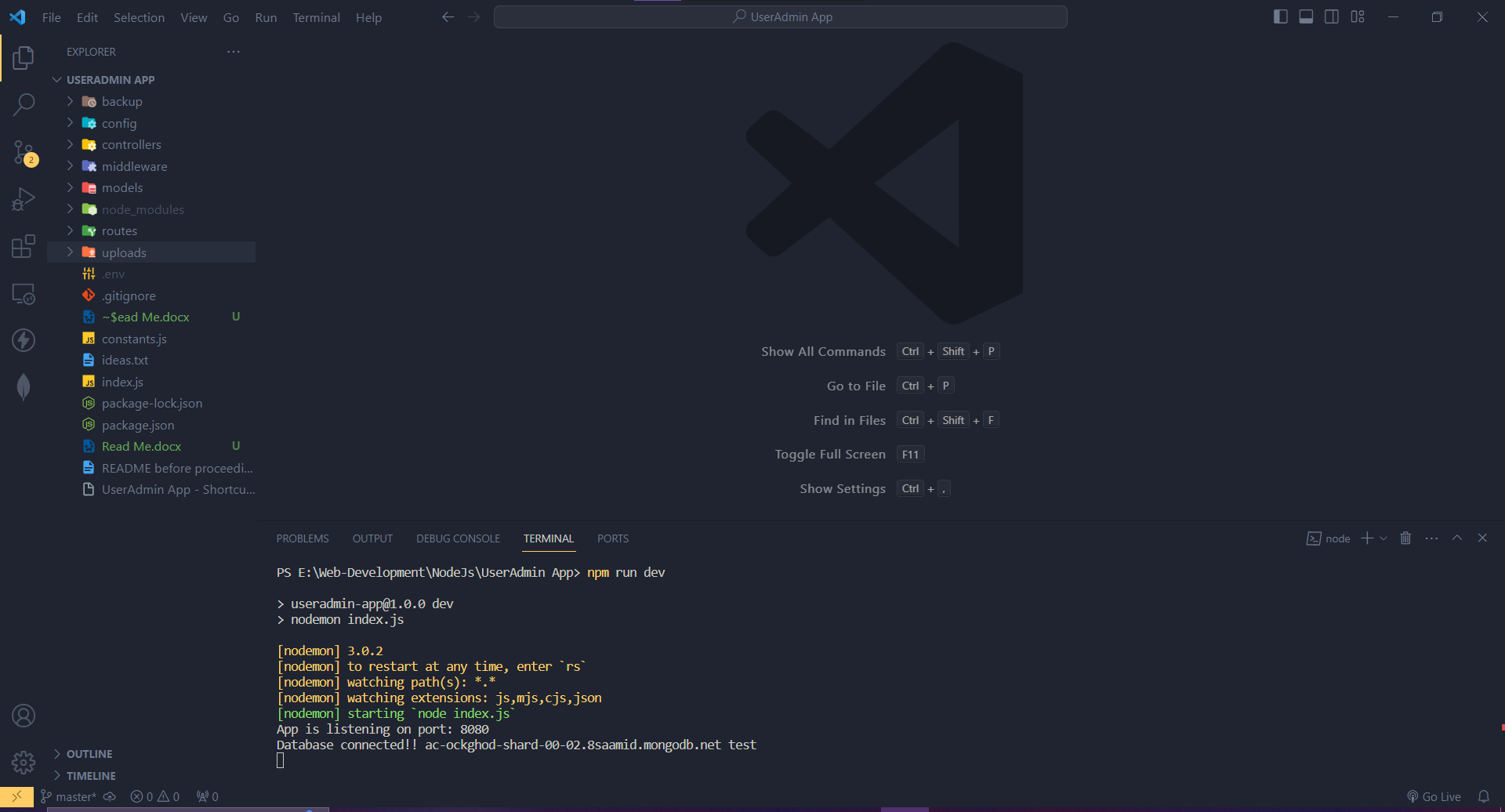
Greetings!

This assignment is done with utmost dedication and desire for getting selected. It follows each and every requirement mentioned in the pdf. Here is a short demo of how to access the various backend endpoints.

Hope you like it !!

**How to start server:**

Type “**npm run dev**” into the terminal of project directory



API can be tested using POSTMAN. Following is the demonstration of how to use the various routes:

The routes are divided into 2 categories:

