Fabfitfun :

Coding Challenge Documentation

By

Ali Sohaib Malik

# Requirements For Coding Challenge:

1. Create a user model. Consider what basic information needs to be captured, how these users should be uniquely identified and create a data object to represent this. As part of this step, please seed your data store with at least 3 users.   
2. Create an endpoint to add a new user to the system based on the model created in the first story.  
3. Create an endpoint to retrieve a user's information from the service based on a unique identifier.   
4. Create an endpoint to retrieve all users from the service.   
5. Create an endpoint to delete a user from the service.  
6. Create an endpoint that returns all users that meet certain criteria. i.e. return all users whose names are Joe. Think search engine.   
7. Create an endpoint to edit a user.  
8. Create a way to assign roles or permissions to a user. Then, protect a specific endpoint based so that only users of a particular status can utilize it.   
9. Choose your own adventure. As engineers, we constantly come up with ideas for improvement, or see gaps in planning. Please take this user story as a chance to add something that you think the service could use, whether it's an endpoint, a performance improvement, or something totally unrelated that you would like to show off!

1. USER MODEL :-

Location and File Name :

***com.challenge.fabfitfun.domain.User.java***

**Basic Structure : -**

Long id;

String firstName;

String middleName;

String lastName;

Date dateOfBirth; //Format On Entry: YYYY-MM-dd

String address;

String role; //Value can be USER or ADMIN

Initial Entries may be found in SQL file at the following location in project:-

***src/main/resources/Data.sql***

Initial Entries consist of three users where one has the ADMIN role.

***INSERT******INTO******USER*** *(FIRST\_NAME, MIDDLE\_NAME, LAST\_NAME, DATE\_OF\_BIRTH, ADDRESS, ROLE)* ***VALUES*** *('Daniel', '','Broukhim', SYSDATE, 'Los Angeles', 'ADMIN');*

***INSERT******INTO******USER*** *(FIRST\_NAME, MIDDLE\_NAME, LAST\_NAME, DATE\_OF\_BIRTH, ADDRESS, ROLE)* ***VALUES*** *('Katie', 'Rosen','Kitchens', SYSDATE, 'Los Angeles', 'USER');*

***INSERT******INTO******USER*** *(FIRST\_NAME, MIDDLE\_NAME, LAST\_NAME, DATE\_OF\_BIRTH, ADDRESS, ROLE)* ***VALUES*** *('Michael', '','Broukhim', SYSDATE, 'Los Angeles', 'USER');*

1. ADD USER (Restricted By Role as per requirement 8) :-

Note : Only ADMIN users can add new users.

URL for adding new users, Request type should be POST :

(POST) Server\_IP : Server\_Port /api/v1/user/{**id**}

The id should belong to an ADMIN user.

In the body of the request please provide the information regarding the user to be added like below:-

