***REST\_API-ASSIGNMENT-SOLUTION***

You are required to create an Employee Management Rest Api based Web application, where you will be developing CRUD(Create,Read,Update and Delete) functionality along with Sorting and some concepts of security.

Your Rest Api should be secure and should have different endpoints for different operations.

The following are the screenshots of the various functionalities added in the project APIEMPLOYEEMANAGEMENT.

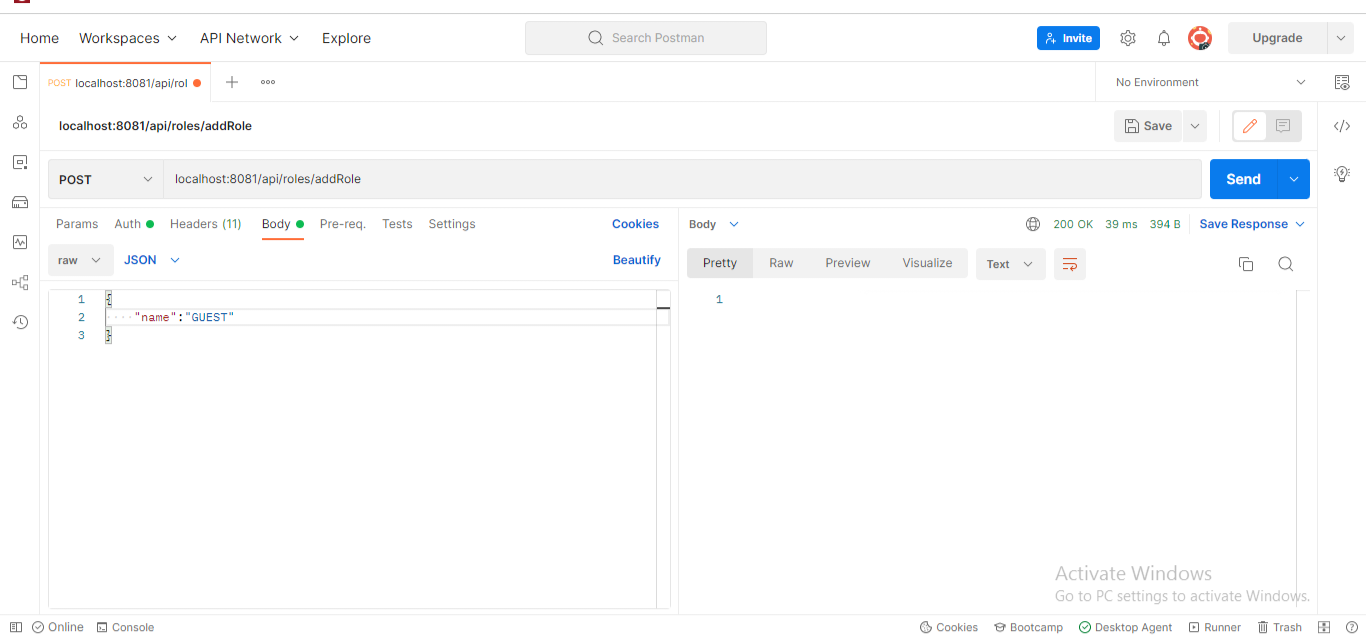
The functionalities are divided in two parts presenting the two kinds of users with different allowed access properties that are **ADMIN and USER**.

🡪The admin is allowed to carry out all kinds of functionalities whereas the user is only allowed to just view the list or search.

