLAYER_OF_THE_MATCH VARCHAR(50),
VENUE VARCHAR(150),
NEUTRAL_VENUE BOOL DEFAULT 0,
TEAM1 VARCHAR(50),
TEAM2 VARCHAR(50),
TOSS_WINNER VARCHAR(50),
TOSS_DECISION VARCHAR(20),
WINNER VARCHAR(50),
RESULT VARCHAR(20),
RESULT_MARGIN INT,
SUPER_OVER ENUM('Y','N') DEFAULT 'N',
DL_METHOD ENUM('NA','D/L') DEFAULT 'NA',
UMPIRE1 VARCHAR(50),
UMPIRE2 VARCHAR(50),
PRIMARY KEY(MATCH_ID)
);
```

```
CREATE TABLE BALL_BY_BALL(
MATCH_ID INT NOT NULL,
INNING_NO INT,
OVER_NO INT,
BALL INT,
BATSMAN VARCHAR(50),
NON_STRIKER VARCHAR(50),
BOWLER VARCHAR(50),
BATSMAN_RUNS INT,
EXTRA_RUNS INT,
TOTAL_RUNS INT,
NON_BOUNDARY BOOL DEFAULT 0,
IS_WICKET BOOL DEFAULT 0,
DISMISSAL_KIND VARCHAR(20),
PLAYER_DISMISSED VARCHAR(50),
FIELDER VARCHAR(50),
EXTRAS_TYPE VARCHAR(20),
```

```
BATTING_TEAM VARCHAR(50),
BOWLING_TEAM VARCHAR(50),
PRIMARY KEY(MATCH_ID,INNING_NO,OVER_NO,BALL),
FOREIGN KEY(MATCH_ID) REFERENCES MATCHES(MATCH_ID)
);
```

```
CREATE TABLE USERS (
ID INT AUTO_INCREMENT PRIMARY KEY,
USERNAME VARCHAR(255) NOT NULL,
PASSWORD VARCHAR(255) NOT NULL );
```

```
CREATE TABLE UserCount (
Count INT DEFAULT 0 );
```

```
CREATE TRIGGER after_insert_user
AFTER INSERT ON Users
FOR EACH ROW
BEGIN
UPDATE UserCount SET Count = Count + 1;
END;
```

```
CREATE TRIGGER after_delete_user
AFTER DELETE ON Users
FOR EACH ROW
BEGIN
UPDATE UserCount SET Count = Count - 1;
END;
```

```
CREATE PROCEDURE CalculateTeamWinPercentage()
BEGIN
CREATE TEMPORARY TABLE IF NOT EXISTS
result_of_calculate_team_win_percentage (
    TeamName VARCHAR(255),
    WinPercentage DECIMAL(5,2)
);
```

```
INSERT INTO result_of_calculate_team_win_percentage
SELECT
    TeamName,
```

```

        (SUM(CASE WHEN Winner = TeamName THEN 1 ELSE 0 END) / COUNT(*)) *
100 AS WinPercentage
FROM (
    SELECT Team1 AS TeamName, Winner FROM MATCHES
    UNION ALL
    SELECT Team2 AS TeamName, Winner FROM MATCHES
) AS Teams
GROUP BY TeamName
ORDER BY WinPercentage DESC;
SELECT * FROM result_of_calculate_team_win_percentage;
END //

```

```

CREATE PROCEDURE GetTopBatsmenByRuns(IN seasonYear INT)
BEGIN
    SELECT BB.BATSMAN, SUM(BB.BATSMAN_RUNS) AS TotalRuns
    FROM BALL_BY_BALL BB
    JOIN MATCHES M ON BB.MATCH_ID = M.MATCH_ID
    WHERE BB.BATSMAN_RUNS > 0 AND YEAR(M.DATE) = seasonYear
    GROUP BY BB.BATSMAN
    ORDER BY TotalRuns DESC
    LIMIT 10;
END //

```

