**Railway Ticket Booking System**

**Introduction**

* The Railway Ticket Booking project aims to provide a simple and efficient system for users to purchase train tickets from London to France. The project focuses on automating the ticket purchasing process, allocating seats to users, and providing functionality to view, modify, and remove user details.

**Purpose**

* The primary purpose of the project is to streamline the process of booking train tickets for users traveling from London to France. By offering a straightforward API, users can purchase tickets, receive receipts, view allocated seats, modify their seat assignments, and remove their details from the system**.**

**Goals:**

* Automated Ticket Purchase: Allow users to submit ticket purchase requests with essential details, such as departure location, destination, user information, and preferred train section.
* Seat Allocation: Automatically allocate seats to users based on the chosen train section (Section A or Section B). The system ensures efficient utilization of available seats.
* User Interaction APIs: Provide APIs for users to view their receipt details, see users and seats in a specific train section, modify their seat assignments, and remove their details from the system.
* Data Storage: Utilize a HashMap-based in-memory storage mechanism to store user details and ticket information.

**High-Level Architecture:**

The project employs the Spring Boot framework to create a RESTful API. Key components of the architecture include:

**1.Spring Boot Application:**

* Serves as the main application container.
* Manages the routing of HTTP requests to the corresponding controller methods.

**2.Controller Layer:**

* Consists of the TicketController class with various methods to handle different API endpoints.
* Maps incoming HTTP requests to appropriate business logic.

**3.Business Logic:**

* Manages the core functionality of ticket purchasing, seat allocation, user data storage, and modification/removal of user details.

**4.Data Storage:**

* Utilizes a HashMap to store user details and ticket information in memory.
* Provides a simple and quick storage solution for the scope of the project.

**5.API Endpoints:**

* Defines several RESTful API endpoints for ticket purchase, receipt viewing, user and seat information retrieval, user modification, and user removal.

**6.User and Ticket Classes:**

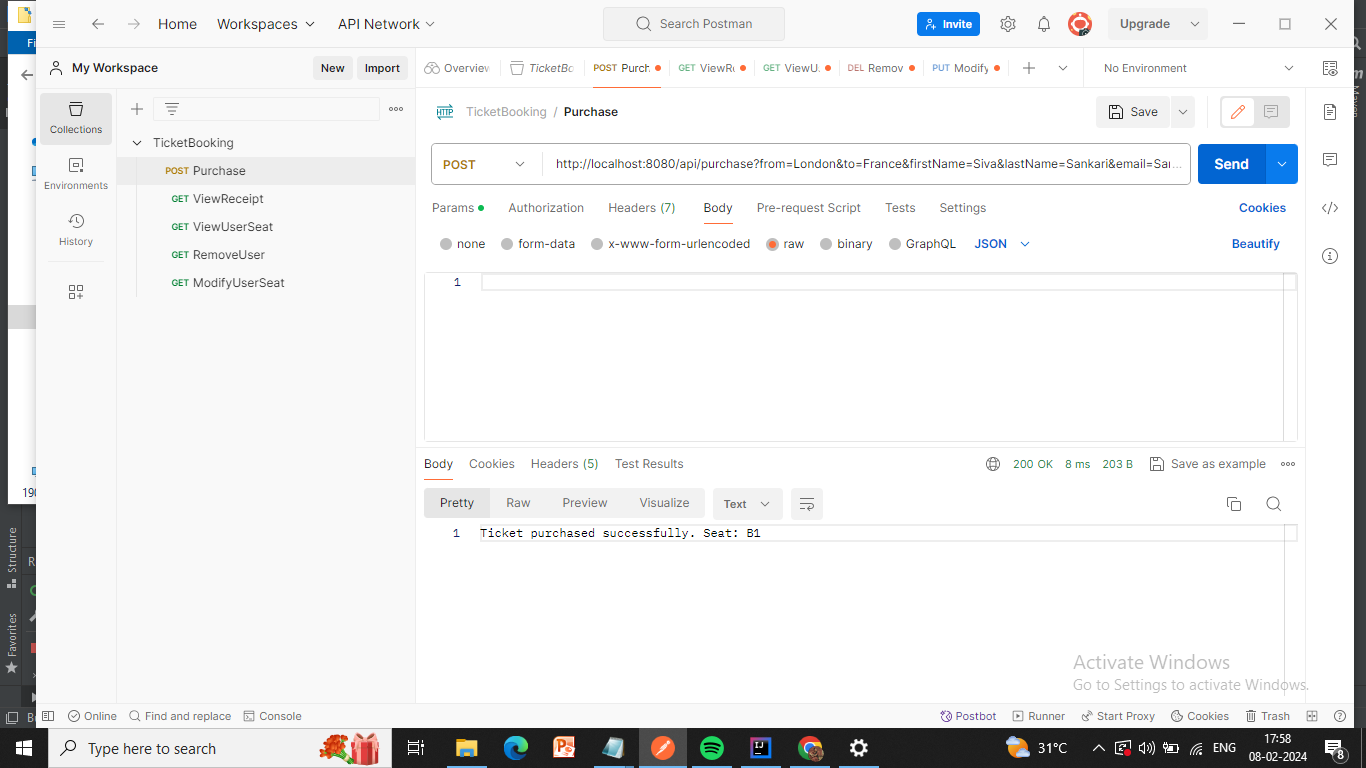
* Internal classes within TicketController representing User and Ticket entities.