{

"firstName" : "First",

"middleName" : "Middle",

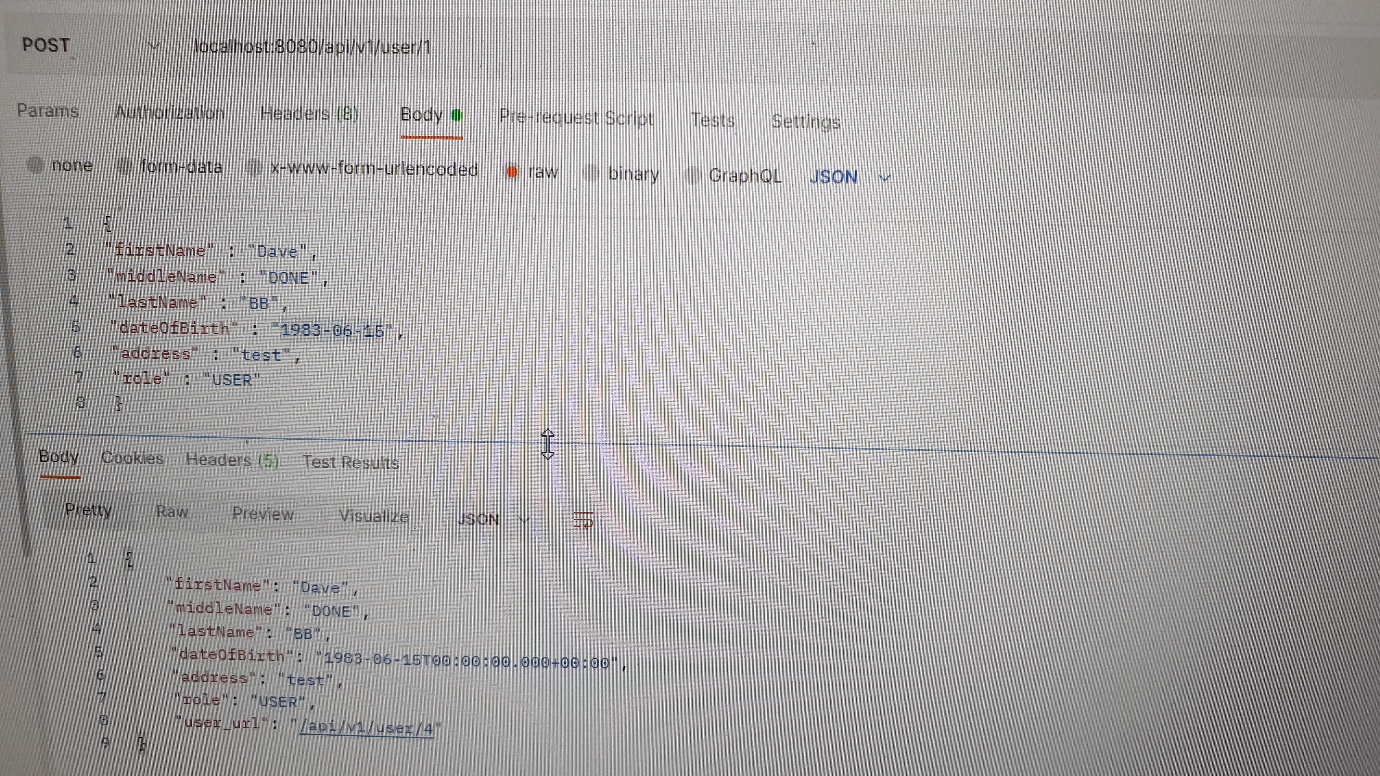
"lastName" : "Last",

"dateOfBirth" : "1947-8-25",

"address" : "LAX",

"role" : "ADMIN"