**The following are the screenshots of the functionalities accessed by the admin:**

1. We are able to add roles in the database dynamically in the db.

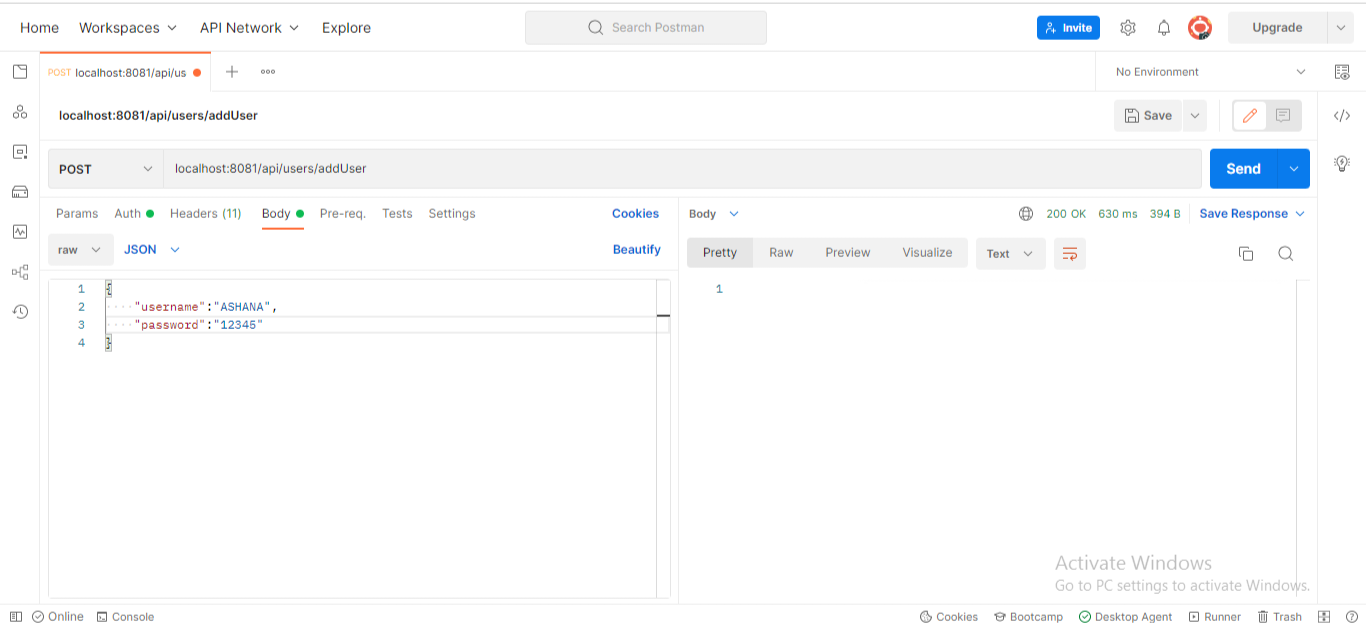
<URL:localhost:8081/api/roles/addRole>



In the screenshot, we have the “name” specifying a role which can be assigned to the user that will be used for authentication purposes while interacting with the API.

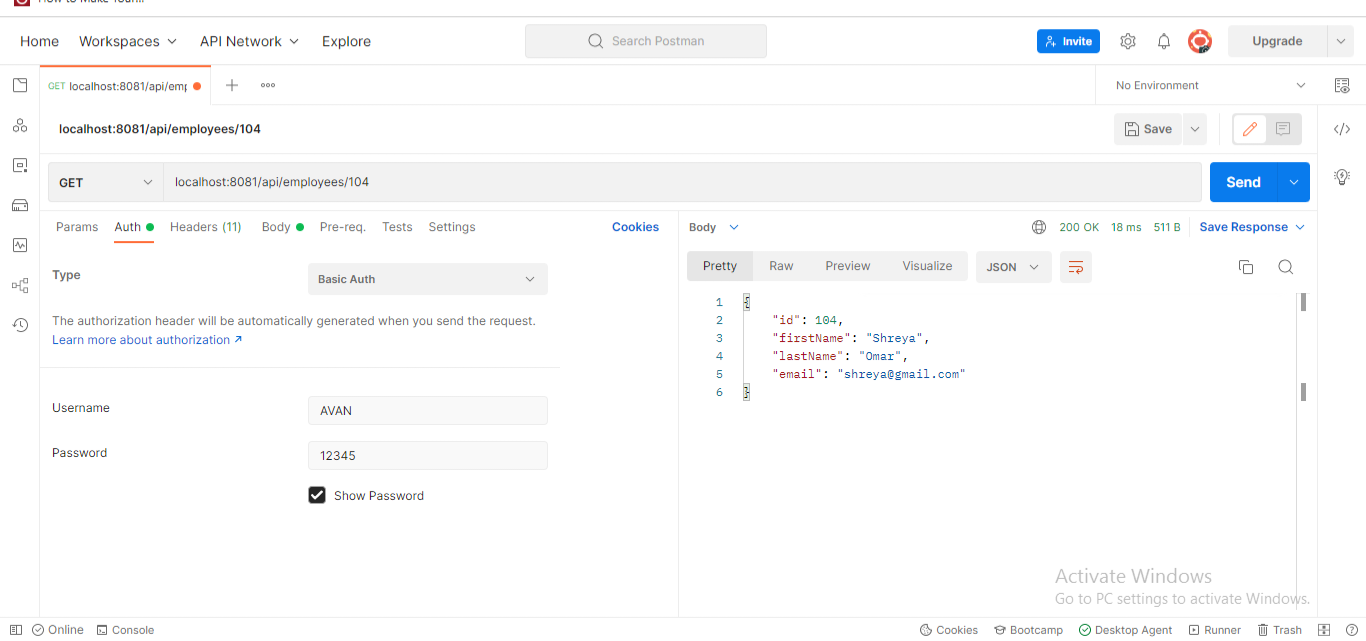
1. Your application should be able to add Users in the db which can be used for authentication purposes.

