**Airline API Documentation**

Welcome to the Airline API documentation. This API allows users to register, authenticate, search for scheduled flights, and view search history. Below are the endpoints and functionalities provided by the API.

**Endpoints**

**1. Register Endpoint**

* **URL:** **/api/auth/register**
* **Method:** POST
* **Description:** Registers a new user with the API.
* **Request Body:**
* {
* "firstName": "string",
* "lastName": "string",
* "email": "string",
* "password": "string",
* "confirmPassword": "string"
* }
* **Response:** Returns a JWT token upon successful registration.

**2. Login Endpoint**

* **URL:** **/api/auth/login**
* **Method:** POST
* **Description:** Authenticates an existing user with the API.
* **Request Body:**
* {
* "email": "string",
* "password": "string"
* }
* **Response:** Returns a JWT token upon successful login.

**3. Search Flights Endpoint**

* **URL:** **/api/Search**
* **Method:** POST
* **Description:** Searches for scheduled flights based on a search term for authenticated user.
* **Authorization Header:** Bearer Token (JWT token obtained from registration or login)
* **Response:** Returns matched scheduled flight information based on the search term.

**4. Search History Endpoint**

* **URL:** **/api/Search/searchHistory**
* **Method:** GET
* **Description:** Retrieves search history for the authenticated user.
* **Authorization Header:** Bearer Token (JWT token obtained from registration or login)
* **Response:** Returns the search history of the authenticated user.

**Prerequisites**

Before using the API, ensure the following prerequisites are met:

1. **Database Configuration:** Configure the **AirlineConnectionString** in the appsettings.json to point to an instance of SQL Server. The API will use code-first migrations to create the database automatically when the application is run for the first time.
2. **Full-Text Search Feature:** Ensure that the full-text search feature is enabled on the SQL Server instance to enable efficient searching of flight schedules.

**Sample Queries**

After configuring the connection string and running the application, you can execute the following sample queries on the AirlineDB created by the application.

1. **List Registered Users:**
2. Select \* from AspNetUsers;
3. **List Roles:**

; Select \* from AspNetRoles;