```

CREATE PROCEDURE GetTopBowlersByWickets(IN seasonYear INT)
BEGIN
    SELECT
        BB.BOWLER AS Bowler,
        COUNT(CASE WHEN BB.IS_WICKET = 1 THEN 1 ELSE NULL END) AS
TotalWickets
    FROM
        BALL_BY_BALL BB
    JOIN
        MATCHES M ON BB.MATCH_ID = M.MATCH_ID
    WHERE
        BB.IS_WICKET = 1 AND YEAR(M.DATE) = seasonYear
    GROUP BY
        BB.BOWLER
    ORDER BY
        TotalWickets DESC LIMIT 10;
END //

```

## CRUD Operations Screenshots

### Create New User

#### Before Creation of User

The screenshot shows a web browser window titled "Admin Page" with the URL "127.0.0.1:5000/admin". The main content is the "IPL ANALYTICS DASHBOARD ADMIN PAGE".

**Users**

ID	Username	Password
1	admin	admin

**Create User**

Username:

Password:

**Insert User**

**Update User**

User ID to Update:

New Username:

New Password:

## After Creation of User

The screenshot shows a web browser window titled "Admin Page" at the URL "127.0.0.1:5000/admin". The main title of the page is "IPL ANALYTICS DASHBOARD ADMIN PAGE". Below it, there is a section titled "Users" containing a table with two rows of data:

ID	Username	Password
1	admin	admin
19	kaushik	kaushik123

Below the table, there are two forms: "Create User" and "Update User".

**Create User**

Username:

Password:

**Insert User**

**Update User**

User ID to Update:

New Username:

The new user can log in with the below credentials

Username:- kaushik

Password:- kaushik123

The admin page would not be visible to the new user and is exclusive only to the admin user.

## Read Details

The contents of the table throughout are displayed which is a Read operation.

## Update User Details

### Before Updation

The screenshot shows a web browser window titled "Admin Page" with the URL "127.0.0.1:5000/admin". The page contains two main sections: "Create User" and "Update User".

**Create User Section:**

- A table with columns "ID", "Username", and "Password". It contains two rows:
  - Row 1: ID 1, Username admin, Password admin
  - Row 2: ID 19, Username kaushik, Password kaushik123
- A "Create User" form with fields for "Username" and "Password", and a green "Insert User" button.

**Update User Section:**

- A "User ID to Update" input field containing the value "19".
- A "New Username" input field containing the value "kaushik\_2023".
- A "New Password" input field containing the value "\*\*\*\*\*".
- A green "Update User" button.