**Result**

**POST :** <http://localhost:8080/api/purchase?from=London&to=France&firstName=Siva&lastName=Sankari&email=Sankari12!@example.com&section=B>

The user purchase the ticket by providing user first name, last name, emailed and section using POST method

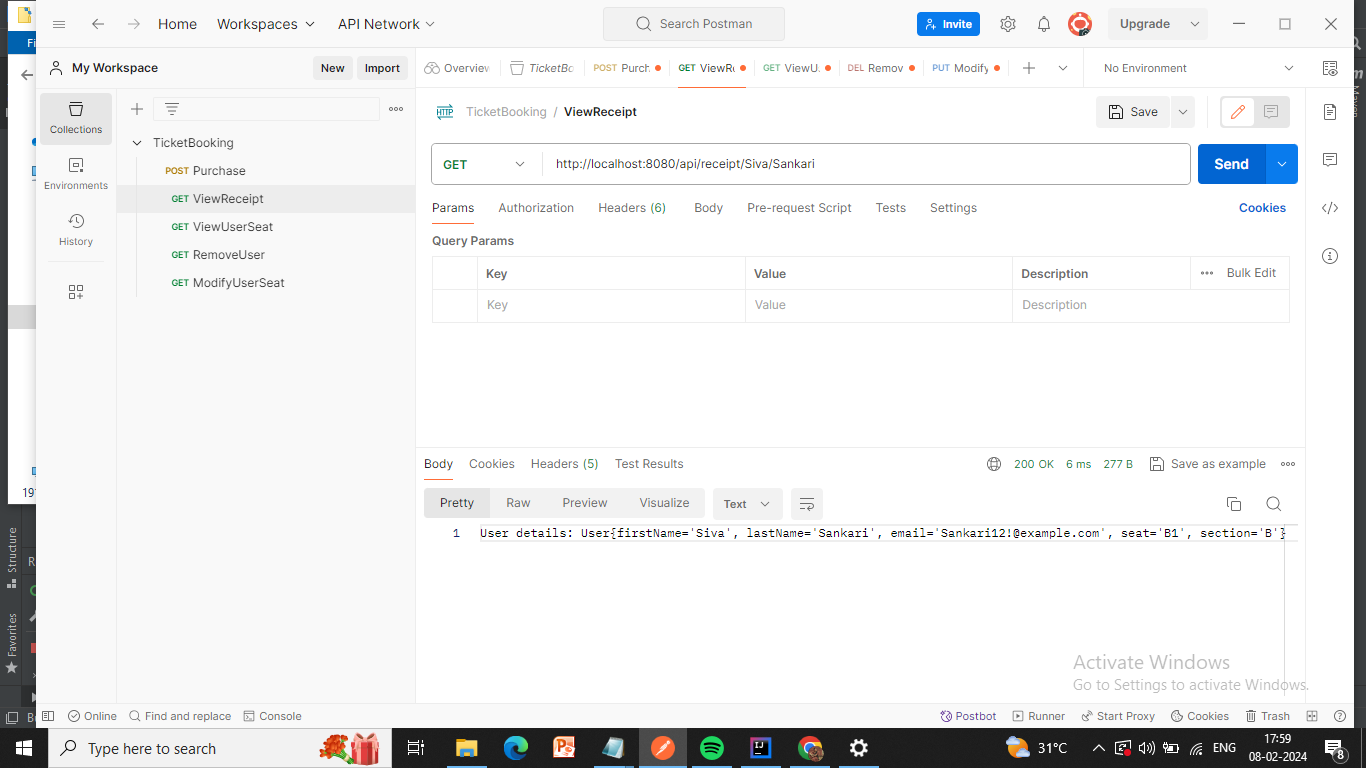
OUTPUT :