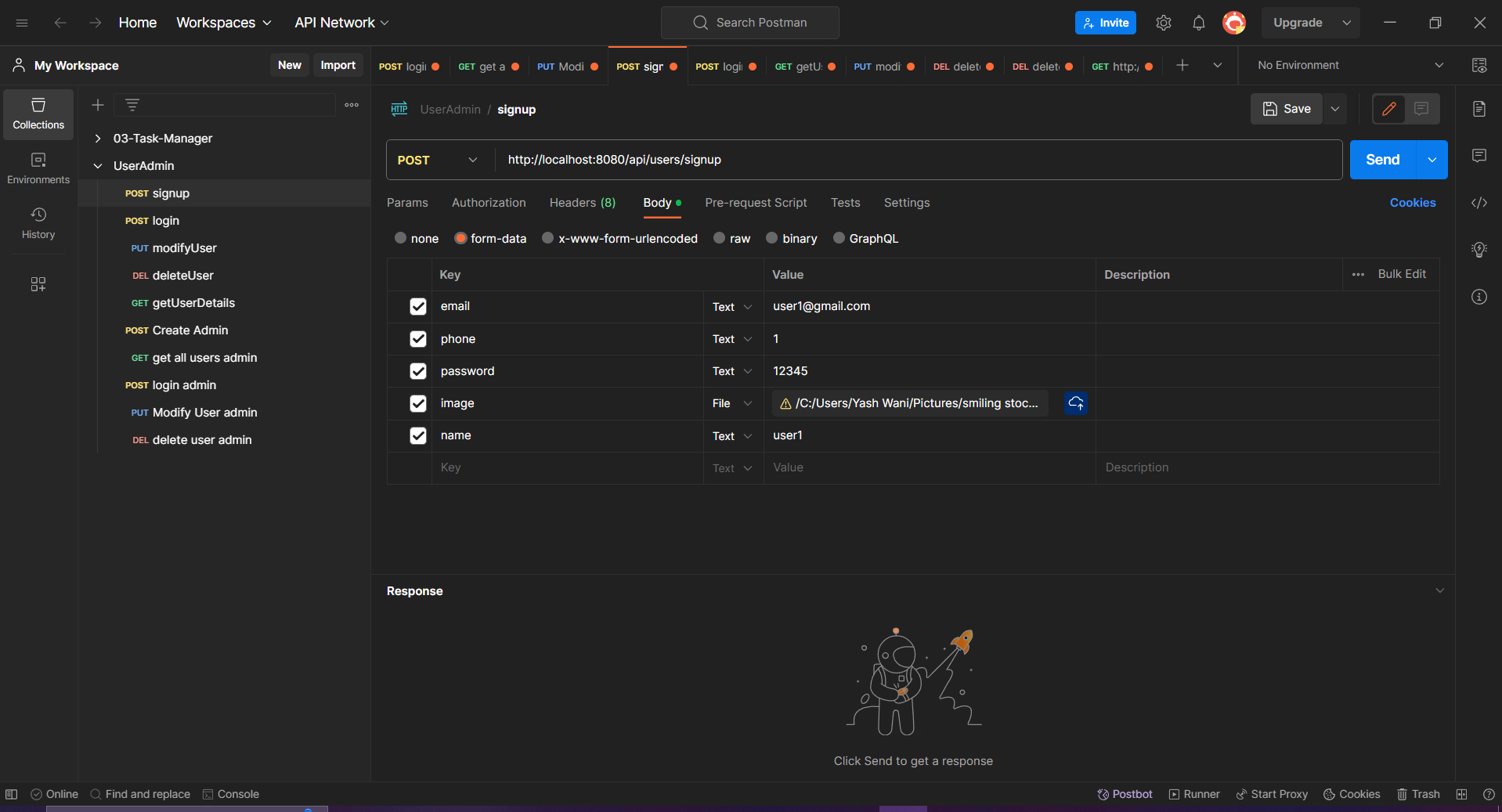
1. User routes
2. Admin routes

**1] User Routes**

**a) Sign up User**

http://localhost:8080/api/users/signup

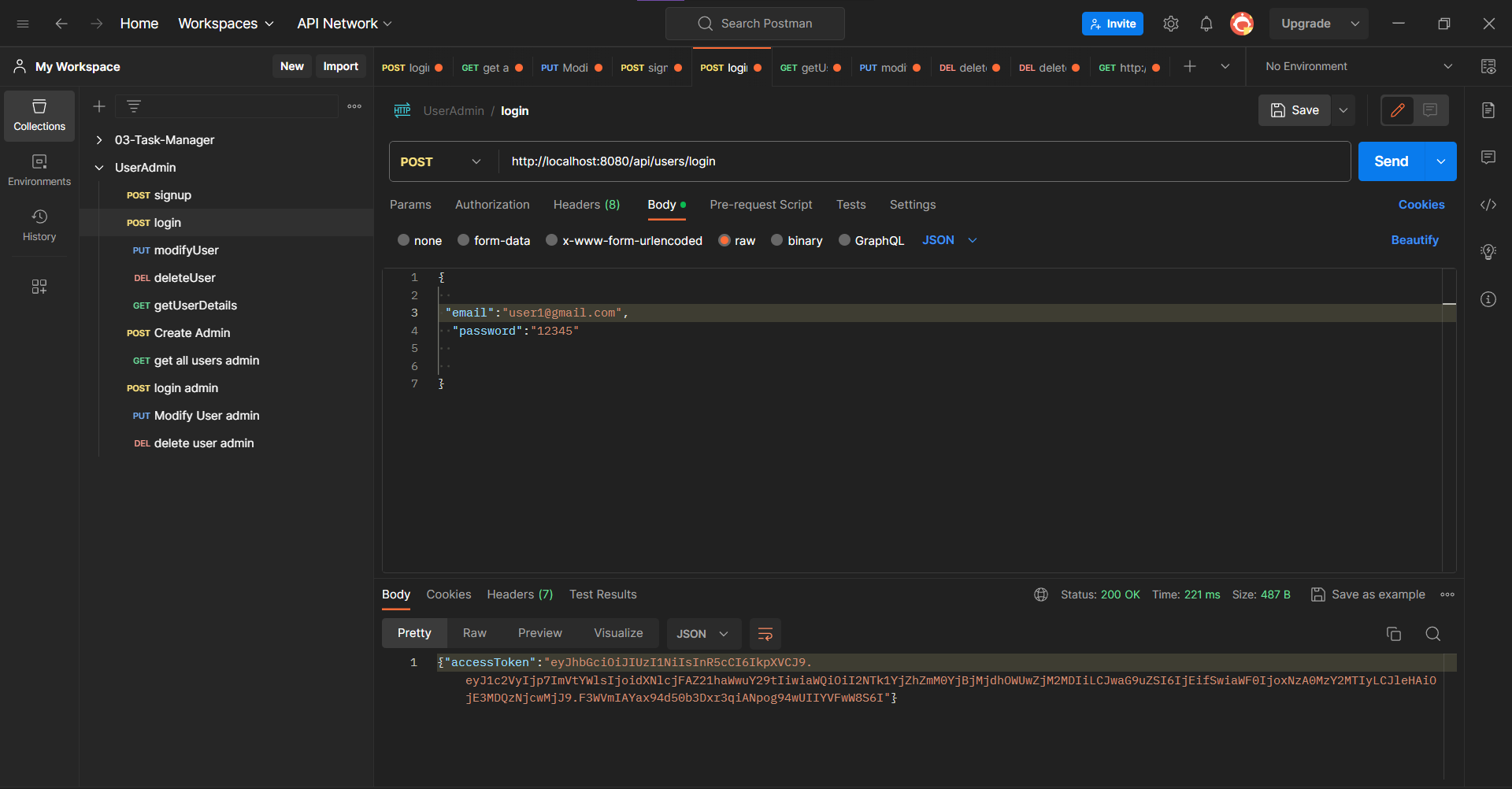
A user can signup by providing name, password, profile image and (email/phone/both). The data will be sent as a multipart form.