## After Updation

The screenshot shows a web browser window titled "Admin Page" with the URL "127.0.0.1:5000/admin". The main content is titled "IPL ANALYTICS DASHBOARD ADMIN PAGE" in red. It features a "Users" section with a table showing two entries:

ID	Username	Password
1	admin	admin
19	kaushik_2023	kaushik1234

Below the table are "Create User" and "Update User" forms. The "Create User" form has fields for "Username" and "Password", and a green "Insert User" button. The "Update User" form has fields for "User ID to Update" and "New Username".

The new user must now login with the updated credentials.

Username:- kaushik\_2023

Password:- kaushik1234

## Delete User

### Before Deletion

The screenshot shows a web browser window titled "Admin Page" with the URL "127.0.0.1:5000/admin". The main content is titled "IPL ANALYTICS DASHBOARD ADMIN PAGE" in red. Below it, a section titled "Users" displays a table with two rows:

ID	Username	Password
1	admin	admin
19	kaushik_2023	kaushik1234

Below the table, there is a "Create User" form with fields for "Username" and "Password", and a green "Insert User" button.

At the bottom, there is an "Update User" section with fields for "User ID to Update" and "New Username".

Admin Page

127.0.0.1:5000/admin

Username:

Password:

Insert User

**Update User**

User ID to Update:

New Username:

New Password:

Update User

**Delete User**

User ID to Delete:

19

Delete User

This screenshot shows the 'Admin Page' interface at '127.0.0.1:5000/admin'. It features three main sections: 'Insert User' (with fields for Username and Password and a green 'Insert User' button), 'Update User' (with fields for User ID to Update, New Username, and New Password, and a green 'Update User' button), and 'Delete User' (with a field for User ID to Delete containing '19' and a green 'Delete User' button). The background shows a blurred list of bookmarks.

## After Deletion

Admin Page

127.0.0.1:5000/admin

# IPL ANALYTICS DASHBOARD ADMIN PAGE

## Users

ID	Username	Password
1	admin	admin

**Create User**

Username:

Password:

Insert User

**Update User**

User ID to Update:

New Username:

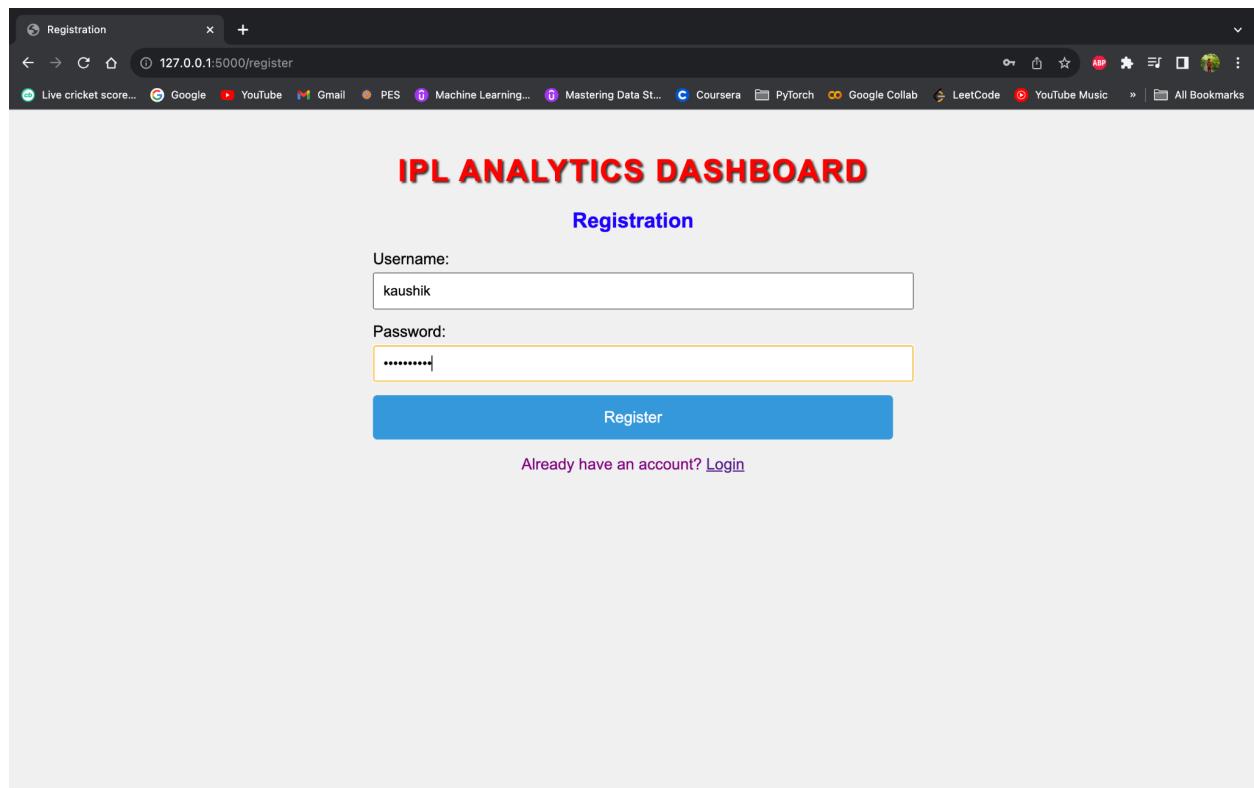
New Password:

This screenshot shows the 'IPL ANALYTICS DASHBOARD ADMIN PAGE' at '127.0.0.1:5000/admin'. The title is displayed prominently in red. Below it, a table titled 'Users' lists one entry: ID 1, Username 'admin', and Password 'admin'. Underneath the table are the 'Create User' and 'Update User' forms, which are identical to those in the previous screenshot but are currently inactive. The background shows a blurred list of bookmarks.