1. **List Flight Schedules:**

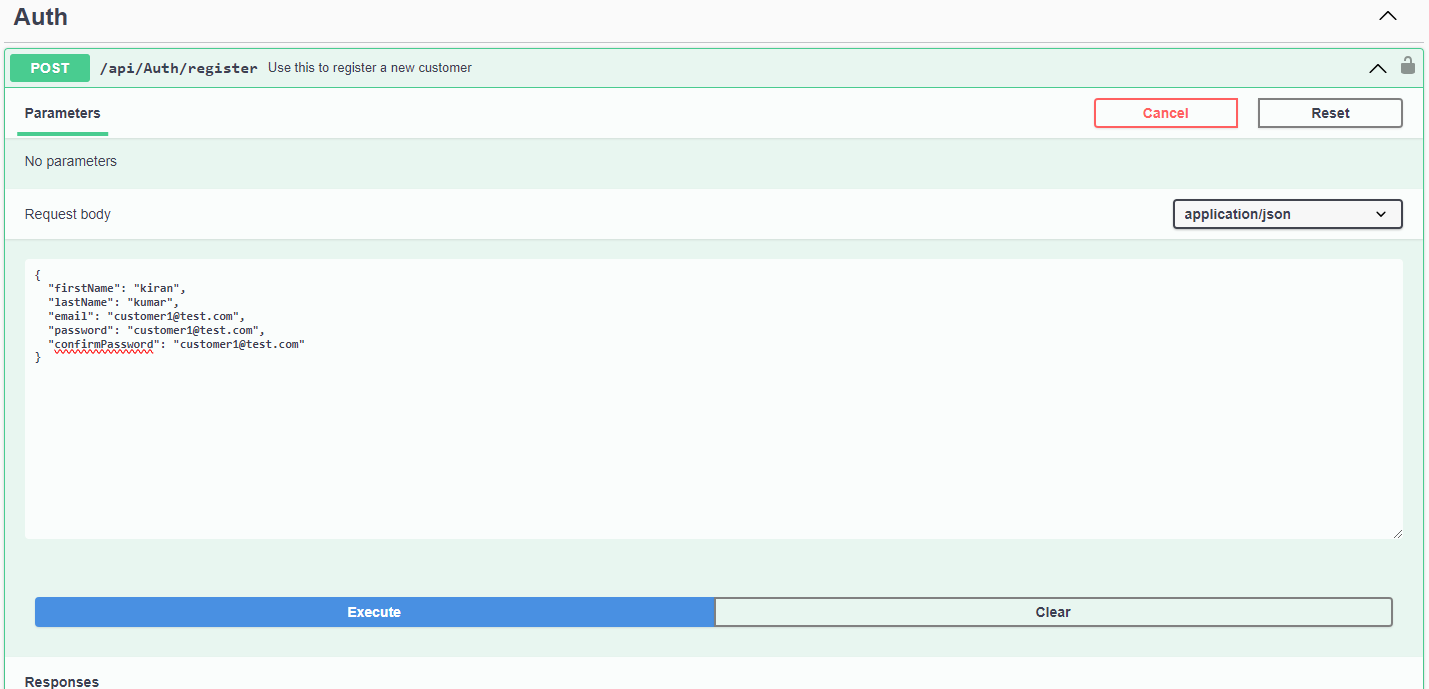
; select \* from FlightSchedules

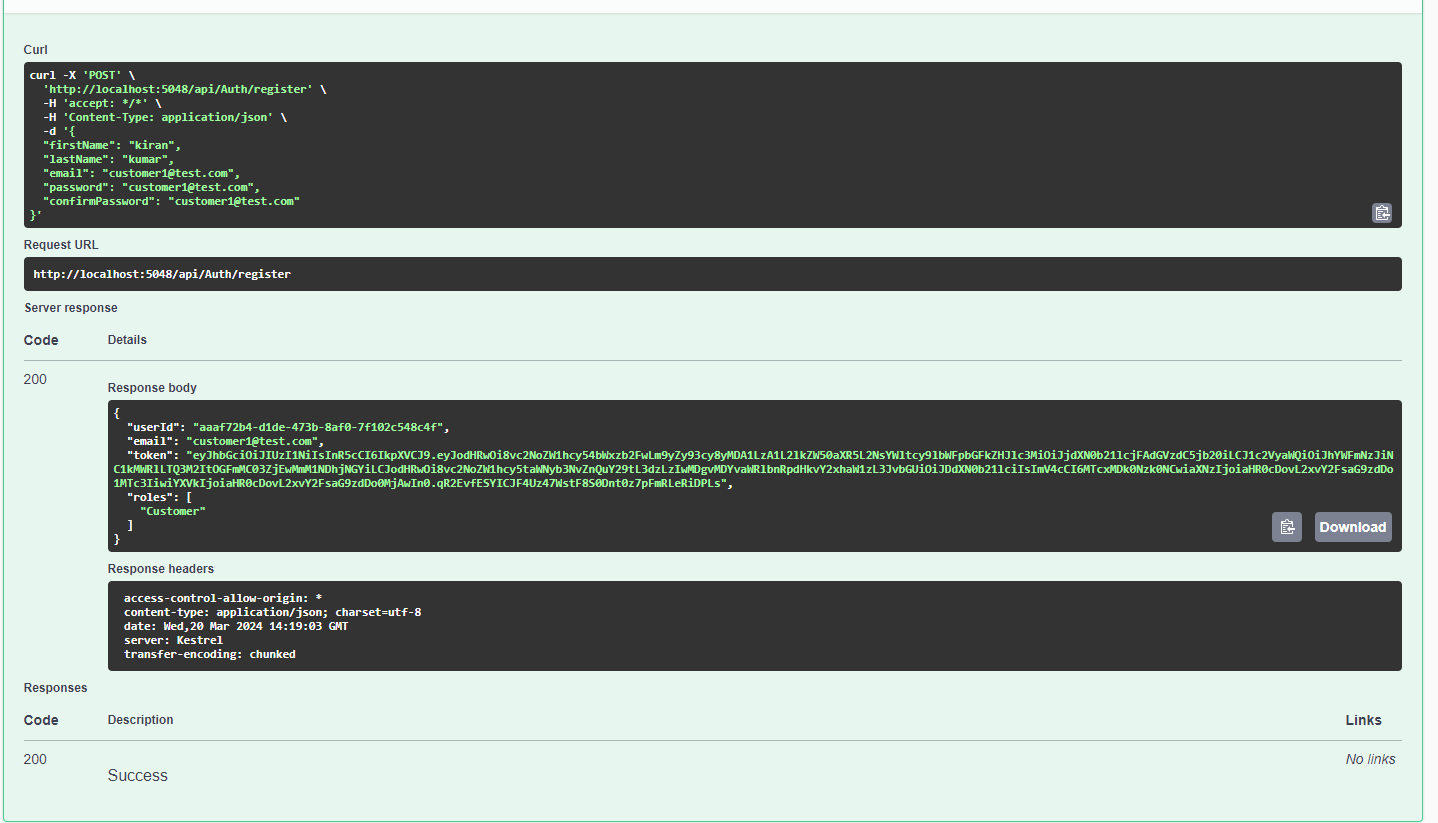
**Using the API**

1. **Register New User:** Use the register endpoint with a sample JSON payload to register a new user.
2. **Login:** Use the login endpoint with email and password to authenticate an existing user and obtain an authentication token.
3. **Search Flights:** Use the search endpoint to search for scheduled flights based on a search term. Pagination is implemented here so user can pass page number and page size information. User can also optionally pass multiple additional criteria to filter the data based on passed criteria’s.Sorting is also supported based on relevance(default), departuretime and arrivaltime.Provide the obtained authentication token in the Authorization header.
4. **View Search History:** Use the history endpoint to view the search history of the authenticated user. Provide the obtained authentication token in the Authorization header.