}

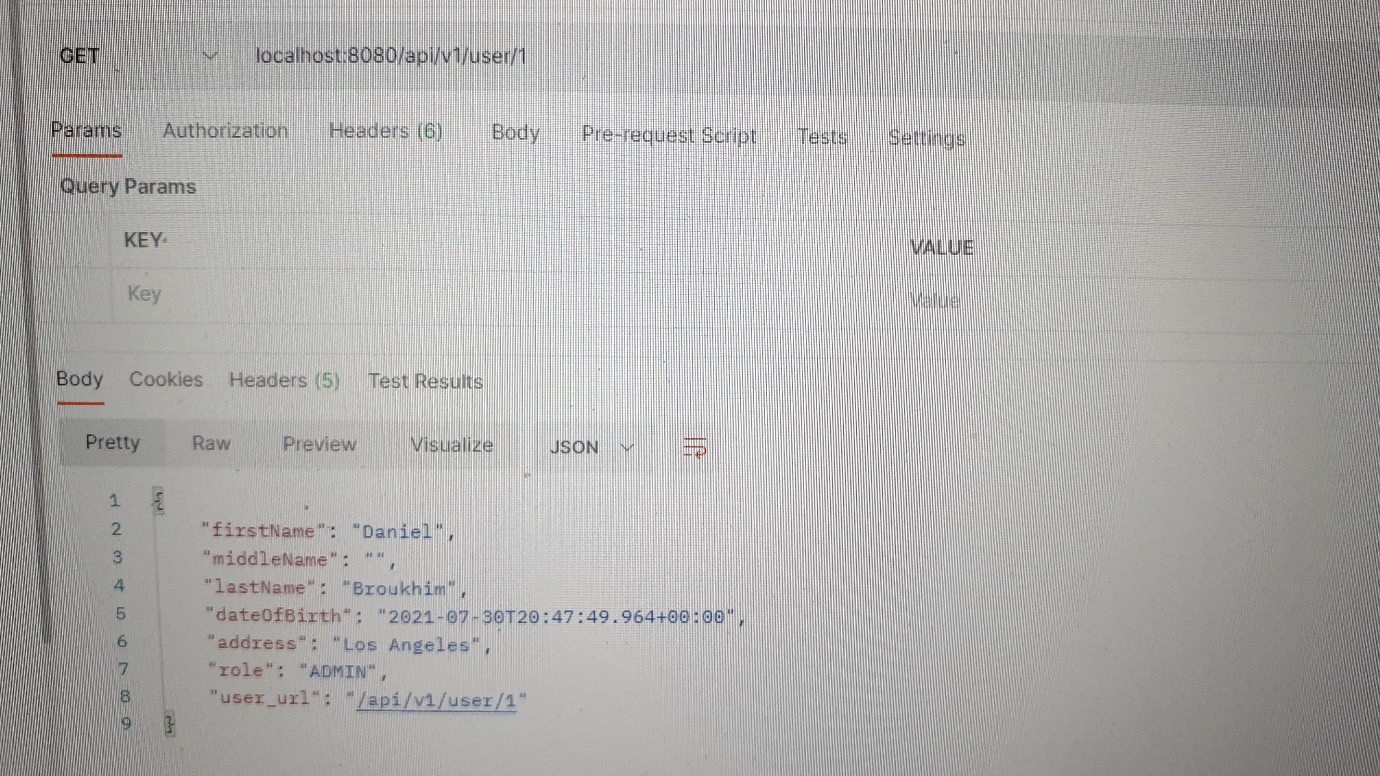


1. GET USER BY ID :-

URL for adding new users, Request type should be GET :

GET) Server\_IP : Server\_Port /api/v1/user/{**id**}

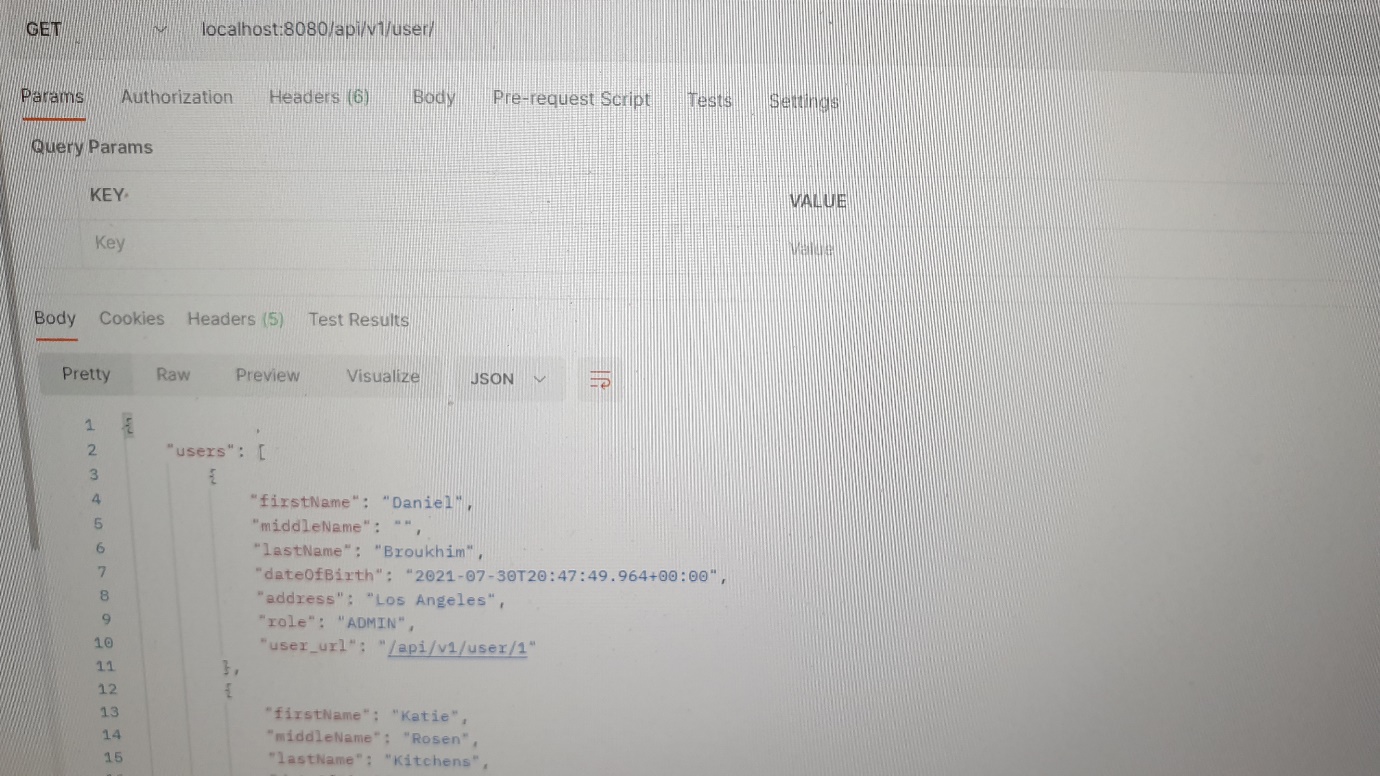
The id should belong to the user to be retrieved.



