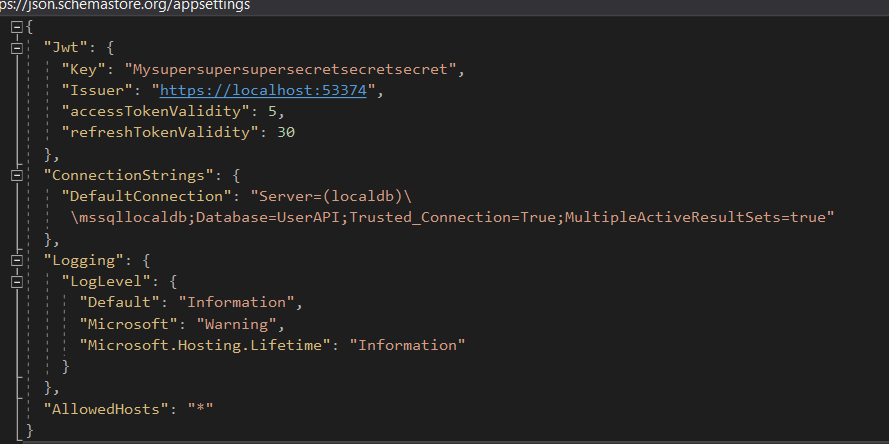
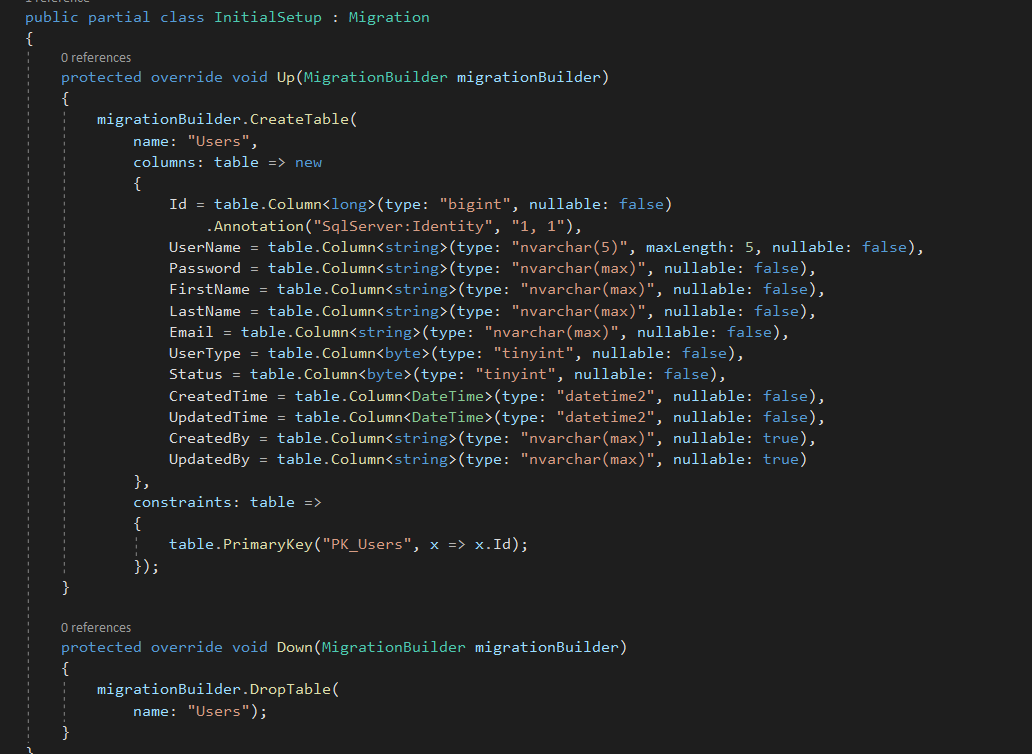
1. Setup
   1. Settings

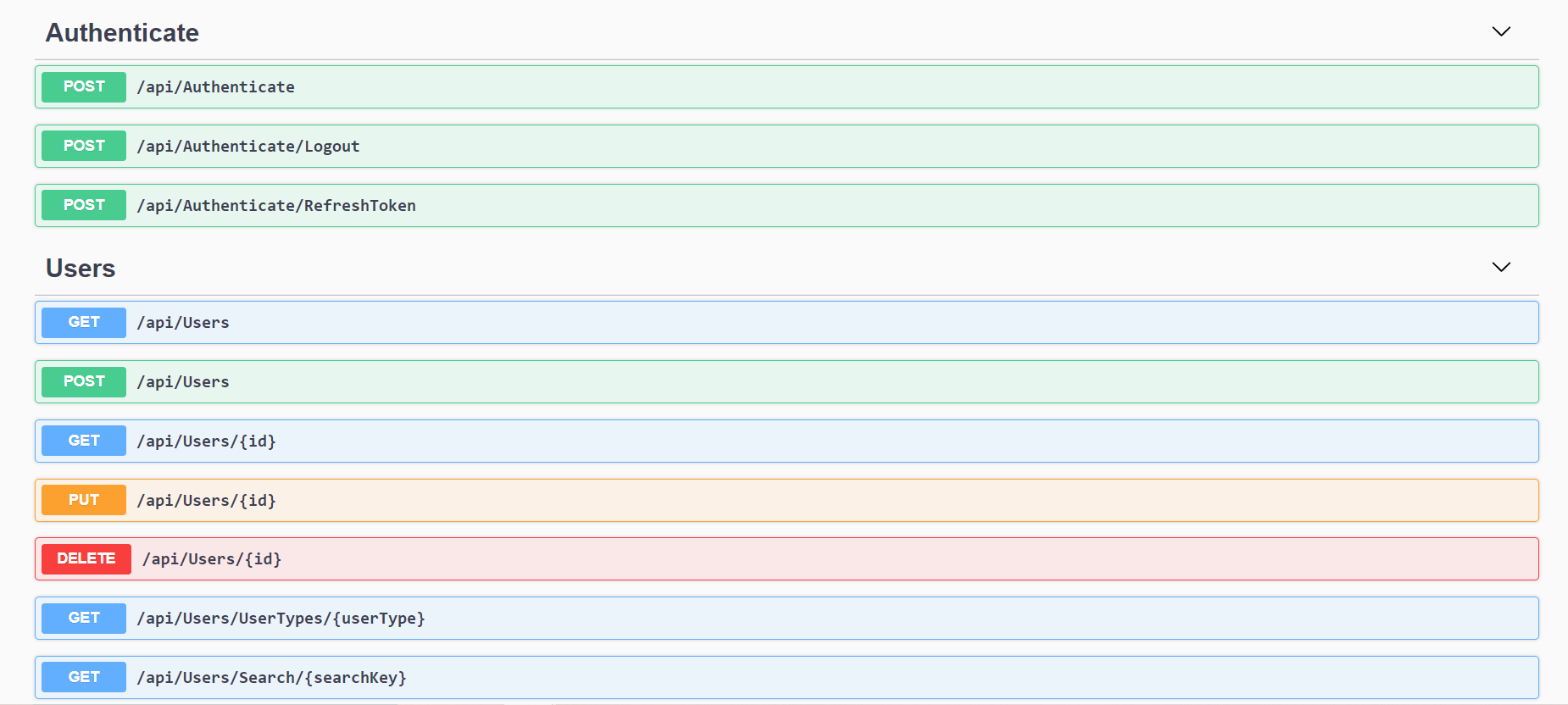


* 1. Migration



* 1. Run Update-Database in package console to create the data structures to the database.
  2. Swagger

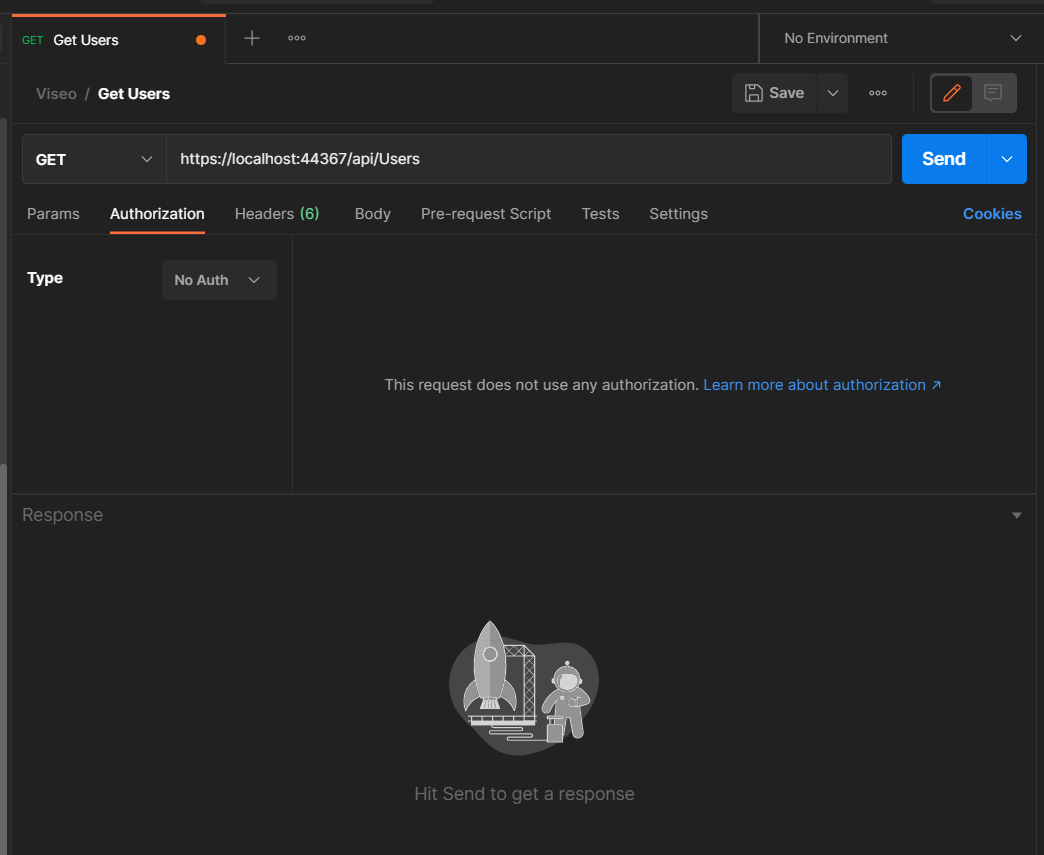
<https://localhost:44367/swagger/index.html>

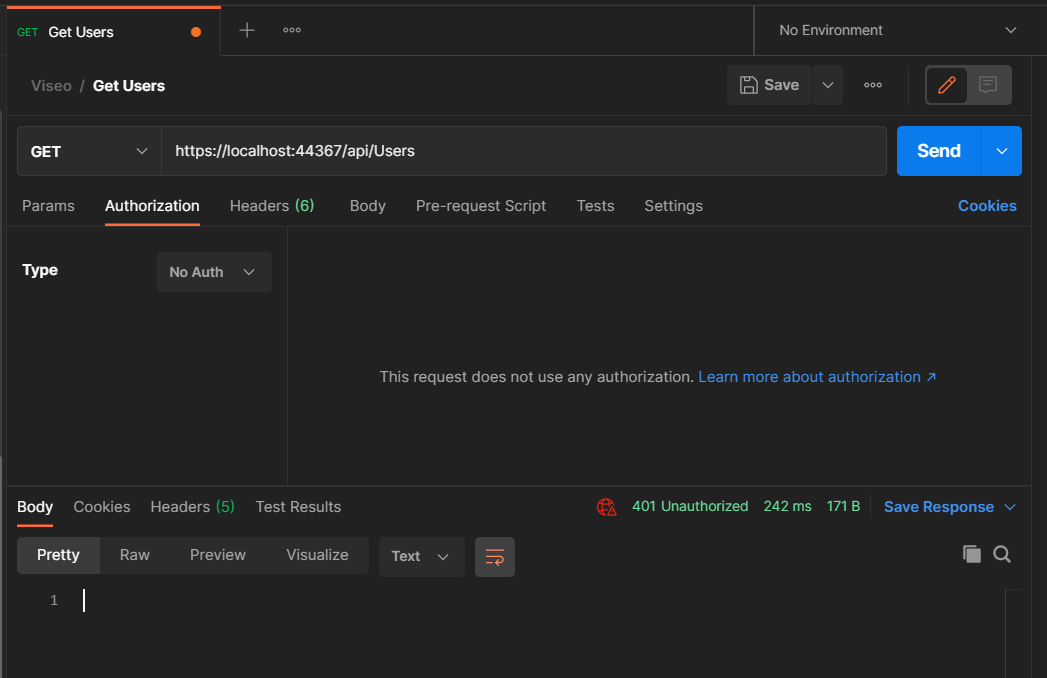


1. Authentication - <https://localhost:44367/api/Authenticate>
   1. User is not authenticated.

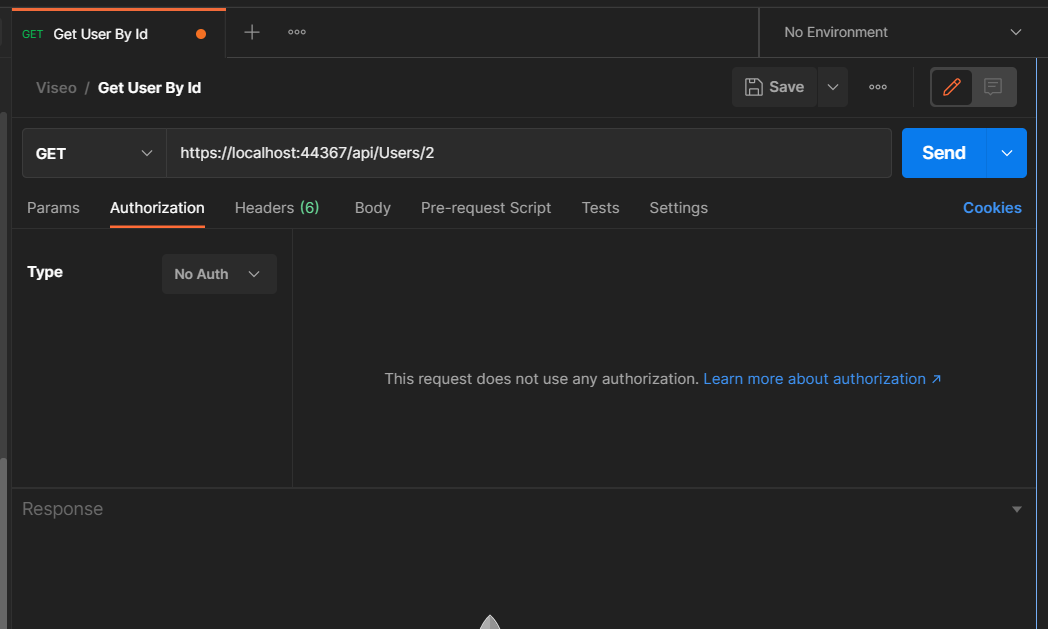
Expected Result: User should have no access and is unauthorized.

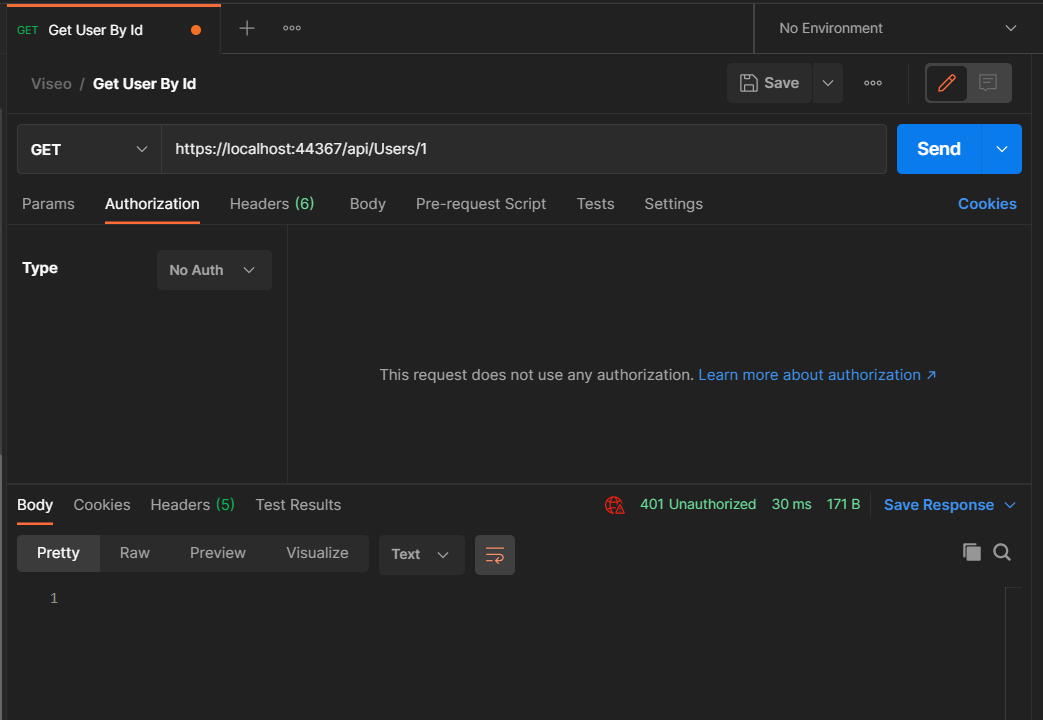
* + 1. Get Users - <https://localhost:44367/api/Users>