**b) Login User**

http://localhost:8080/api/users/login

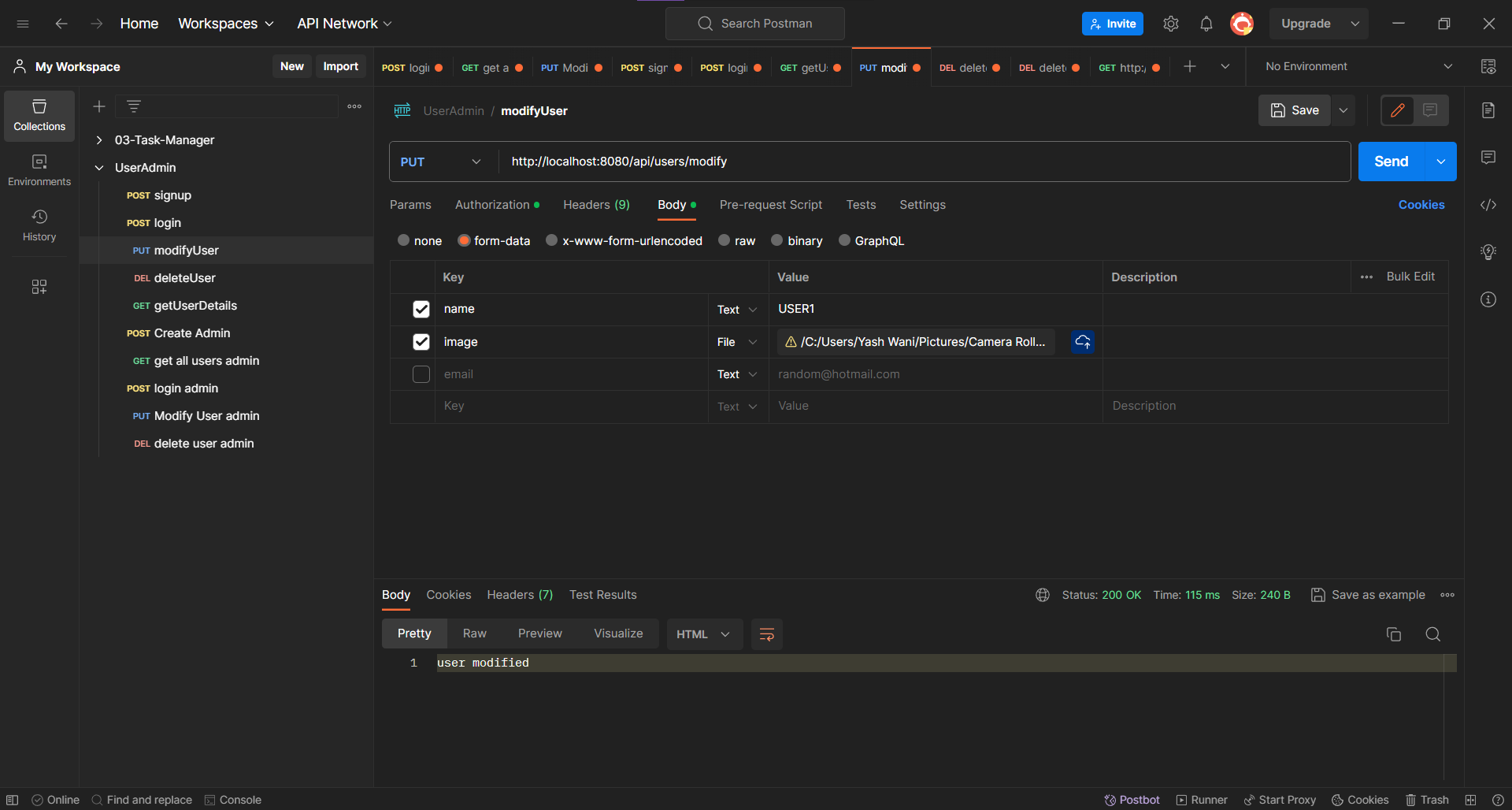
A user can login using email/phone and password. Upon logging in, a JWT token will be returned as response. This JWT token can be sent as a bearer token to access protected routes of the user. Login credentials should be sent as JSON data. Note that the JWT token generated by the user cannot be used to access any of the admin routes and system will not allow you to do so to ensure security.