1. GET ALL USERS :-

URL for adding new users, Request type should be GET :

(GET) Server\_IP : Server\_Port /api/v1/user/

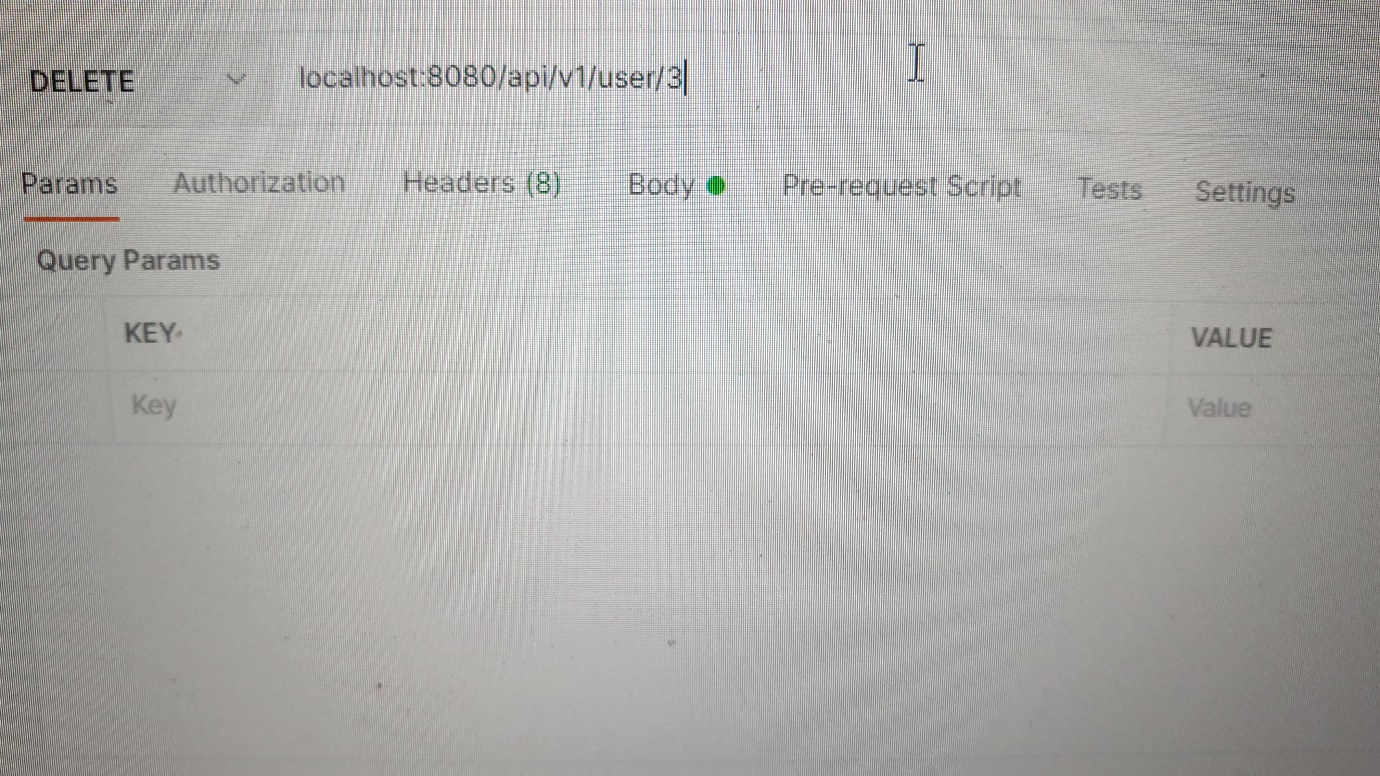


1. DELETE USER BY ID :-

URL for deleting a user, Request type should be DELETE :