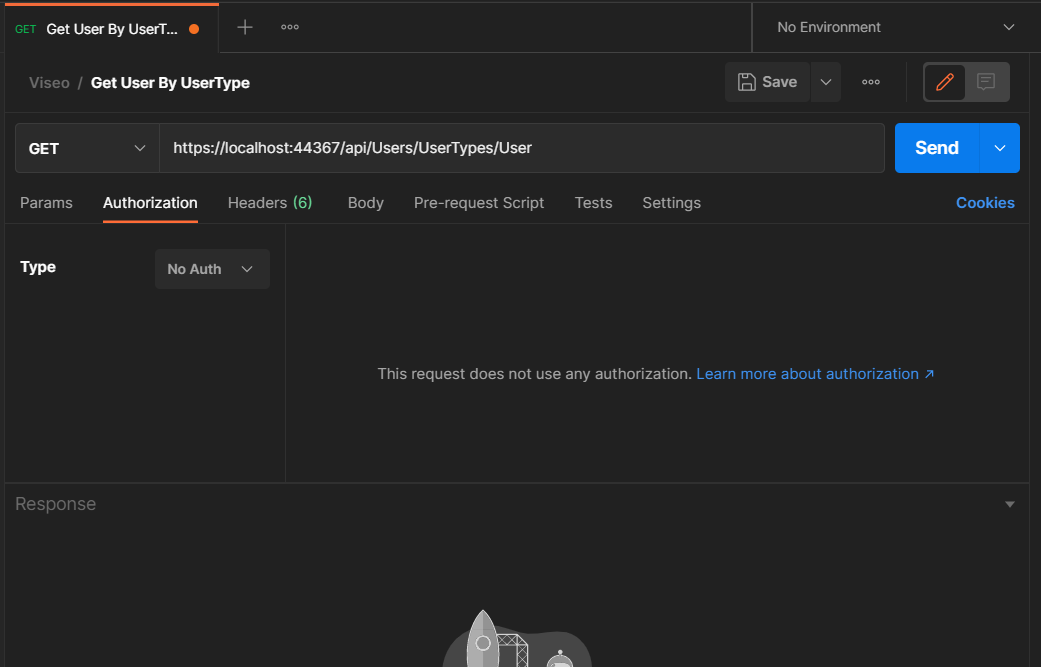


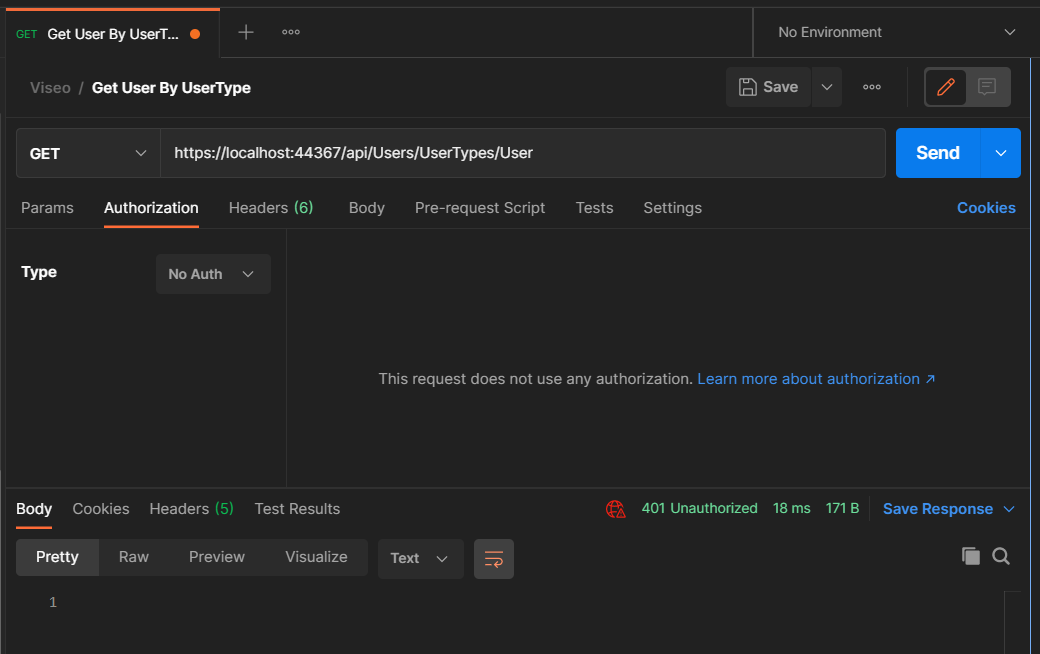
* + 1. Get User By Id



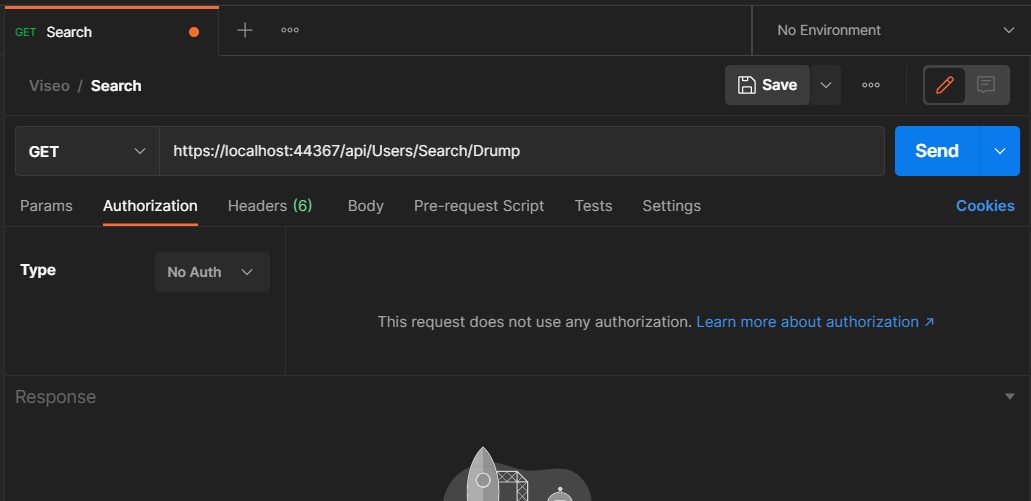


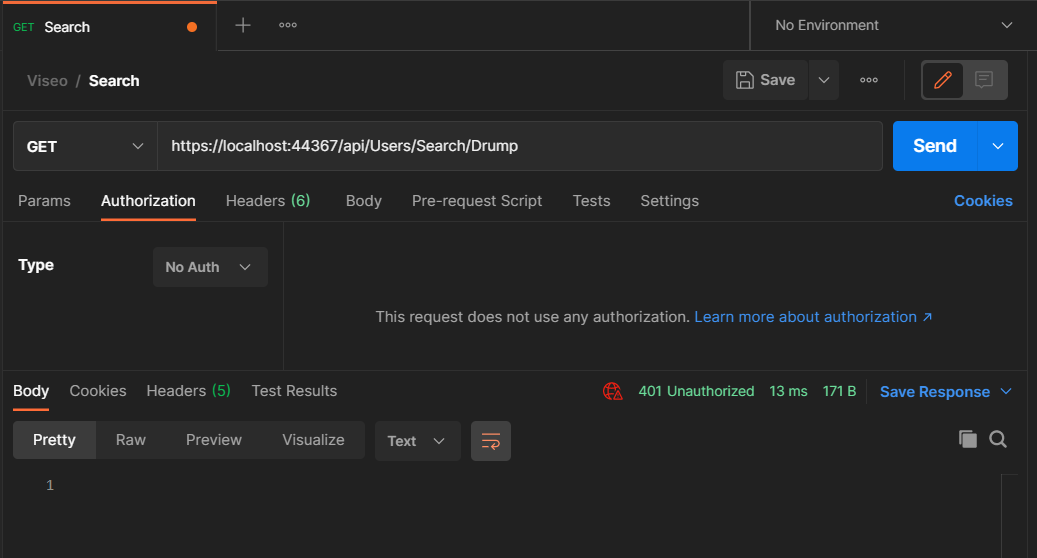
* + 1. Get User by User Type - <https://localhost:44367/api/Users/UserTypes/User>



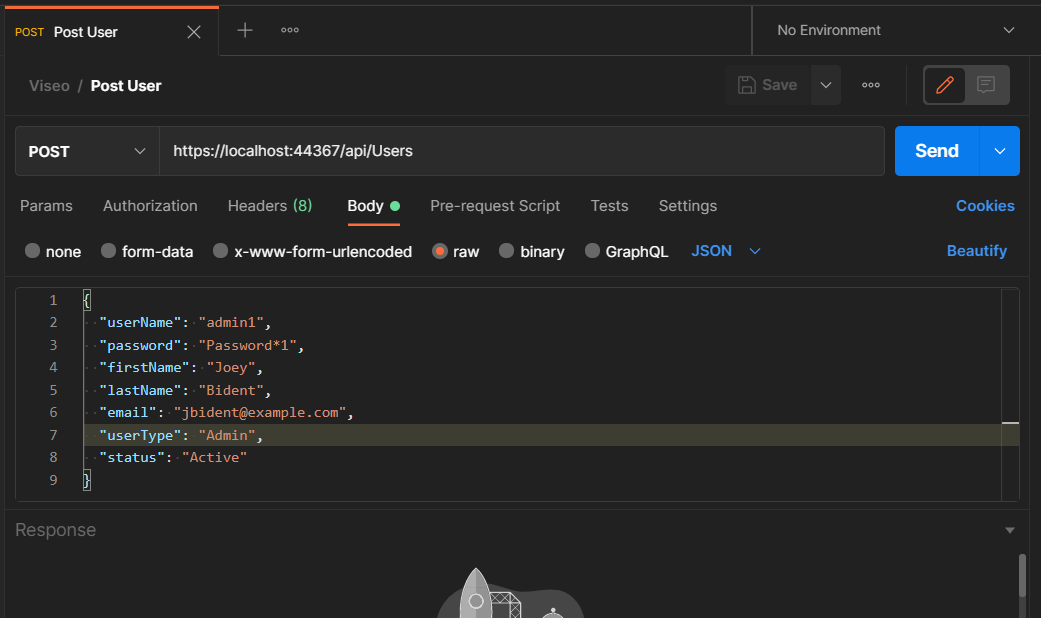


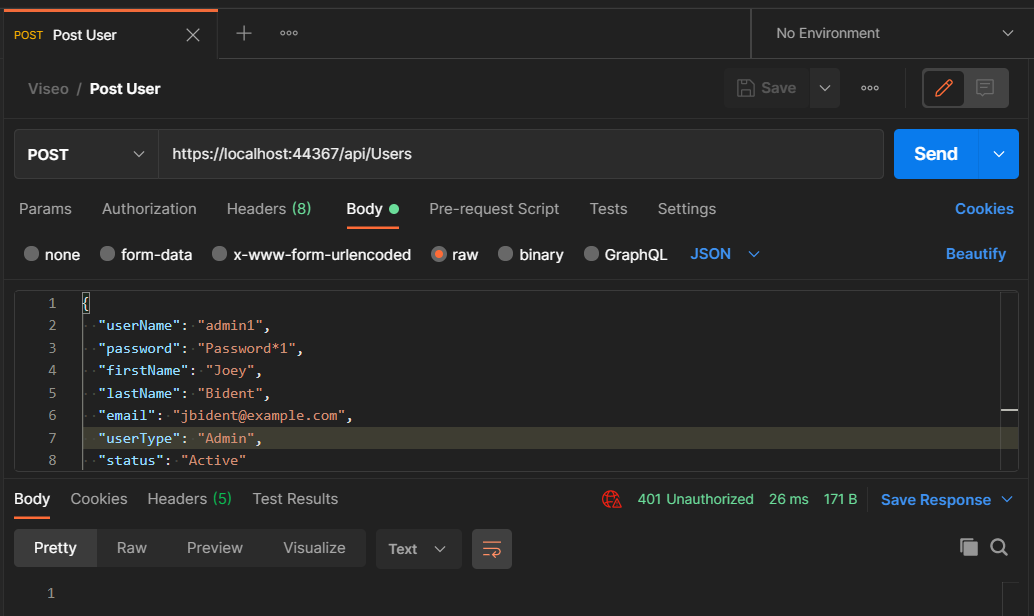
* + 1. Search - <https://localhost:44367/api/Users/Search/Drump>



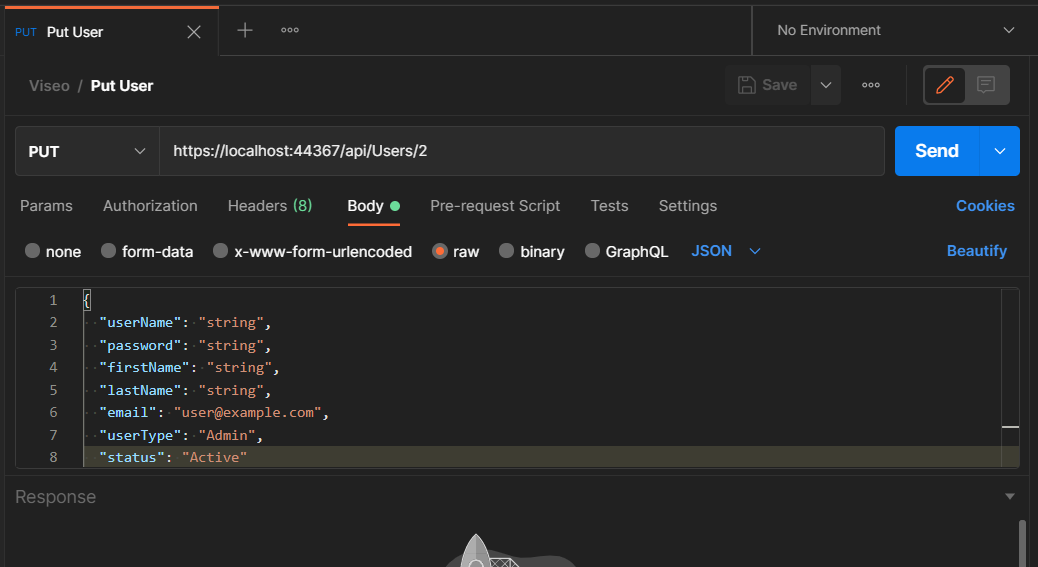


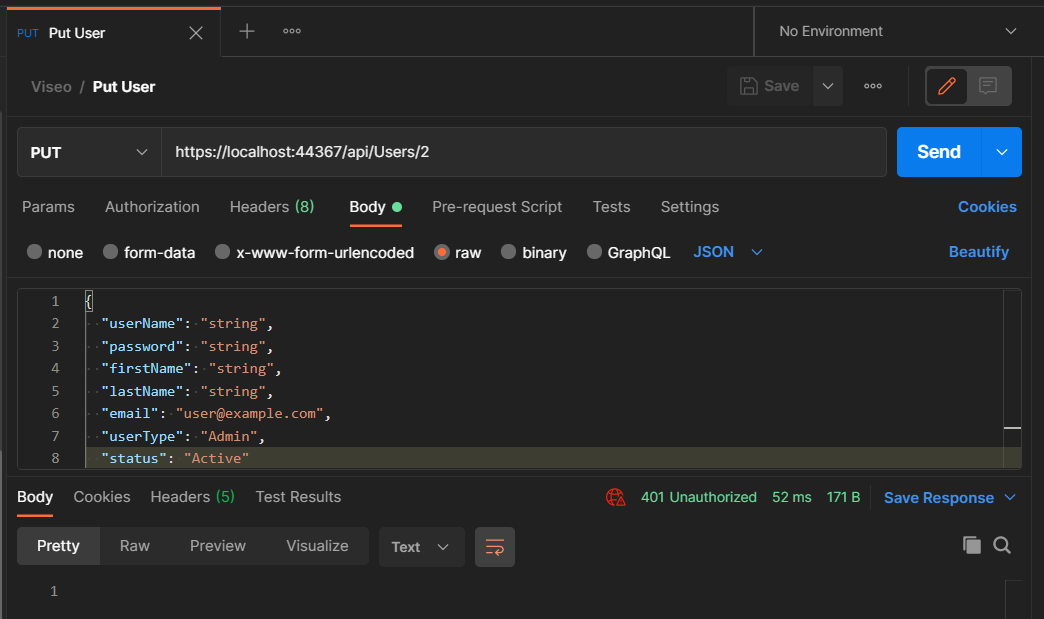
* + 1. Post User



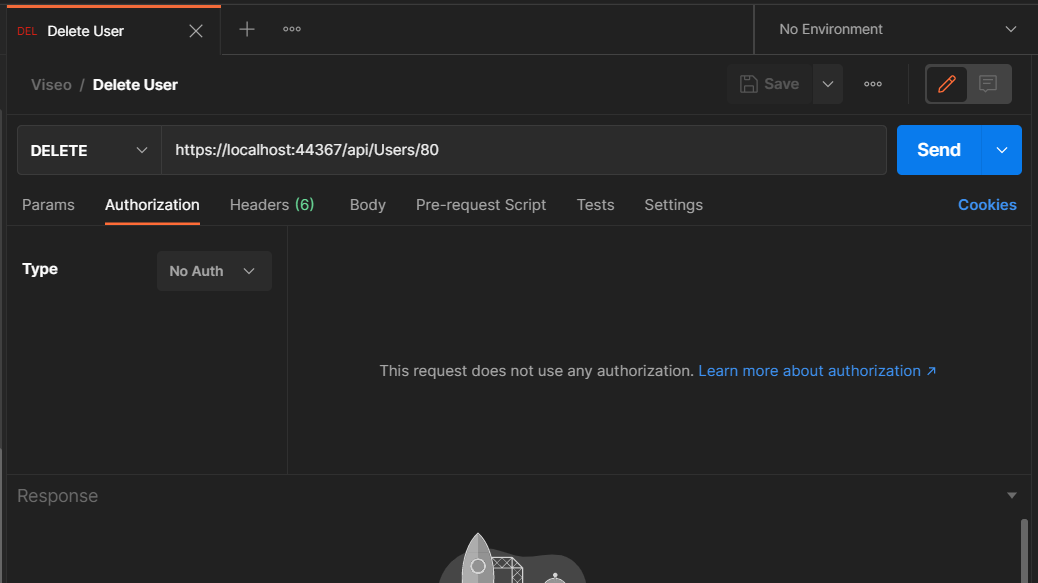


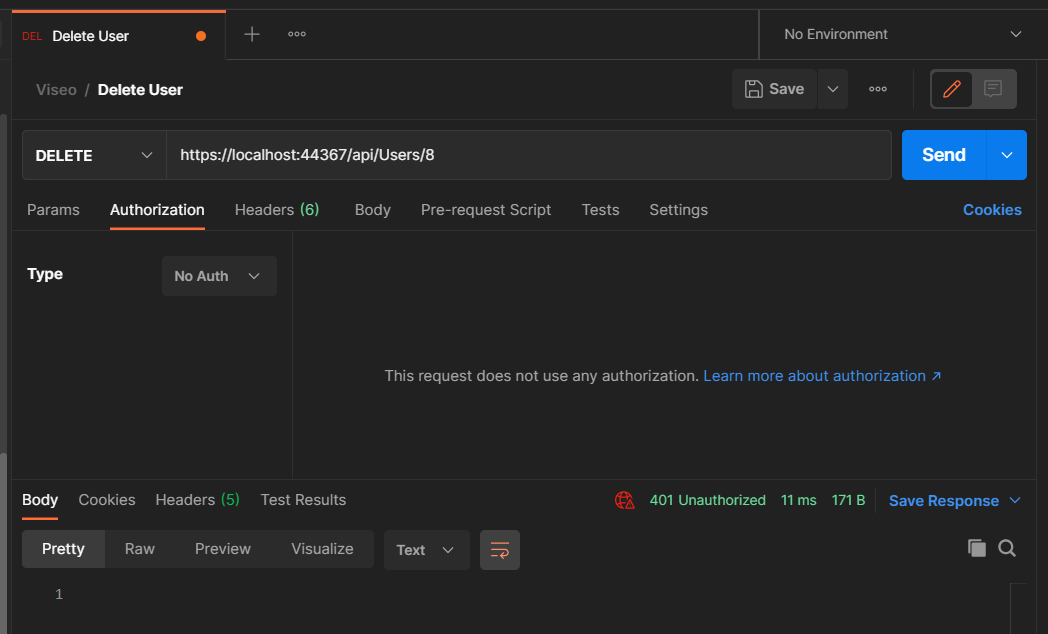
* + 1. Put User





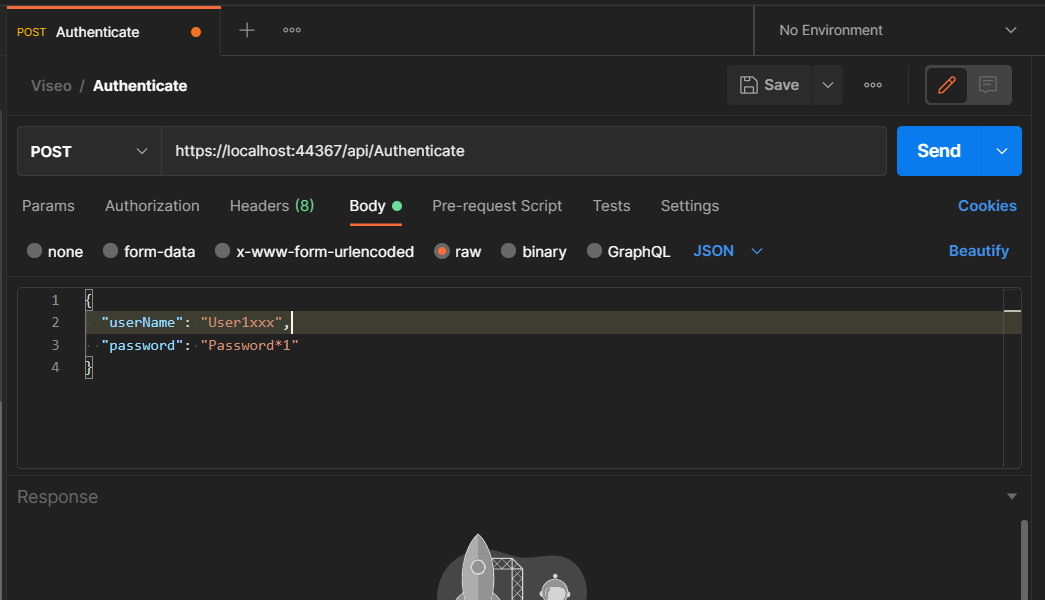
* + 1. Delete User





* 1. User enter invalid credential.