<URL:localhost:8081/api/users/addUser>



1. Your application should provide endpoint to fetch or get an employee record specifically based on the id of that employee.

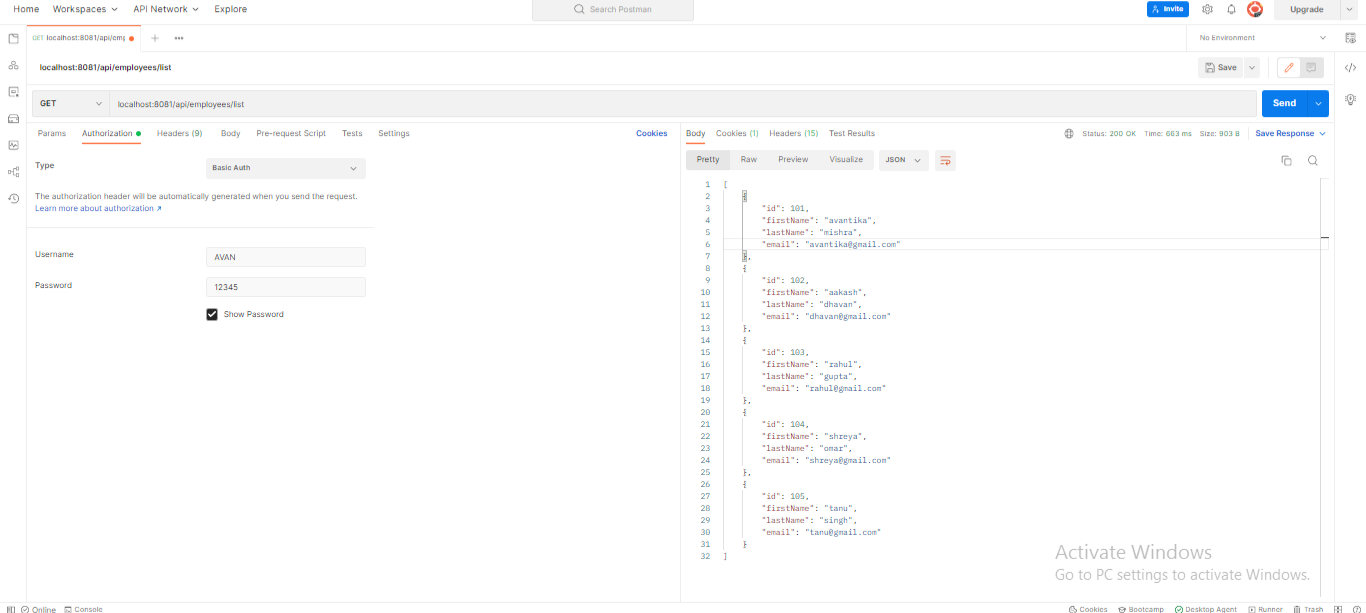
<URL:localhost:8081/api/employees/104>



Here 104 is the Employee-Id.