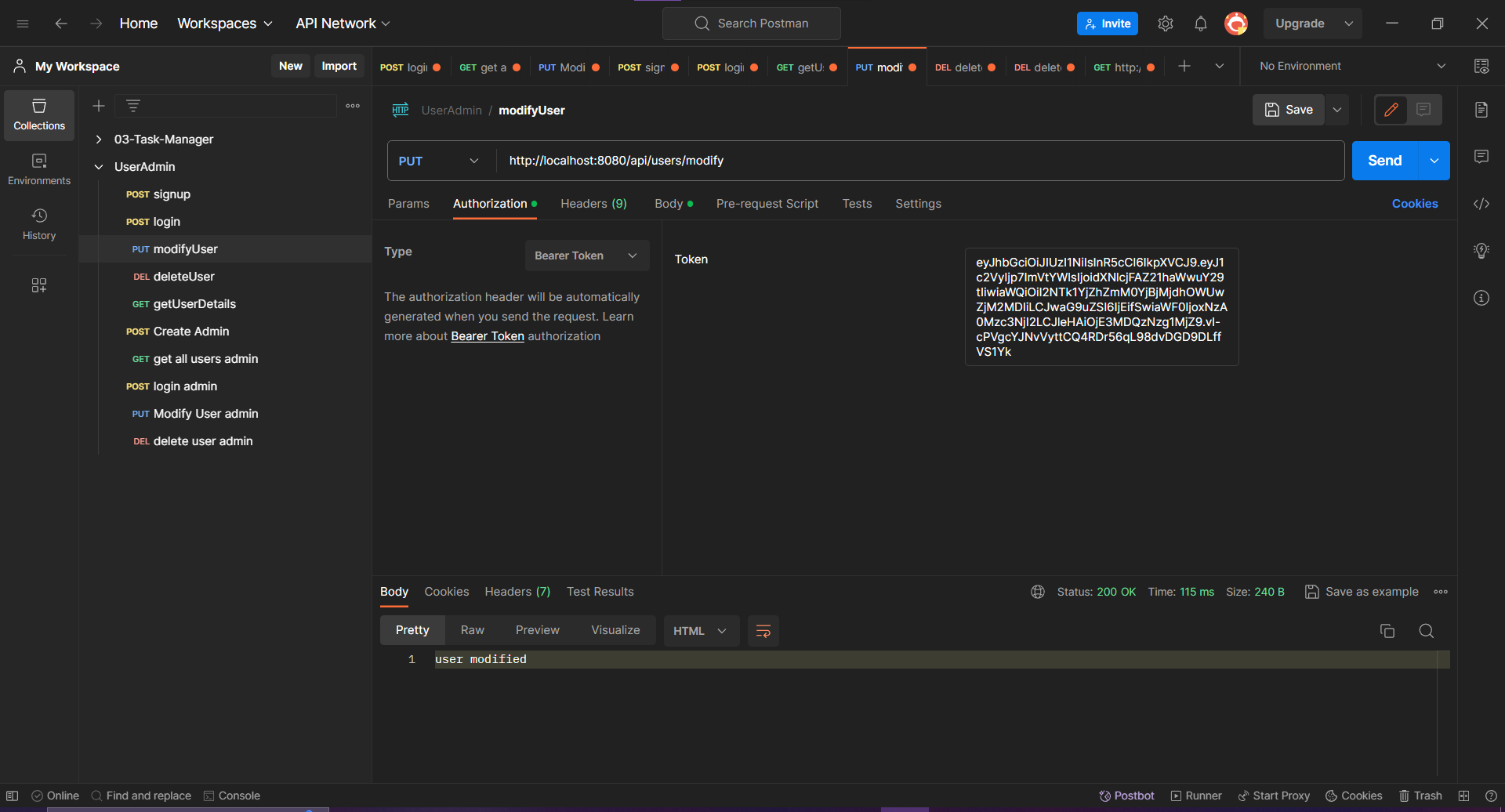


**c) Modify User**

http://localhost:8080/api/users/modify

User can only modify his profile photo and user name. To access this route, the token generated above needs to be put as bearer token. Modify user is also a multipart form.