****

**GET**: <http://localhost:8080/api/receipt/Siva/Sankari>

An API that shows the details of the receipt for the user

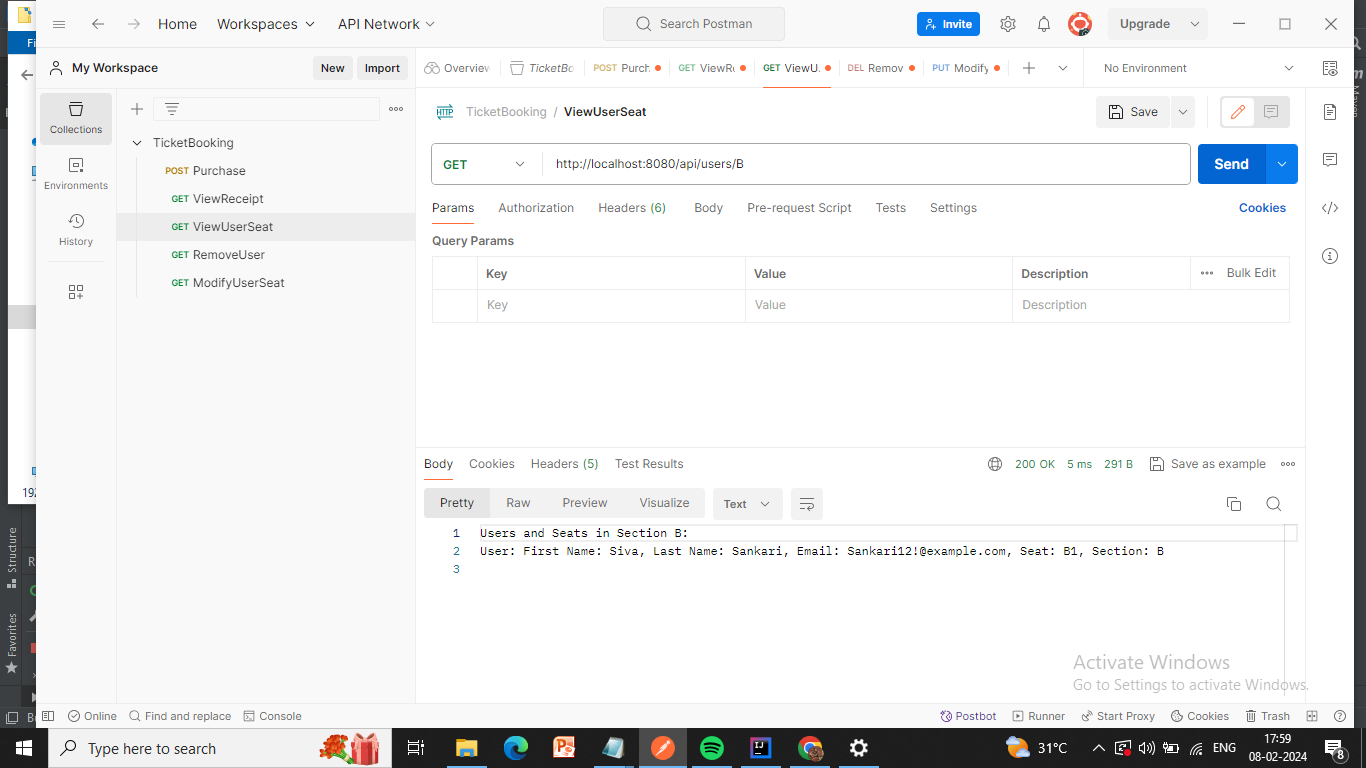
OUTPUT :