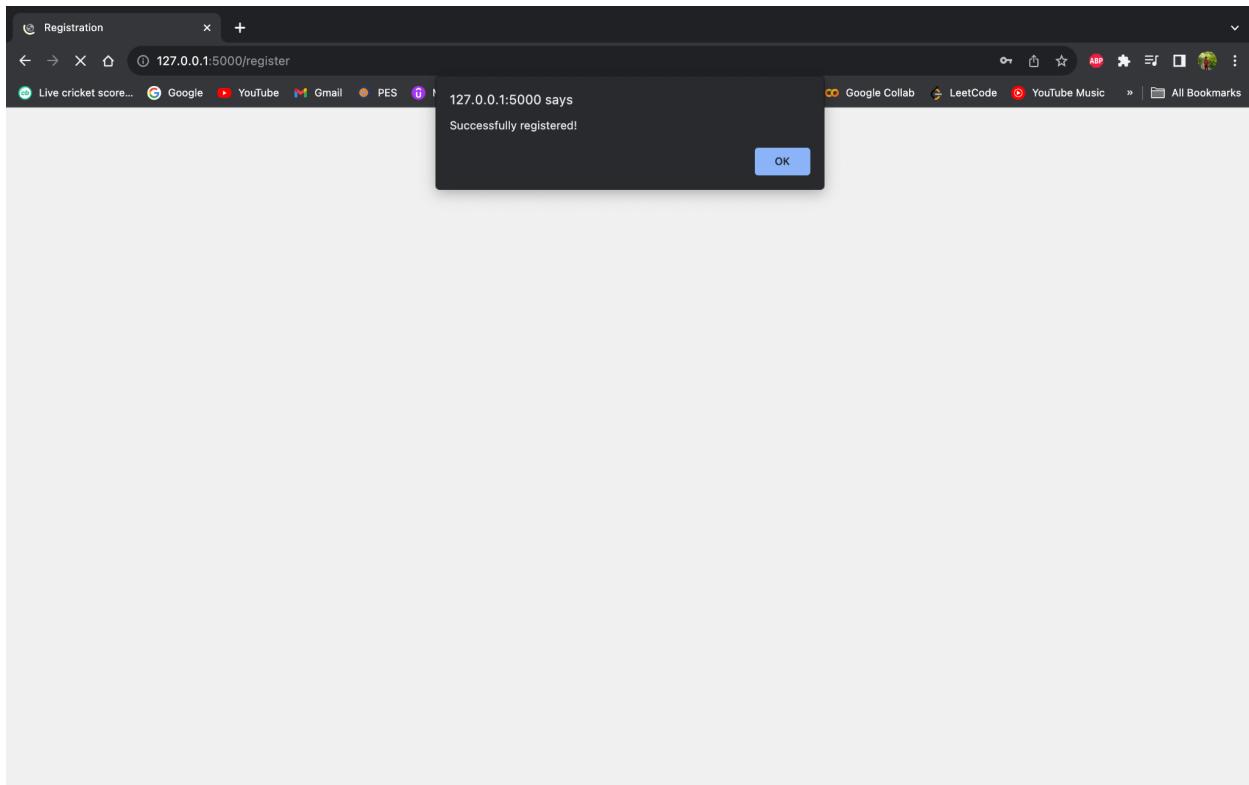
## List of Functionalities and its associated Query screenshots from front end

### 1. User Authentication

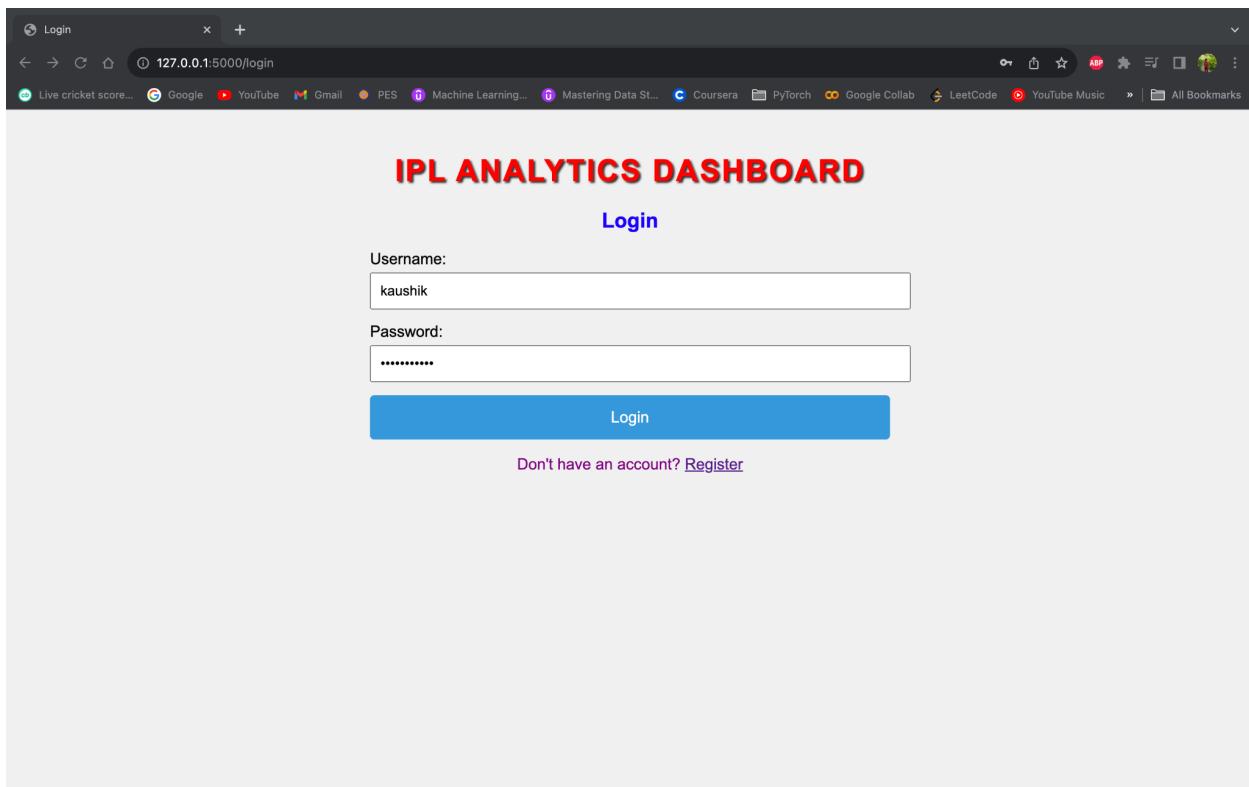
#### User Registration

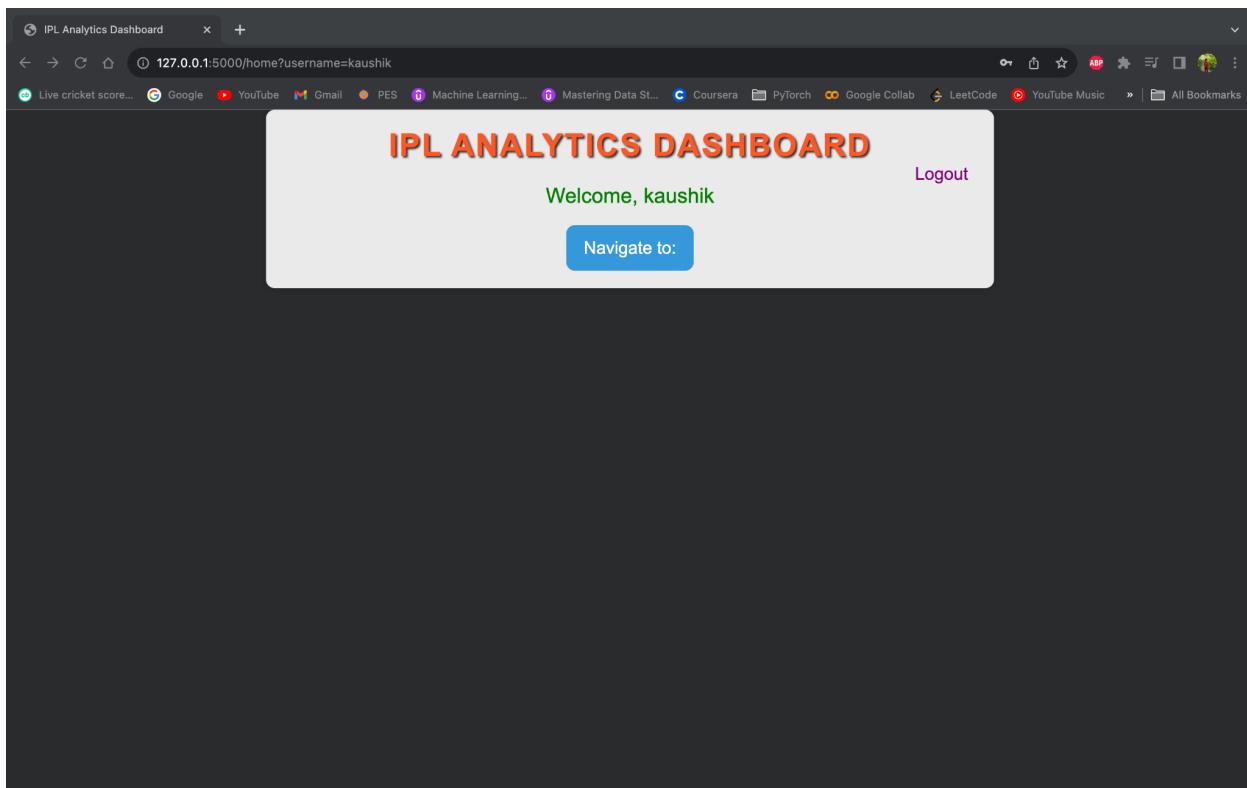
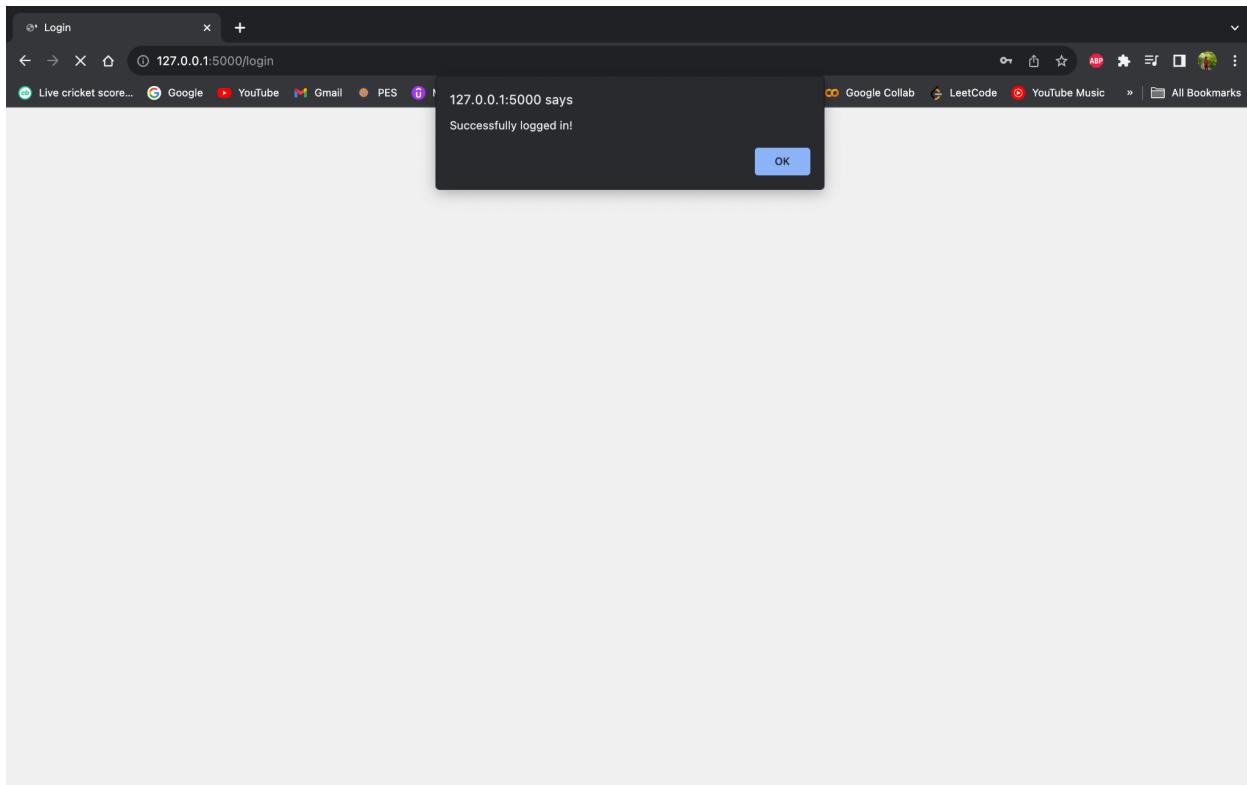


The screenshot shows a web browser window titled "Registration" with the URL "127.0.0.1:5000/register". The page has a header "IPL ANALYTICS DASHBOARD" and a sub-header "Registration". It contains two input fields: "Username:" with the value "kaushik" and "Password:" with the value "\*\*\*\*\*". Below the password field is a blue "Register" button. At the bottom, there is a link "Already have an account? [Login](#)". The browser's address bar and various bookmarks are visible at the top.



## User Login

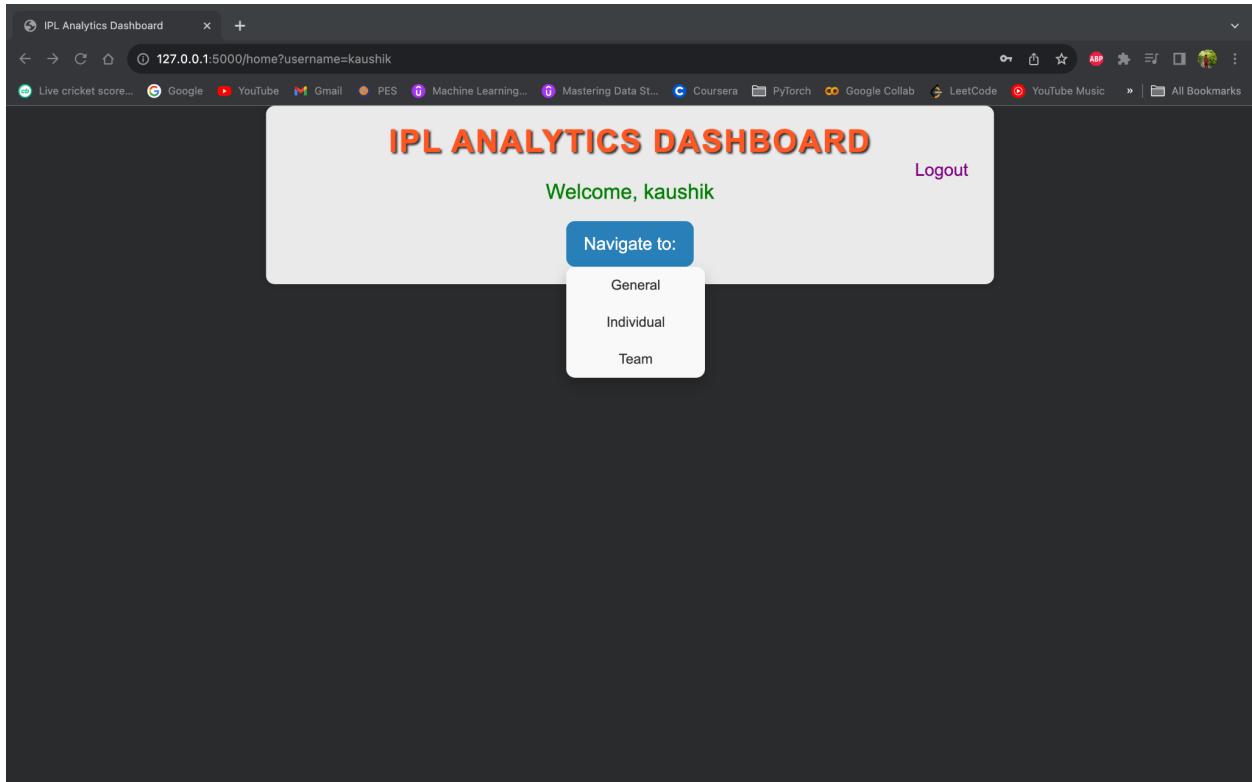




## 2. Admin Functionality

- Explained above under CRUD operations

## 3. Dashboard Visualizations

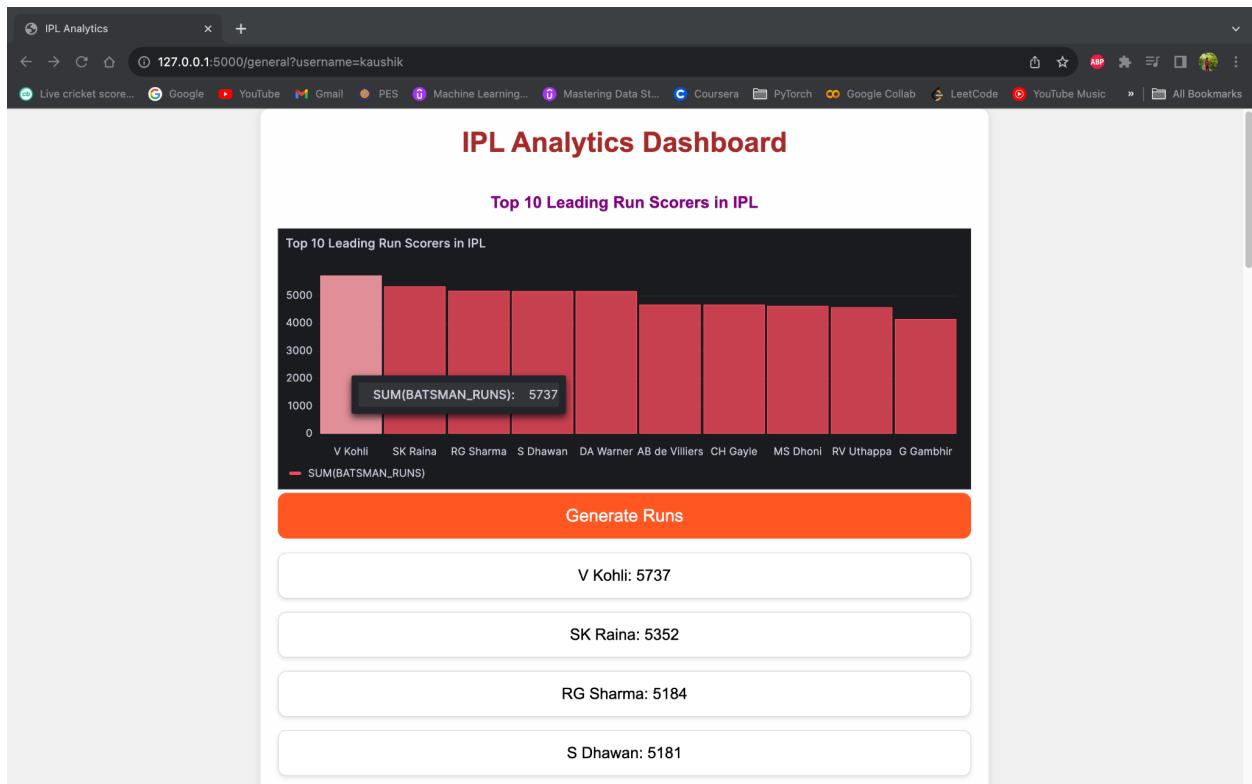


General, Individual and Team Statistics visualizations are available.

## General Statistics

### 1) Top 10 Leading Run Scorers in IPL

On hovering over the graph of each player, you get the statistic value associated to the particular player.

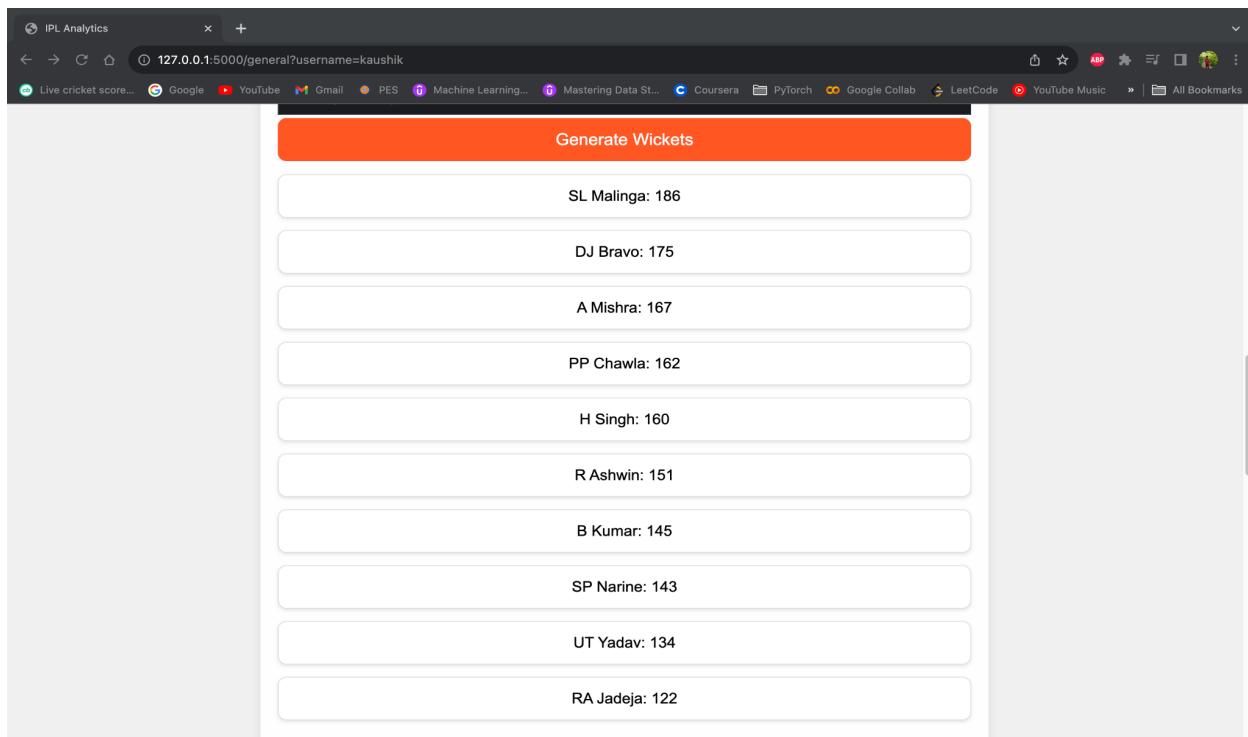
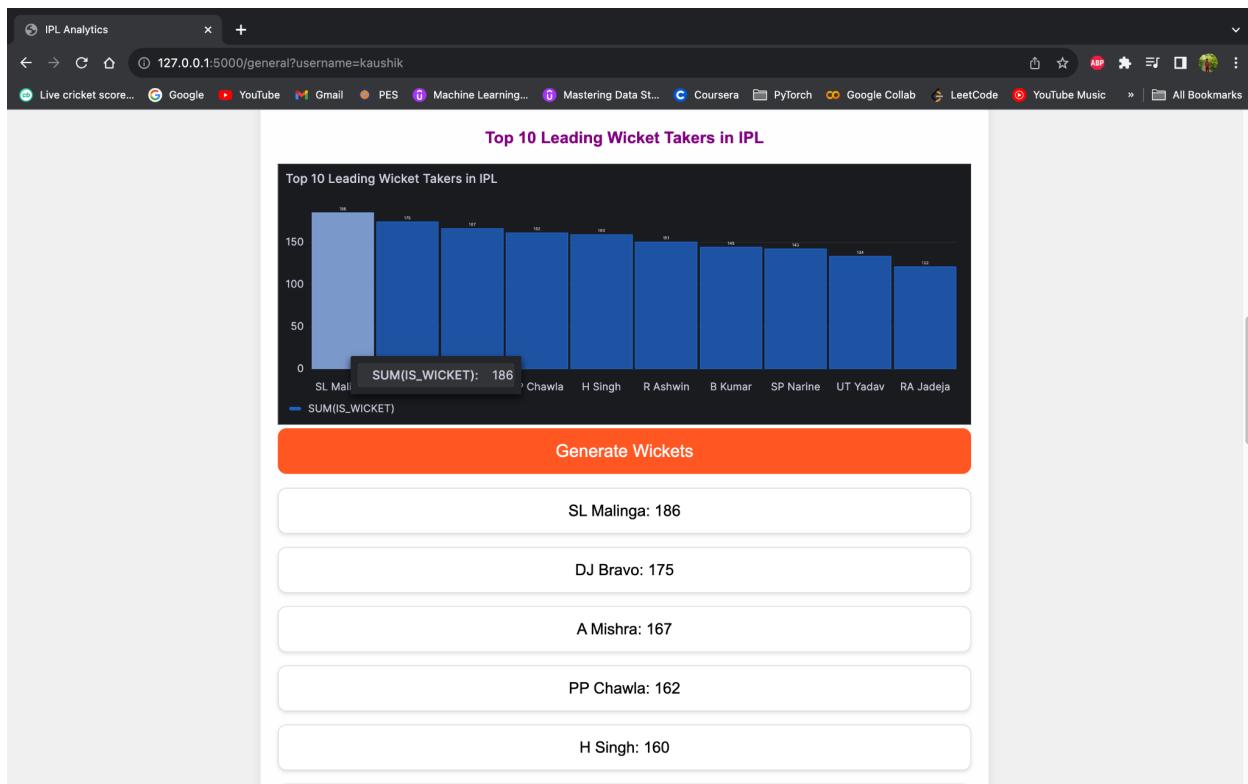


The button on clicking will display the same result in a key-value pair format.

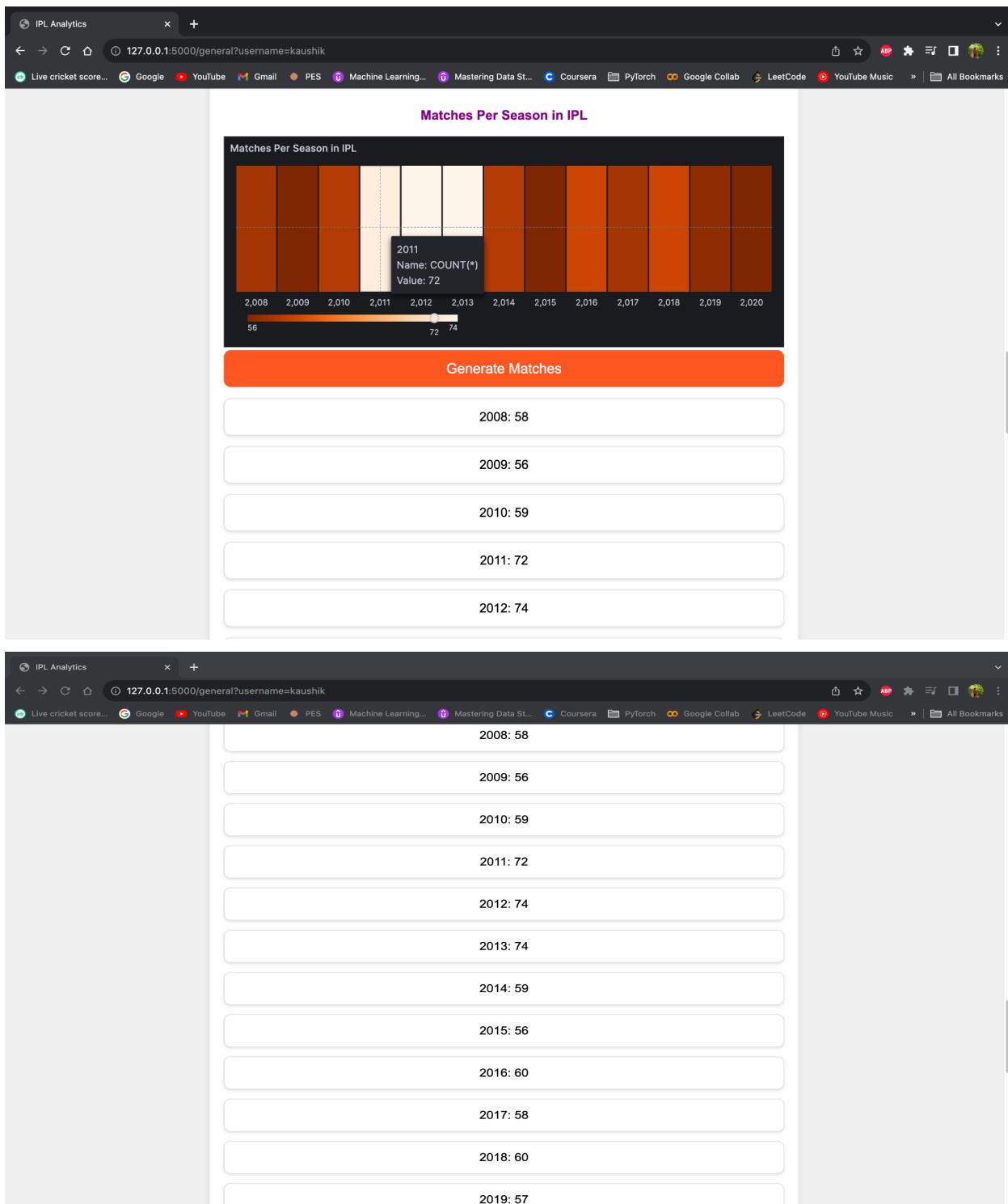
A screenshot of a web browser window titled "IPL Analytics". The URL in the address bar is "127.0.0.1:5000/general?username=kaushik". Below the address bar, there is a toolbar with various icons for live cricket scores, Google, YouTube, Gmail, PES, Machine Learning, Mastering Data Science, Coursera, PyTorch, Google Collab, LeetCode, YouTube Music, and All Bookmarks. A red bar at the top of the main content area contains the text "SUM(BATSMAN\_RUNS)". Below this bar is a large orange button labeled "Generate Runs". To the left of the button, there is a small red icon. The main content area displays a list of ten cricket players with their total runs: V Kohli: 5737, SK Raina: 5352, RG Sharma: 5184, S Dhawan: 5181, DA Warner: 5173, AB de Villiers: 4688, CH Gayle: 4682, MS Dhoni: 4632, RV Uthappa: 4596, and G Gambhir: 4172. Each player's name and run total are displayed in a separate white box with a thin gray border.

Player	Total Runs
V Kohli	5737
SK Raina	5352
RG Sharma	5184
S Dhawan	5181
DA Warner	5173
AB de Villiers	4688
CH Gayle	4682
MS Dhoni	4632
RV Uthappa	4596
G Gambhir	4172