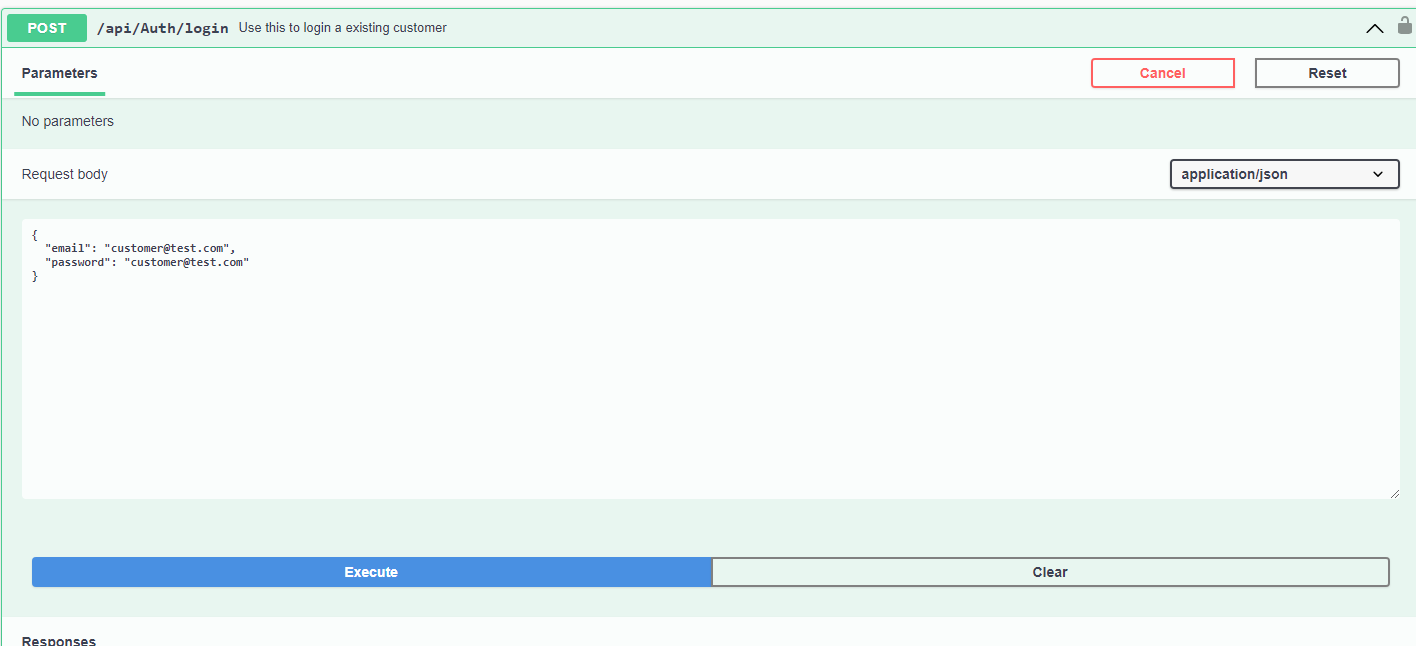
**Some of the screenshots of API**

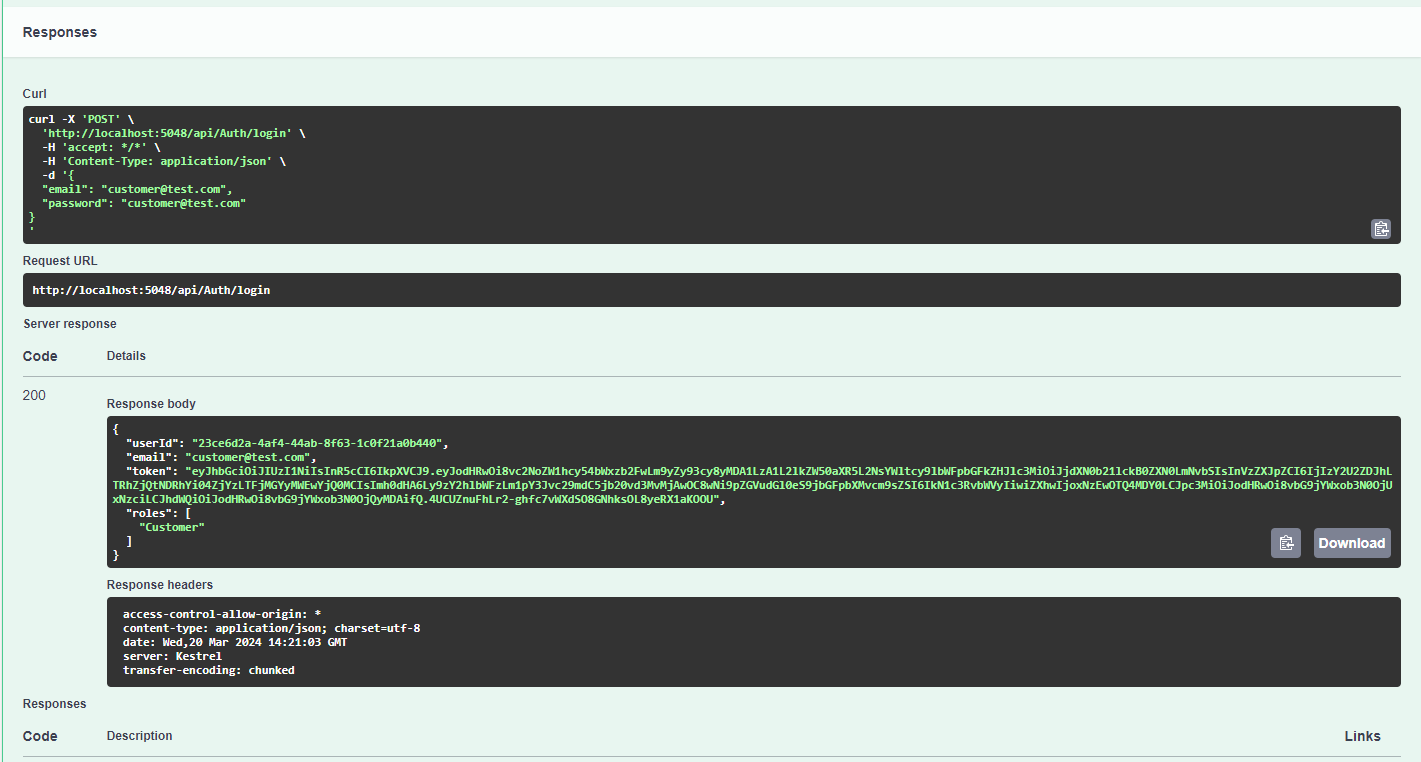
Register User to get token on successful registration