(DELETE) Server\_IP : Server\_Port /api/v1/user/{**id**}

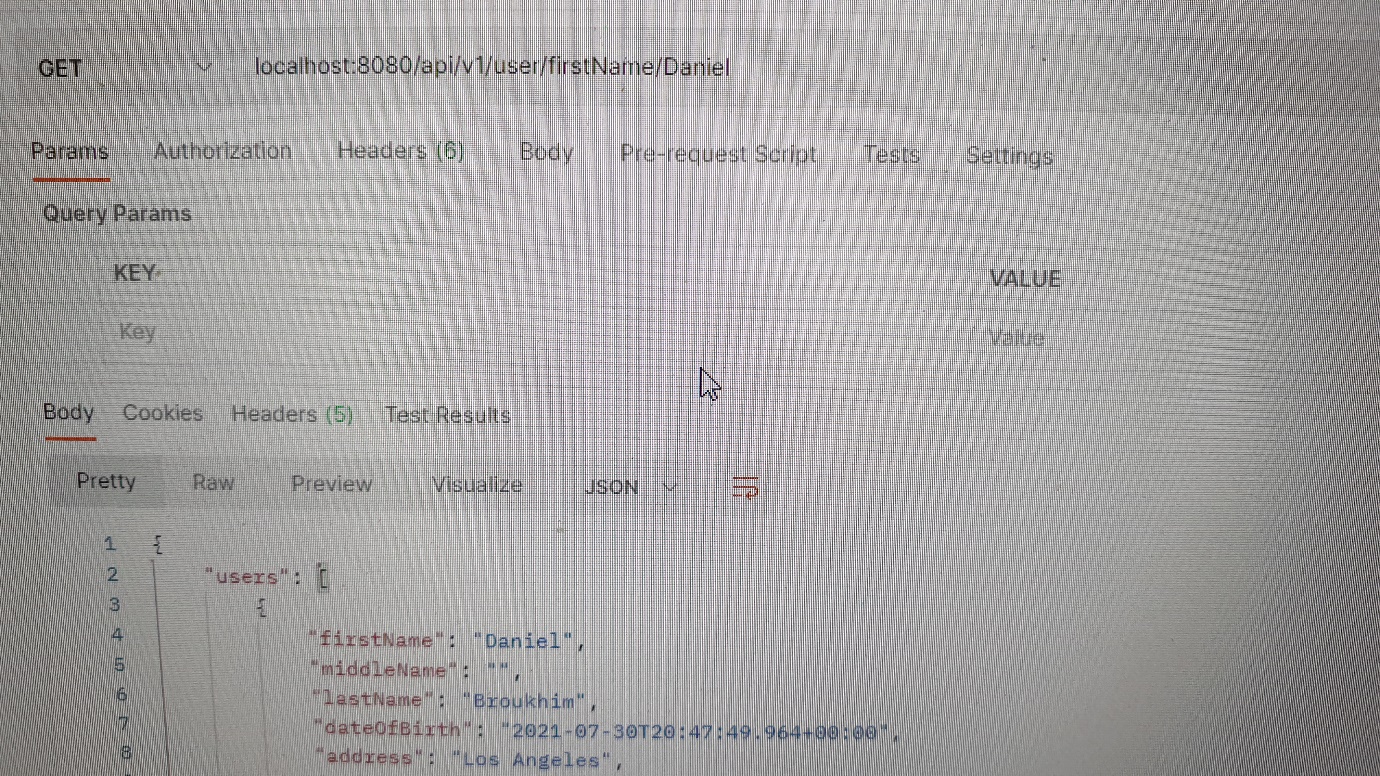
The id should belong to the user to be deleted.