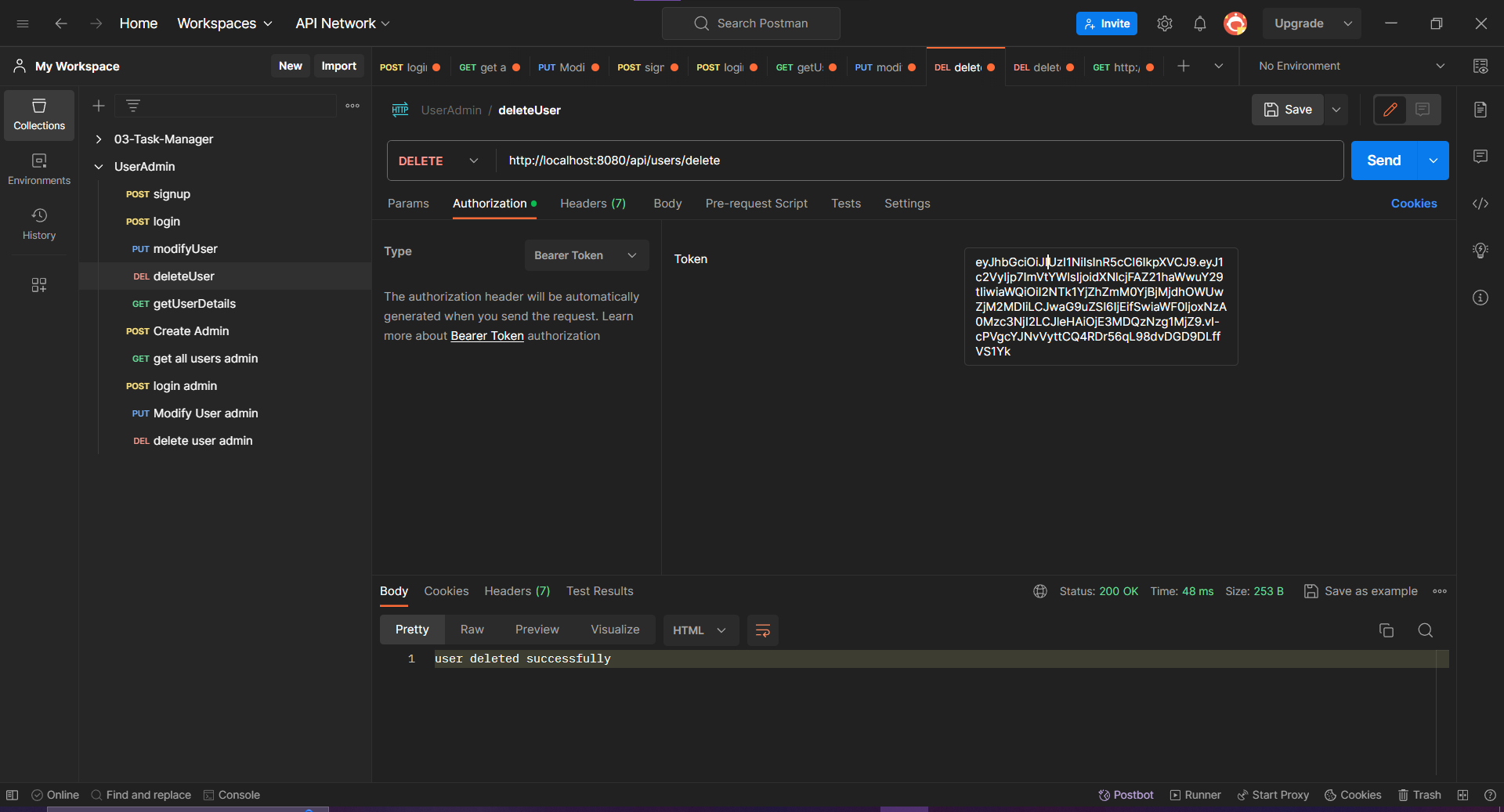




**d) Delete User**

http://localhost:8080/api/users/delete

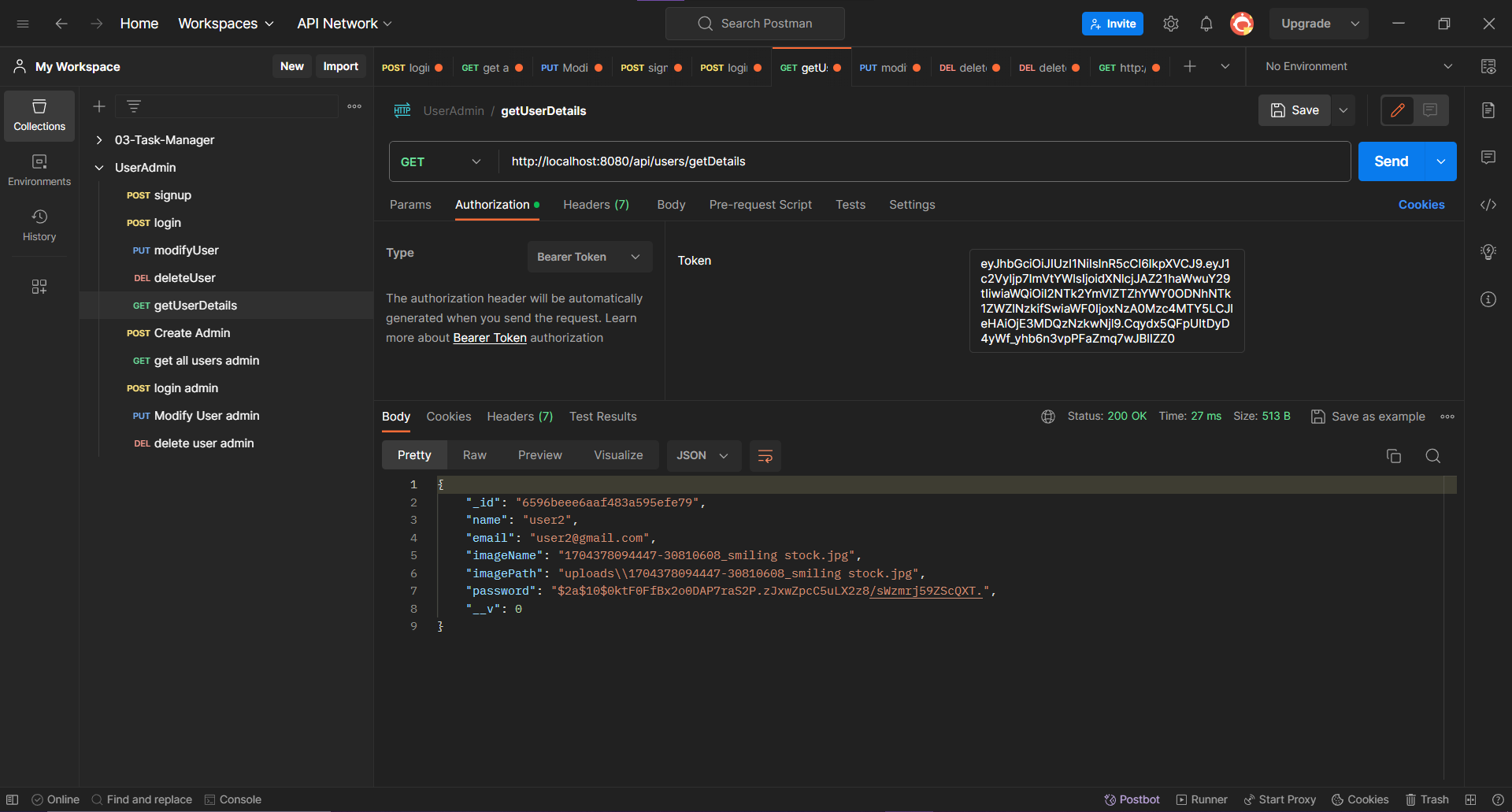
We only need to send JWT token generated at the time of login while accessing this endpoint. The user corresponding to the data in the payload of the token will get deleted.



**e) Get User Details**

http://localhost:8080/api/users/getDetails

JWT token generated should be passed as bearer token to get details of user corresponding to that user.