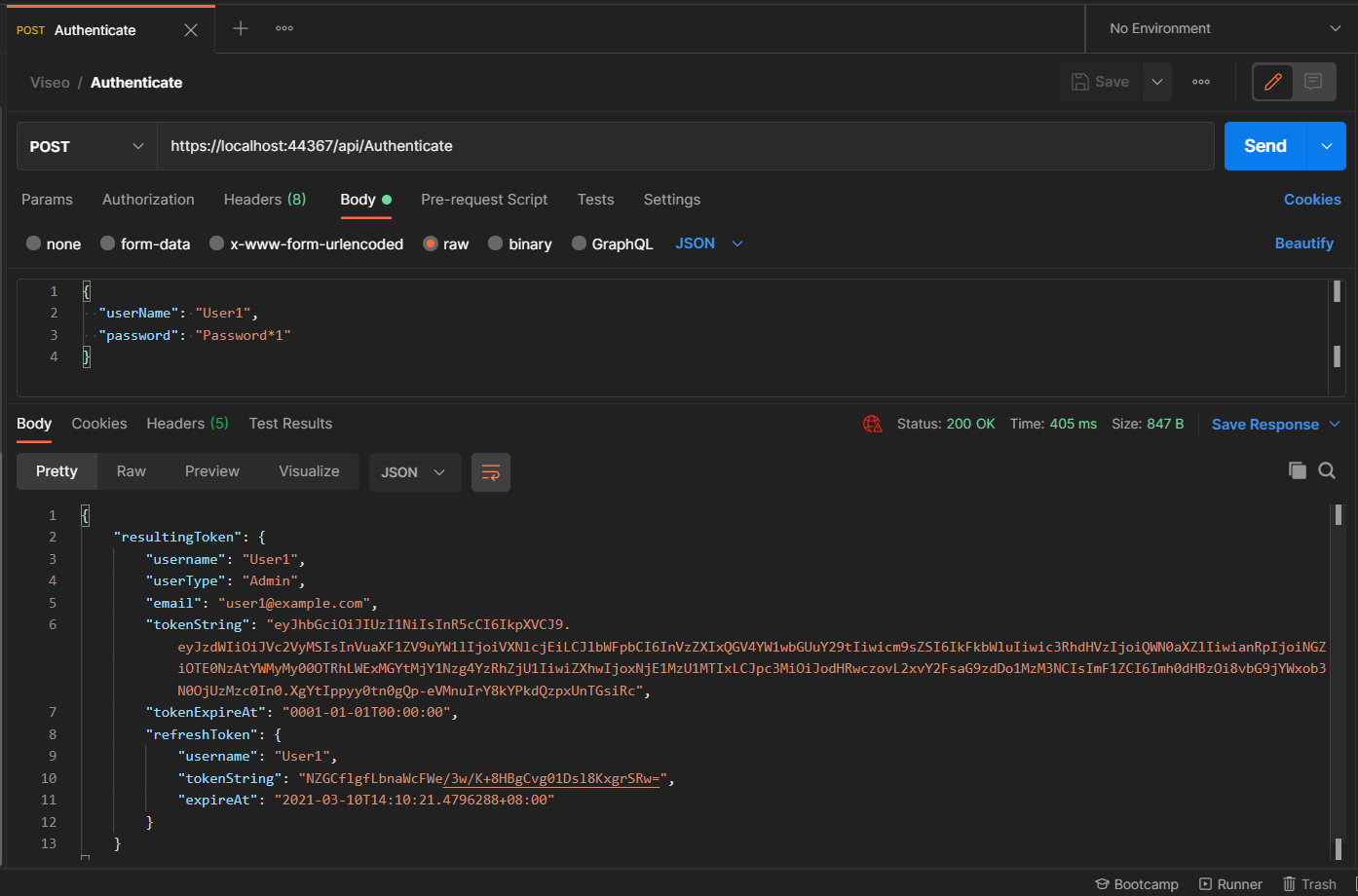
Expected Result: Notify the user that the credential is invalid.



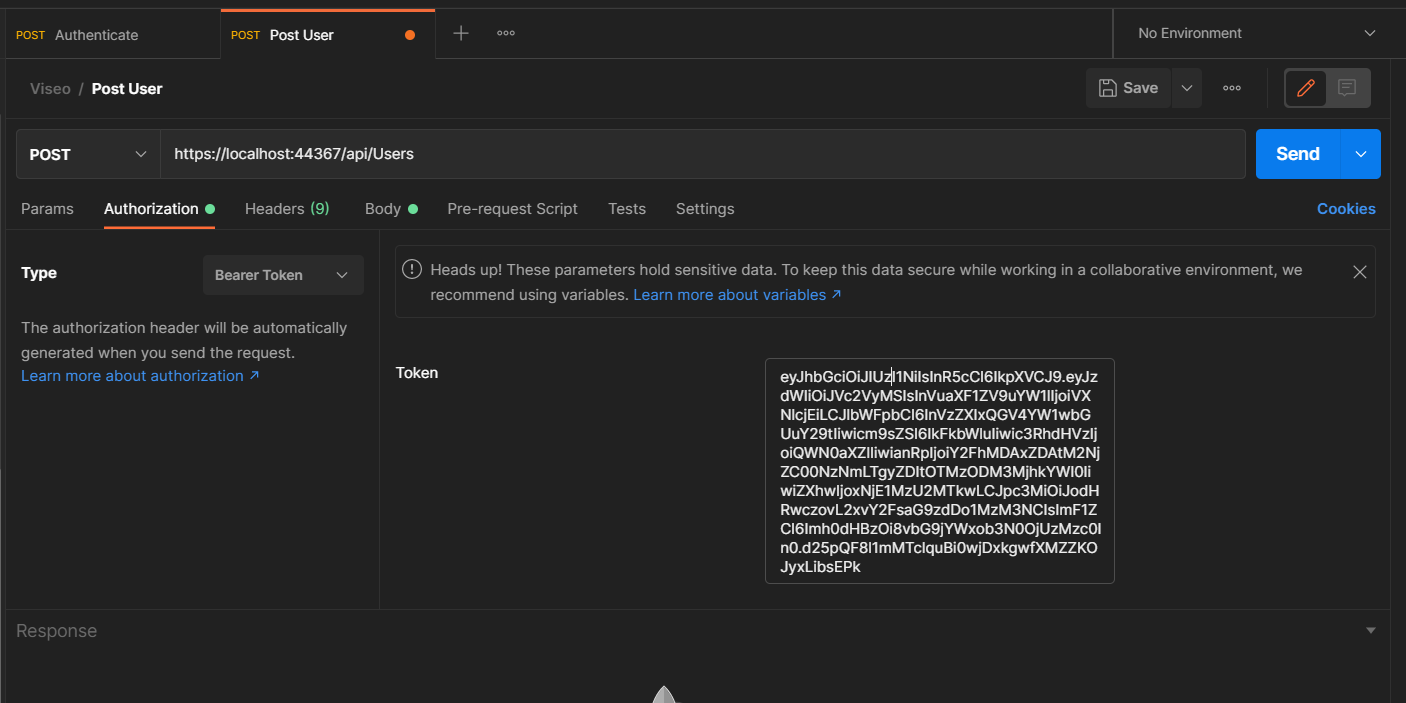


* 1. User is authenticated.

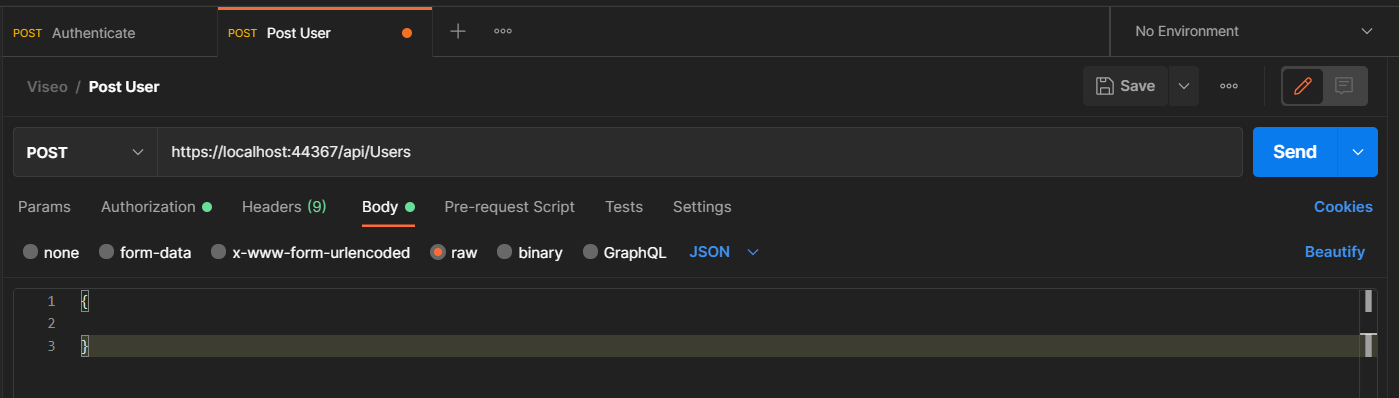
Expected Result: Return the details of the user including the access and refresh tokens.



1. Posting

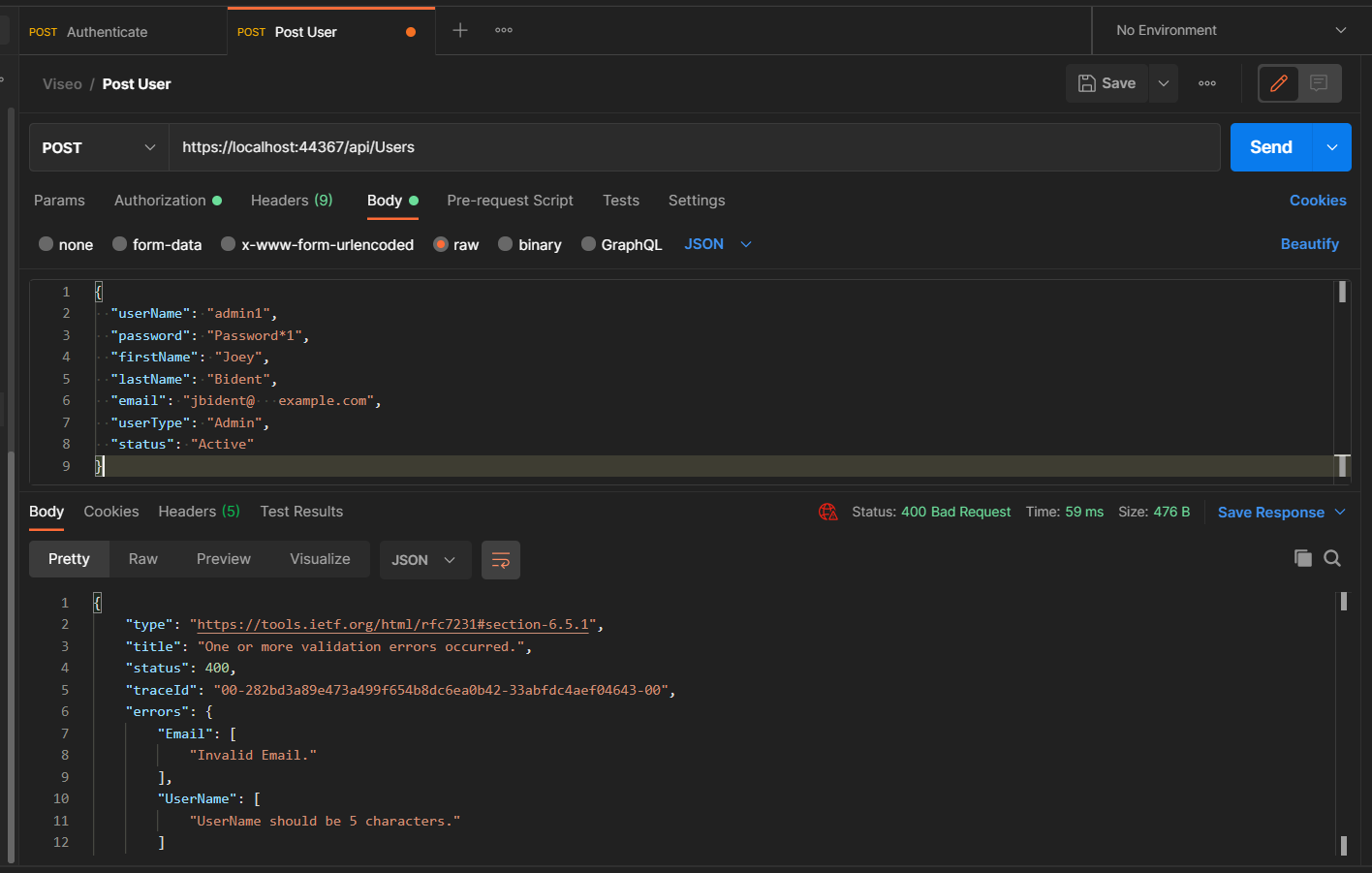


* 1. Validation
     1. All fields are required.

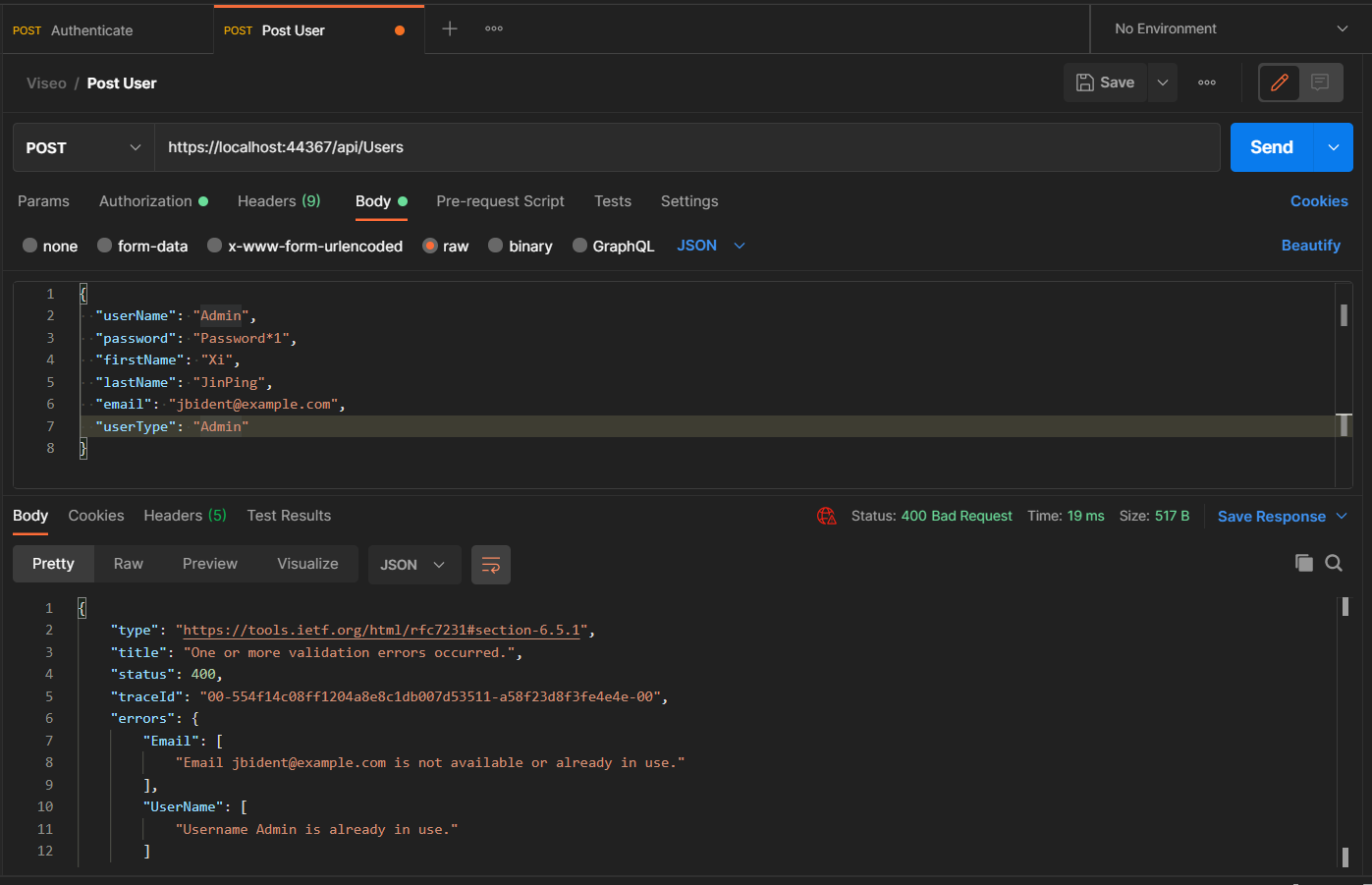




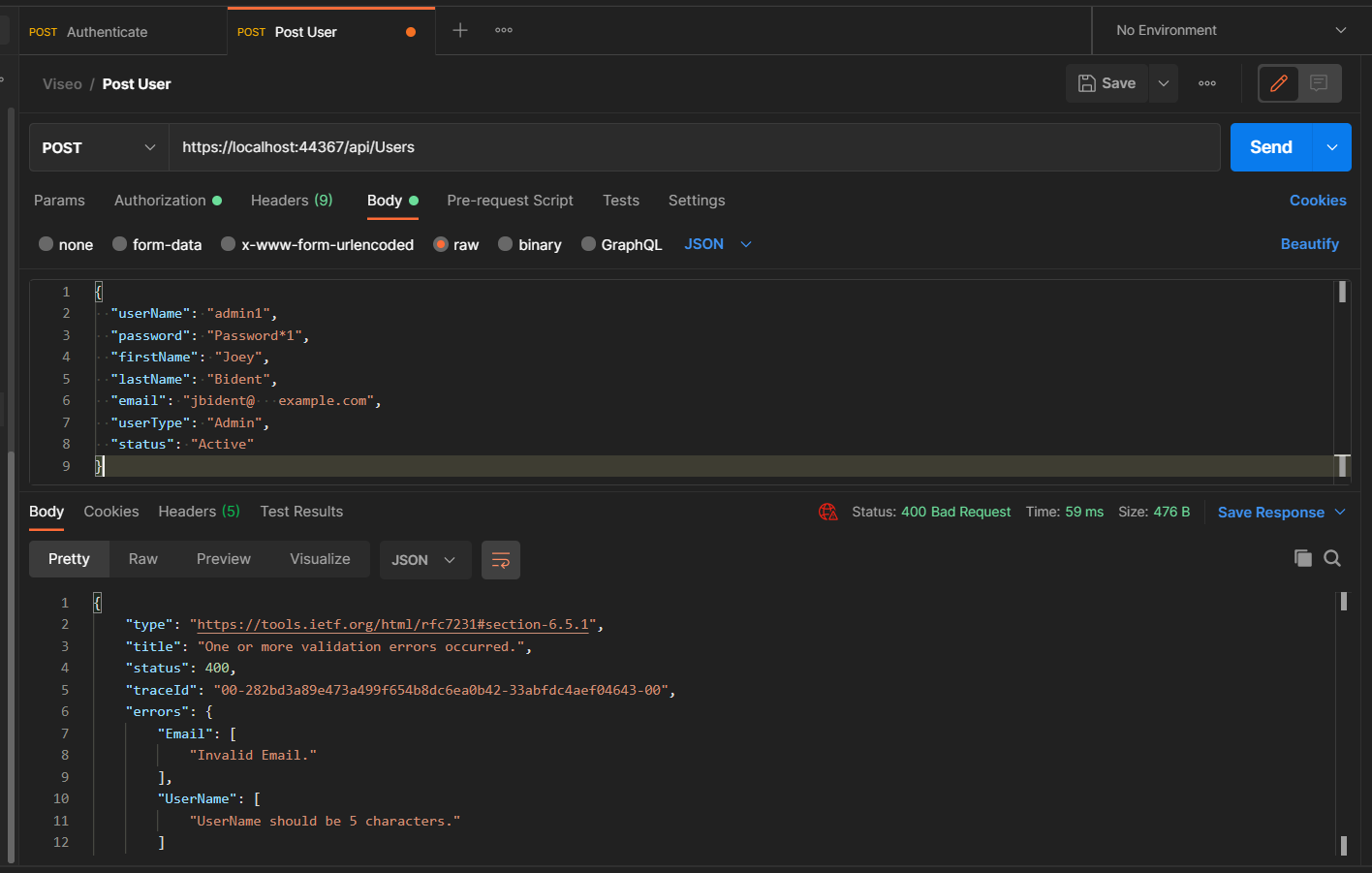
* + 1. Email should be valid.