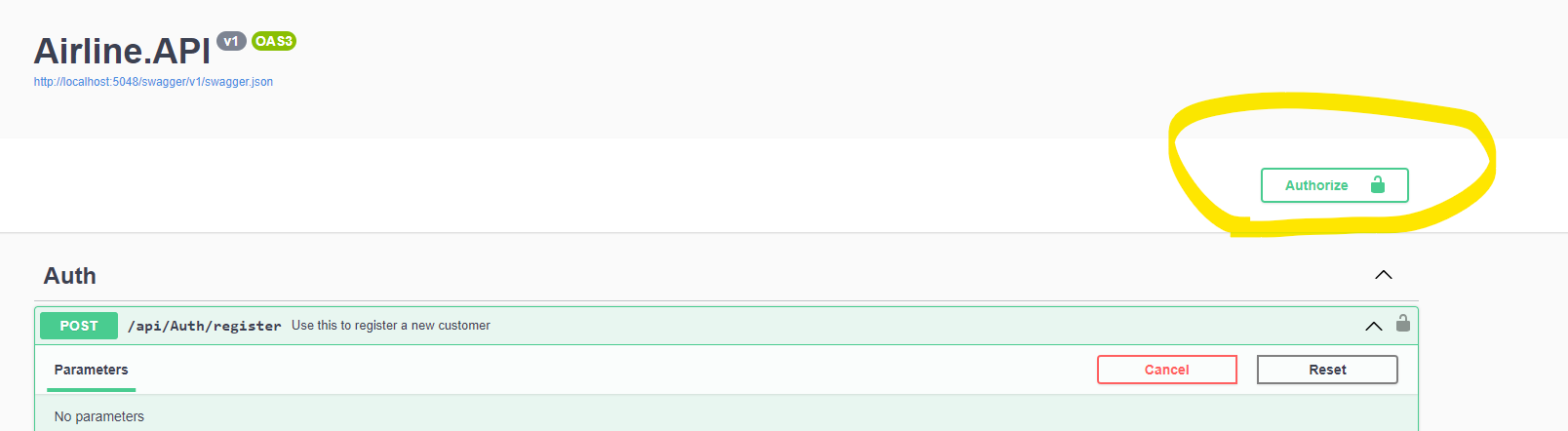


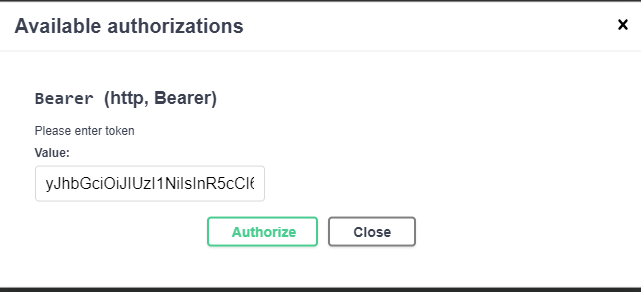
Login existing user to get token