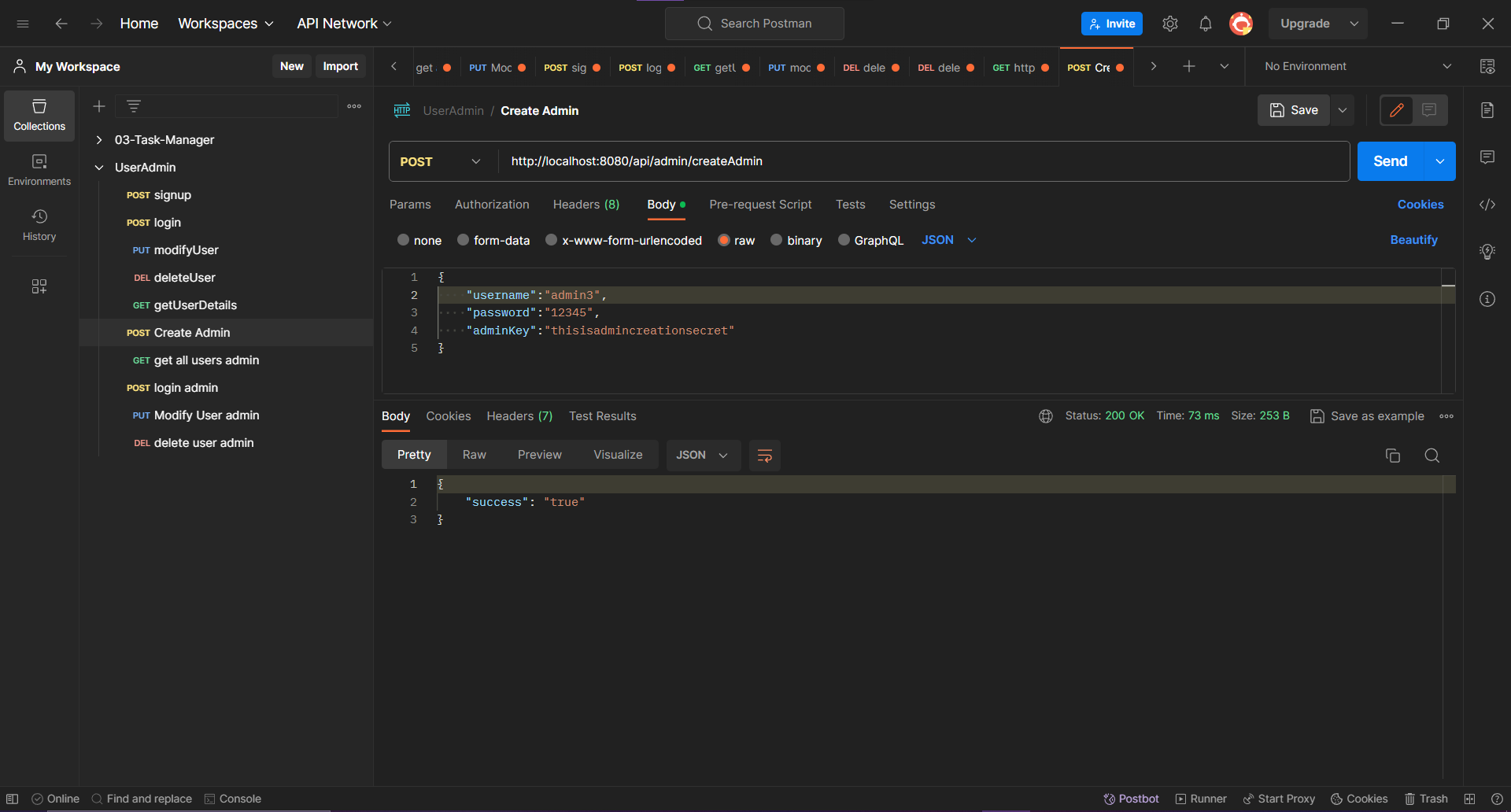


**2] Admin Routes**

**a) Create Admin**

http://localhost:8080/api/admin/createAdmin

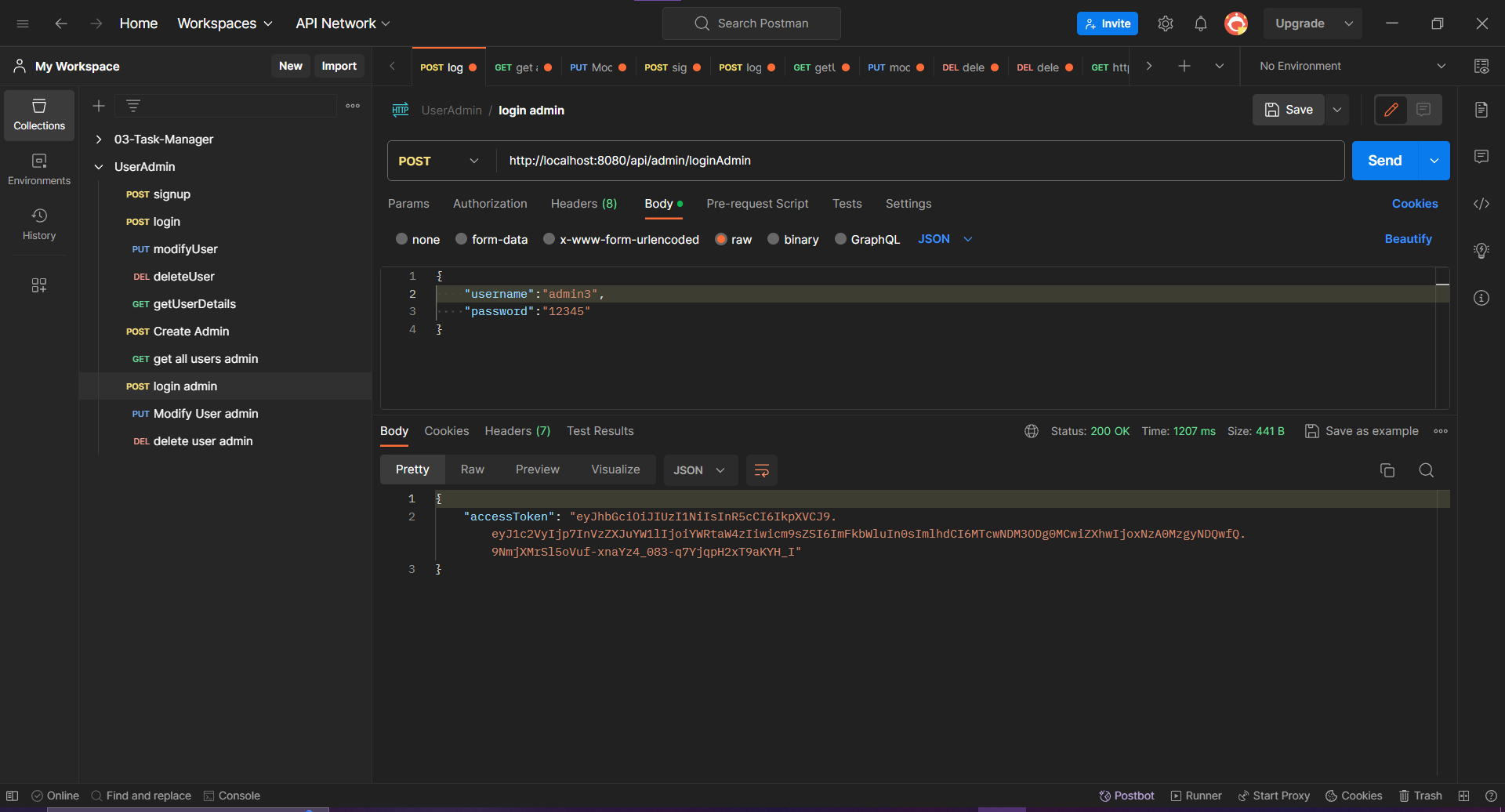
Used to create admin accounts. To create an admin account, one must enter username, password and a secret admin key. The secret admin key is stored only in environment variables of the server and only known by the administrators of the organization using the API. These details are passed as JSON data. The adminKey is: “thisisadmincreationsecret”.