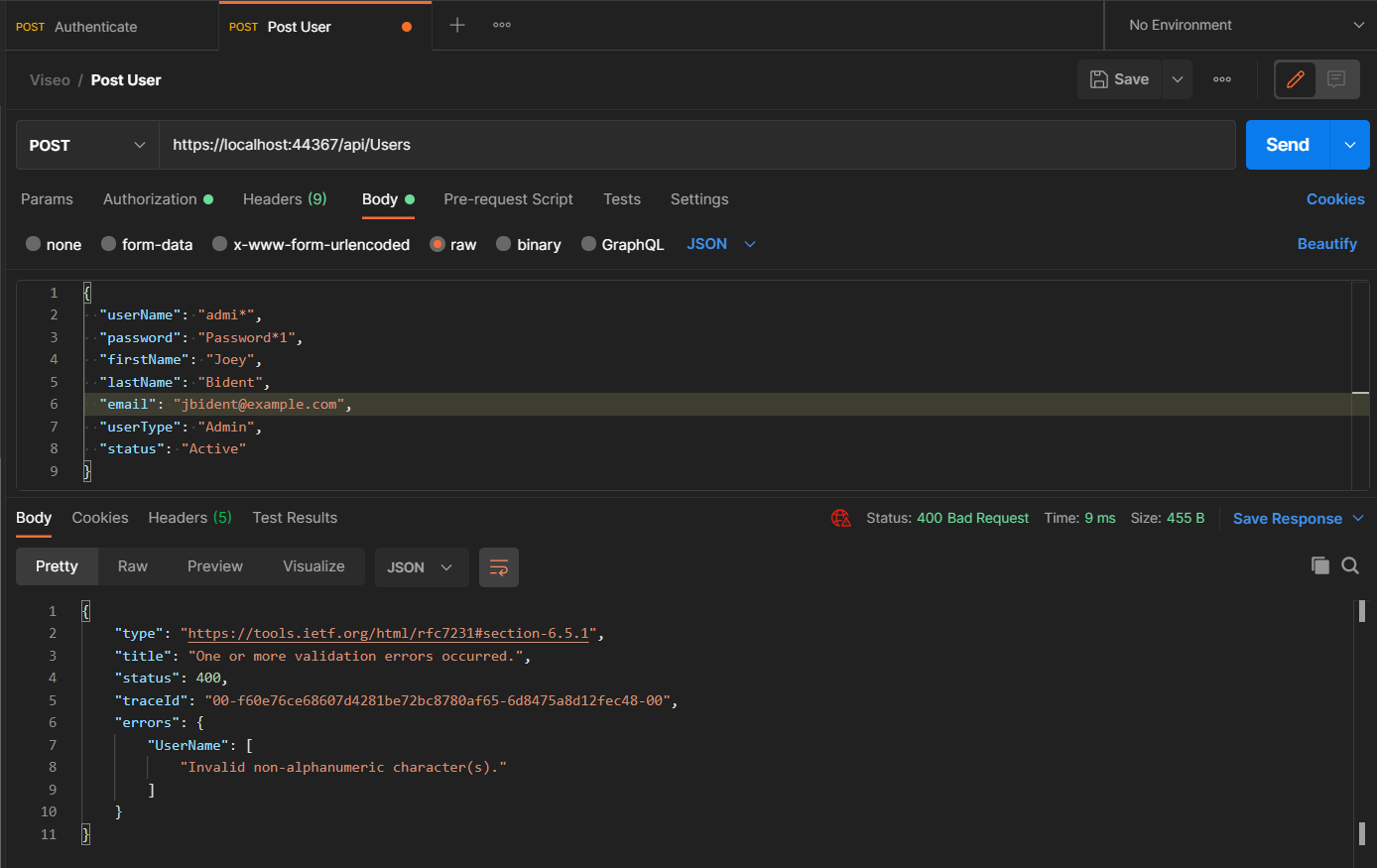
* + 1. Email should be unique.



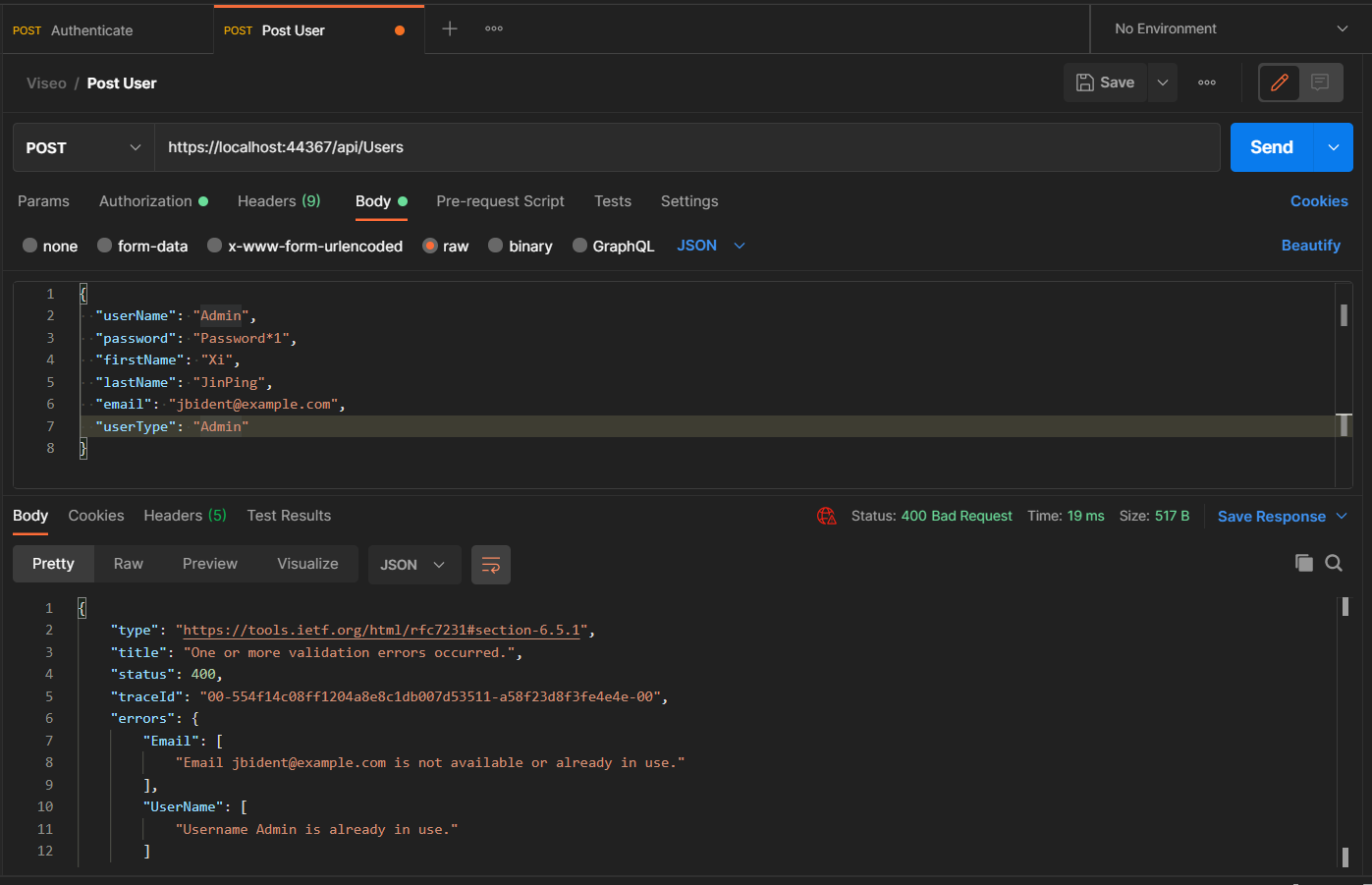
* + 1. Username should have a size of 5.



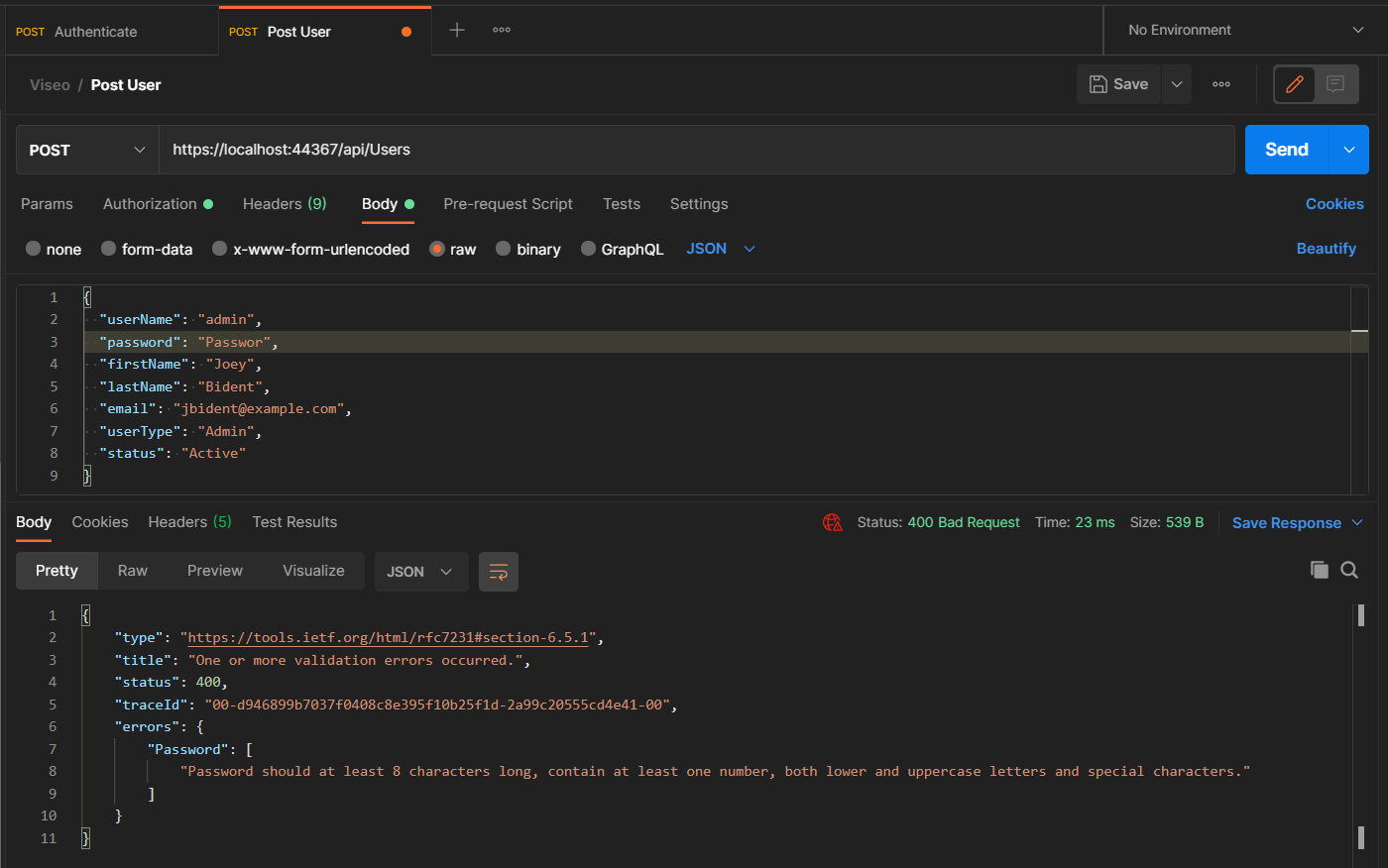
* + 1. Username should be alphanumeric.



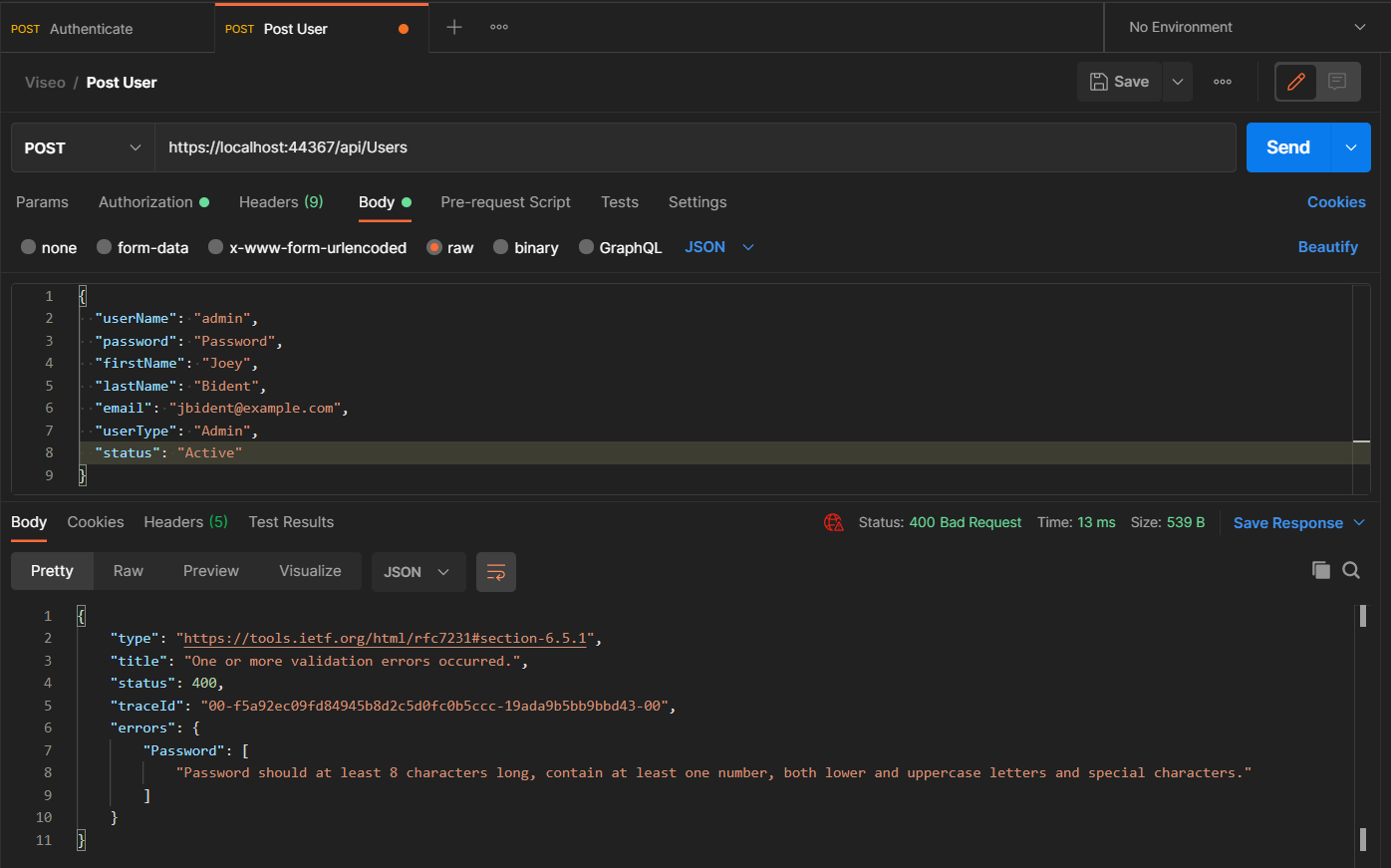
* + 1. Username should be unique.



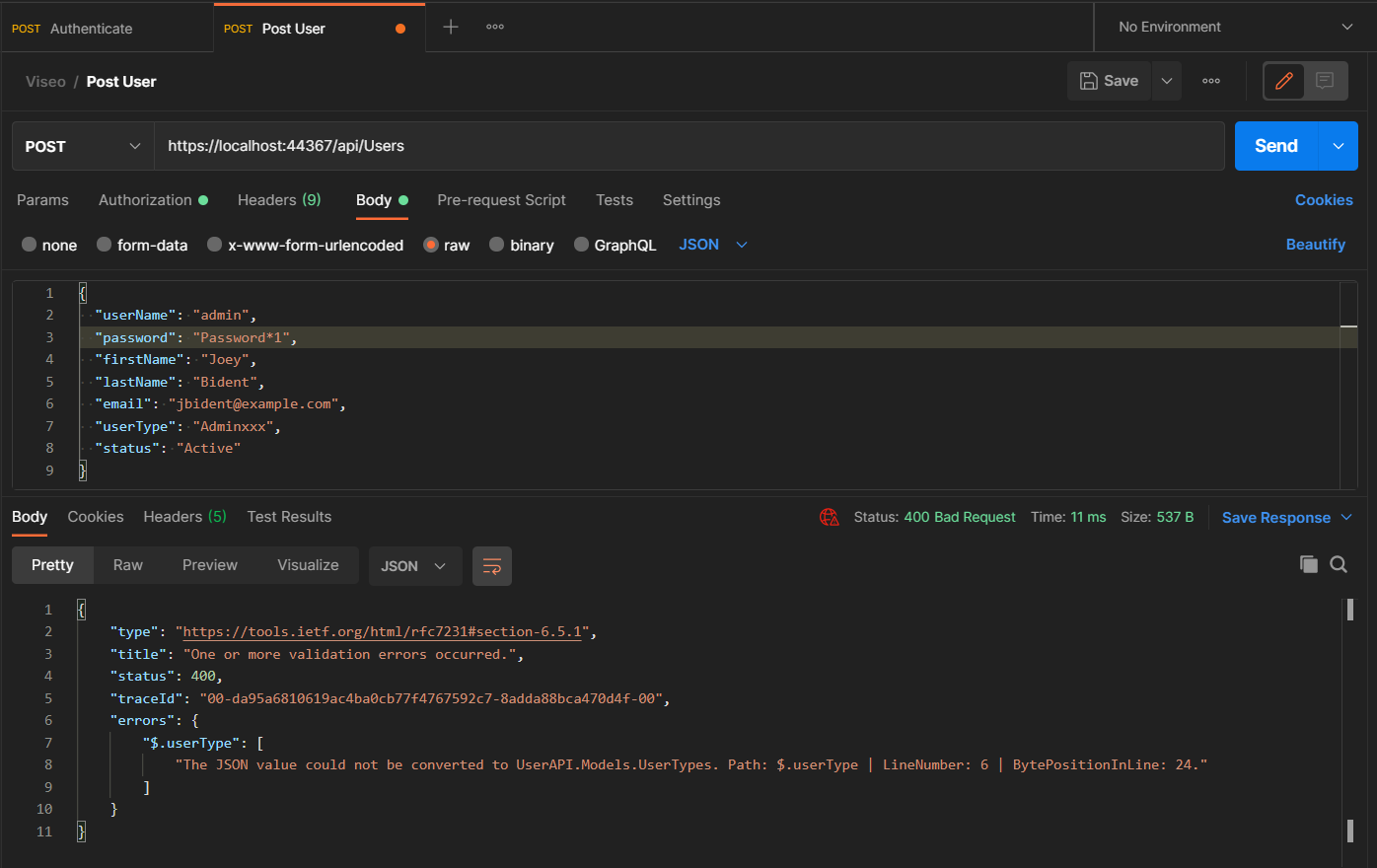
* + 1. Password should contain at least 8 characters.

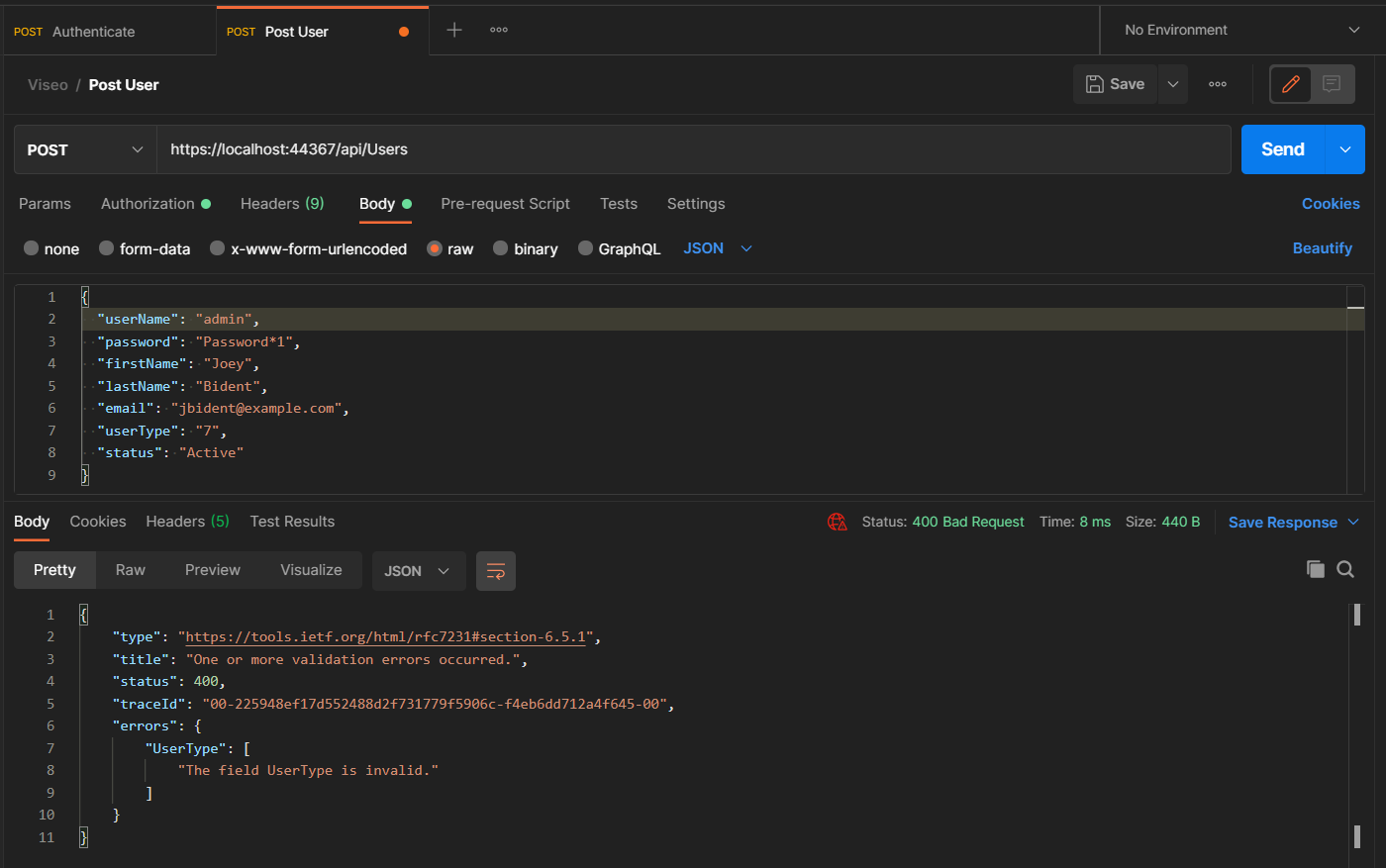


* + 1. Password should contain at least one number and both lower and uppercase letters and special characters.