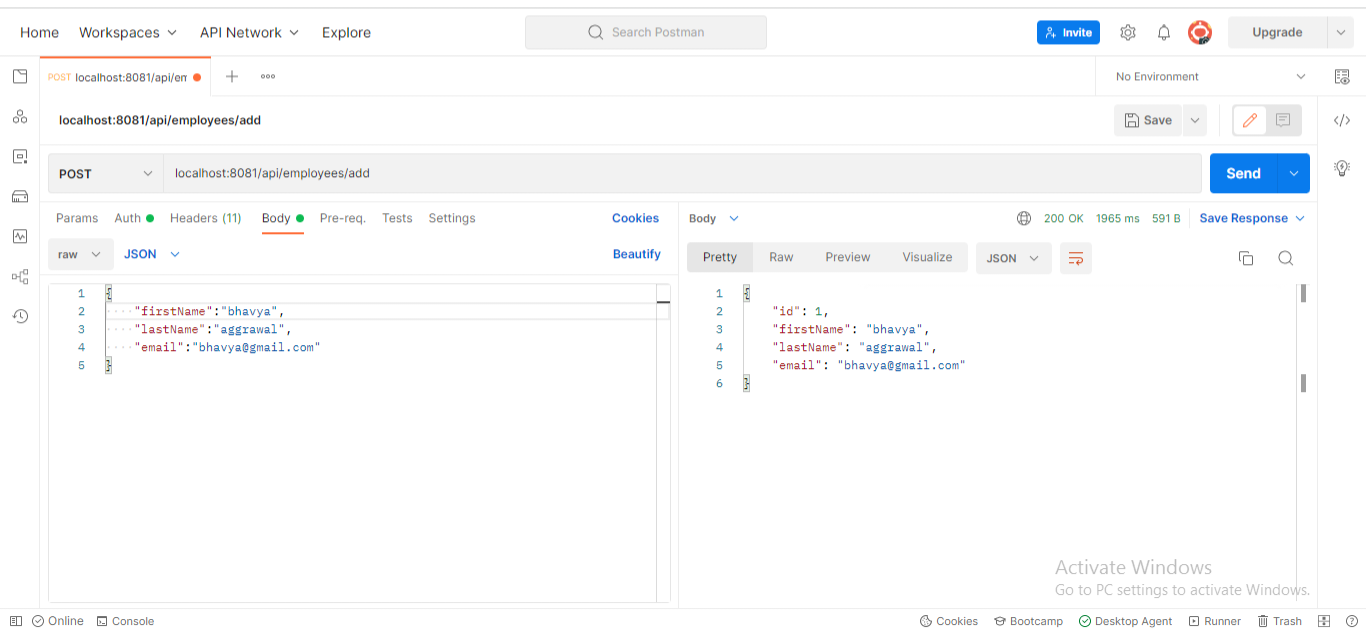
1. Your application should provide an endpoint to list all the employees stored in the database.

<URL:localhost:8081/api/employees/list>



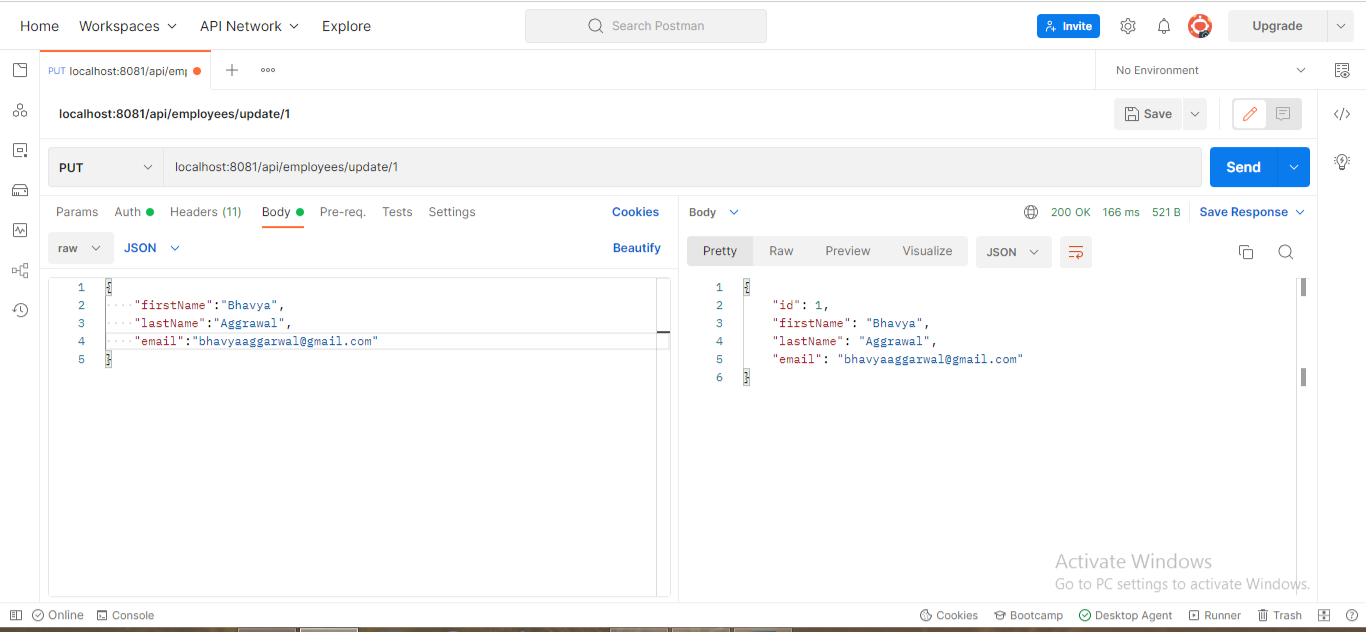
1. Now your application should be able to add employees data in the db if and only if the authenticated user is **ADMIN**.