**b) Login Admin**

http://localhost:8080/api/admin/loginAdmin

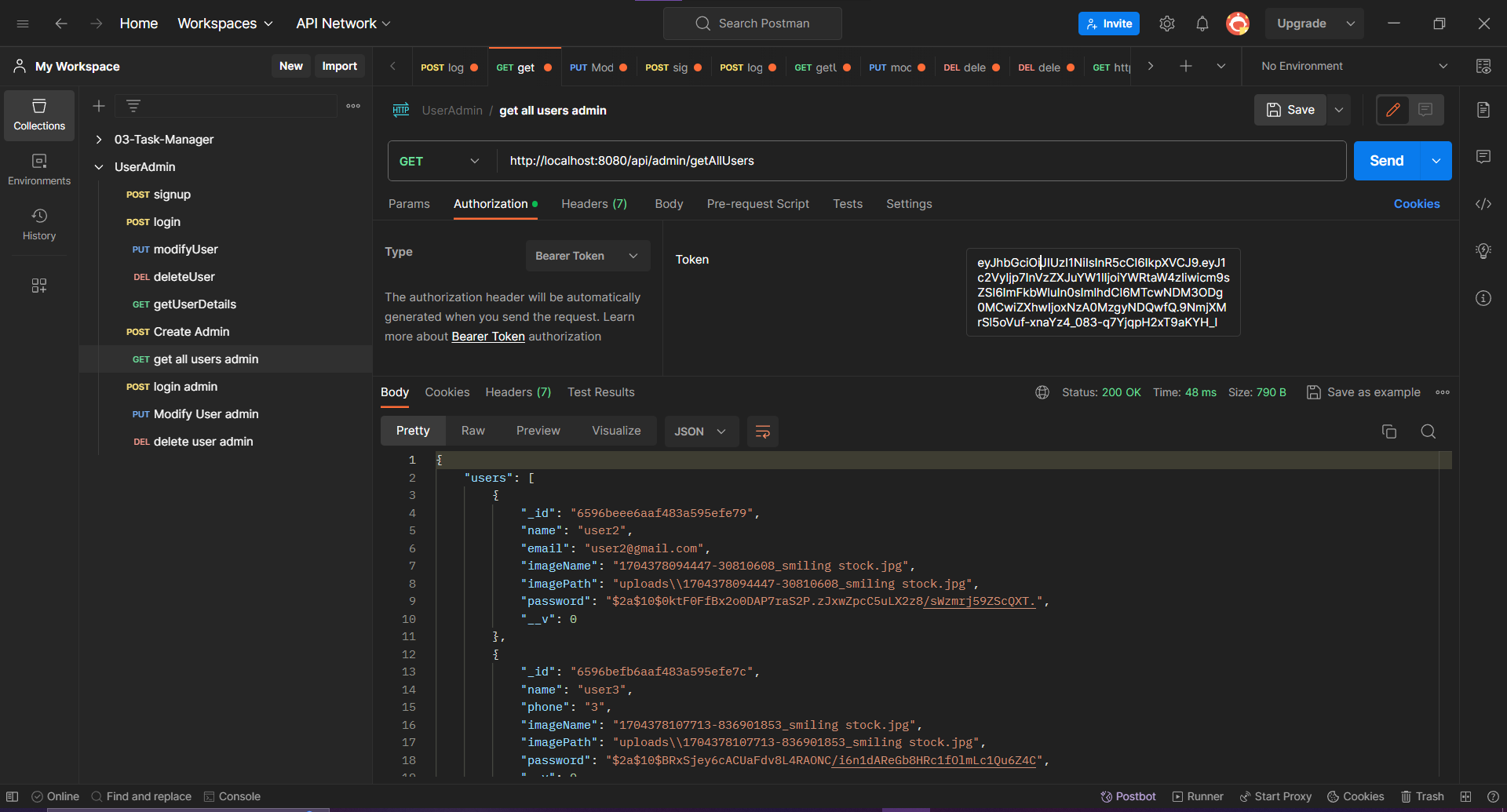
To login admin, send the username and password used for admin creation as JSON data. Upon logging in, you will get an access token. This access token is specific for admins. Access tokens used by users cannot be used to access admin routes.



**c) Admin getting details of all users**

http://localhost:8080/api/admin/getAllUsers

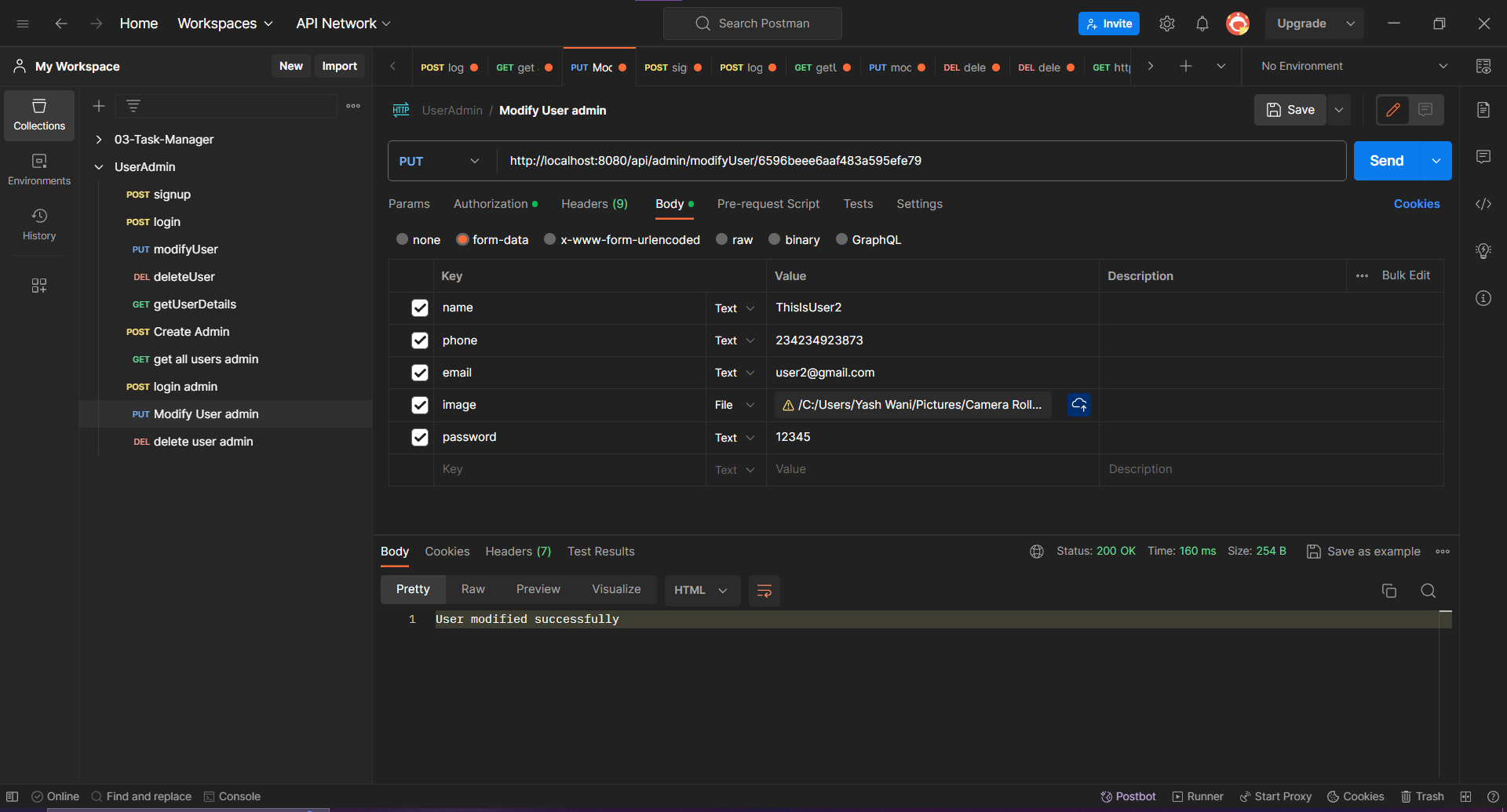
Pass the token generated while signing in as admin to access this route. For this, pass the token as a bearer token.