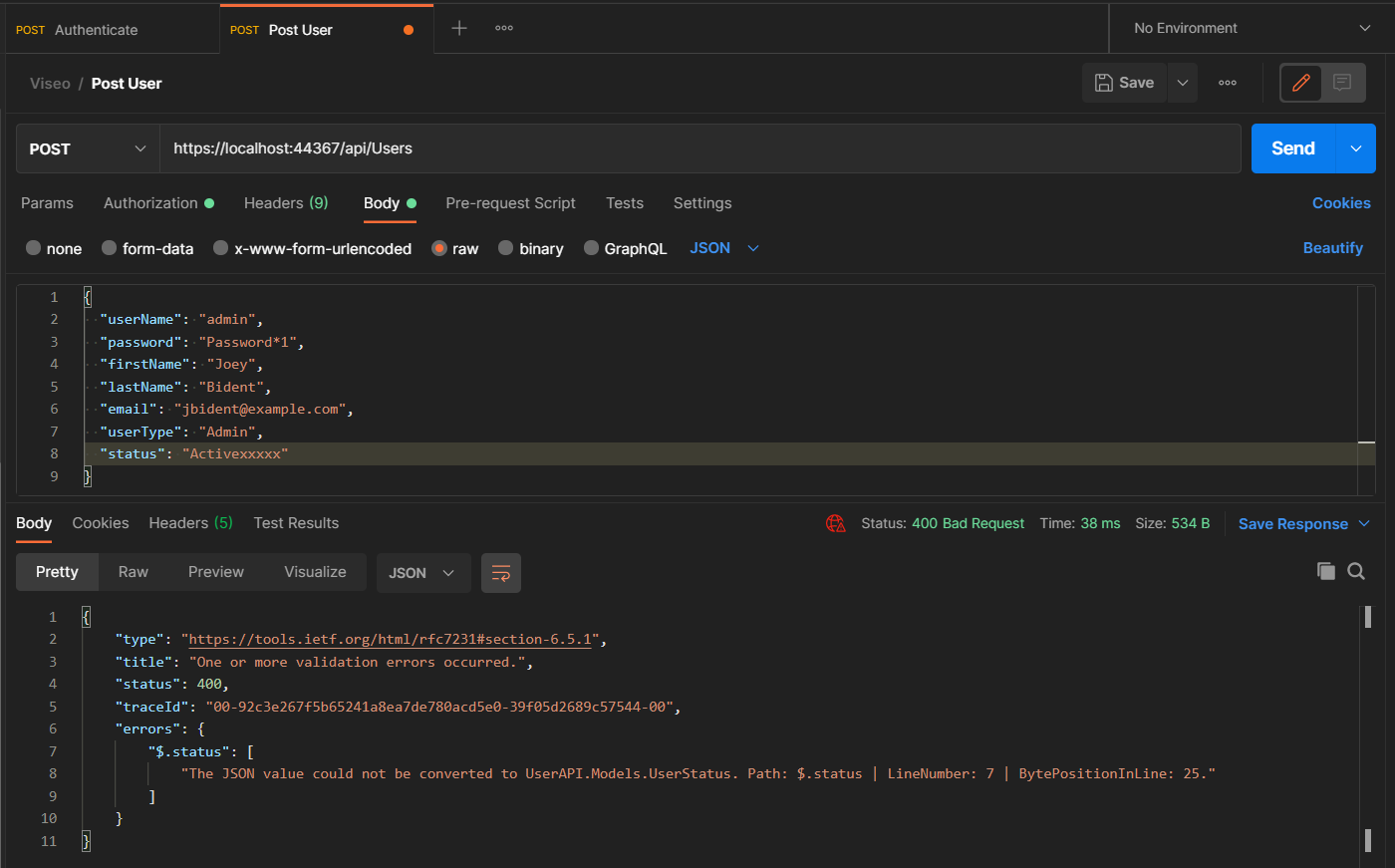


* + 1. User Type value should be correct.

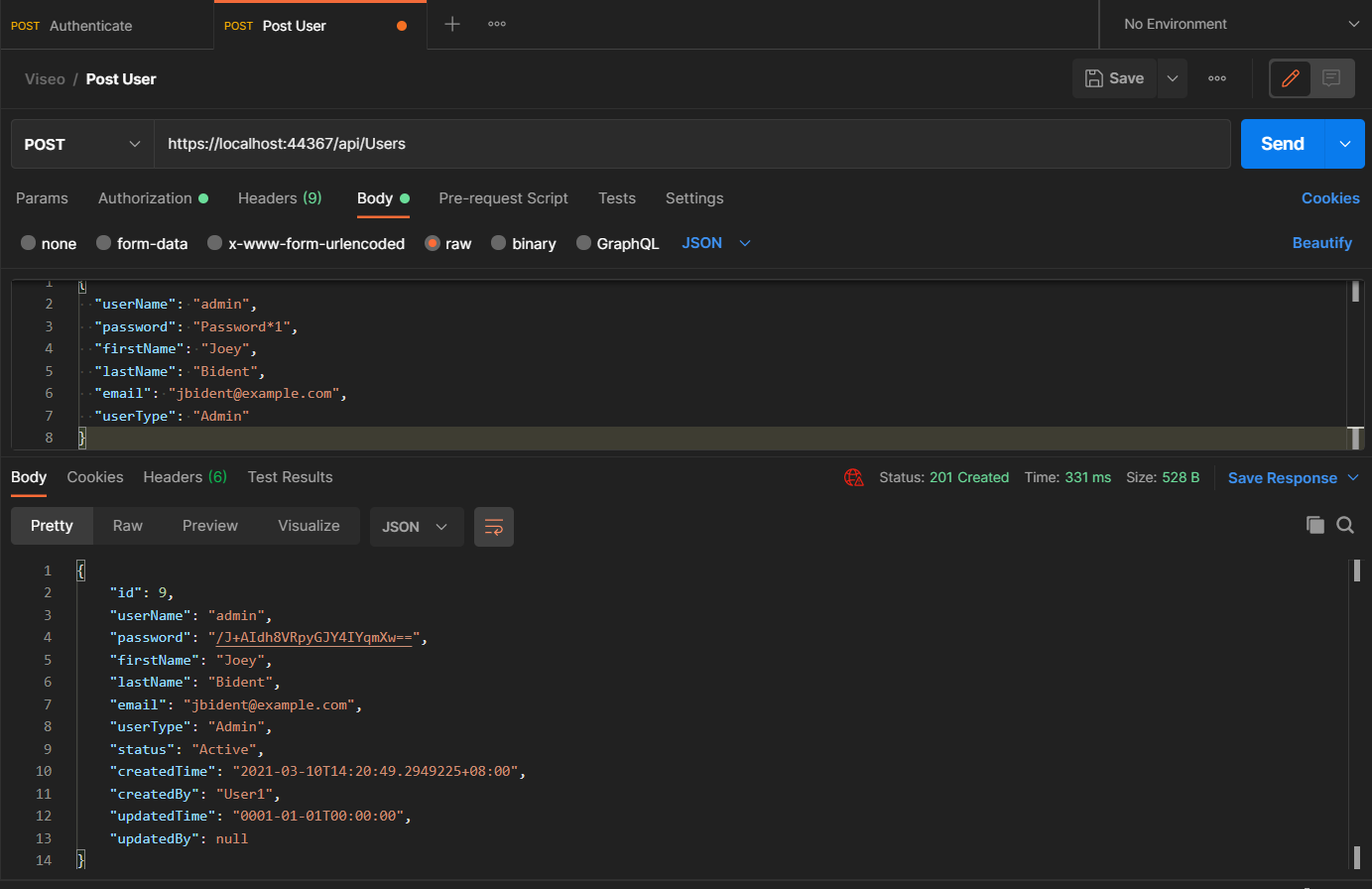




* + 1. Status value should be correct.



* 1. Successful save.



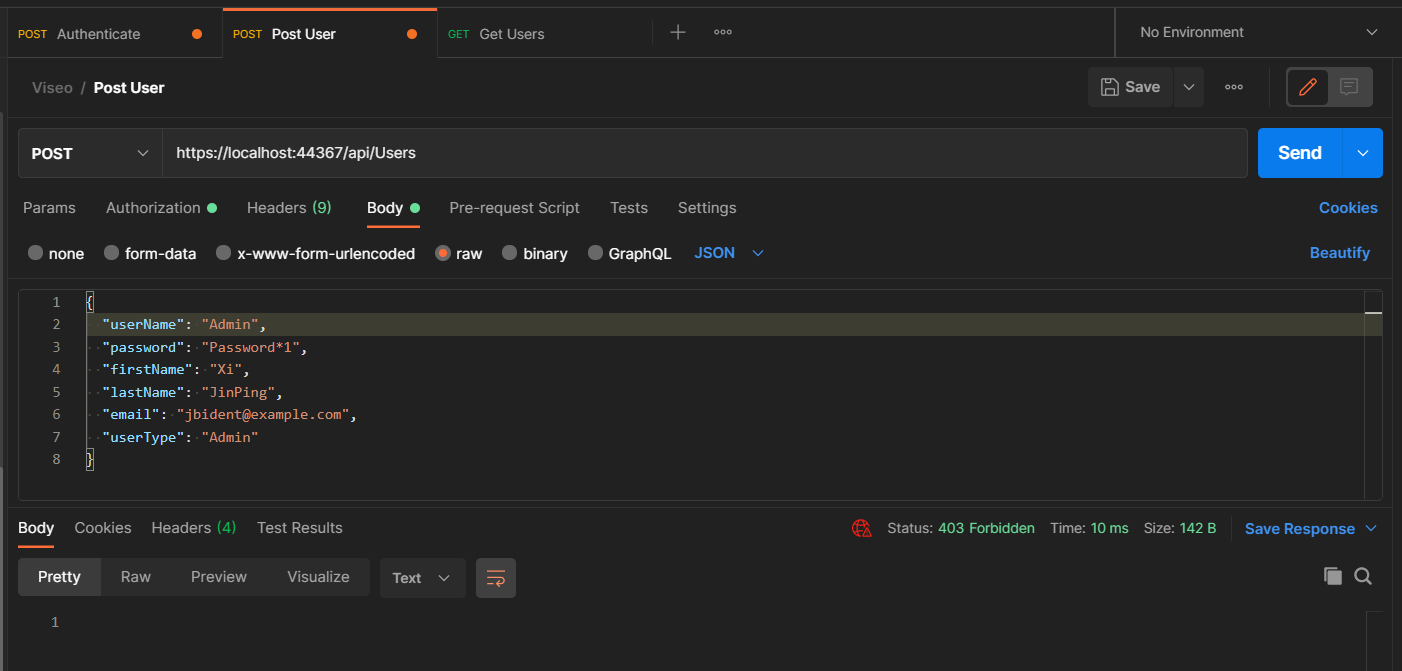
Results:

* Password is encrypted.
* Set default status to Active.
* Created by and created date is populated with the owner.
  1. User with “User” user type should not be able to POST data.

Authenticate as User.



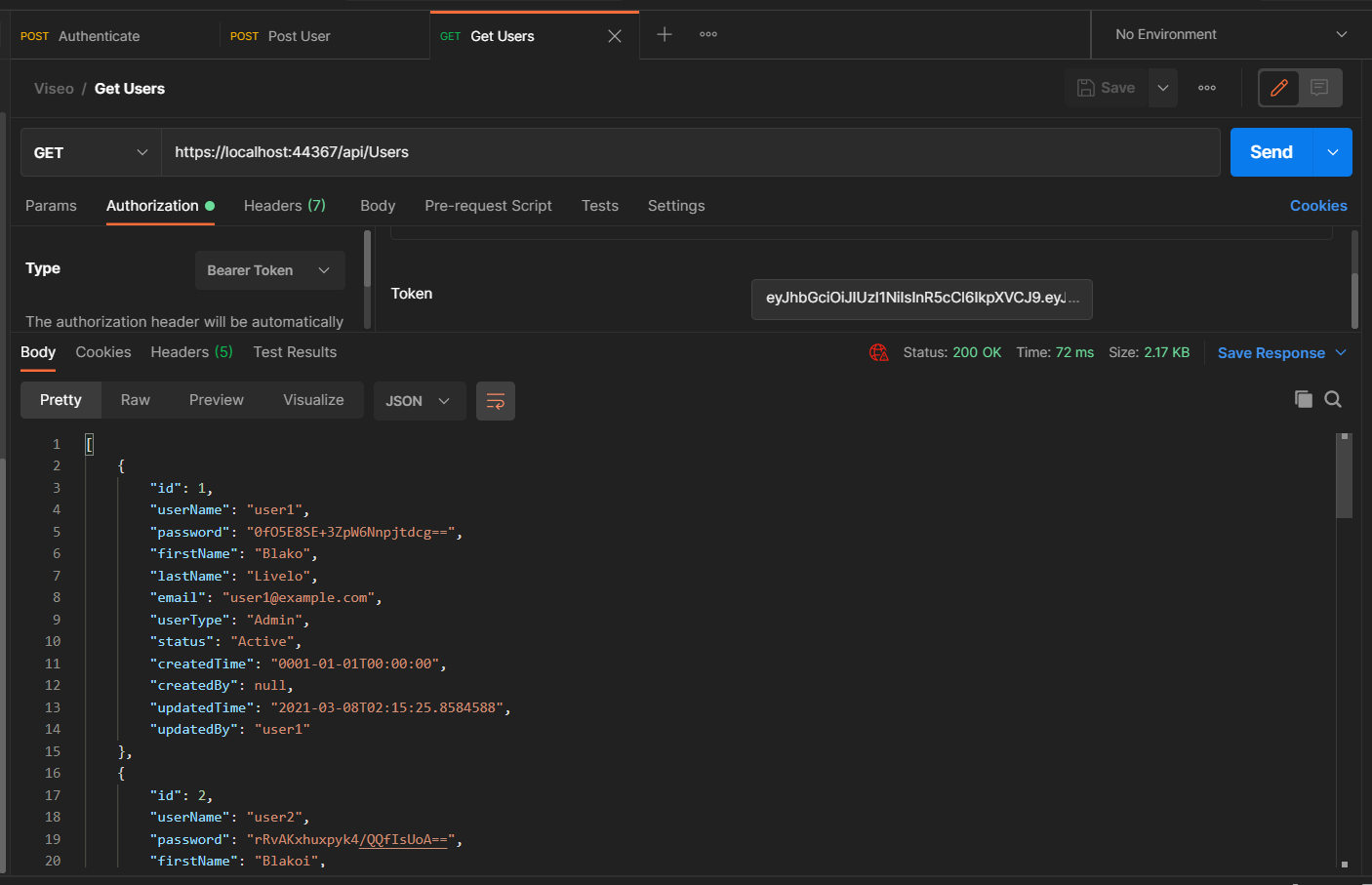
The user type “User” is forbidden.



1. Getting



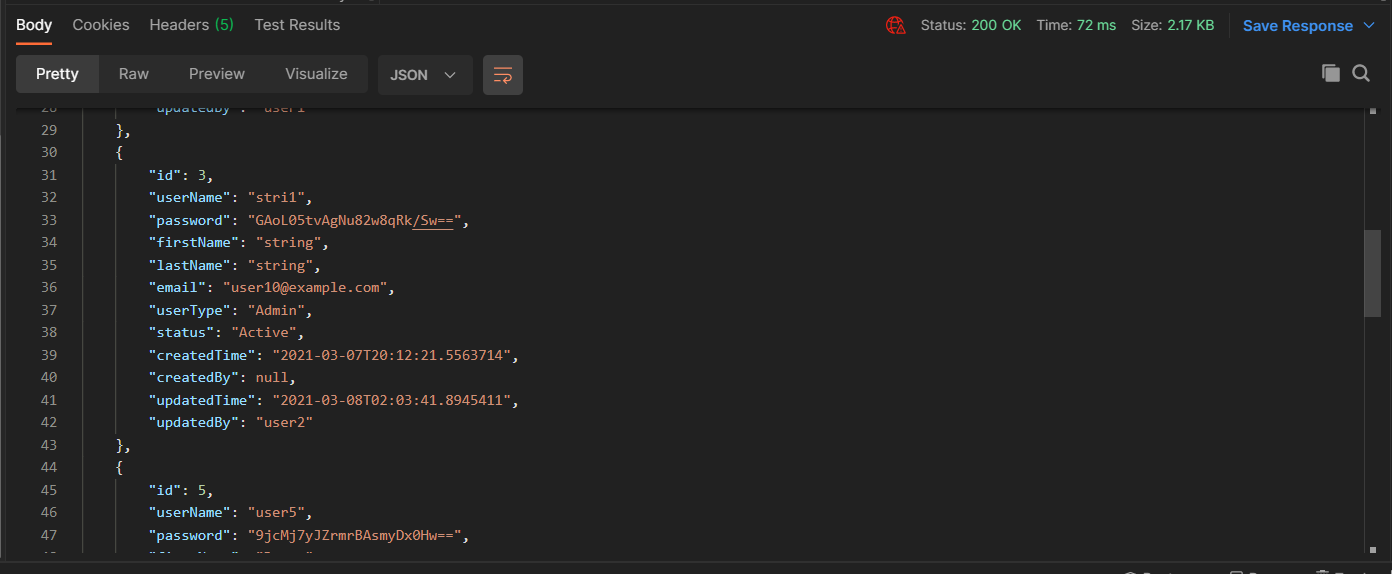
* 1. Get Users

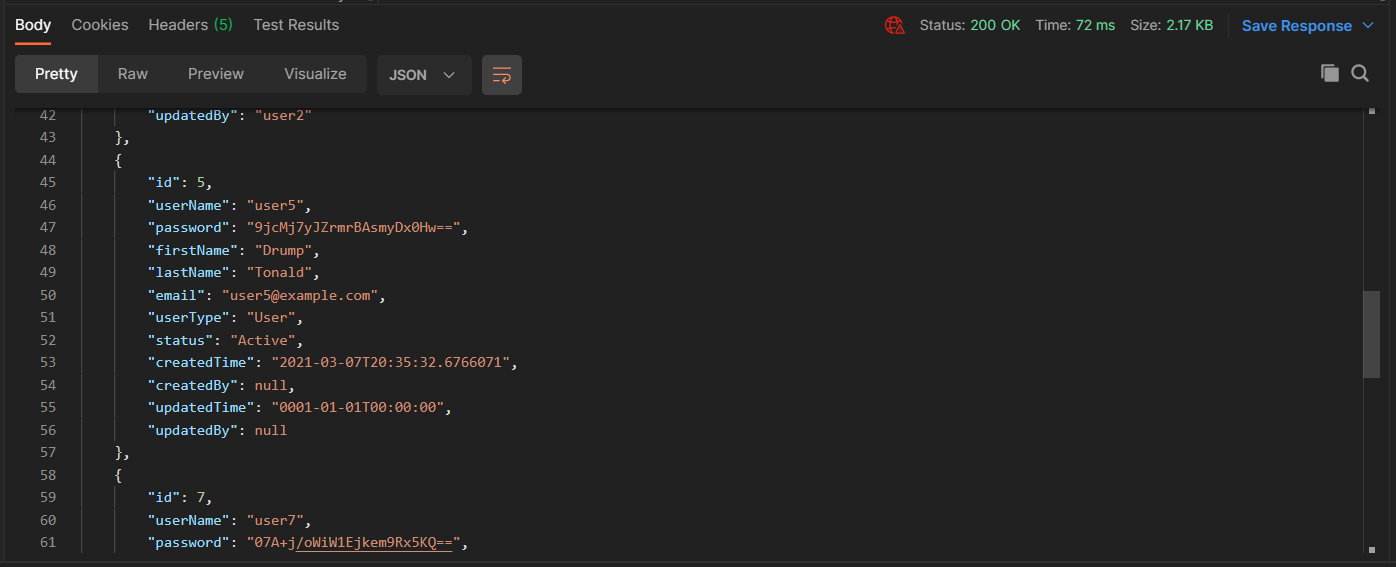


Results:

* Deleted records should not appear.