<URL:localhost:8081/api/employees/add>



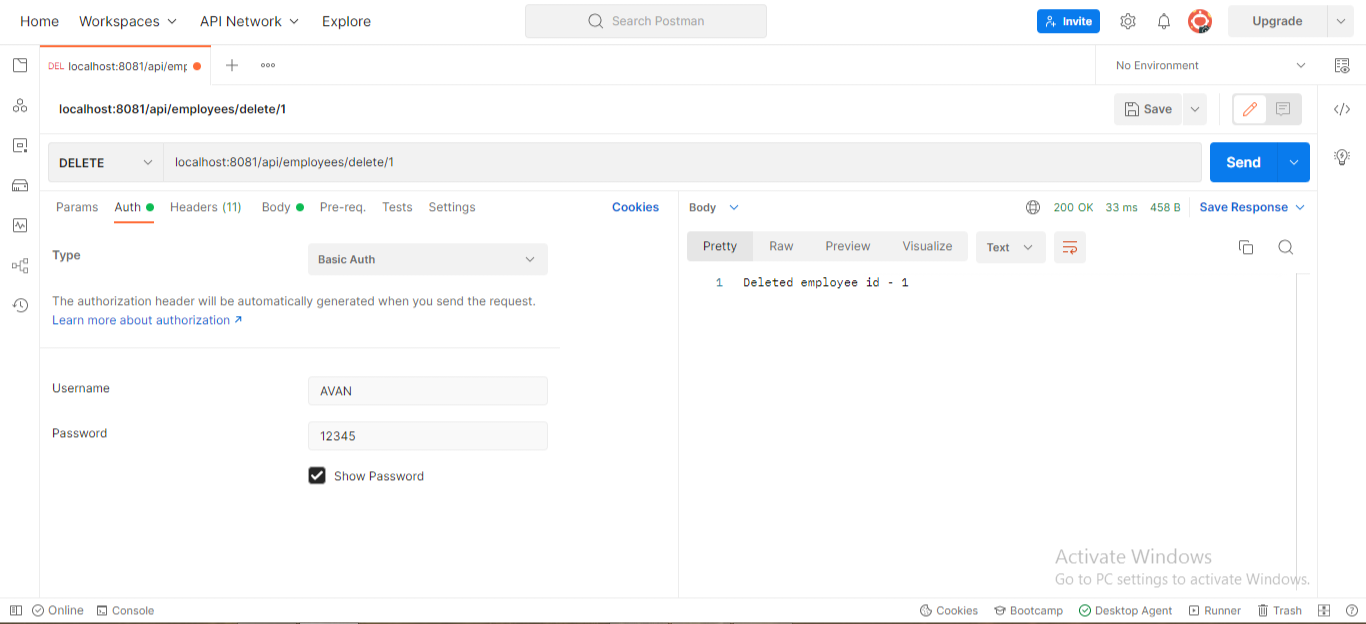
1. Your application should provide an endpoint to update an existing employee record with the given updated json object. <URL:localhost:8081/api/employees/update/1>.

In the process the employee “bhavya aggarwal” is updated to “Bhavya Aggarwal” with the email updated from “bhavya@gmail.com” to “bhavyaaggarwal@gmail.com”.



1. Your application should also provide an endpoint to delete an existing employee record based on the id of the employee (1 is the employee-Id).