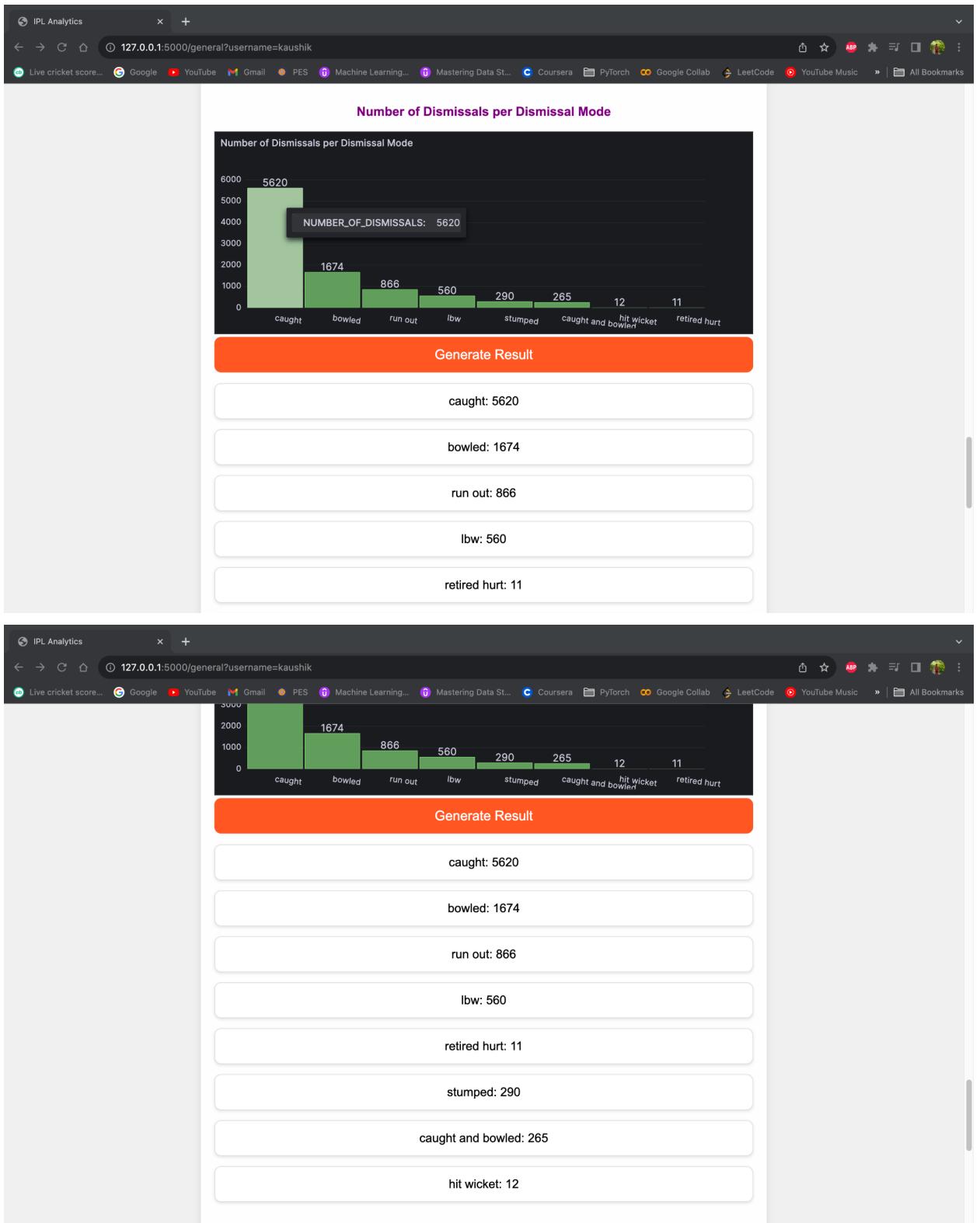
## 2) Top 10 Wicket Takers in IPL



### 3) Matches Per Season in IPL

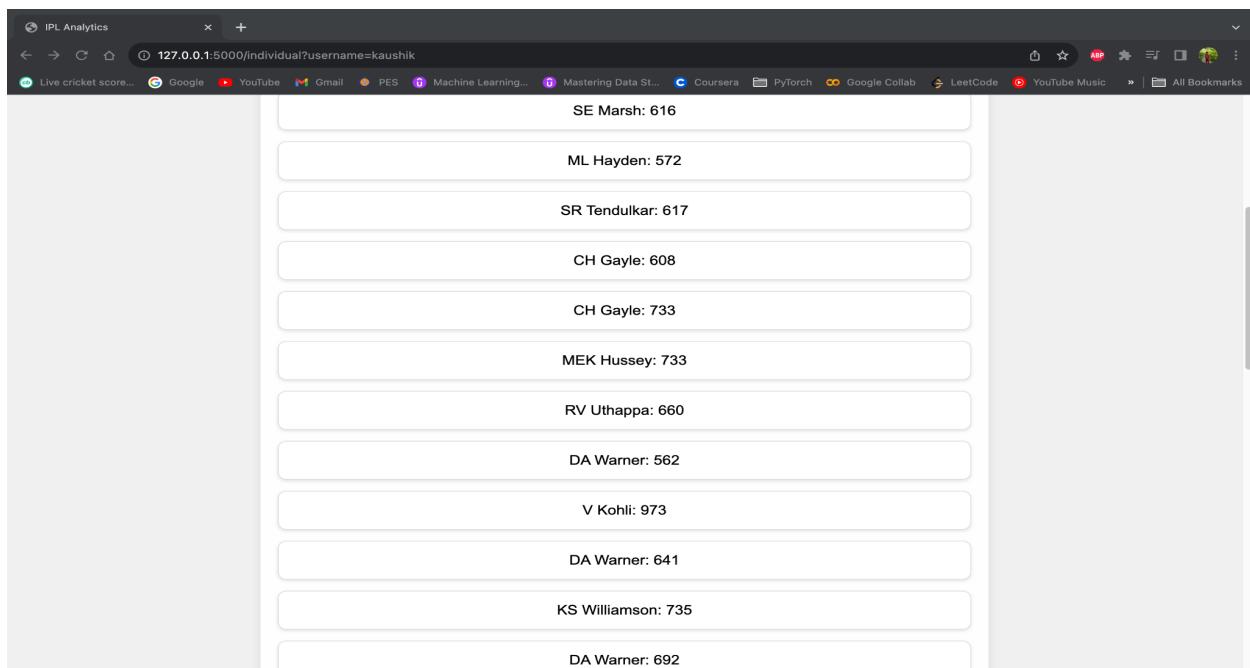
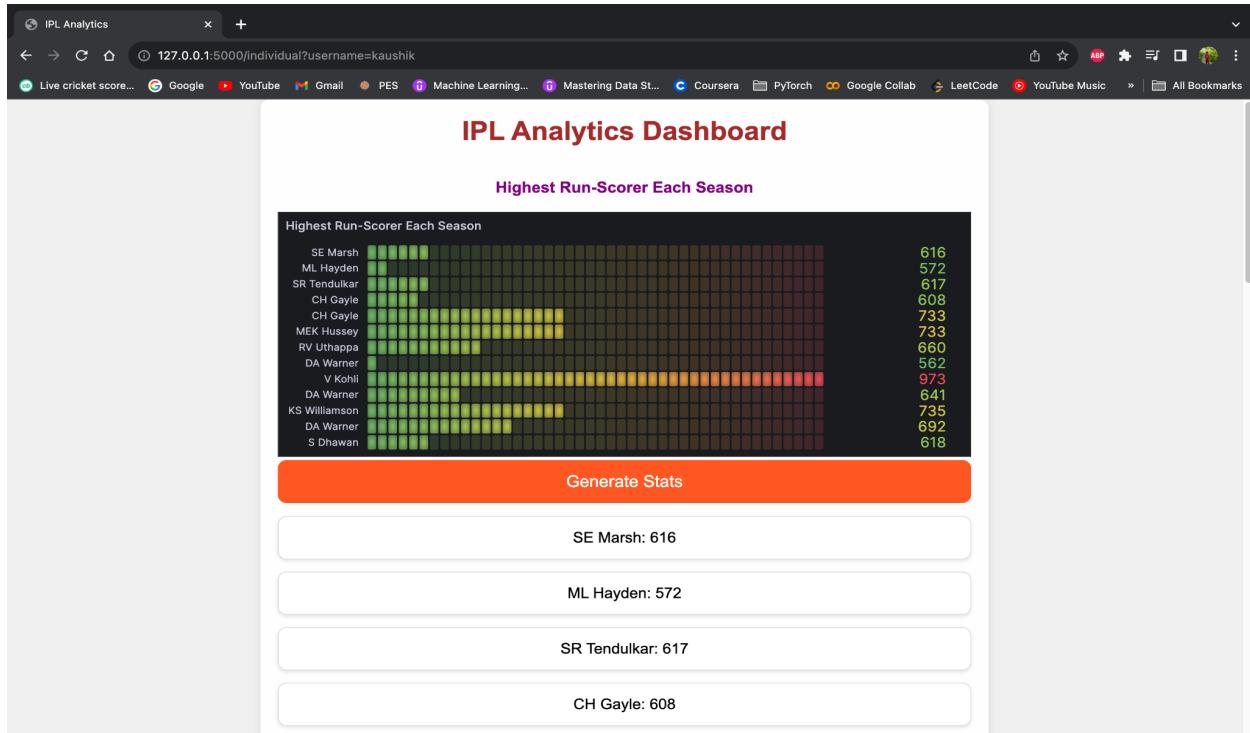


#### 4) Number of Dismissals per Dismissal Mode

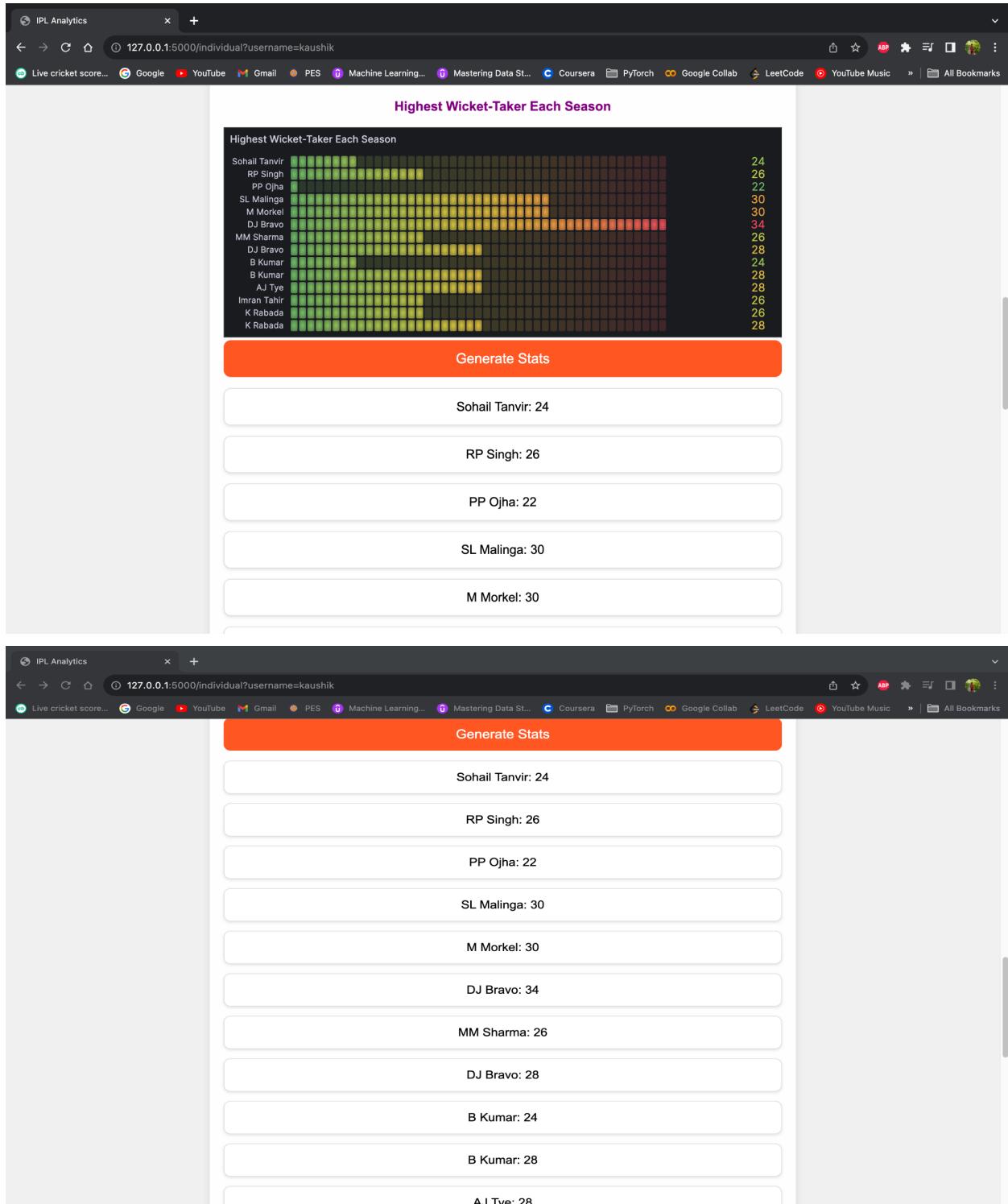


## Individual Statistics

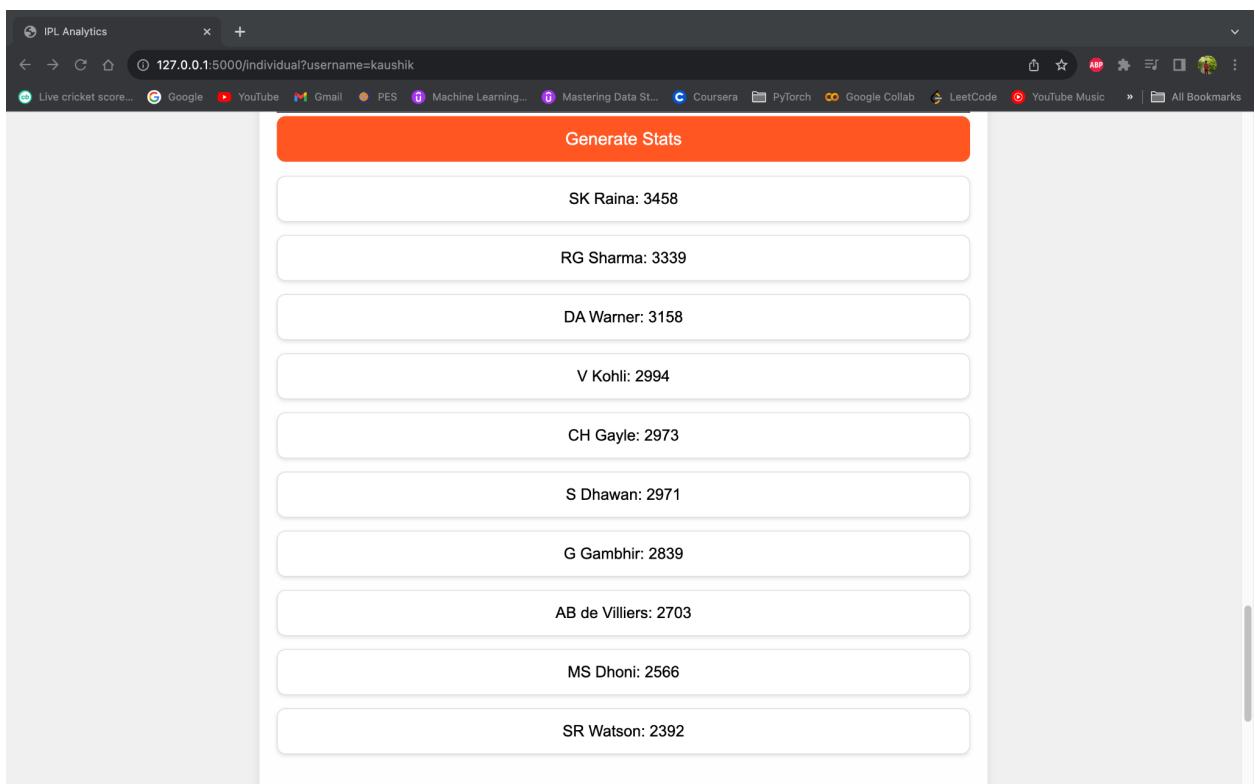
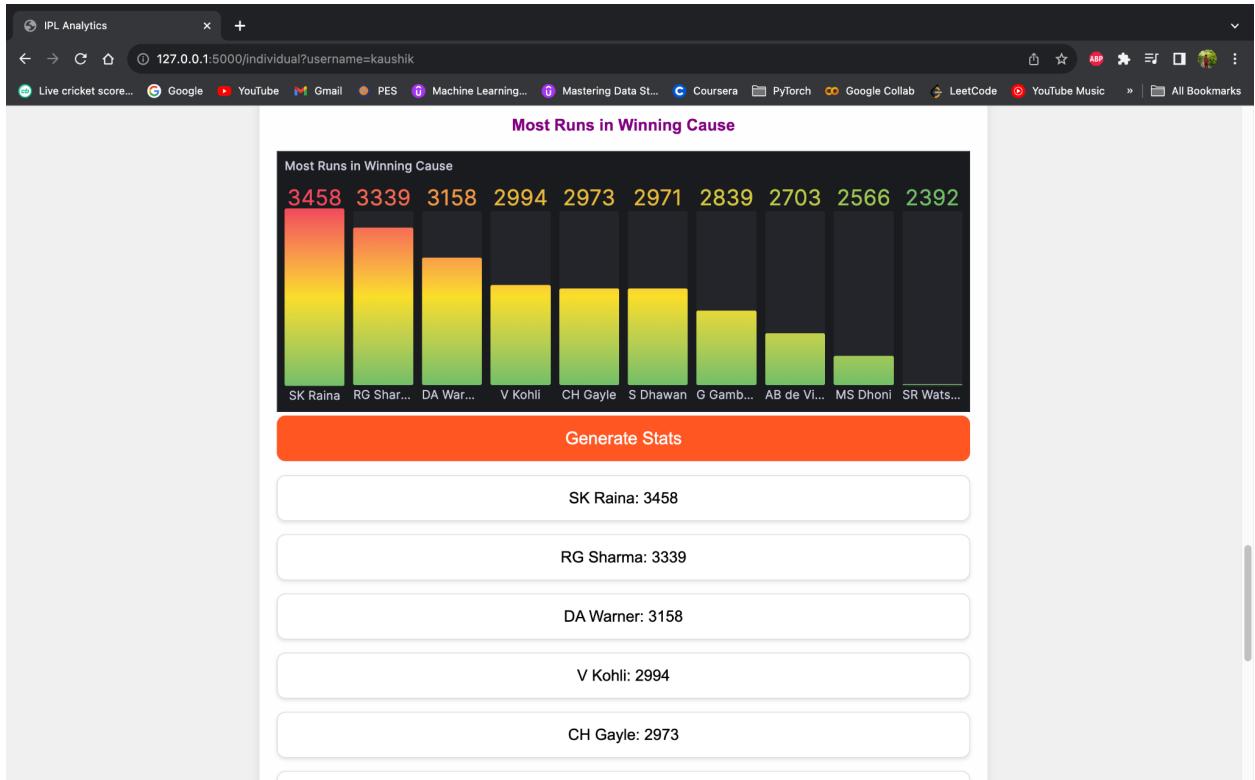
### 1) Highest Run Scorer Each Season



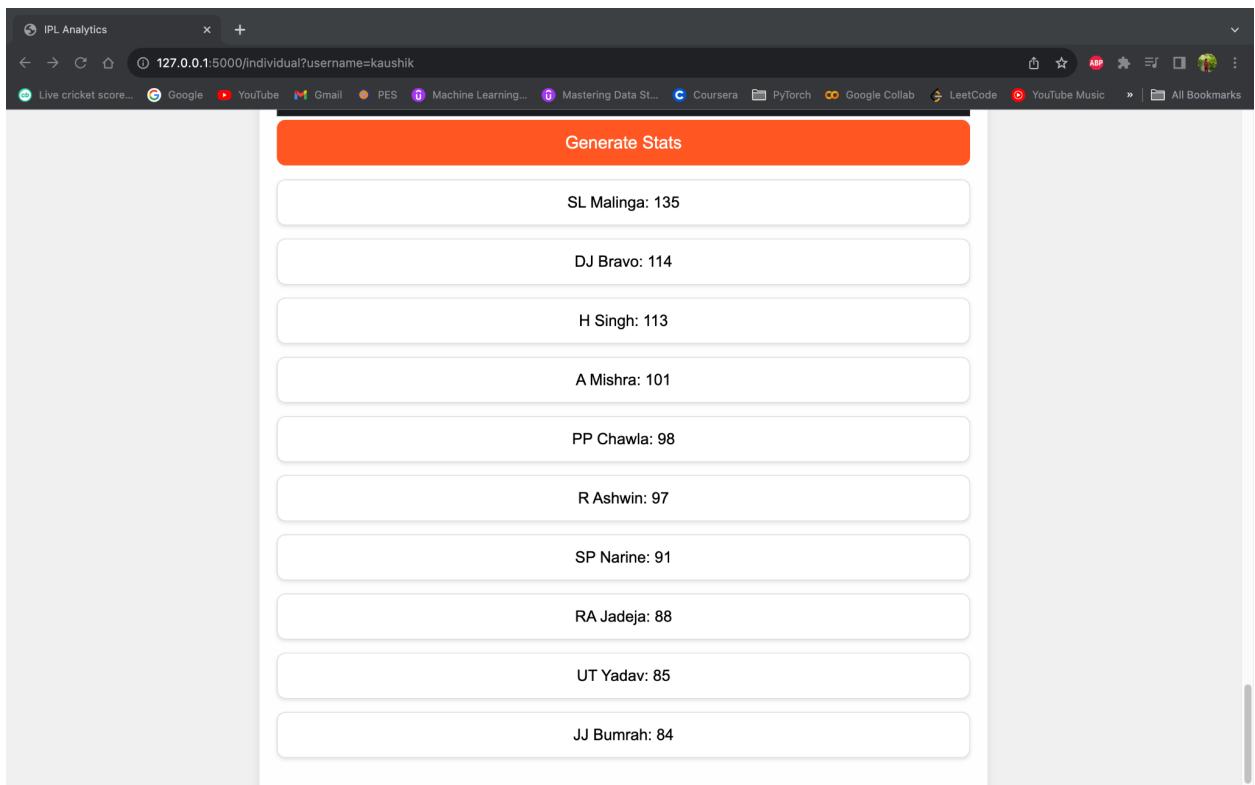
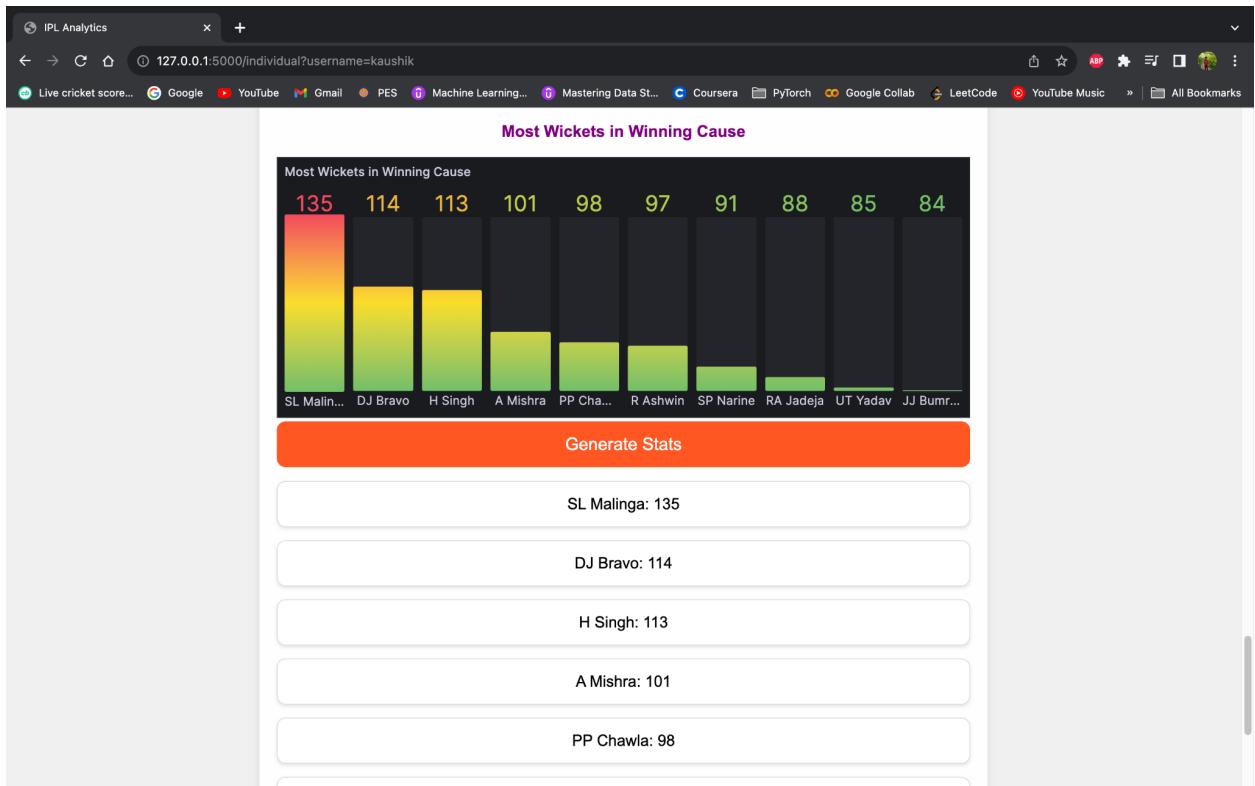
## 2) Highest Wicket-Taker Each Season



### 3) Most Runs in Winning Cause

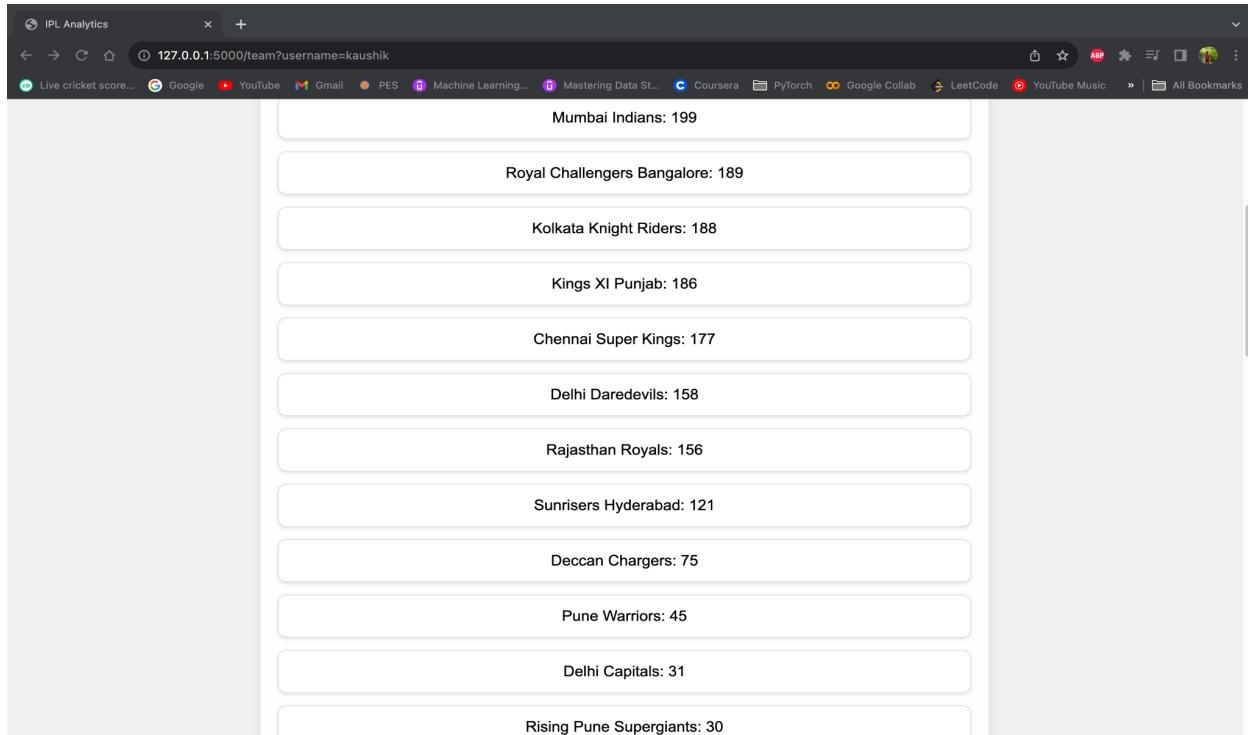
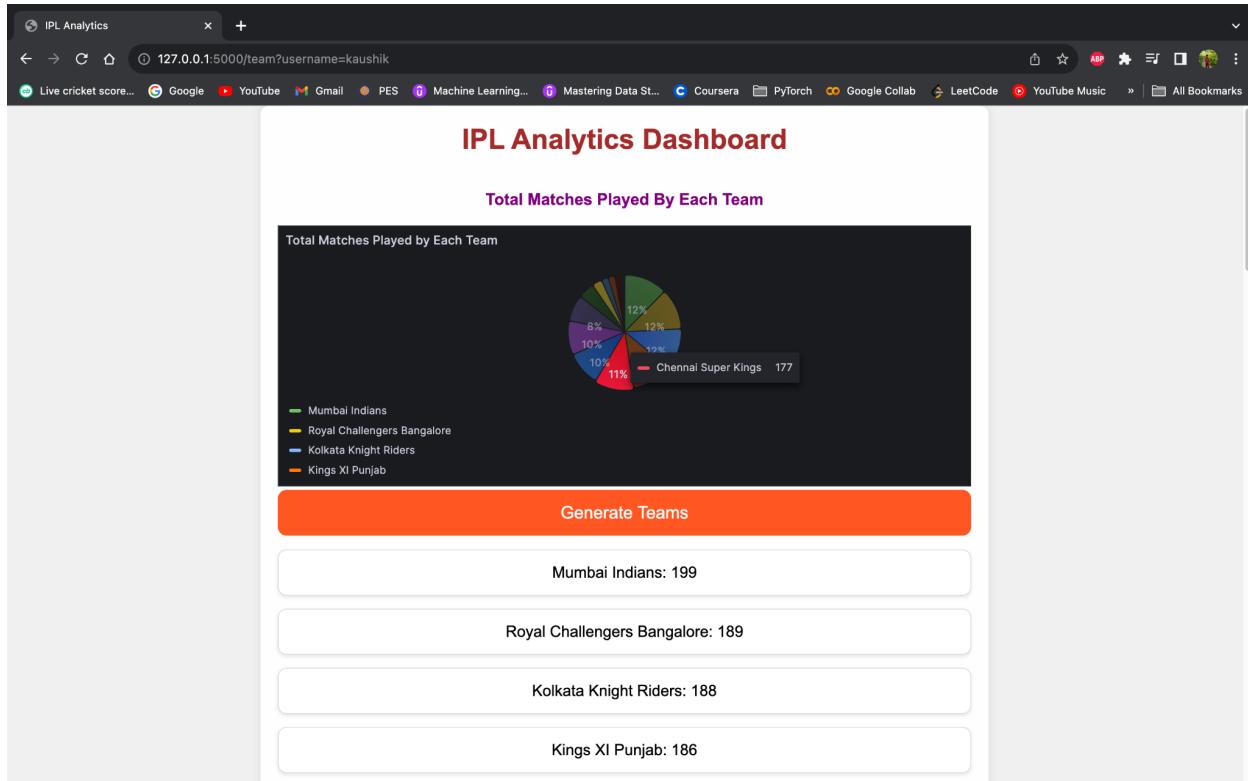


#### 4) Most Wickets in Winning Cause

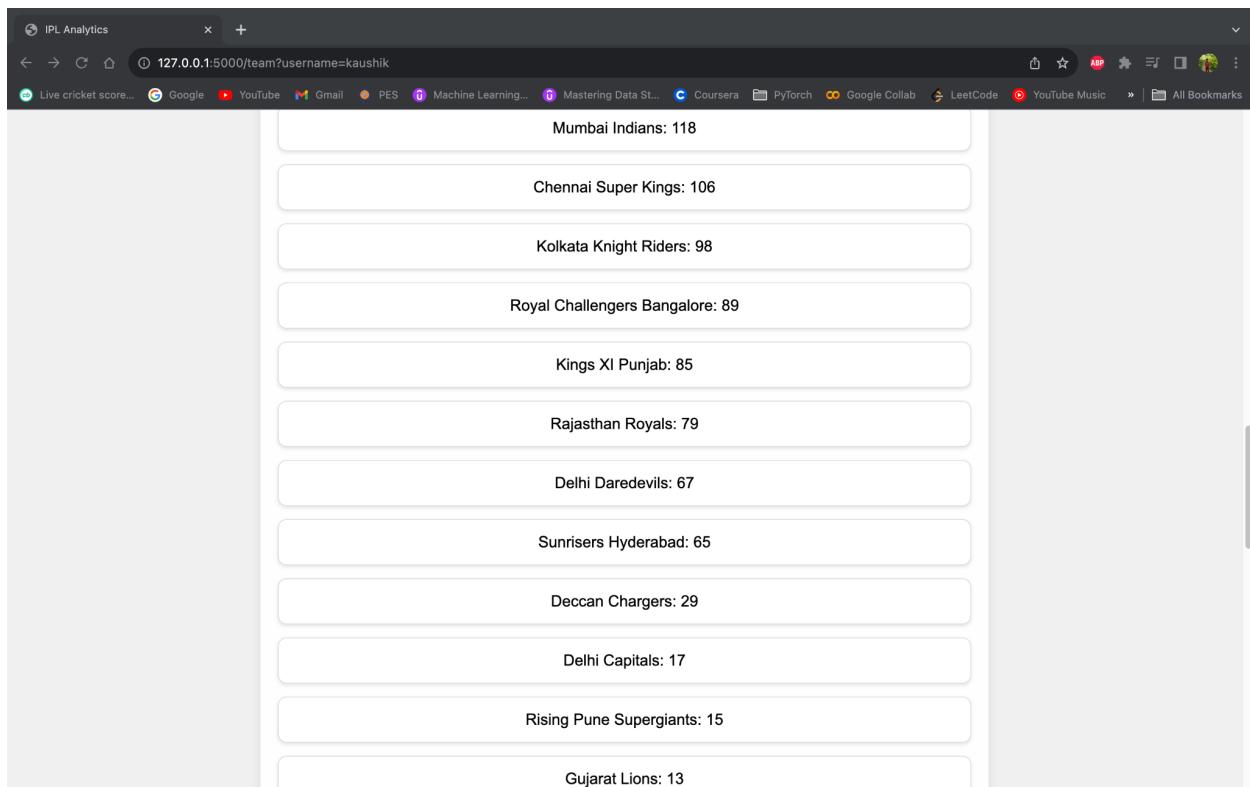
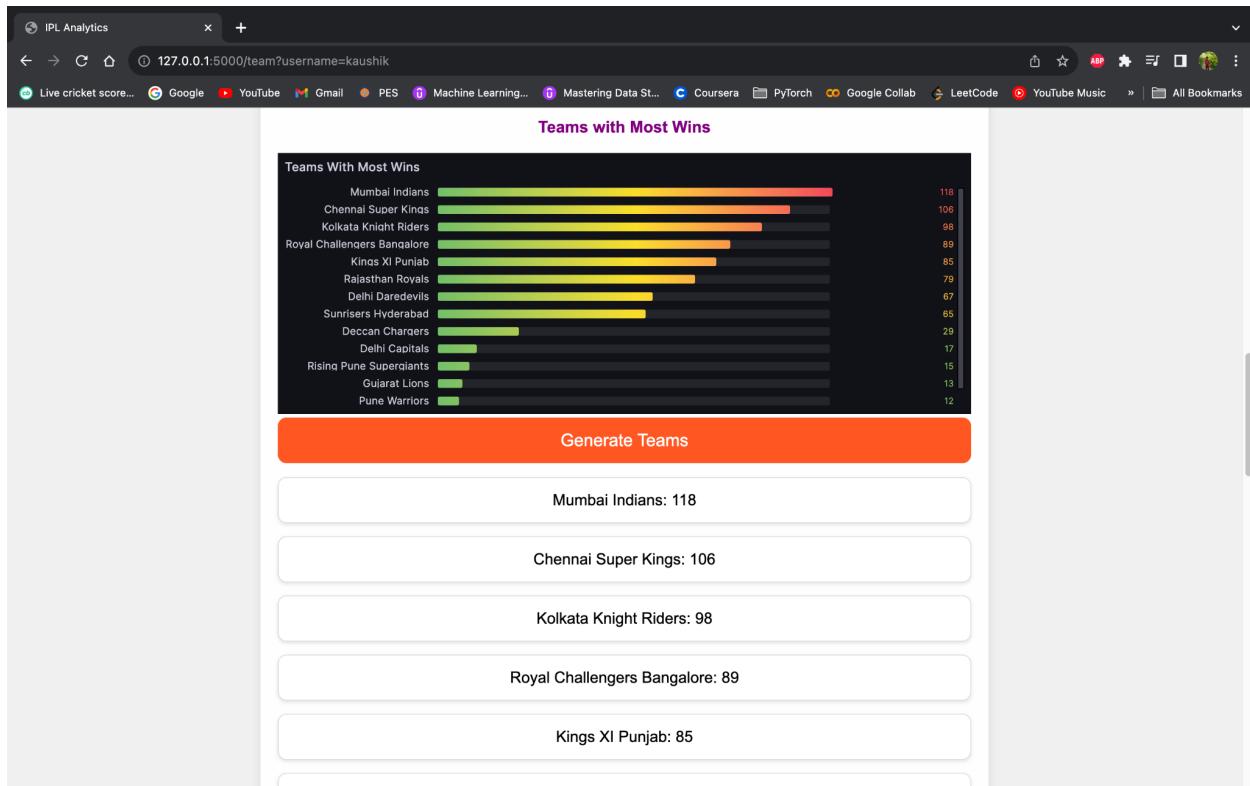


## Team Statistics

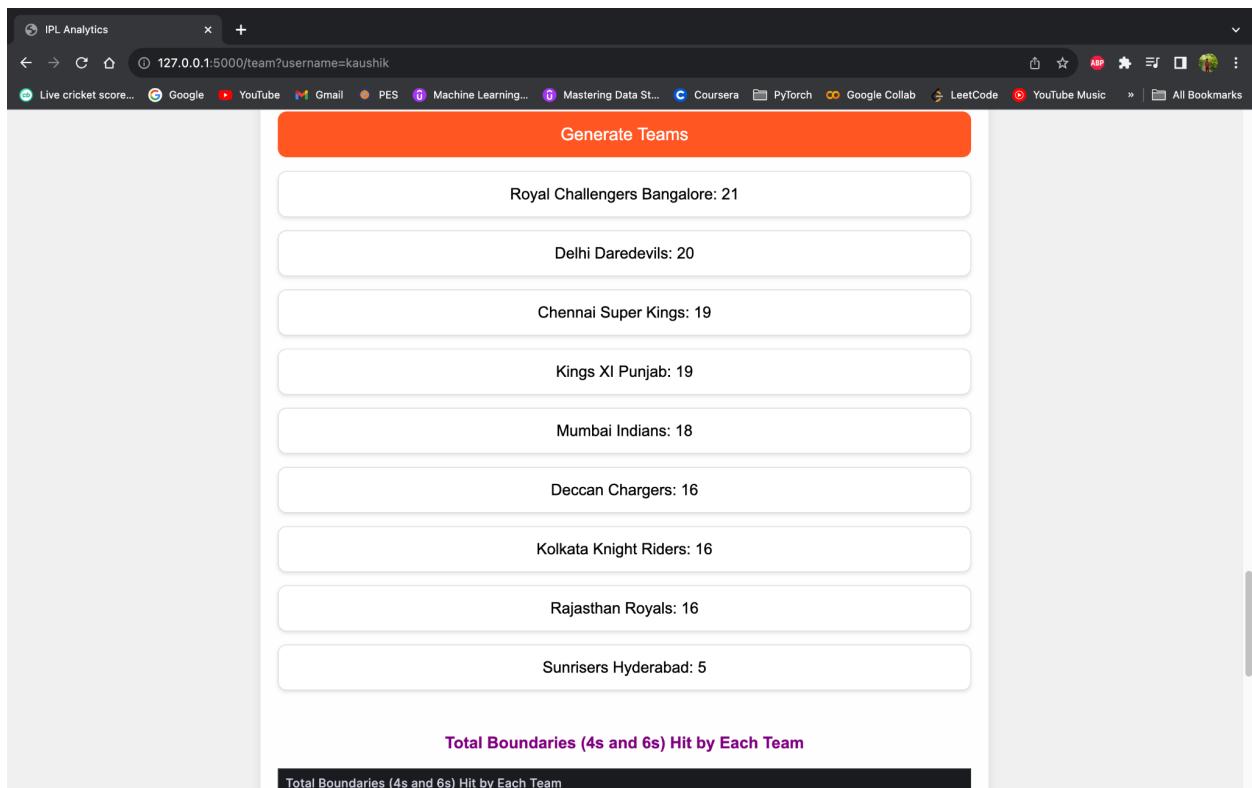
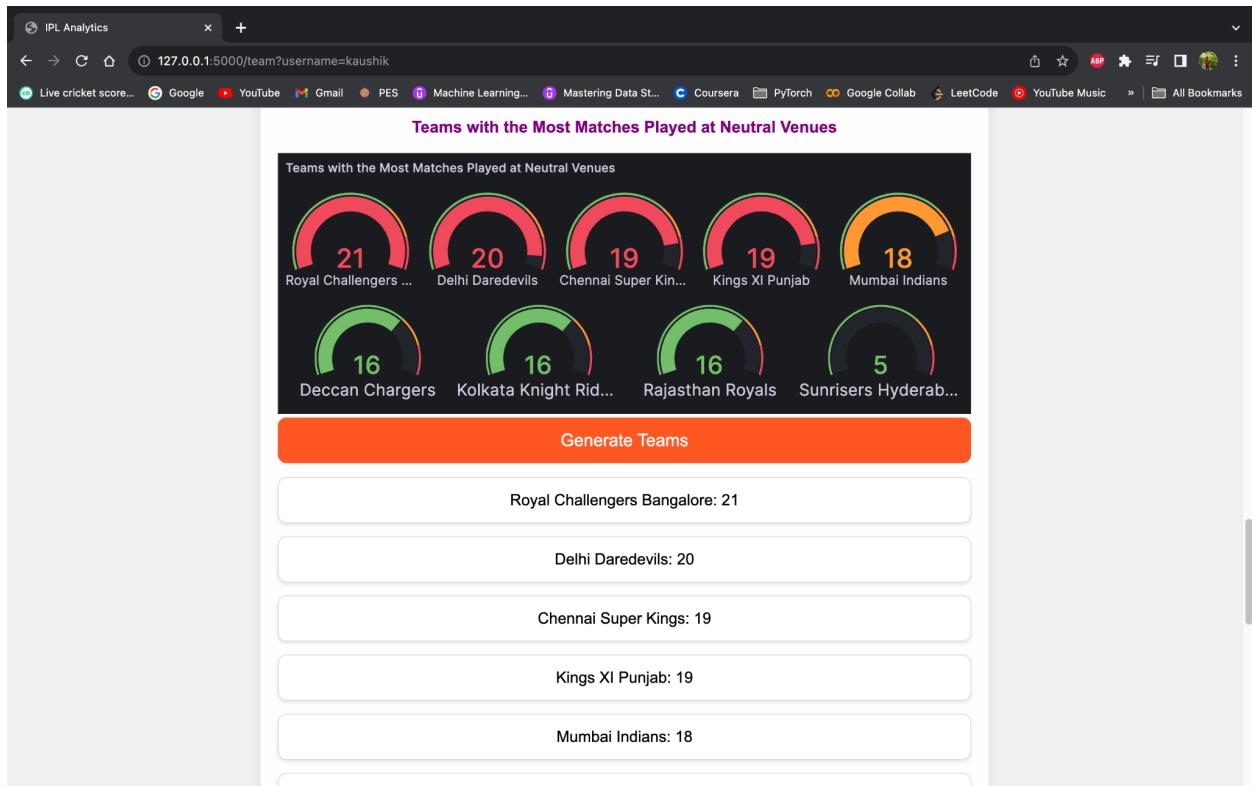
### 1) Total Matches Played By Each Team



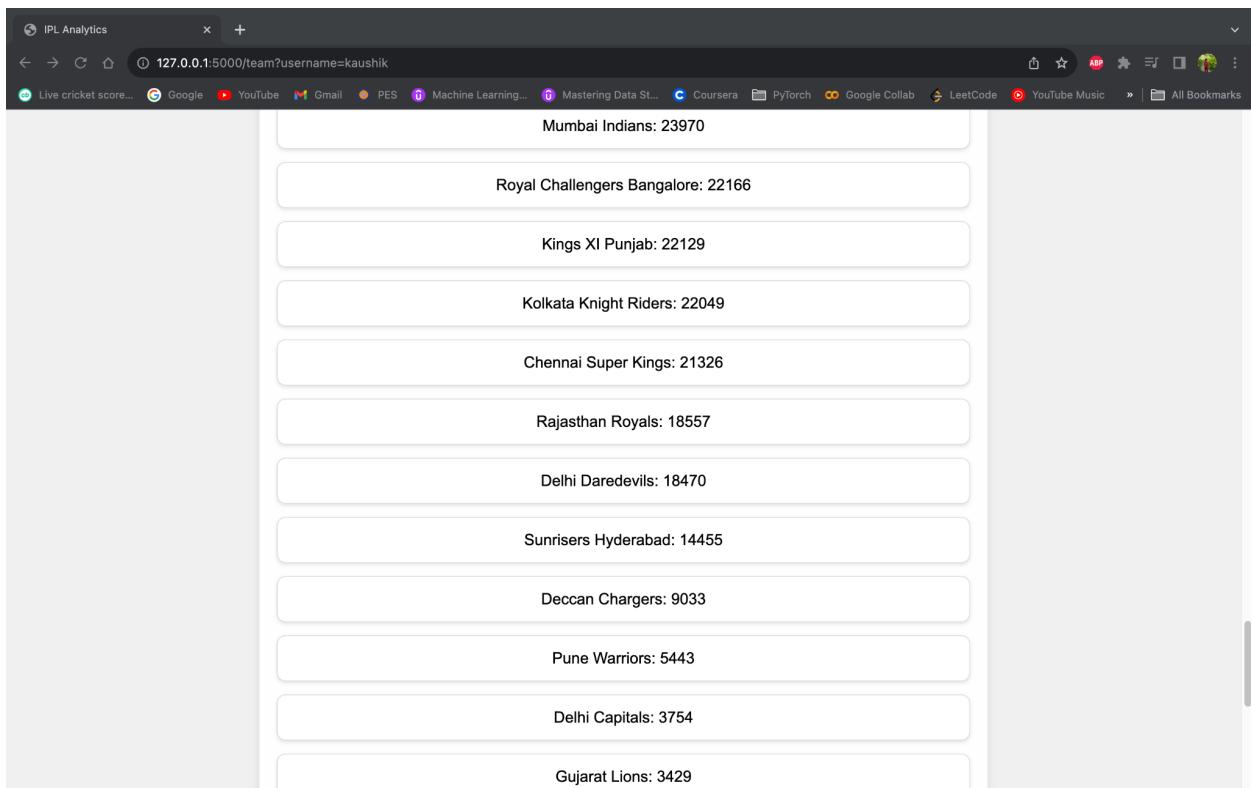
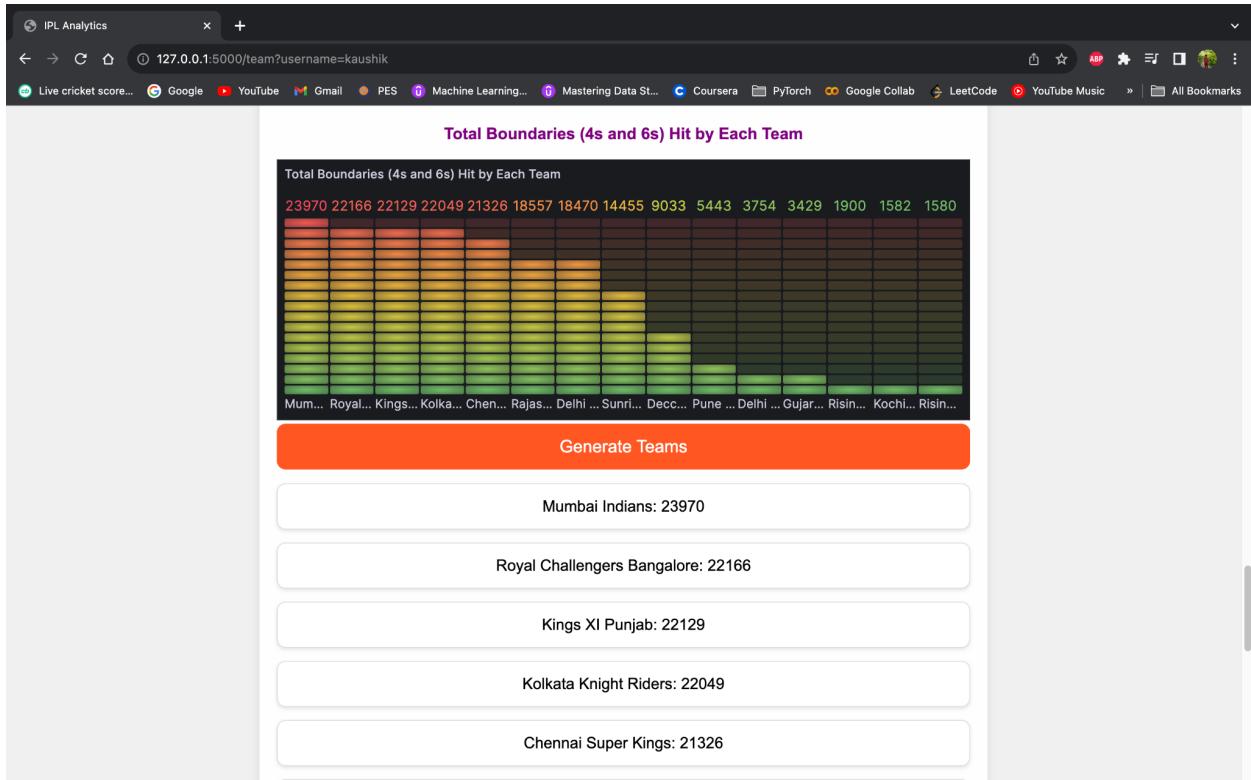
## 2) Teams with Most Wins



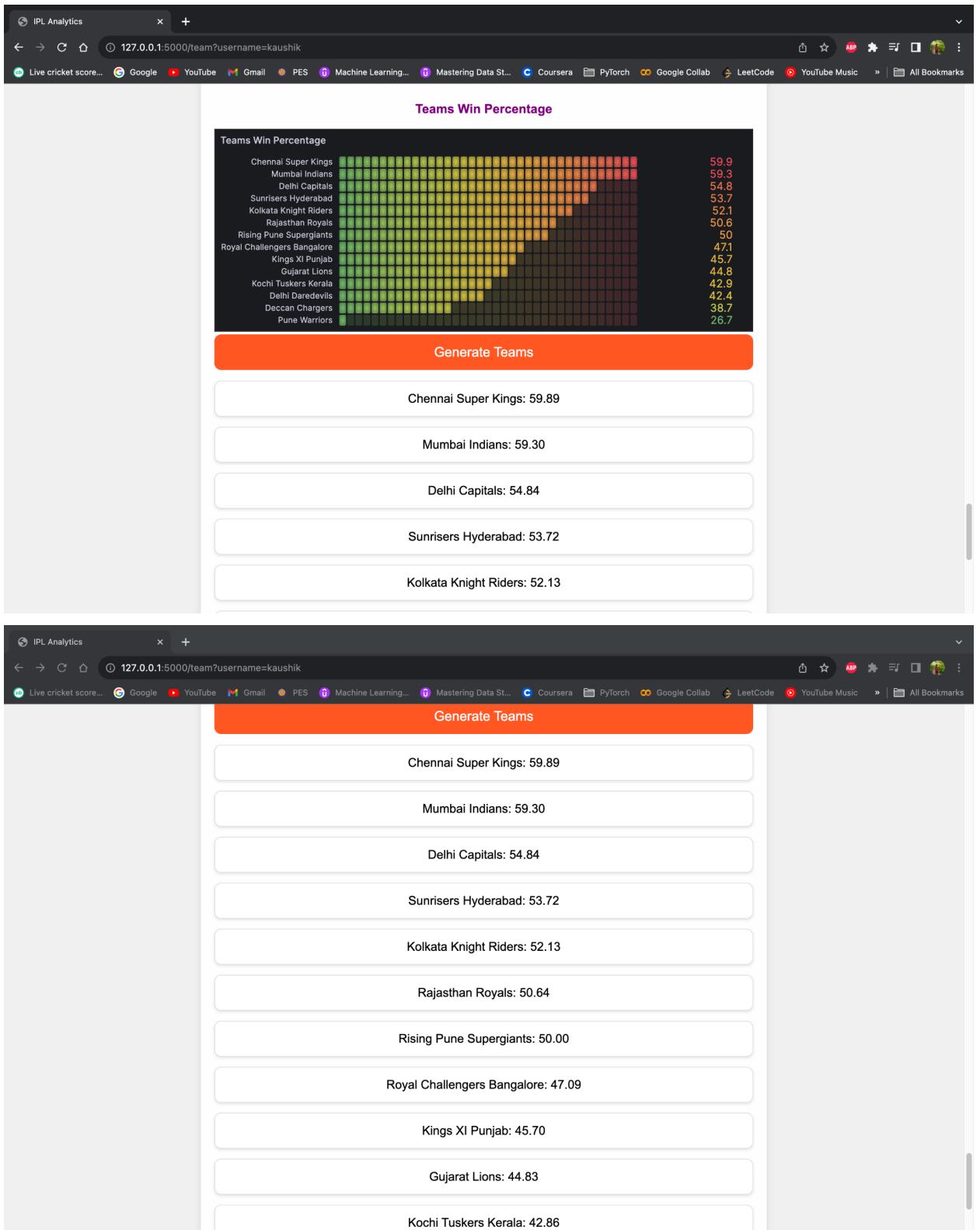
### 3) Teams with the Most Matches Played at Neutral Venues



#### 4) Total Boundaries (4's and 6's) Hit By Each Team



## 5) Teams Win Percentage



## SQL Queries Screenshots:-

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows the project structure under "PES CSE (WORKSPACE)". Key files include `app.py`, `general.html`, `individual.html`, and `team.html`.
- Code Editor:** The main pane displays Python code for an IPL Analytics Dashboard. The code includes SQL queries for selecting data from tables like `BALL_BY_BALL`, `Bowler`, and `Dismissal_Kind`. It also handles data transformation and rendering.
- Terminal:** The bottom right corner shows a terminal window titled "python3.9 - IPL Analytics Dashboard" with some log output.
- Bottom Status Bar:** Shows the current file is `app.py`, line 347, column 34, with 4 spaces, encoding UTF-8, and Python 3.9.16 ('base' conda).

The screenshot shows a Jupyter Notebook workspace titled "PES CSE (Workspace)". The left sidebar lists files and folders, including "app.py", "general.html", "individual.html", and "team.html". The main area displays Python code for an "IPL Analytics Dashboard". The code includes two queries: one for the highest run-scorer each season and another for the highest wicket-taker each season. The code uses SQLite and Python's built-in modules like `sqlite3` and `datetime`. The right side of the screen shows a vertical stack of other open notebooks or code snippets.