Record 4 and 6 is not is the result.

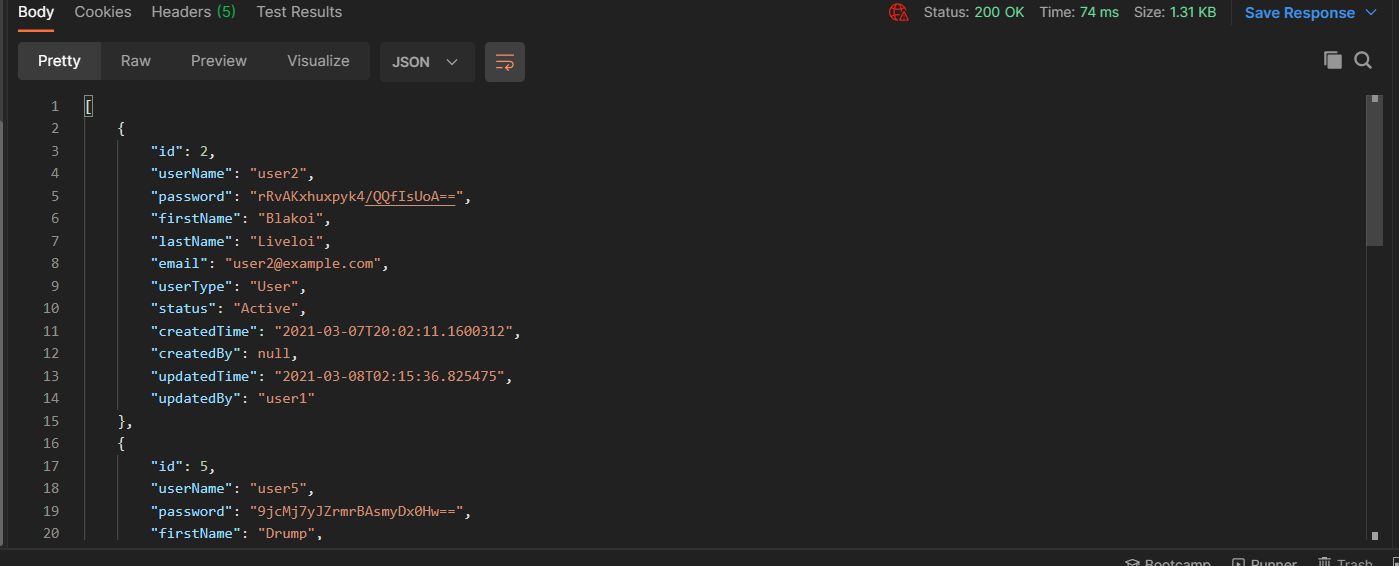




* User with “User” user type should be able to see the records with “User” user type.

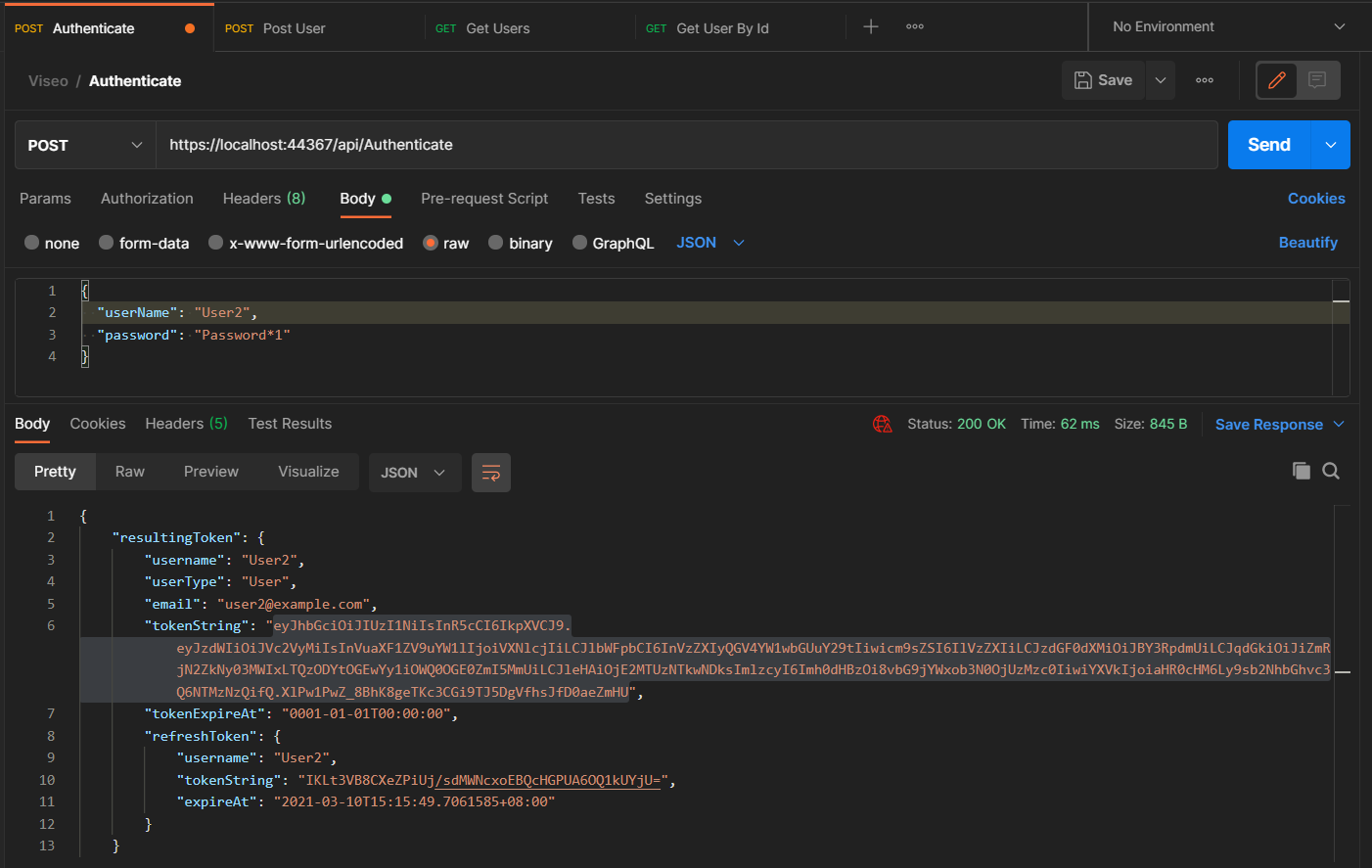
Login as User then run Get User.

Result: Record 1 and record 3 is not in the list.

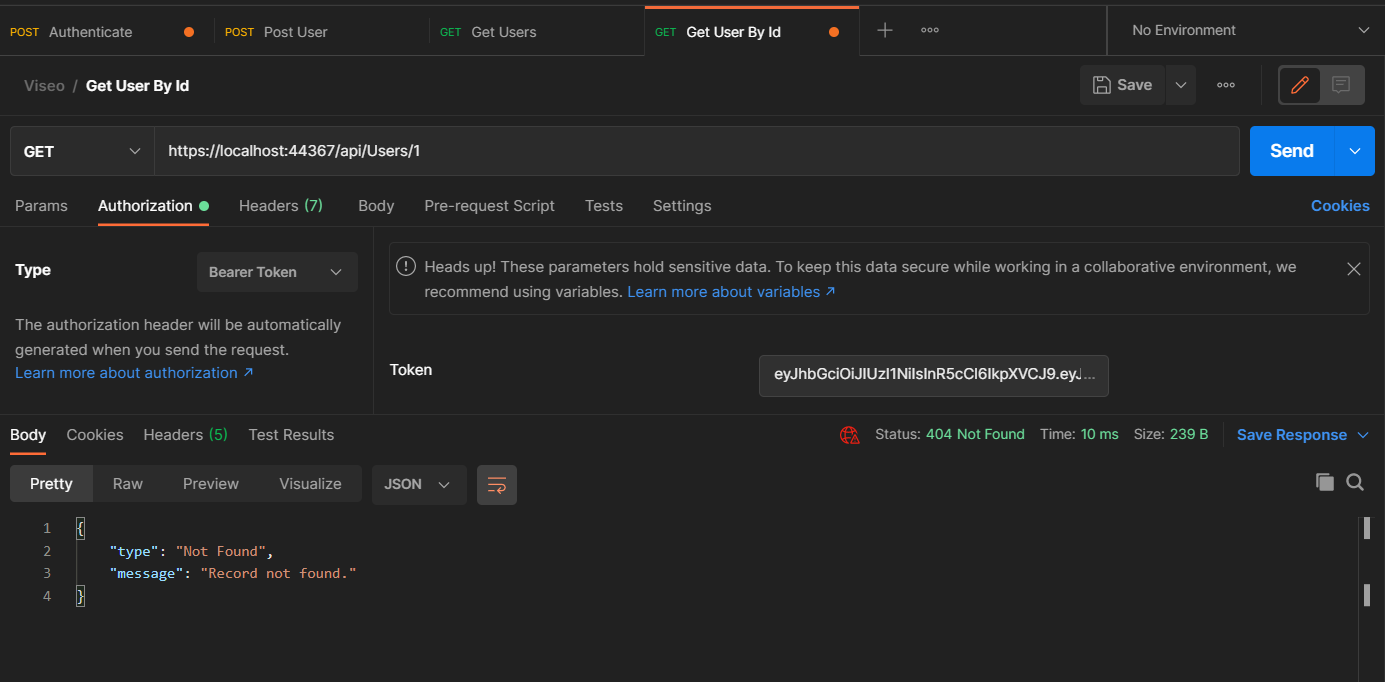


* 1. Get Users By Id.

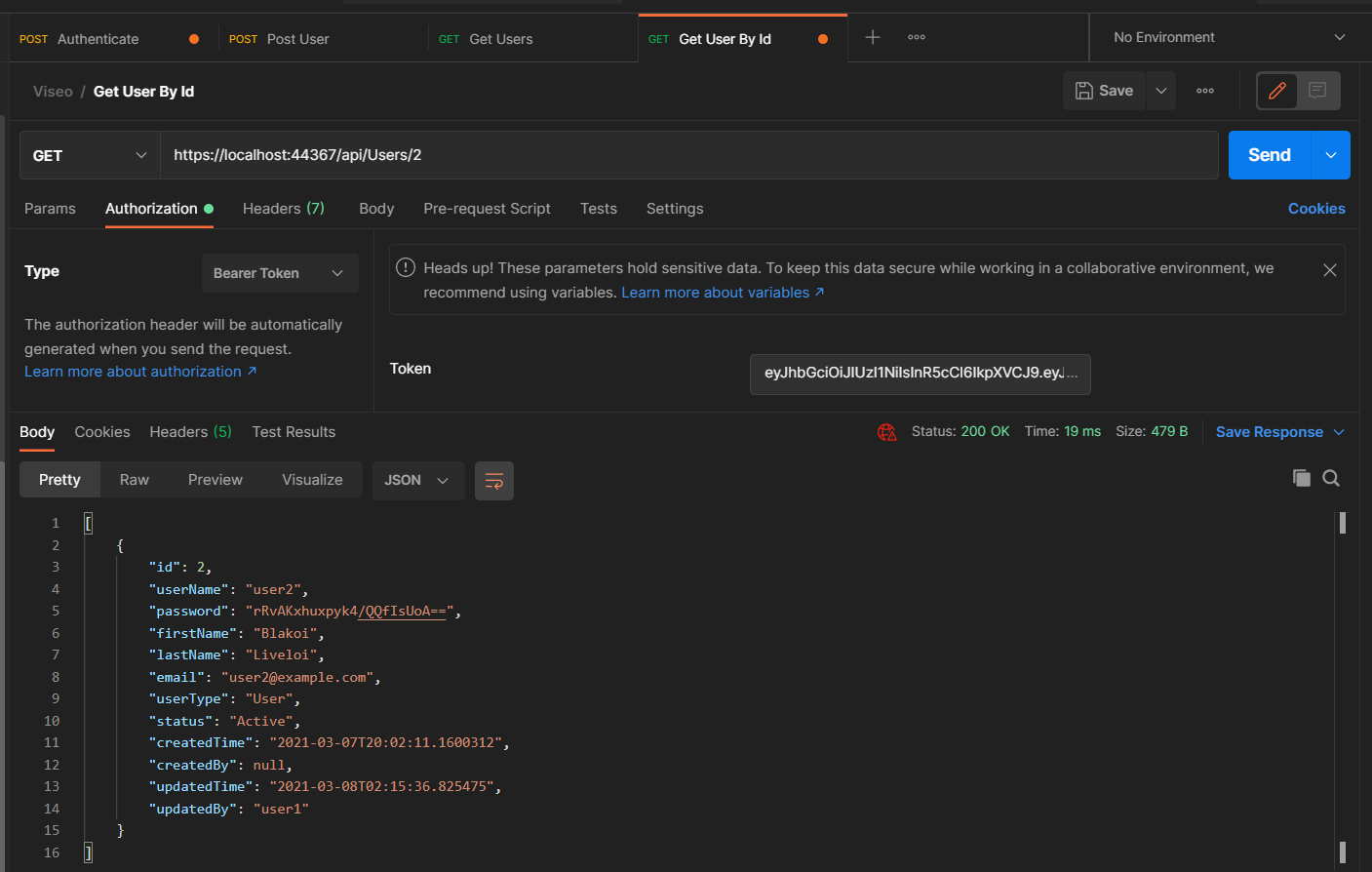
Login as User.



View Admin user. Cannot view record 1 because record 1 is Admin.

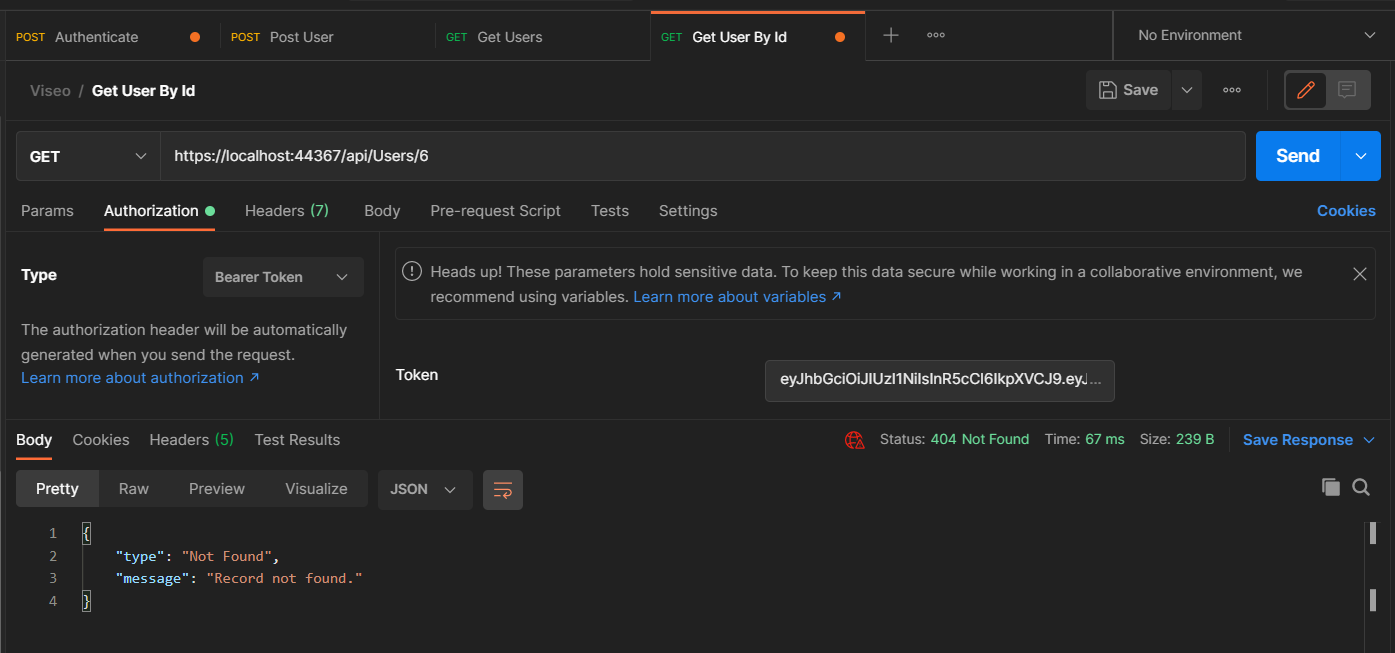


View user with User Type “User”.



View Deleted user.

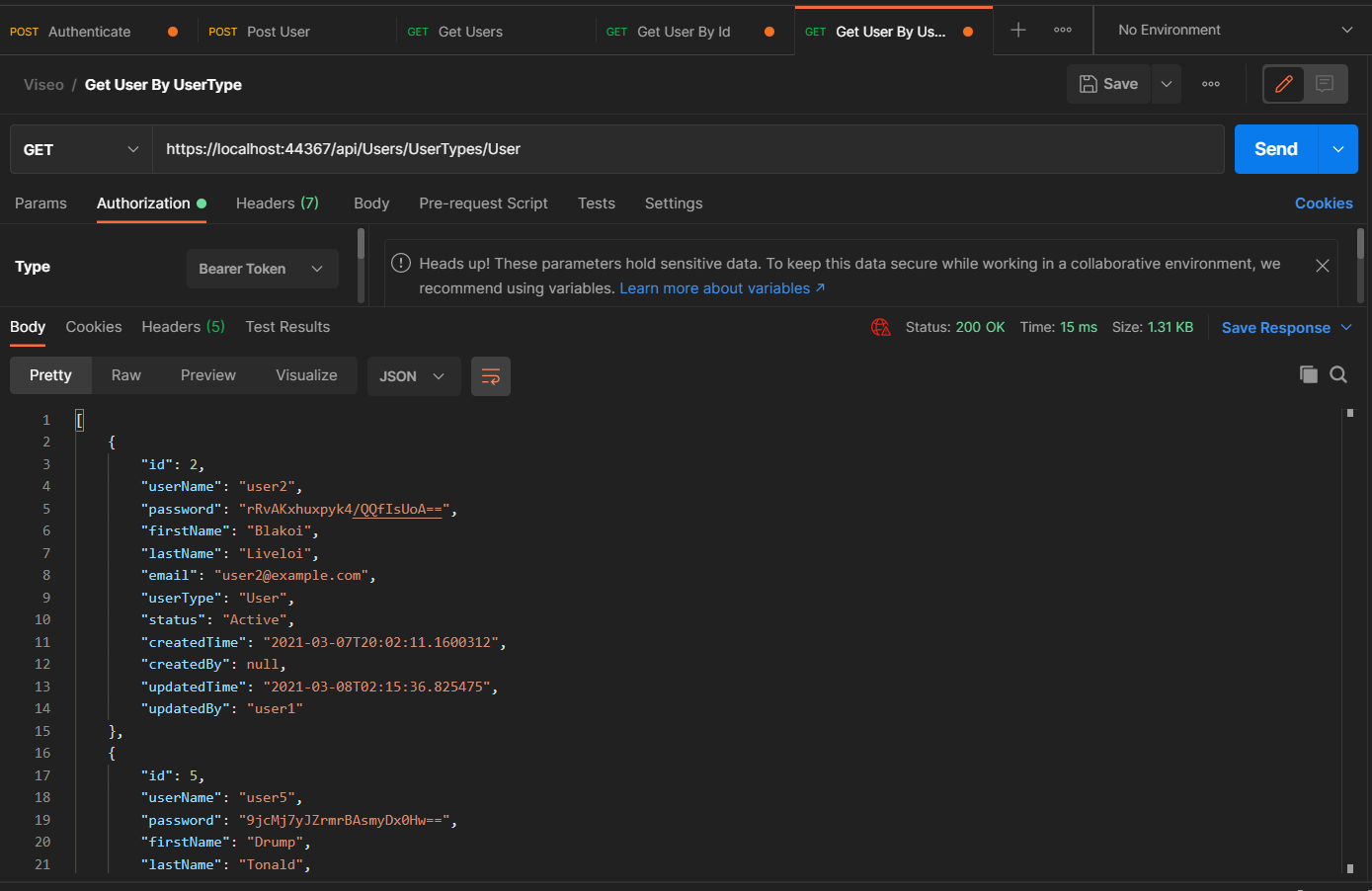
User Type: User and Admin.



* 1. Get Users By User Type.