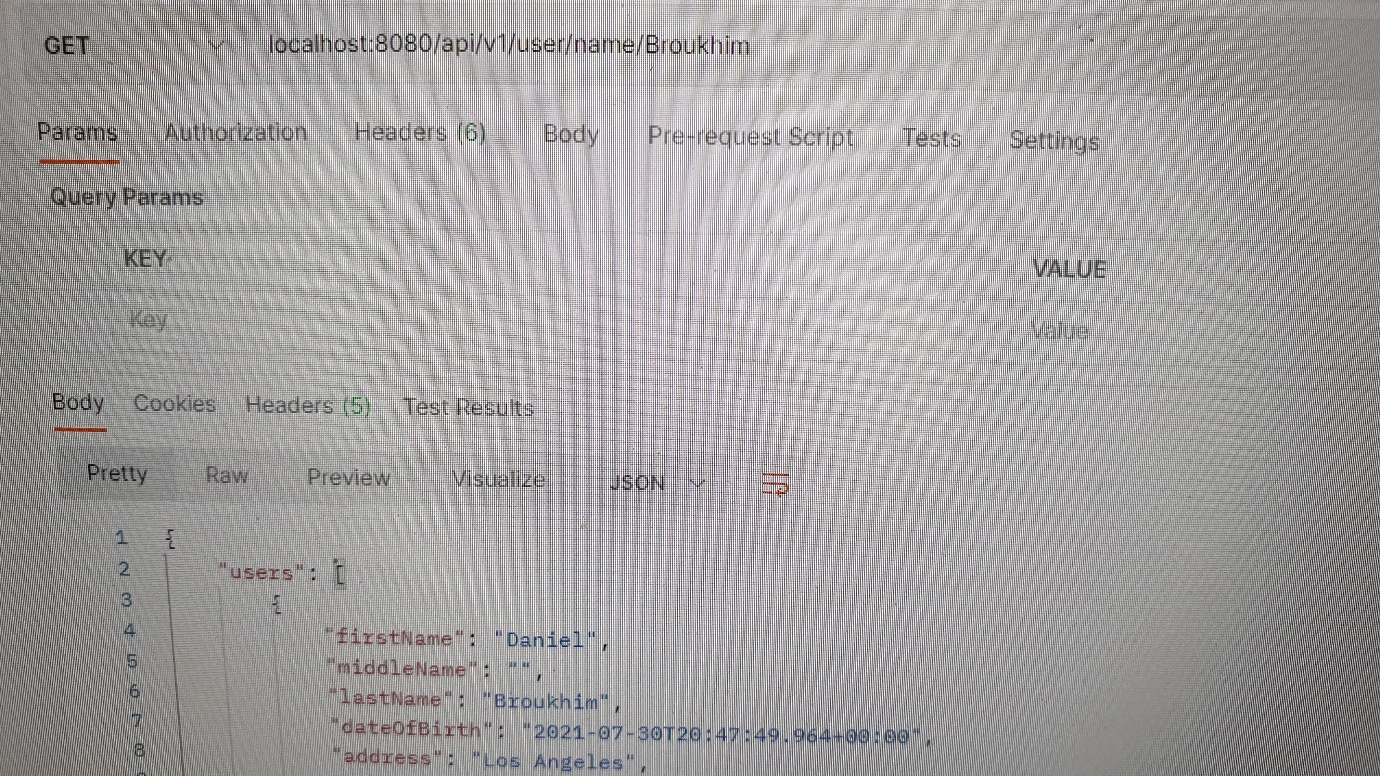
1. SEARCH :-

There are four calls to search users by.

* Search By First Name
  + URL for searching by First Name, Request type should be GET :
  + (GET) Server\_IP : Server\_Port /api/v1/user/firstName/{firstName}
* Search By Middle Name
  + URL for searching by Middle Name, Request type should be GET :
  + (GET) Server\_IP : Server\_Port /api/v1/user/middleName/{middleName}
* Search By Last Name
  + URL for searching by Last Name, Request type should be GET :
  + (GET) Server\_IP : Server\_Port /api/v1/user/lastName/{lastName}



* Search By Name (any)
  + URL for searching by Name (any), Request type should be GET :
  + (GET) Server\_IP : Server\_Port /api/v1/user/name/{name}



1. EDIT USER :-

URL for editing users, Request type should be PUT :

(PUT) Server\_IP : Server\_Port /api/v1/user/{**id**}

The id should belong to the user to be edited.