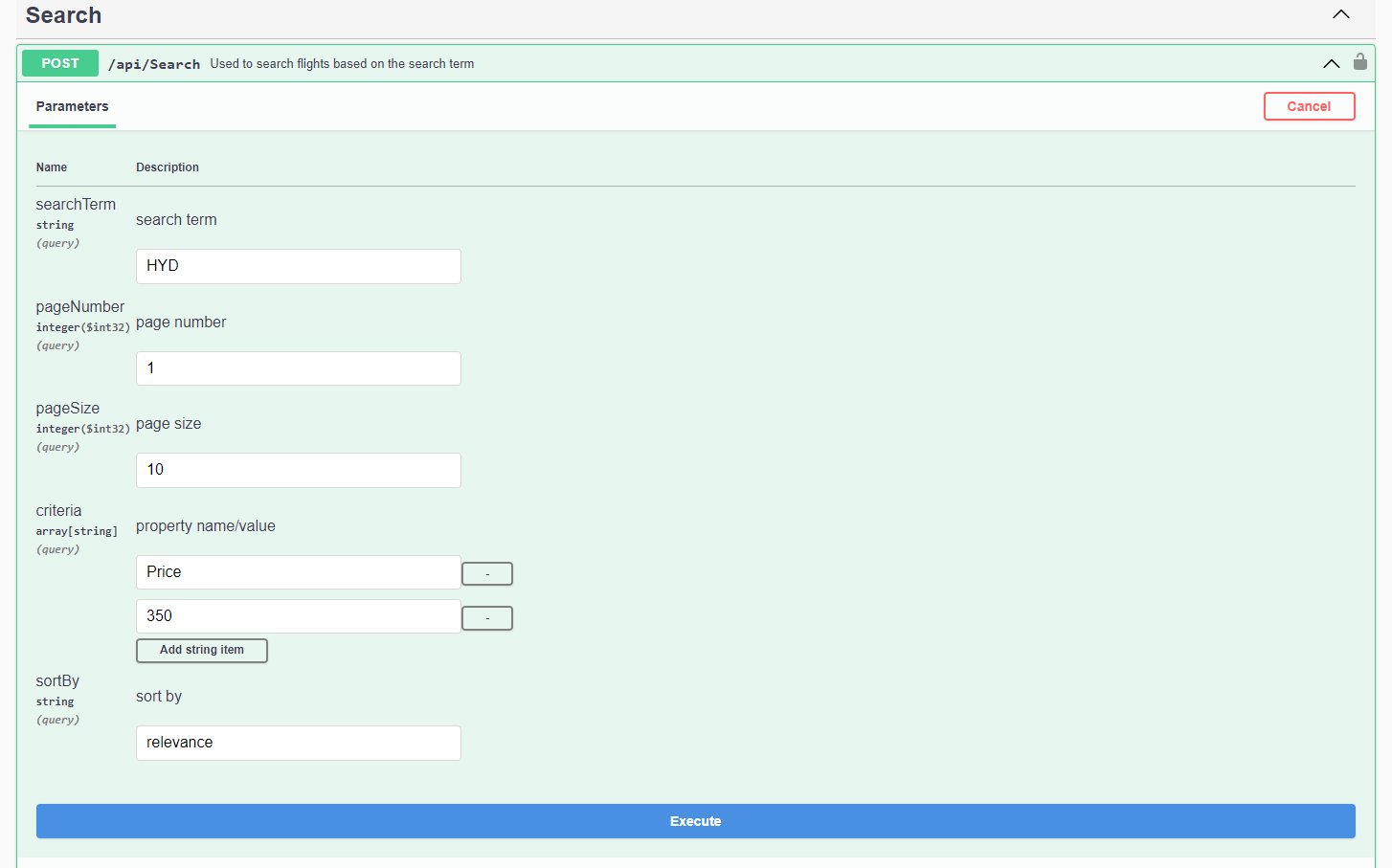


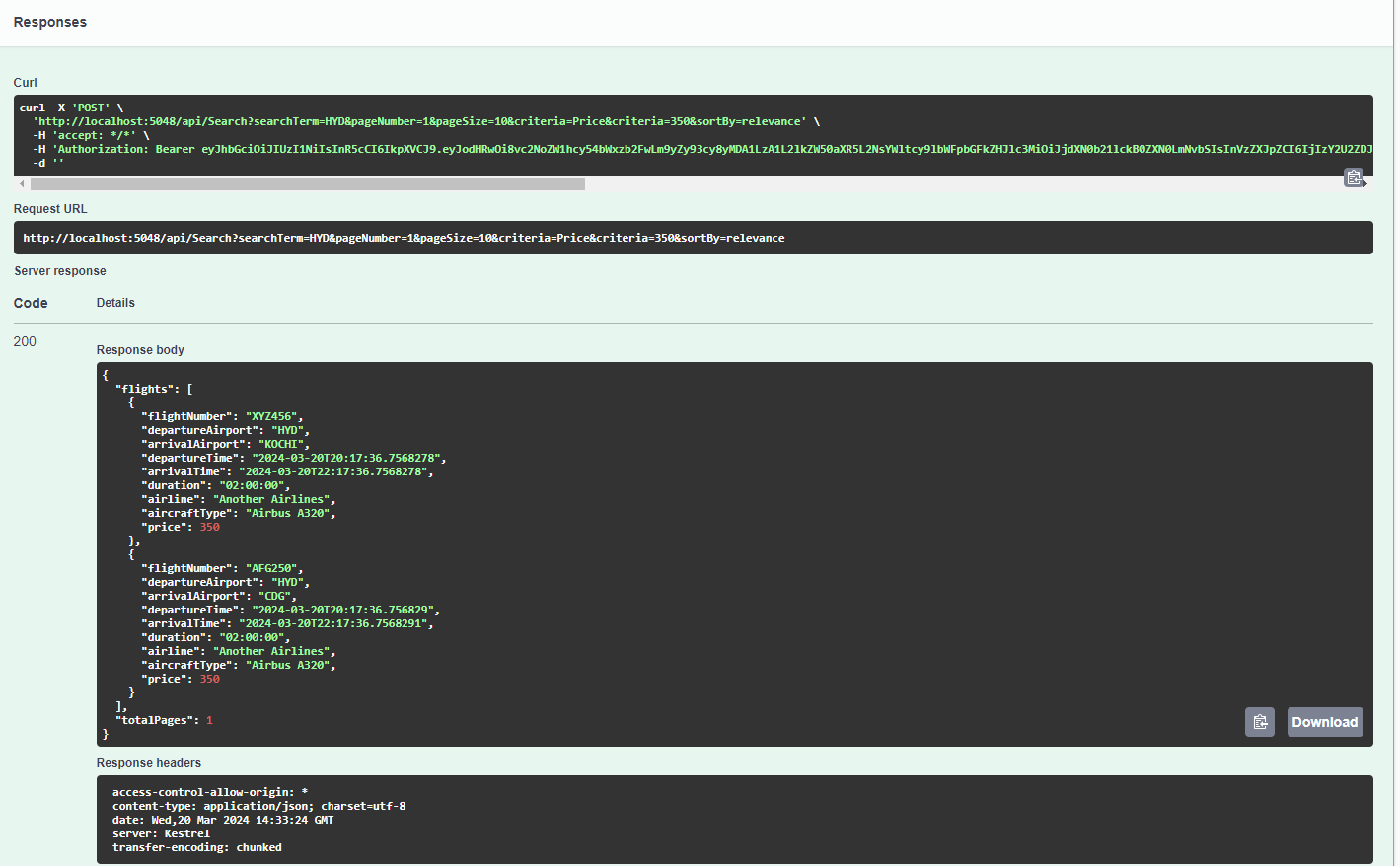
Use token to authenticate user into the system to make calls to search and search history endpoint





Making call to search endpoint





Checking search history of the logged in user