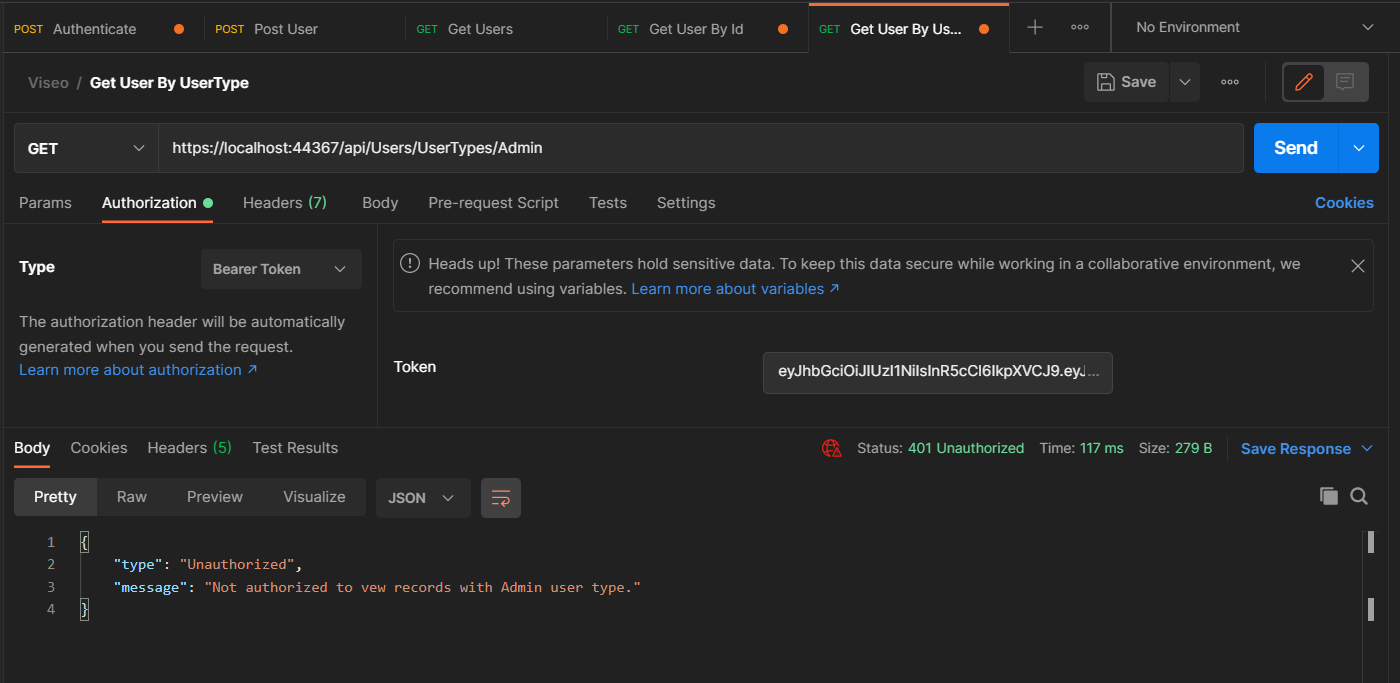
View “User” user type.

User Type: User



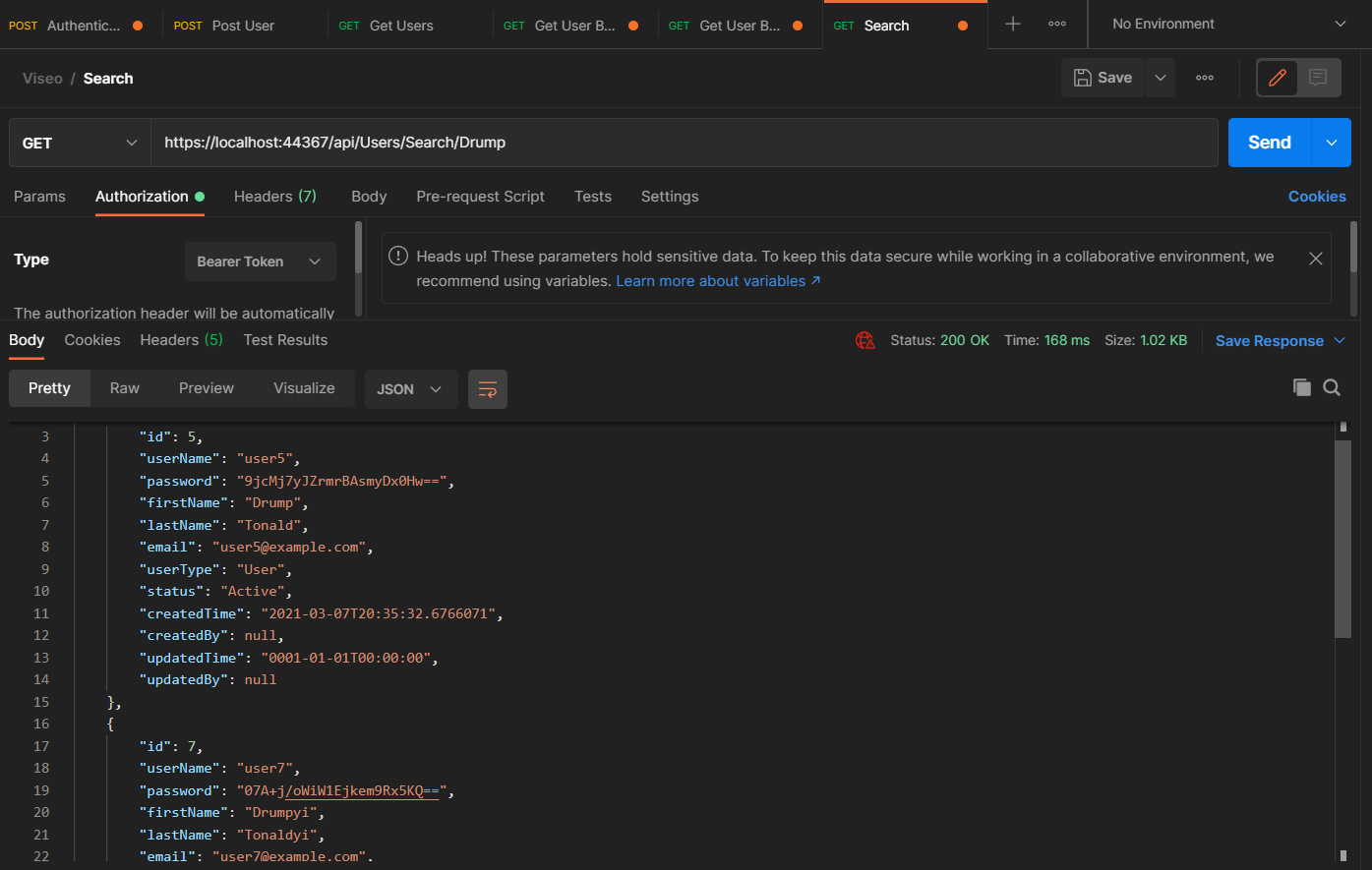
View “Admin” user type.

User Type: User

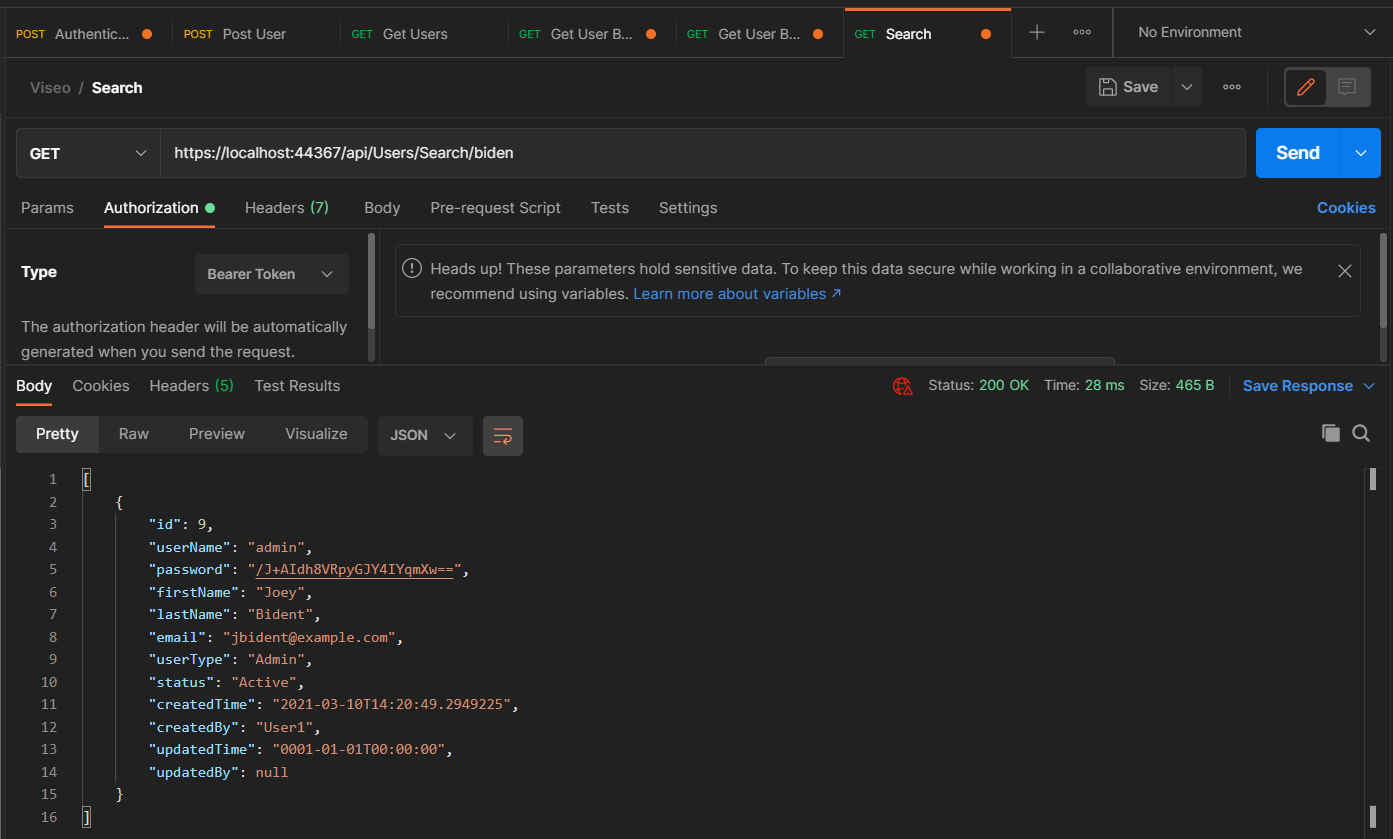


* 1. Search

Search for “Drump”:

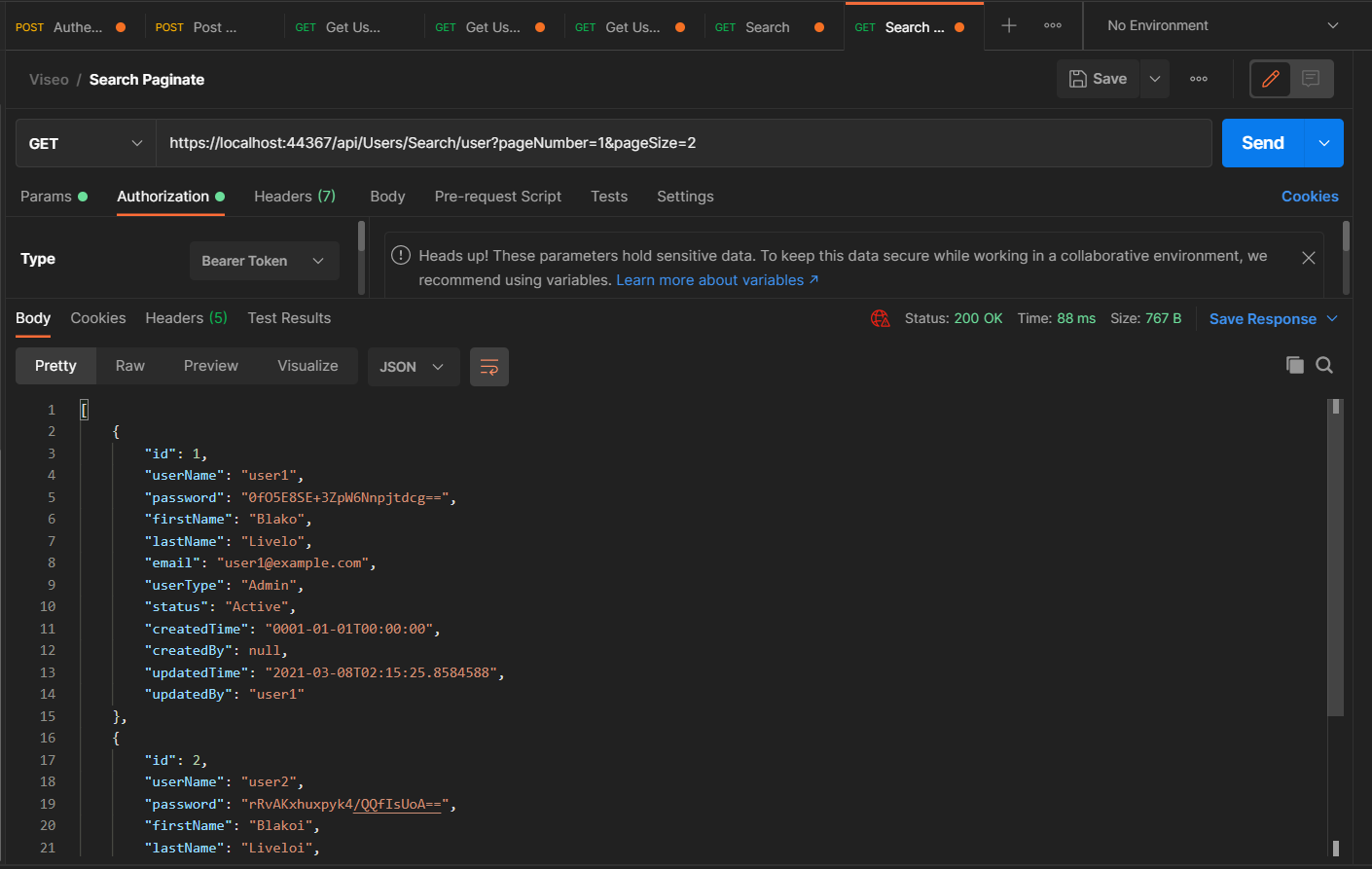


Search for “Biden”.



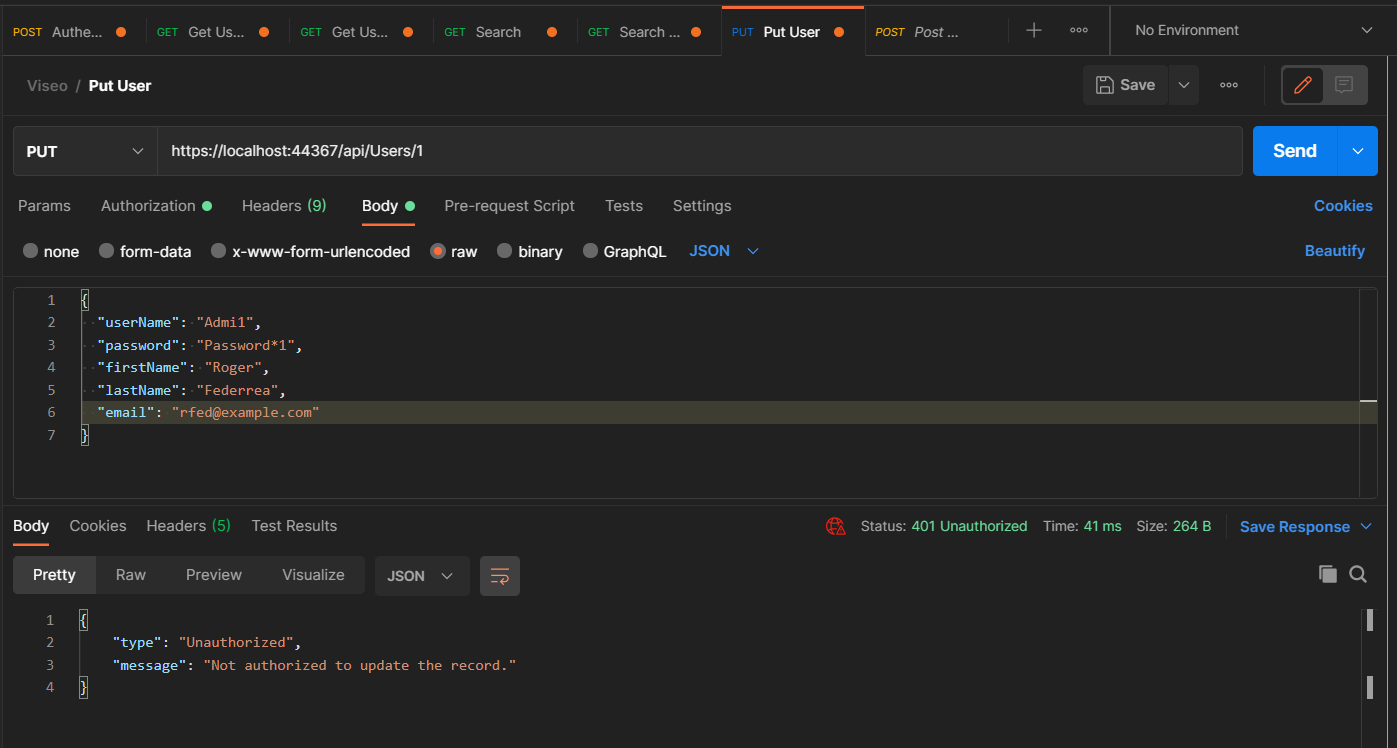
* 1. Search with pagination

Show page 1 and page size 2 (2 record in 1 page).

