SDE API Round - IRCTC

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 3 hours to develop the API.
3. You can use Google to find solutions, however, copying an existing project is not allowed.
4. At the end of the 3 hours, you will be given get 15 minutes to make your submission on Google Forms & to push your project into a GitHub repository
5. Submission form : API Round Submissions 2023
6. Our team will evaluate your API submission and the result will be published at 1 PM on the same day.
7. For any other queries feel free to join this meet link : Meet

Problem Statement

Hey there, Mr. X. You have been appointed to design a railway management system like IRCTC, where users can come on the platform and check if there are any trains available between 2 stations.

The app will also display how many seats are available between any 2 stations and the user can book a seat if the availability > 0 after logging in. Since this has to be real-time and multiple users can book seats simultaneously, your code must be optimized enough to handle large traffic and should not fail while doing any bookings.

If more than 1 users simultaneously try to book seats, only either one of the users should be able to book. Handle such race conditions while booking.

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

1. Admin - can perform all operations like adding trains, updating total seats in a train, etc.
2. Login users - can check availability of trains, seat availability, book seats, get booking details, etc.

Tech Stack:

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

Requirements

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

```
[POST] /api/signup

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

Response Data : {
  "status": "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/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. **Add a New Train**

   An endpoint for the admin to create a new train with a source and destination

```
[POST] /api/trains/create

Request Data : {
  "train_name": "Express Train",
  "source": "Station A",
  "destination": "Station B",
  "seat_capacity": 100,
  "arrival_time_at_source": "14:00:00",
  "arrival_time_at_destination": "20:30:00"
}

Response Data : {
  "message": "Train added successfully",
  "train_id": "9876543210"
}
```

4. **Get Seat Availability**

   Create an endpoint for the users where they can enter the source and destination and fetch all the trains between that route with their availabilities

```
[GET] /api/trains/availability?source=SOURCE&destination=DESTINATION

Request Data : {}

Params: {
  "source": str
  "destination": str
}

Response Data : [
  {
    "train_id": "9876543210",
    "train_name": "Express Train",
    "available_seats": 75
  },
  {
    "train_id": "9876543211",
    "train_name": "Express Train 2",
    "available_seats": 0
  }
]
```

5. **Book a Seat**
   Create an endpoint for the users to book a seat on a particular train

```
[POST] /api/trains/{train_id}/book

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

Request Data : {
  "user_id": "1234567890",
  "no_of_seats": 2
}

Response Data : {
  "message": "Seat booked successfully",
  "booking_id": "5432109876",
  "seat_numbers": [5,6]
}
```

6. **Get Specific Booking Details**
   Create an endpoint for the users to book a seat on a particular train

```
[GET] /api/bookings/{booking_id}

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

Request Data : {}

Response Data : {
  "booking_id": "5432109876",
  "train_id": "9876543210",
  "train_name": "Express Train",
  "user_id": "1234567890",
  "no_of_seats": 1
  "seat_numbers": [7],
  "arrival_time_at_source": "2023-01-01 14:00:00",
  "arrival_time_at_destination": "2023-01-01 20:30:00"
}
```

Mandatory requirement:

1. You need to protect all the admin API endpoints with an **API key** that will be known only to you and the admin so that no one can add false data to your system.
2. For booking a seat and getting specific booking details, you need to send the **Authorization Token** received in the login endpoint.