```
    # Query 1: Highest Run-Scorer Each Season
    cursor.execute("""SELECT PlayerName, RunsScored
    FROM (
        SELECT YEAR(M.DATE) AS Year, BB.BATSMAN AS PlayerName, SUM(BB.BATSMAN_RUNS) AS RunsScored,
        RANK() OVER(PARTITION BY YEAR(M.DATE) ORDER BY SUM(BB.BATSMAN_RUNS) DESC) AS RunRank
        FROM BALL_BY_BALL BB
        JOIN MATCHES M ON BB.MATCH_ID = M.MATCH_ID
        WHERE YEAR(M.DATE) BETWEEN 2008 AND 2020
        GROUP BY Year, PlayerName
    ) AS SeasonLeaders
    ORDER BY RunRank = 1
    ORDER BY Year""")
    runs_results = cursor.fetchall()

# Query 2: Highest Wicket-Taker Each Season
cursor.execute("""SELECT PlayerName, WicketsTaken
FROM (
    SELECT YEAR(M.DATE) AS Year, BB.BOWLER AS PlayerName, COUNT(*) AS WicketsTaken,
    RANK() OVER(PARTITION BY YEAR(M.DATE) ORDER BY COUNT(*) DESC) AS WicketRank
    FROM BALL_BY_BALL BB
    JOIN MATCHES M ON BB.MATCH_ID = M.MATCH_ID
    WHERE YEAR(M.DATE) BETWEEN 2008 AND 2020
    AND BB.IS_WICKET = 1
    GROUP BY Year, PlayerName
) AS SeasonLeaders
ORDER BY WicketRank = 1
ORDER BY Year""")
```

The screenshot shows a Jupyter Notebook interface with the following details:

- EXPLORER** sidebar:
  - OPEN EDITORS**: app.py, general.html, individual.html, team.html.
  - PES CSE (WORKSPACE)**: \_\_pycache\_\_, static/css, admin.css, cricket-stadiu..., cricket-stadiu..., default.css, home.css, login.css, register.css, styles.css, templates, admin.html, default.html, general..., home.html, individual..., login.html, logout.html, register.html, team.html.
  - app.py**: The main Python script for the dashboard.
  - DotSlash**, Flask, Flutter, Java, Machine Learning, NLP-Audio Process..., OpenCV, OUTLINE, TIMELINE: Other projects or notebooks.
- CELLS** tab: The current cell is cell 284.
- CODE** tab: The code for the dashboard is displayed, including queries for Season Leaders, Highest Wicket-Takers, and Most Runs in Winning Cause.
- PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE**, **TERMINAL**, **PORTS**: Development tools.
- TERMINAL** tab: Shows a history of HTTP requests to the dashboard.
- STATUS**: python 3.9 - IPL Analytics Dashboard.