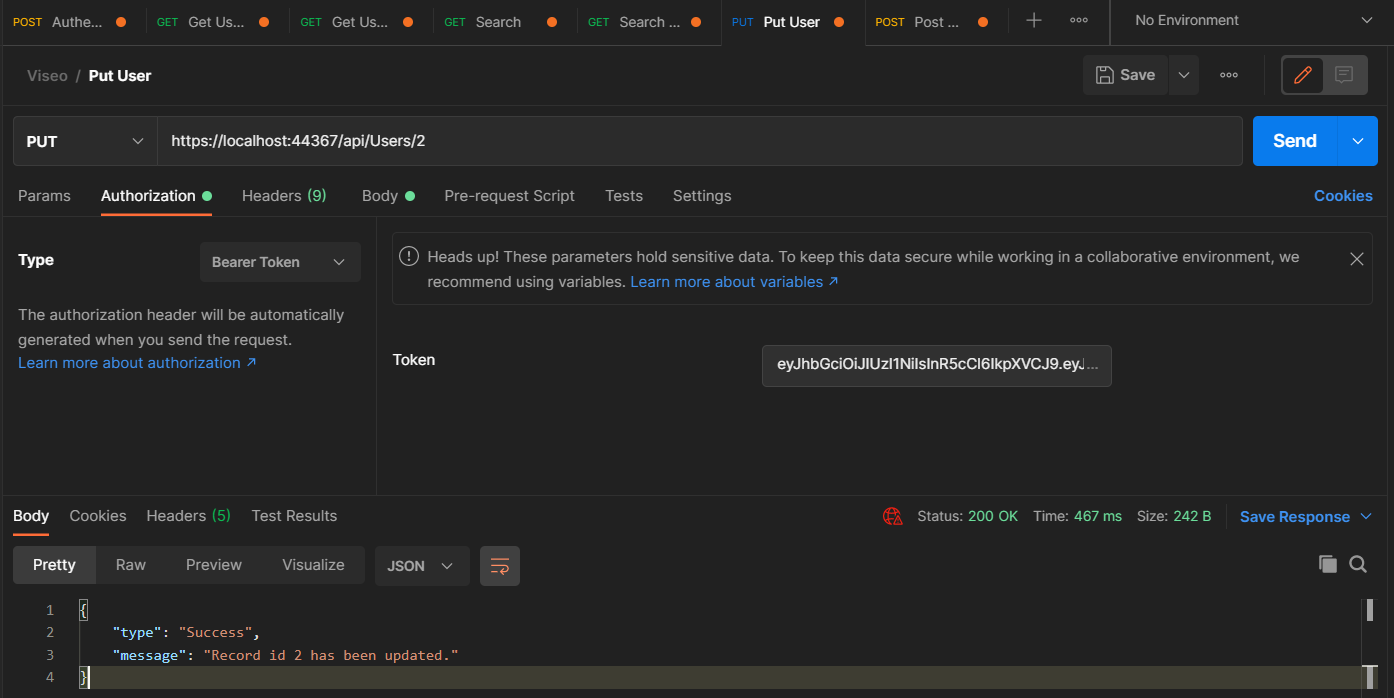


1. Putting

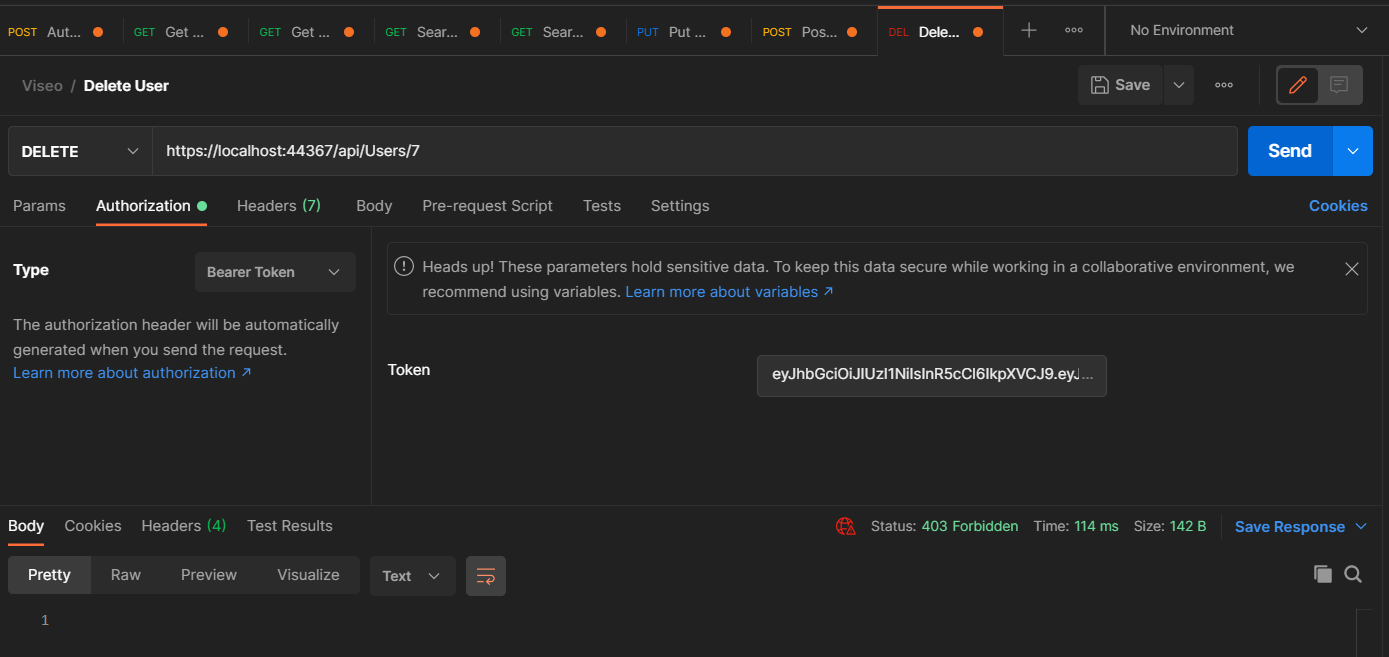
Login as user. User is not authorized to user 1 because it is not his/her record.



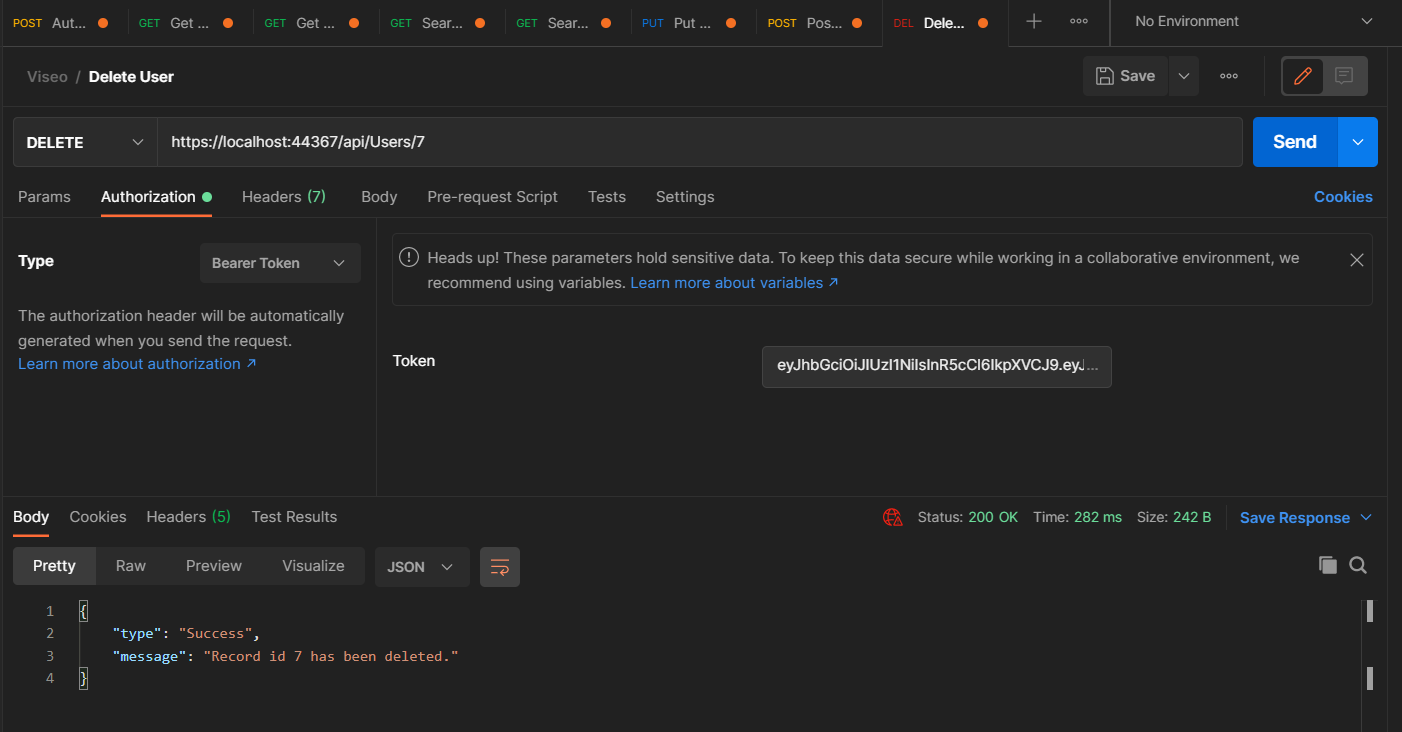
Login as user. Update own record.



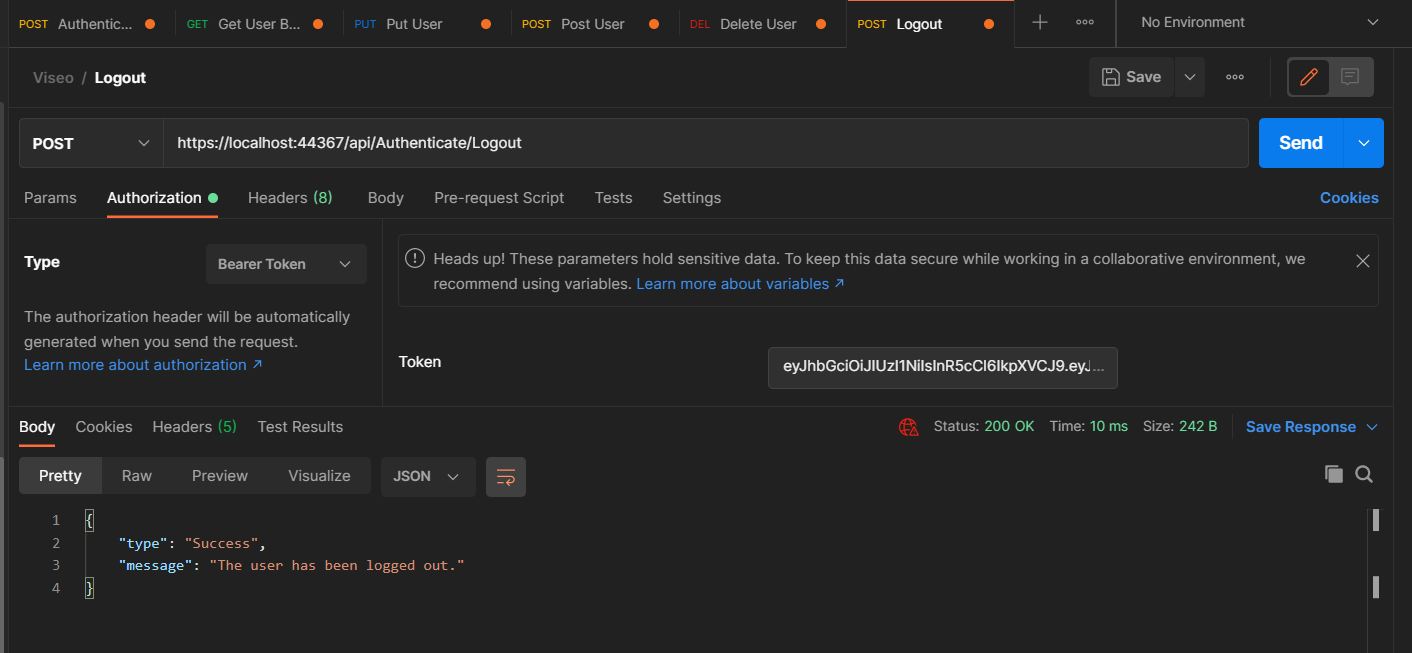
1. Deleting
   1. User Type “User” is not authorize to delete.



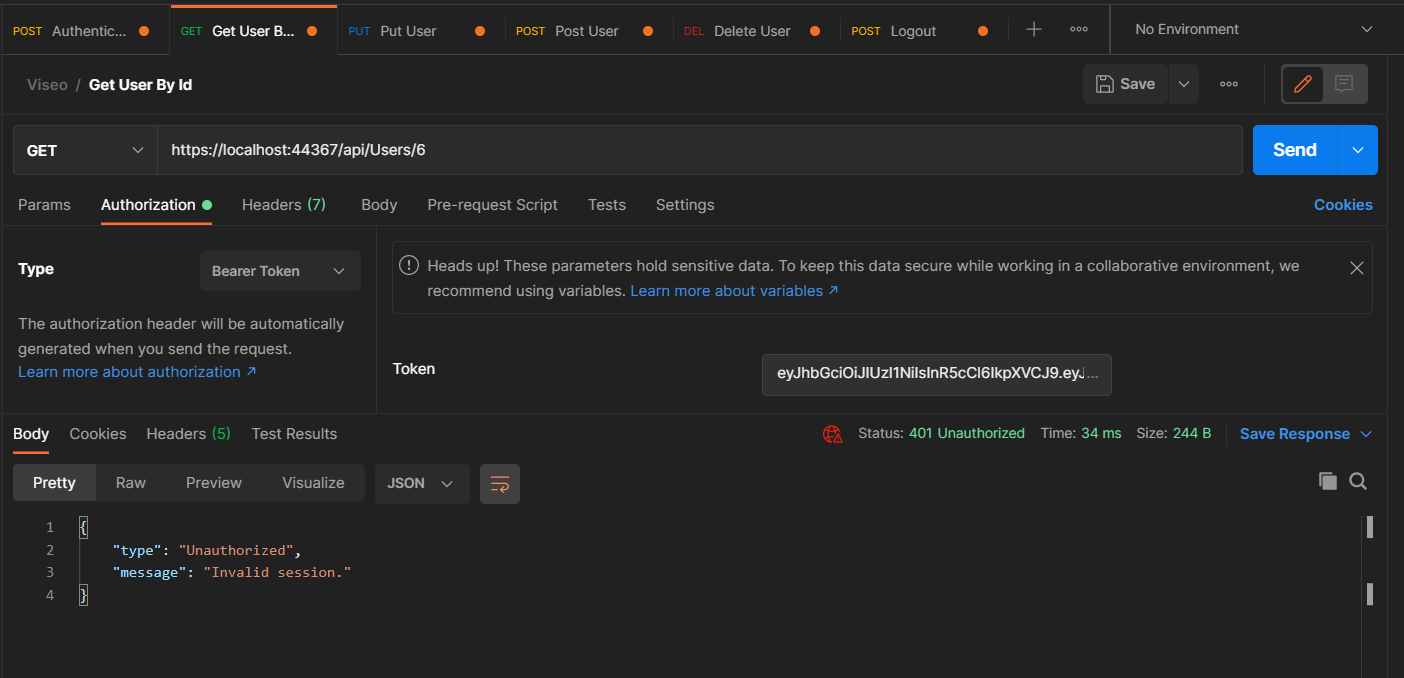
* 1. User Type “Admin” can delete.



1. Logout.

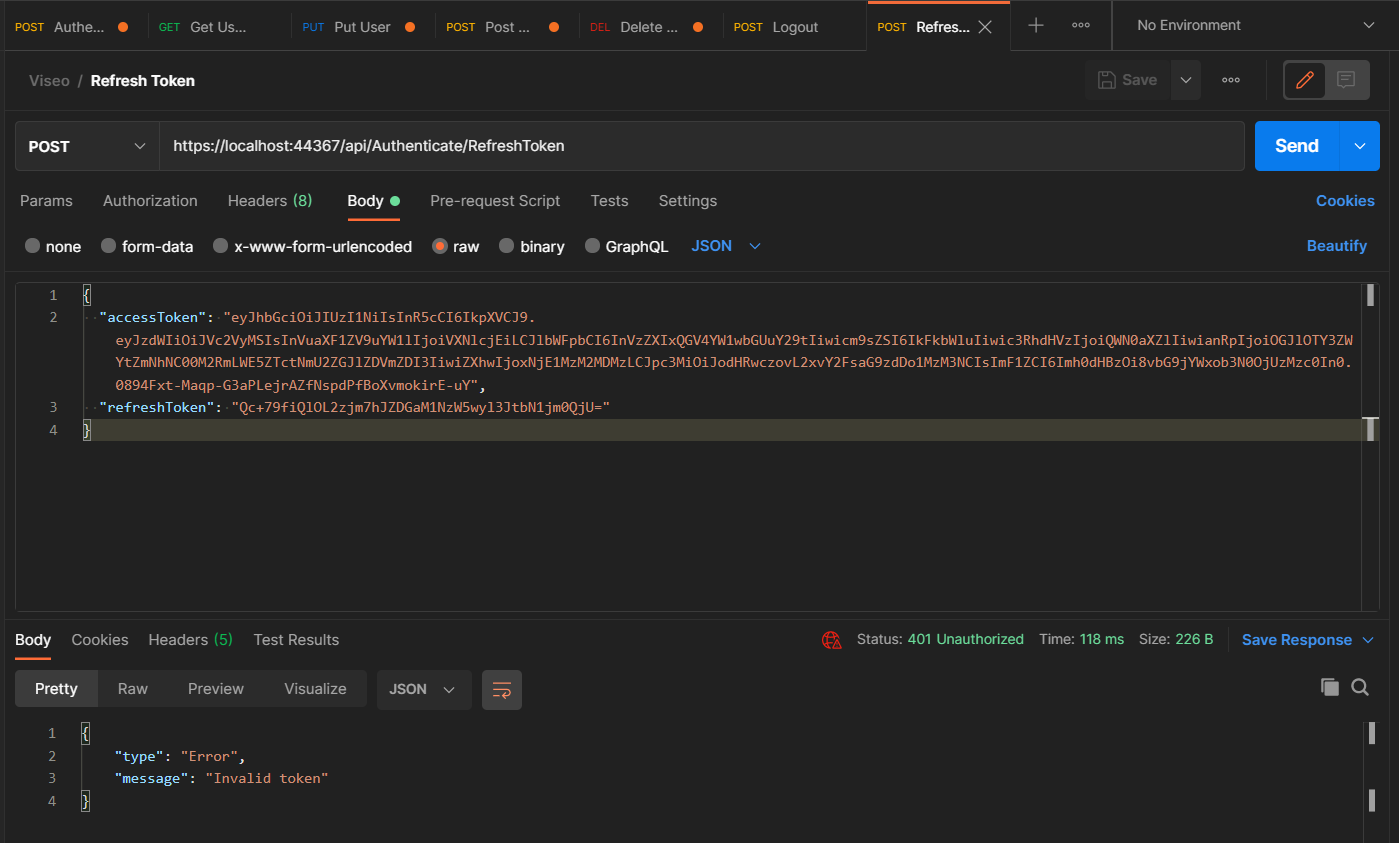


The logged out user is now authorized due to invalid session.

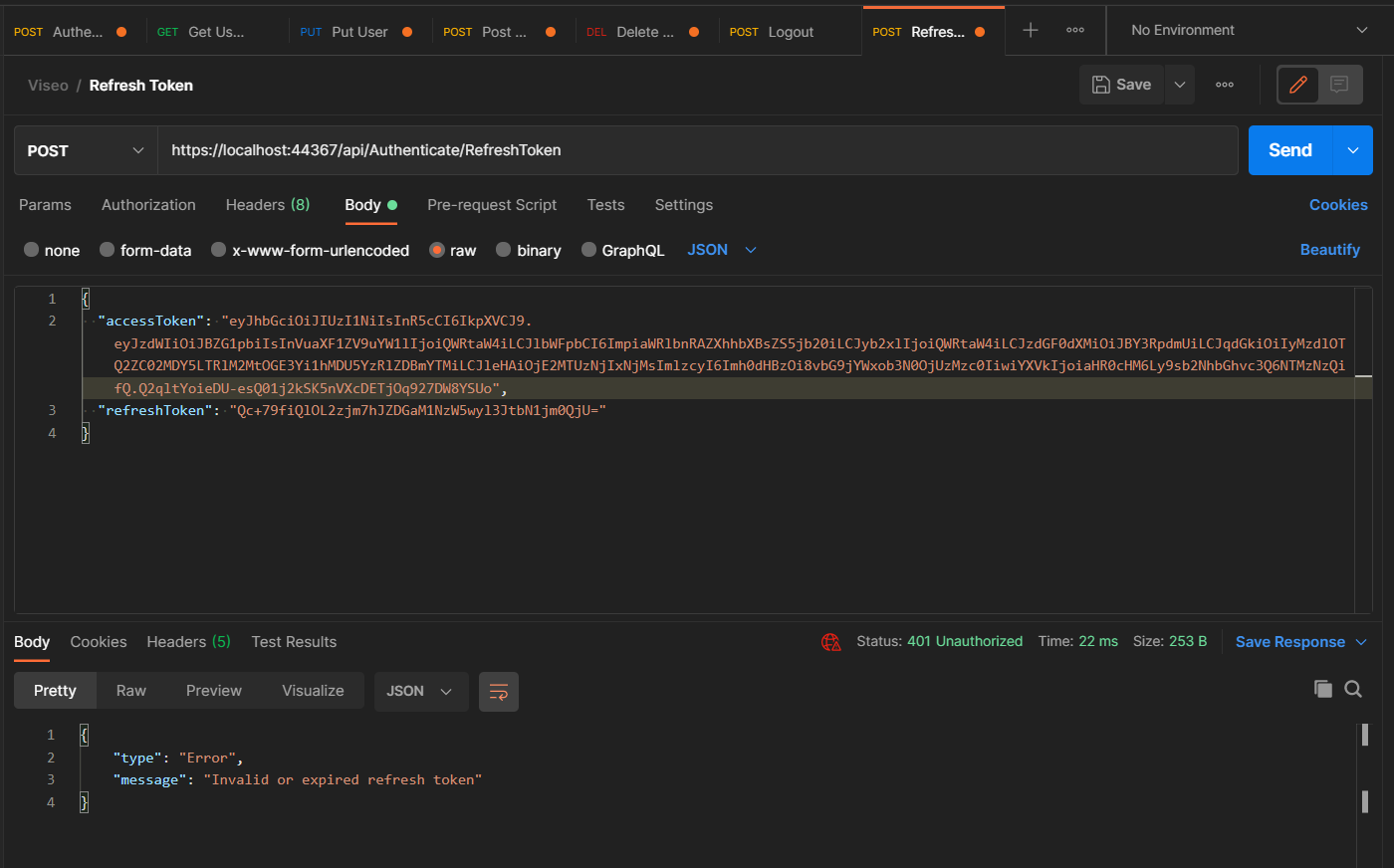


1. Refresh Token

When invalid access token is provided.



When refresh token is invalid or expired.



When all tokens are valid.

