SDE API Round - Cricbuzz Platform

Statement for Candidates:

Below is the problem statement for your project.

1. Your task is to create an API that supports the requirements mentioned in Problem Statement
2. You have 24 hours to develop and submit the API. You'll need to submit the link to the repo (GitHub / bitbucket / gitlab etc) where you've pushed your commits.
3. Ensure that your repository includes a `README.md` file where you've mentioned how to setup, run and run tests (optional) for your code. When in doubt make a plausible assumption and state it explicitly in your `README.md` file.
4. You can use Google to find solutions, however, copying an existing project is not allowed.
5. Do not use ChatGPT / copilot or any tool that generates code for you, otherwise your submission will not be considered for evaluation.
6. Once completed, please make your submission here or on the link shared by the recruiter https://forms.gle/rXYwpVAf4ofxdQQz8

Problem Statement

Hey there, Mr. X. You have been appointed to design a platform like Cricbuzz, wherein guest users can come on the platform and browse across multiple matches and can see either of them in detail.

There is a Role Based Access provision and 2 types of users would exist :

1. Admin - can perform all operations like adding matches, players in the teams, updating stats and scores, etc.
2. Guest - can only view matches and their details.

Tech Stack:

1. Any web server of your choice (Python Flask / Django, NodeJS / ExpressJS , Java/Springboot, etc)
2. Database: MySQL/PostgreSQL (Compulsory)

Requirements

1. **Register Admin**
   Create an endpoint for registering a user.

```
[POST] /api/admin/signup

Request Data : {
  "username": "example_user",
  "password": "example_password",
  "email": "user@example.com"
}

Response Data : {
  "status": "Admin Account successfully created",
  "status_code": 200,
  "user_id": "123445"
}
```

2. **Login User**
   Provide the ability to the user to log into his account

```
[POST] /api/admin/login

Request Data : {
  "username": "example_user",
  "password": "example_password"
}

For successful login
Response Data : {
  "status": "Login successful",
  "status_code": 200,
  "user_id": "12345",
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9"
}

For failure
Response Data: {
  "status": "Incorrect username/password provided. Please retry",
  "status_code": 401
}
```

3. **Create Match**

   Create an endpoint where an admin can add matches

```
[POST] /api/matches

Headers : {
  "Authorization": "Bearer {token}"
}

Request Data : {
  "team_1": "India",
  "team_2": "Australia",
  "date": "2023-07-12",
  "venue": "Sydney Cricket Ground"
}

Response Data : {
  "message": "Match created successfully",
  "match_id": "3"
}
```

4. **Get Match Schedules**

   Create an endpoint for the guest user where all the matches will be fetched

```
[GET] /api/matches

Request Data : {}

Response Data : {
  "matches": [
    {
      "match_id": "1",
      "team_1": "India",
      "team_2": "England",
      "date": "2023-07-10",
      "venue": "Lord's Cricket Ground"
    },
    {
      "match_id": "2",
      "team_1": "Australia",
      "team_2": "New Zealand",
      "date": "2023-07-11",
      "venue": "Melbourne Cricket Ground"
    },
    ...
  ]
}
```

(You can add dummy data of your own for this use case)

5. **Get Match Details**

   Create an endpoint for the guest user where all the details of a particular match will be fetched.

```
[GET] /api/matches/{match_id}

Request Data : {}

Response Data : {
  "match_id": "1",
  "team_1": "India",
  "team_2": "England",
  "date": "2023-07-10",
  "venue": "Lord's Cricket Ground",
  "status": "upcoming",
  "squads": [
    "team_1": [
      {
        "player_id": "123",
        "name": "Virat Kohli"
      },
      {
        "player_id": "456",
        "name": "Jasprit Bumrah"
      },
      ...
    ],
    "team_2": [...]
  ]
}
```

   (You can add dummy data of your own for this use case)

6. **Add a Team Member to a Squad**

   Create an endpoint for the admin to add a player to the team's squad.

```
[POST] /api/teams/{team_id}/squad

Request Data : {
  "name": "Rishabh Pant",
  "role": "Wicket-Keeper"
}

Response Data : {
  "message": "Player added to squad successfully",
  "player_id": "789"
}
```

7. **Get Player Statistics**

   Create an endpoint for the logged in user to fetch player statistics

```
[GET] /api/players/{player_id}/stats

Request Data : {}
```

```

Response Data : {
  "player_id": "123",
  "name": "Virat Kohli",
  "matches_played": 200,
  "runs": 12000,
  "average": 59.8,
  "strike_rate": 92.5
}
```

(You can add dummy data of your own for this use case)

Mandatory requirement:

For all the admin API endpoints you need to send the **Authorization Token** received in the login endpoint.