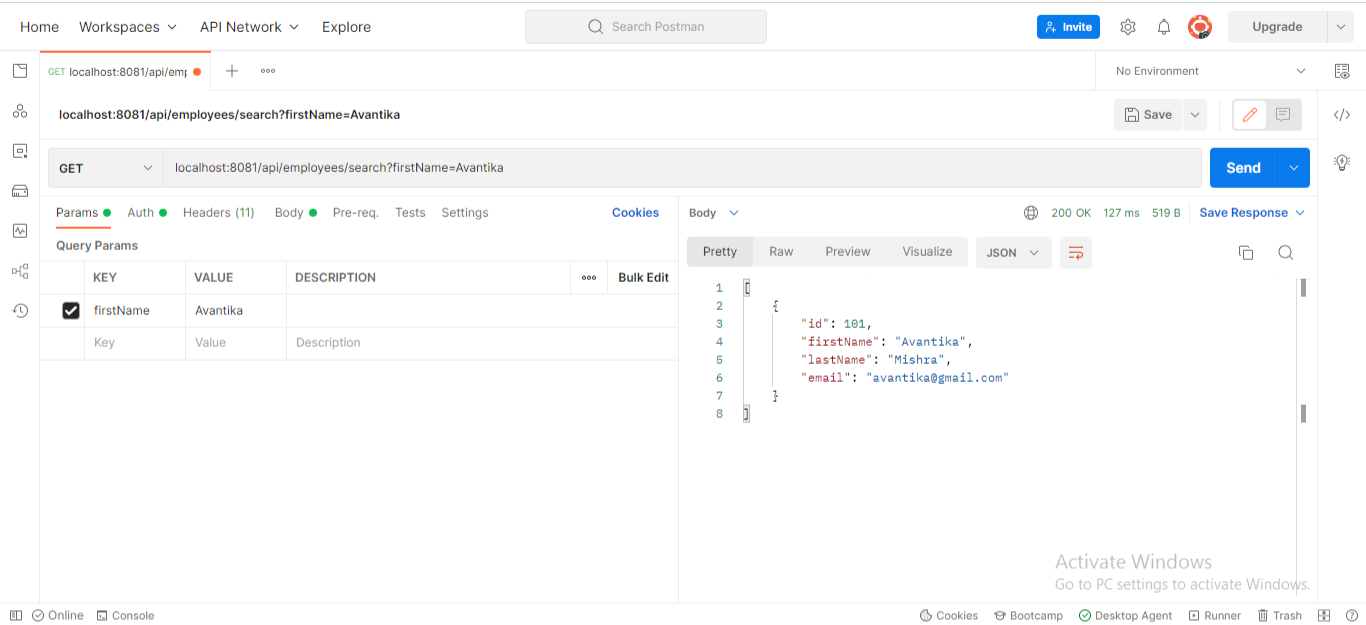
<URL:localhost:8081/api/employees/delete/1>



1. Your application should provide an endpoint to fetch employee by his/her first name and if found more than one record then list them all.

If the firstName is taken as Avantika in the parameters requested, we have the following URL:

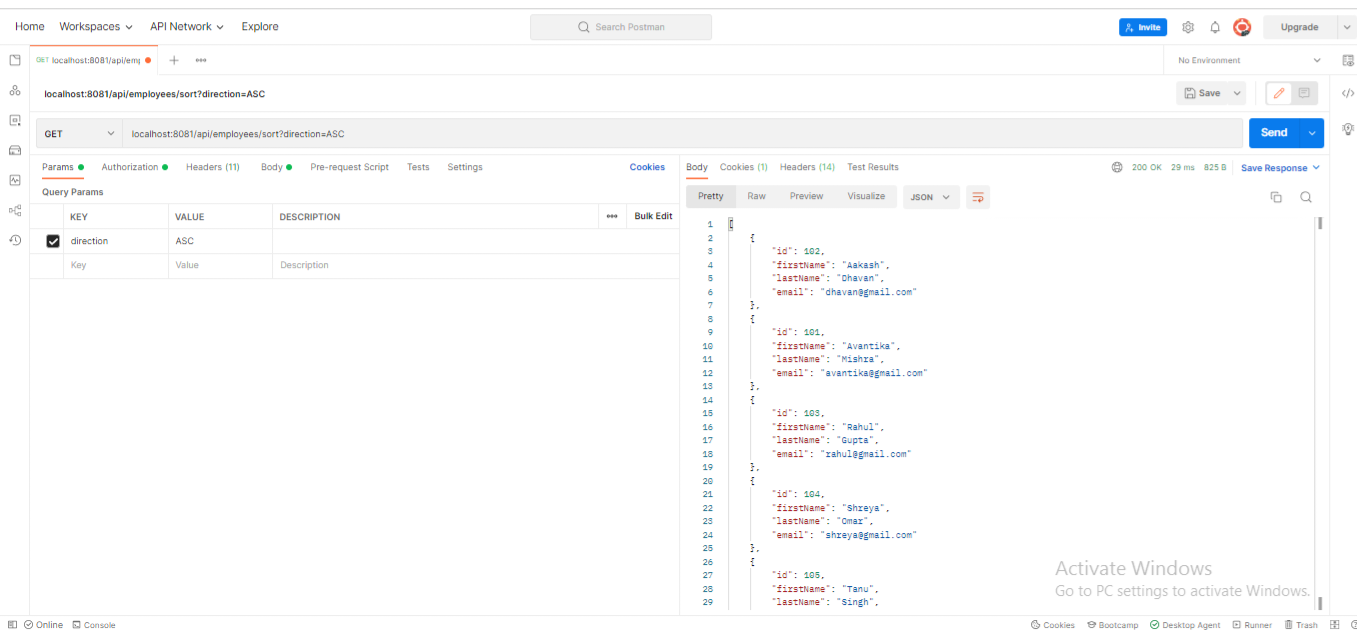
<URL:localhost:8081/api/employees/search?firstName=Avantika>