```
291 # Query 2: Highest Wicket-Taker Each Season
292 cursor.execute("'''SELECT PlayerName, WicketsTaken
293 FROM (
294     SELECT YEAR(M.DATE) AS Year, BB.BOWLER AS PlayerName, COUNT(*) AS WicketsTaken,
295         RANK() OVER(PARTITION BY YEAR(M.DATE) ORDER BY COUNT(*) DESC) AS WicketRank
296     FROM BALL_BY_BALL BB
297     JOIN MATCHES M ON BB.MATCH_ID = M.MATCH_ID
298     WHERE YEAR(M.DATE) BETWEEN 2008 AND 2020
299     AND BB.IS_WICKET = 1
300     GROUP BY Year, PlayerName
301 ) AS SeasonLeaders
302 WHERE WicketRank = 1
303 ORDER BY Year
304 ''')
305 wickets_results = cursor.fetchall()
306
307 # Query 3: Most Runs in Winning Cause
308 cursor.execute("'''SELECT BB.BATSMAN, SUM(BB.BATSMAN_RUNS) AS TotalRuns
309 FROM BALL_BY_BALL BB
310 JOIN MATCHES M ON BB.MATCH_ID = M.MATCH_ID
311 WHERE BB.BATSMAN_RUNS > 0
312 AND YEAR(M.DATE) BETWEEN 2008 AND 2020
313 AND M.WINNER = BB.BATTING_TEAM
314 GROUP BY BB.BATSMAN
315 ORDER BY TotalRuns DESC
316 LIMIT 10'''')
317 winning_runs_results = cursor.fetchall()
318
319 # Query 4: Most Wickets in Winning Cause
320 cursor.execute("'''SELECT BB.BOWLER, COUNT(*) AS WicketsInMatchesWon
321 FROM BALL_BY_BALL BB
322 JOIN MATCHES M ON BB.MATCH_ID = M.MATCH_ID
323 WHERE BB.IS_WICKET = 1
324 AND YEAR(M.DATE) BETWEEN 2008 AND 2020
325 AND M.WINNER = BB.BOWLING_TEAM
326 GROUP BY BB.BOWLER
327 ORDER BY WicketsInMatchesWon DESC
328 LIMIT 10'''')
```

File: app.py

```

CSE > CSE V Sem > SQL > IPL Analytics Dashboard > app.py > team
318
319     # Query 4: Most Wickets in Winning Cause
320     cursor.execute("""SELECT BB.BOWLER, COUNT(*) AS WicketsInMatchesWon
321     FROM BALL_BY_BALL_BB
322     JOIN MATCHES M ON BB.MATCH_ID = M.MATCH_ID
323     WHERE BB.IS_WICKET = 1
324     AND YEAR(M.DATE) BETWEEN 2008 AND 2020
325     AND M.WINNER = BB.BOWLING_TEAM
326     GROUP BY BB.BOWLER
327     ORDER BY WicketsInMatchesWon DESC
328     LIMIT 10
329     """)

330
331     dismissals_results = cursor.fetchall()

332
333     cursor.close()
334     conn.close()

335
336     runs_data = [f'{name}: {runs}' for name, runs in runs_results]
337     wickets_data = [f'{name}: {wickets}' for name, wickets in wickets_results]
338     matches_data = [f'{name}: {runs}' for name, runs in winning_runs_results]
339     dismissals_data = [f'{dismissal_mode}: {count}' for dismissal_mode, count in dismissals_results]

340
341
342     return check_authentication('individual.html', data1=runs_data, data2=wickets_data, data3=matches_data, data4=dismissals_data)

```

Terminal:

```

R: (not attached) Ln 347, Col 34 Spaces: 4 UTF-8 LF ↴ Python 3.9.16 ('base': conda) ⌂ Go Live ⌂

```

File: app.py

```

CSE > CSE V Sem > SQL > IPL Analytics Dashboard > app.py > team
351
352     cursor.execute("""SELECT TEAM, COUNT(*) AS TOTAL_MATCHES_PLAYED
353     FROM (
354         SELECT TEAM1 AS TEAM FROM MATCHES
355         UNION ALL
356         SELECT TEAM2 AS TEAM FROM MATCHES
357     ) AS Teams
358     GROUP BY TEAM
359     ORDER BY TOTAL_MATCHES_PLAYED DESC""")
360
361     count_matches_results = cursor.fetchall()

362
363     # Query 2: Teams with Most Wins
364     cursor.execute("""SELECT WINNER, COUNT(*) AS TOTAL_WINS FROM MATCHES
365     GROUP BY WINNER ORDER BY TOTAL_WINS DESC""")
366
367     wins_results = cursor.fetchall()

368
369     # Query 3: Teams with the Most Matches Played at Neutral Venues
370     cursor.execute("""SELECT TEAM, COUNT(*) AS MATCHES_AT_NEUTRAL_VENUES
371     FROM (
372         SELECT TEAM1 AS TEAM FROM MATCHES WHERE NEUTRAL_VENUE = 1
373         UNION ALL
374         SELECT TEAM2 AS TEAM FROM MATCHES WHERE NEUTRAL_VENUE = 1
375     ) AS NeutralVenueTeams
376     GROUP BY TEAM
377     ORDER BY MATCHES_AT_NEUTRAL_VENUES DESC""")
378
379     neutral_matches_results = cursor.fetchall()

```

Terminal:

```

R: (not attached) Ln 347, Col 34 Spaces: 4 UTF-8 LF ↴ Python 3.9.16 ('base': conda) ⌂ Go Live ⌂

```

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows a project structure under "PES CSE (Workspace)".
  - app.py
  - general.html M
  - individual.html M
  - team.html M
- Code Editor:** Displays Python code for an "IPL Analytics Dashboard".

```
    377     # Query 4: Total Boundaries (4s and 6s) Hit by Each Team
    378     cursor.execute("""SELECT BATTING_TEAM AS Team,
    379                     SUM(CASE WHEN NON_BOUNDARY = 0 THEN 1 ELSE 0 END) AS TOTAL_BOUNDARIES
    380                 FROM BALL_BY_BALL
    381                 GROUP BY Team
    382                 ORDER BY TOTAL_BOUNDARIES DESC""")
```

```
    383
    384     boundaries_results = cursor.fetchall()
```

```
    385
    386     #Query 5: Teams Win Percentage
    387
    388     cursor.callproc('CalculateTeamWinPercentage')
    389     result_cursor = conn.cursor()
    390     result_cursor.execute("SELECT * FROM result_of_calculate_team_win_percentage")
    391
    392     team_win_results = result_cursor.fetchall()
```

```
    393
    394     cursor.close()
    395     conn.close()
```

```
    396
    397     runs_data = [f'{team}: {matches}' for team, matches in count_matches_results]
    398     wickets_data = [f'{team}: {wins}' for team, wins in wins_results]
    399     matches_data = [f'{team}: {matches}' for team, matches in neutral_matches_results]
    400     dismissals_data = [f'{team}: {boundaries}' for team, boundaries in boundaries_results]
```

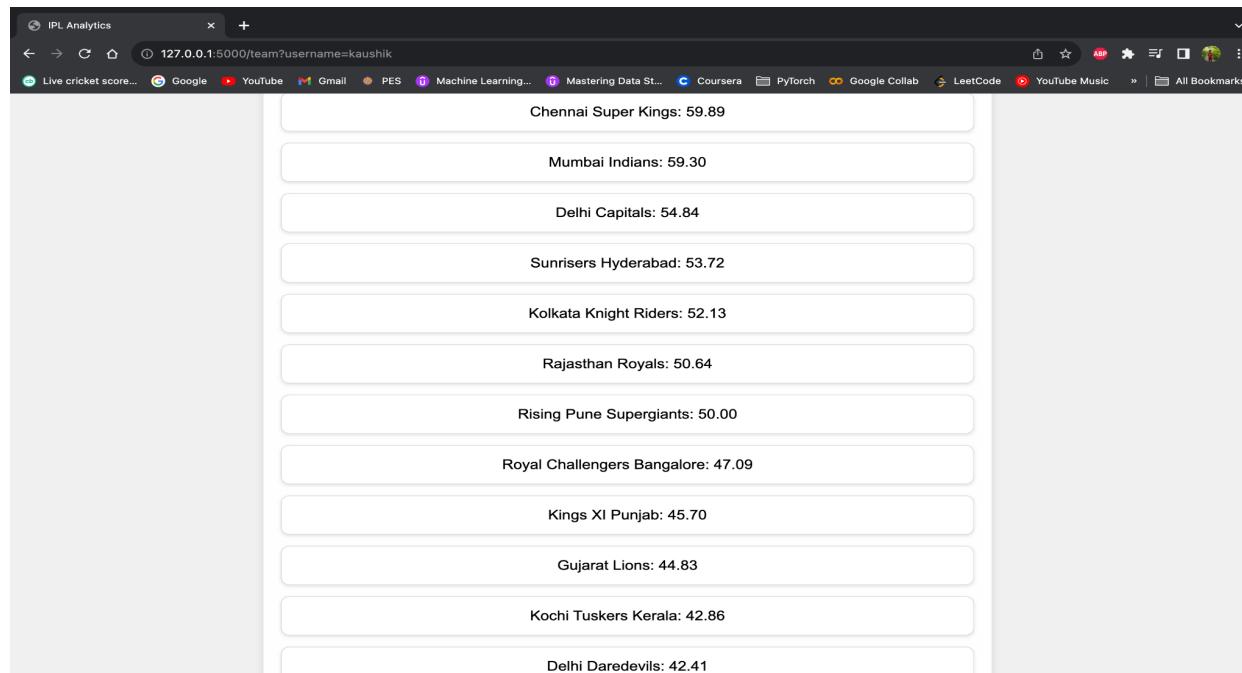
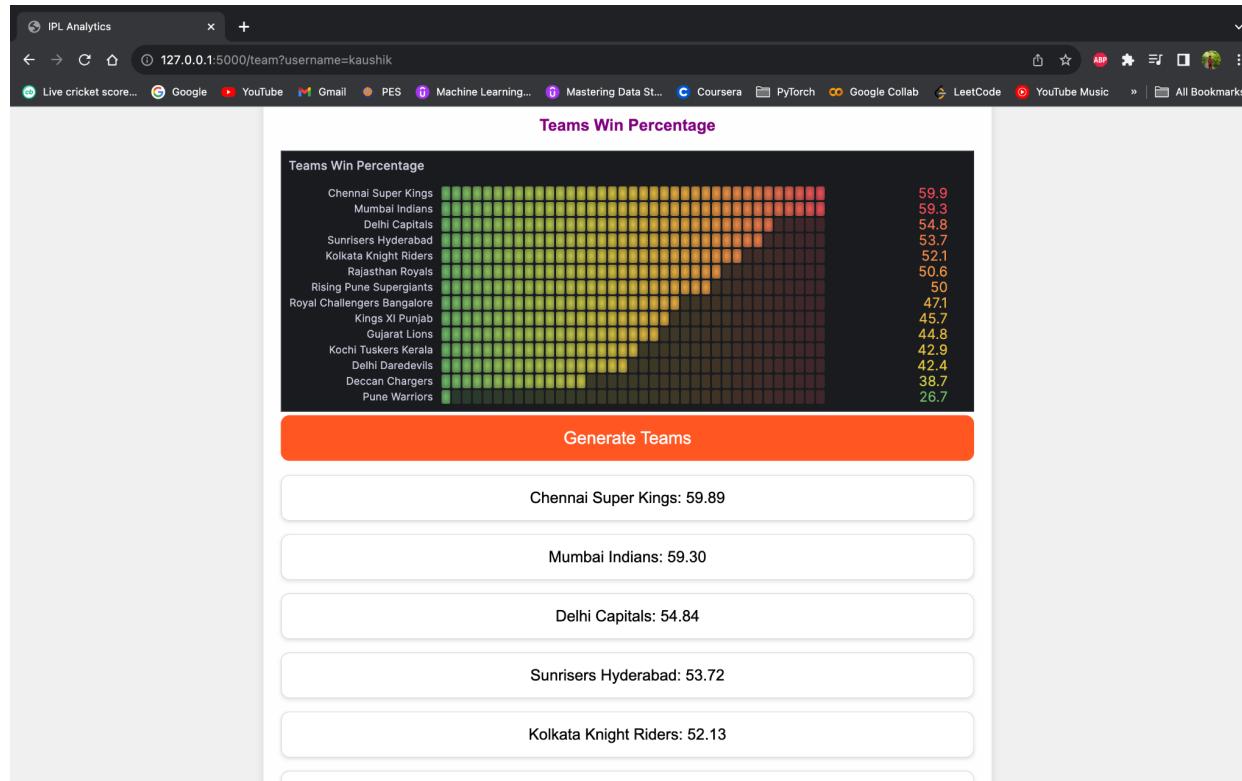
```
    401     team_win_data = [f'{team}: {row[1]}' for row in team_win_results]
```
- Terminal:** Shows a log of HTTP requests from IP 127.0.0.1.

```
127.0.0.1 - - [23/Nov/2023 18:01:31] "GET /static/css/login.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 18:01:38] "GET /home?username=kaushik HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 18:01:38] "GET /static/css/home.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 18:01:40] "GET /static/css/styles.css HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 18:04:00] "GET /static/css/styles.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 18:04:00] "GET /general?username=kaushik HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 18:04:18] "GET /outline?username=kaushik HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 18:30:04] "GET /individual?username=kaushik HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 18:30:04] "GET /static/css/styles.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 18:36:32] "GET /team?username=kaushik HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 18:36:32] "GET /static/css/styles.css HTTP/1.1" 304
```
- Status Bar:** Shows the file is "python3.9 - IPL Analytics Dashboard", line 347, column 34, and the Python version is 3.9.16 ('base' conda).

## Procedures/Functions/Triggers and their Code snippets for invoking them

### Procedures

#### 1) Calculate Win Percentage Ratio



```

OPEN EDITORS
CSE > CSE V Sem > SQL > IPL Analytics Dashboard > app.py > team.html M
3/9 |     SUM(CASE WHEN NUN_BOUNDARY = 0 THEN 1 ELSE 0 END) AS TOTAL_BOUNDARIES
380 |     FROM BALL_BY_BALL
381 |     GROUP BY Team
382 |     ORDER BY TOTAL_BOUNDARIES DESC""")
```

#Query 5: Teams Win Percentage

```

cursor.callproc('CalculateTeamWinPercentage')
result_cursor = conn.cursor()
result_cursor.execute("SELECT * FROM result_of_calculate_team_win_percentage")
team_win_results = result_cursor.fetchall()

cursor.close()
conn.close()
```

```

runs_data = [f'{team}: {matches}' for team, matches in count_matches_results]
wickets_data = [f'{team}: {wins}' for team, wins in wins_results]
matches_data = [f'{team}: {matches}' for team, matches in neutral_matches_results]
dismissals_data = [f'{team}: {boundaries}' for team, boundaries in boundaries_results]
team_win_data = [f'{row[0]}: {row[1]}' for row in team_win_results]
```