**GET** : <http://localhost:8080/api/users/B>

An API that lets user view the users and seat they are allocated by the requested section

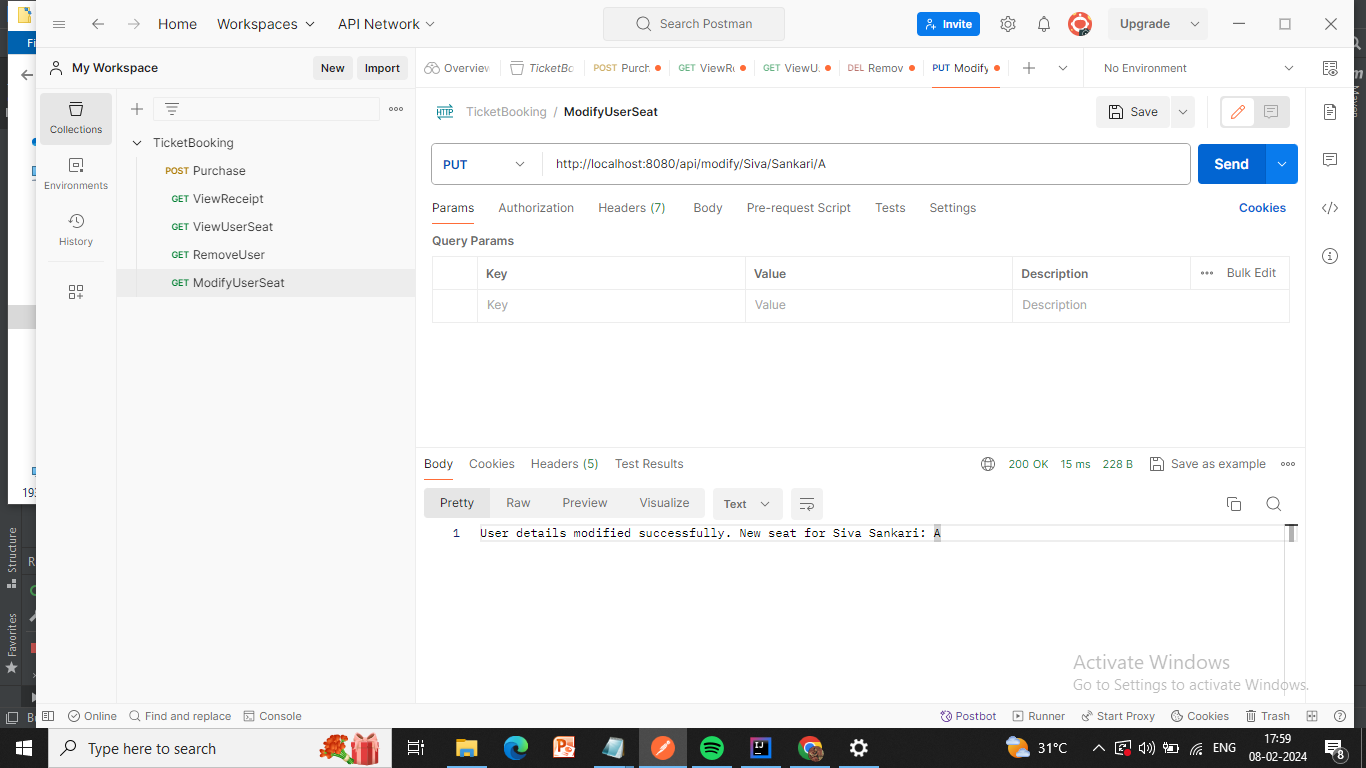
OUTPUT :



**PUT**: <http://localhost:8080/api/modify/Siva/Sankari/A>

An API to MODIFY a user’s seat

OUTPUT :



**DELETE** : <http://localhost:8080/api/remove/Siva/Sankari>

An API to remove a user from the train

OUTPUT :