**d) Admin modifying user details**

http://localhost:8080/api/admin/modifyUser/6596beee6aaf483a595efe79

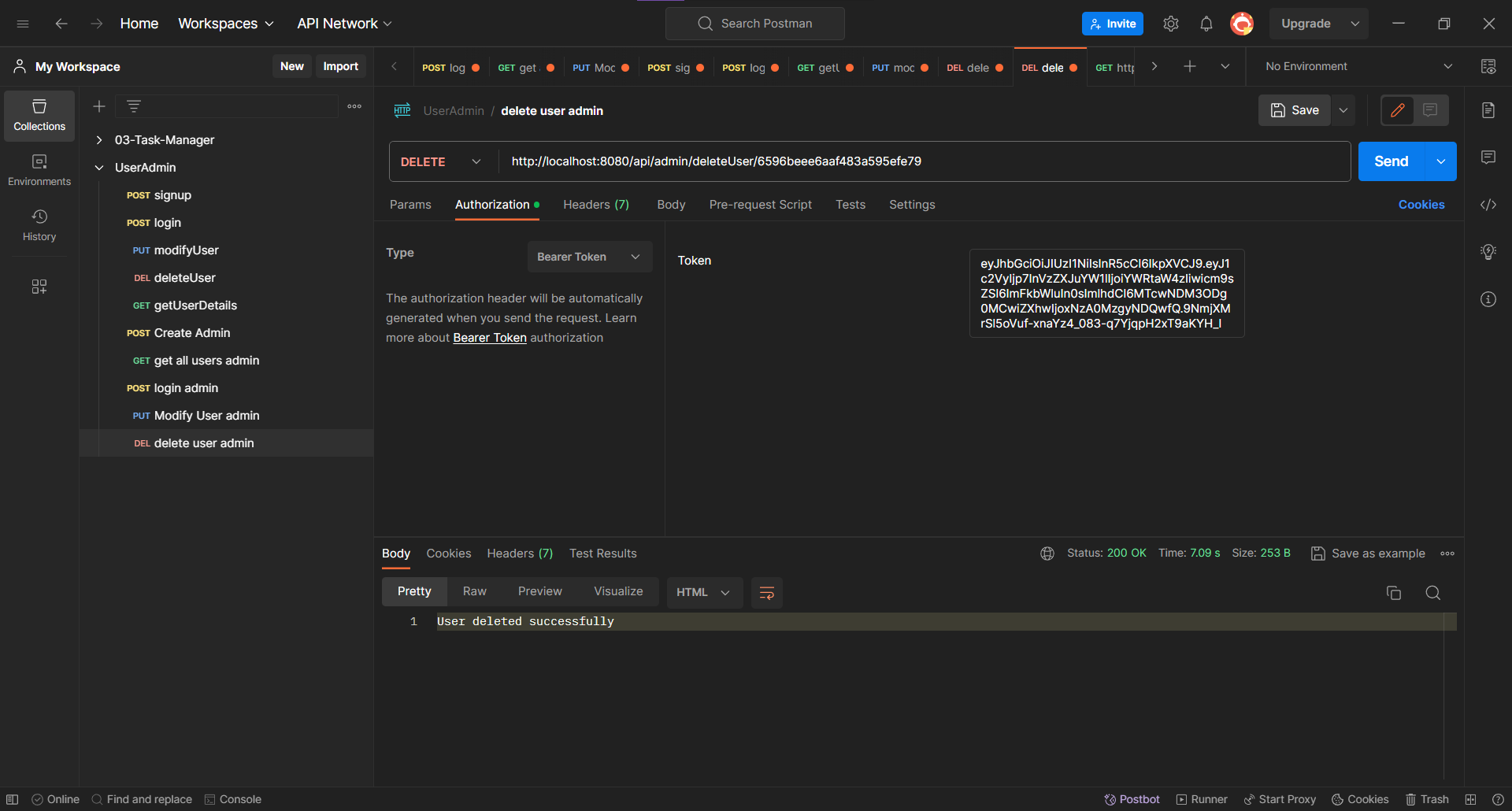
Admin can modify all the details of any user that they want. For this, they must send the id of the user that they want to modify as a query parameter. In the example we will modify details of “6596beee6aaf483a595efe79” user. Note that this is protected route and can only be accessed when JWT token generated at the time of admin login is passed.



**e) Admin deleting user details**

http://localhost:8080/api/admin/deleteUser/6596beee6aaf483a595efe79

Admin can delete any user by passing id as parameter and by sending JWT token as this is protected route.



The data is stored in MongoDB as so:  