```

return check_authentication('team.html', data1=runs_data, data2=wickets_data, data3=matches_data, data4=dismis-
```

## 2) Top 10 Run Scorers in a Particular Season (Example:2016)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

127.0.0.1 - - [23/Nov/2023 19:26:14] "GET /static/css/login.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 19:26:17] "POST /login HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 19:26:18] "GET /static/css/logout.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 19:26:18] "GET /index?username=kushik HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 19:26:18] "GET /static/css/home.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 19:26:21] "GET /general?username=kushik HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 19:29:44] "GET /individual?username=kushik HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 19:29:44] "GET /static/css/styles.css HTTP/1.1" 304 -
127.0.0.1 - - [23/Nov/2023 19:29:46] "GET /team?username=kushik HTTP/1.1" 200 -
127.0.0.1 - - [23/Nov/2023 19:29:46] "GET /static/css/styles.css HTTP/1.1" 304 -

IPL Analytics

127.0.0.1:5000/general

Live cricket score... Google YouTube Gmail PES Machine Learning... Mastering Data St... Coursera PyTorch Google Collab LeetCode YouTube Music All Bookmarks

Generate Result

Top 10 Run Scorers in a Particular Season

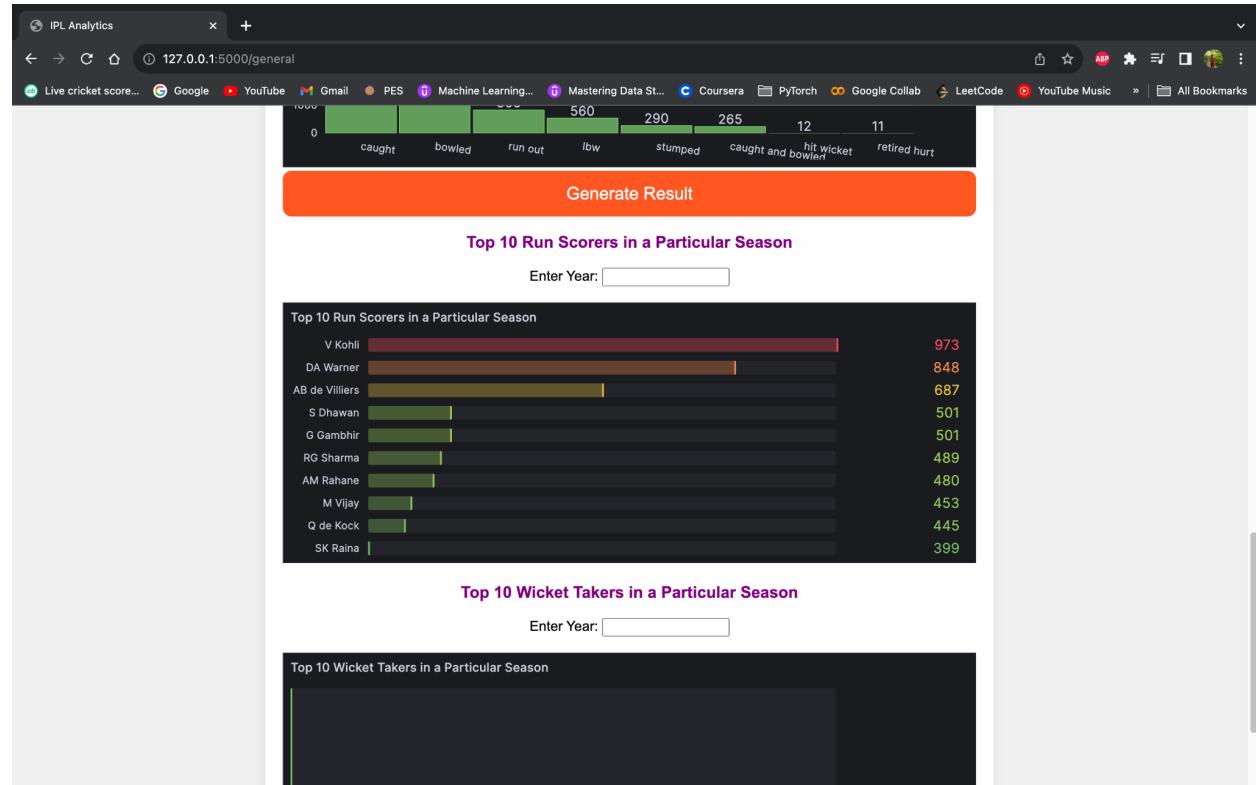
Enter Year: 2016

No data

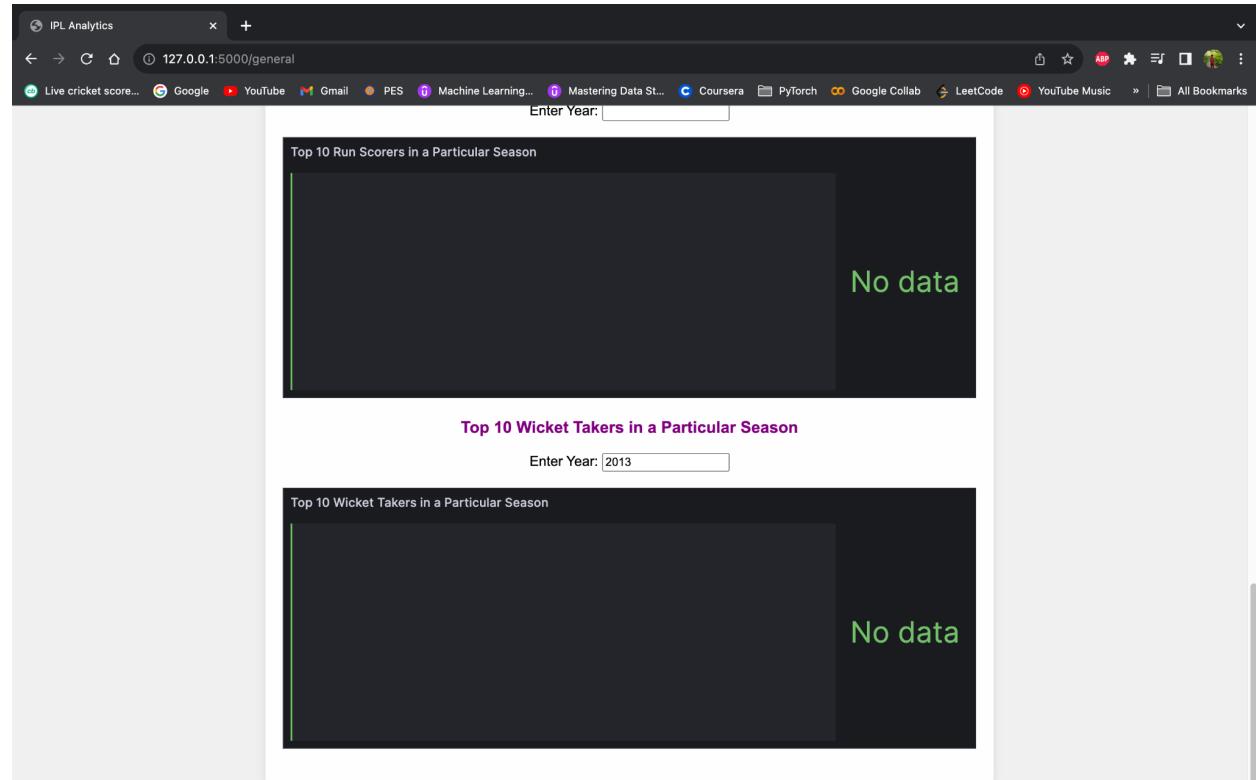
Top 10 Wicket Takers in a Particular Season

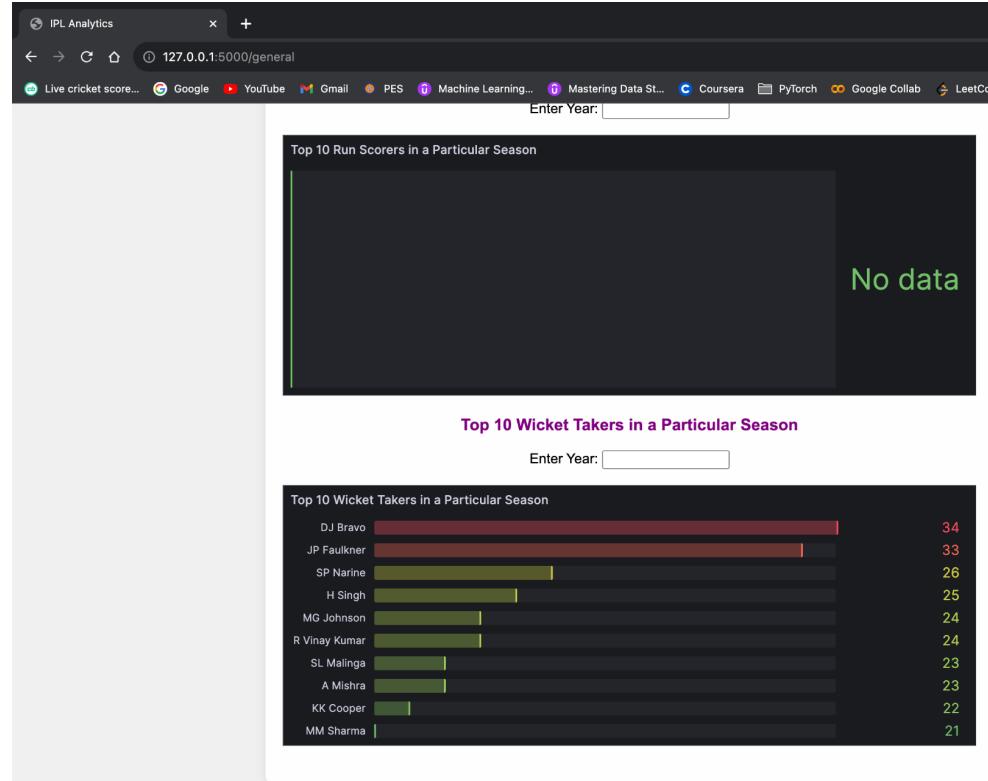
Enter Year:

No data

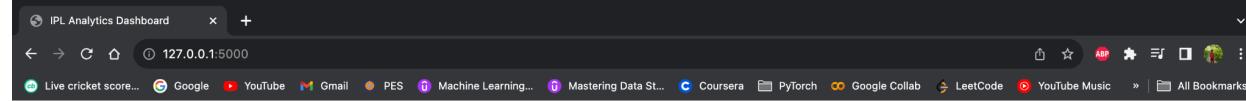


### 3) Top 10 Wickets in a Particular Season (Example:2013)





## Triggers



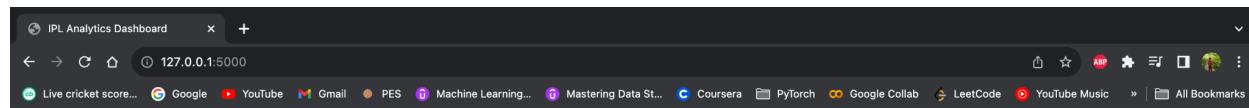
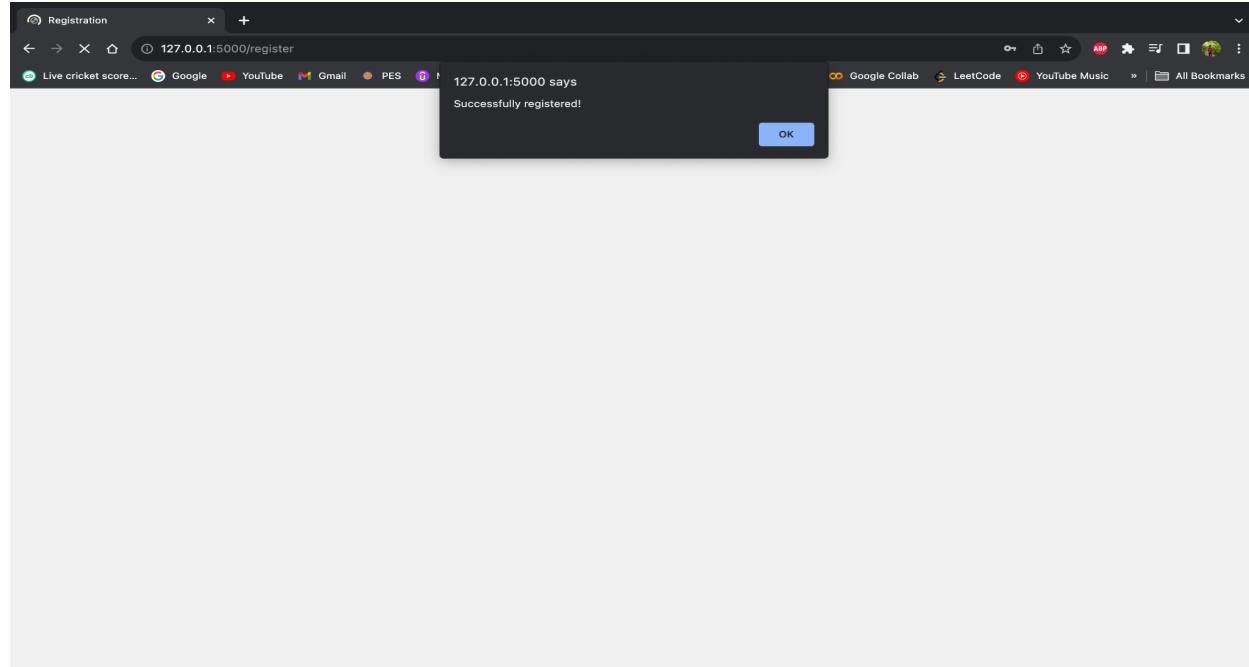
Register

Login

Currently No.Of Users = 2

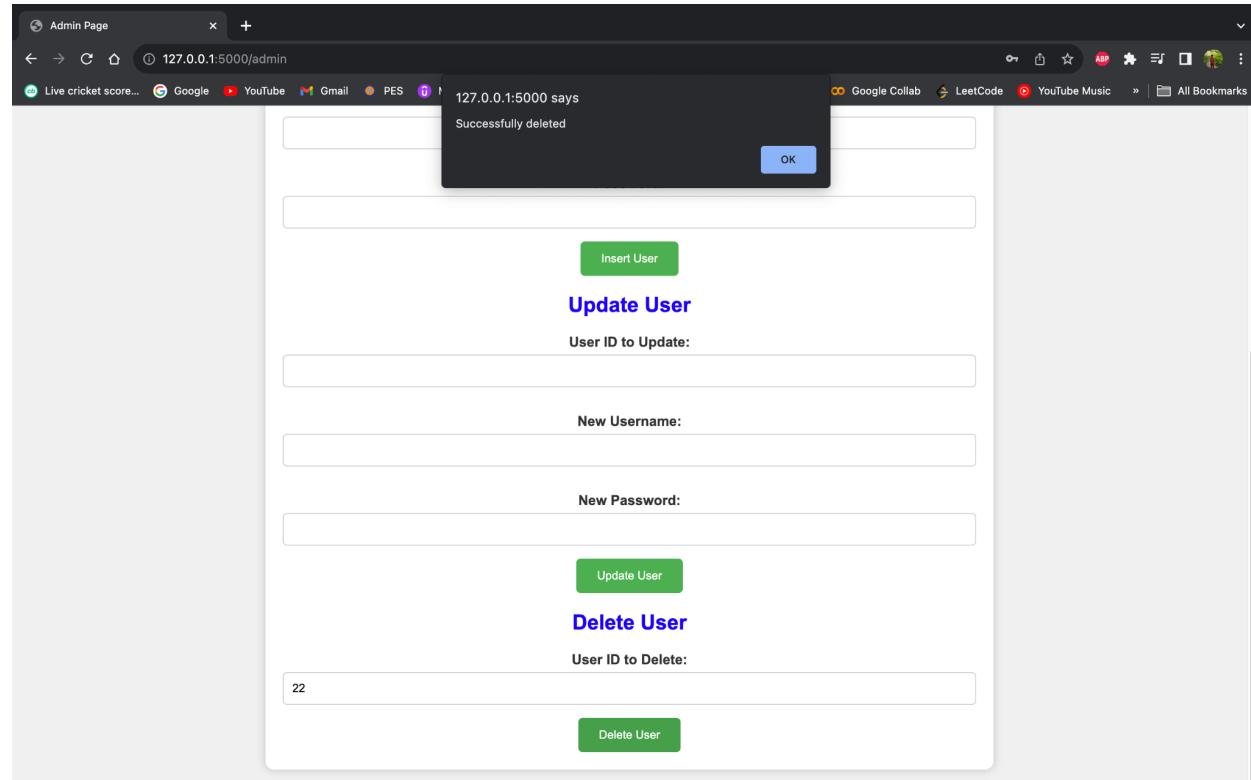
Now, I will register a new account.

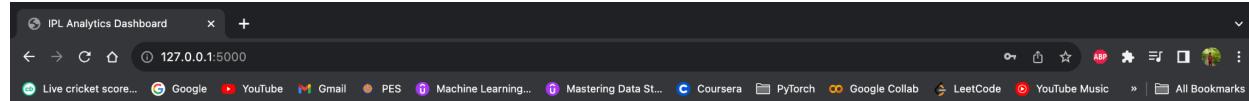
The screenshot shows a web browser window titled "Registration" with the URL "127.0.0.1:5000/register". The page has a header "IPL ANALYTICS DASHBOARD" and a title "Registration". It contains two input fields: "Username:" with the value "yes" and "Password:" with the value "\*\*\*\*\*". Below the fields is a blue "Register" button. At the bottom, there is a link "Already have an account? [Login](#)". The browser's address bar and various bookmarks are visible at the top.



The trigger is executed and the Total Number of Users increases to 3.

Now let's try deleting an account from the admin account





## IPL ANALYTICS DASHBOARD

Total Number of Users : 2

Register

Login

The Total Number of Users is reduced to 2 as we have deleted an account.

The Trigger makes sure that the total number of users registered on the website is updated instantly.