1. Your application should be able to list all employee records sorted on their first name in either ascending order or descending or ascending.

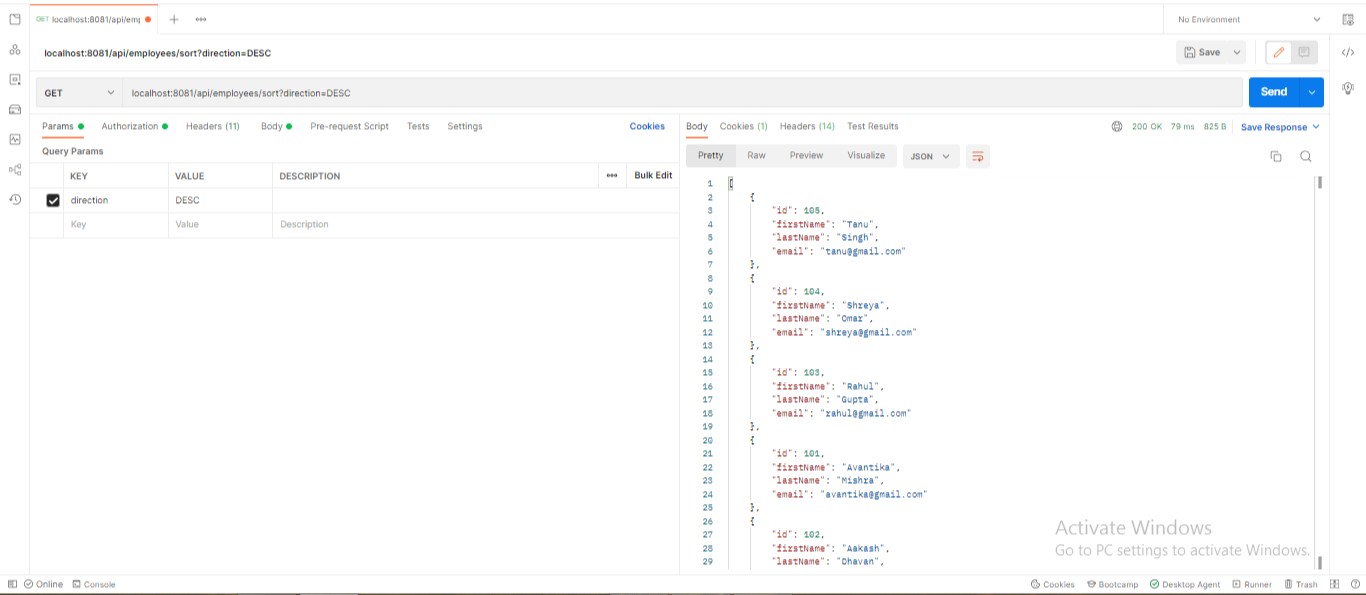
In the following command, we have the sorting direction given in requested parameters as of ascending,

<URL:localhost:8081/api/employees/sort?direction=ASC>



In the following command, we have the sorting direction given in requested parameters as of descending,

<URL:localhost:8081/api/employees/sort?direction=DESC>



The following screenshots show that the second kind of user with the role of a “USER” will not be able to perform any operation on the database and can only view the employees.