In the body of the request please provide the information regarding the edited information abou the user like below:-

{

"firstName" : "First",

"middleName" : "Middle",

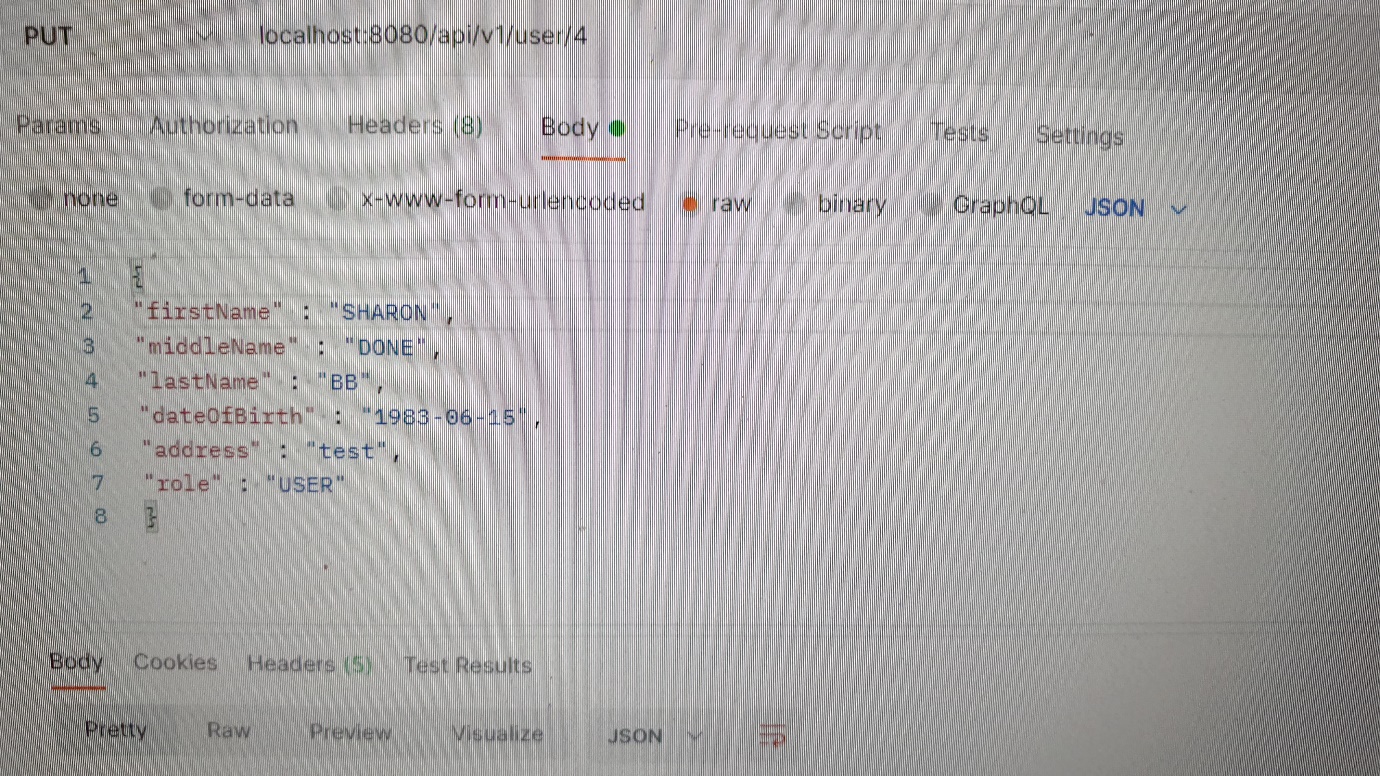
"lastName" : "Last",

"dateOfBirth" : "1947-8-25",

"address" : "LAX",

"role" : "ADMIN"

}



1. NOTES AND TO BE ADDED FEATURES:-

This application is still lacking multiple features to make it more robust and production ready.

These features include:-

Validation Of Data (Not Implemented)

Better Error Handling (Bare-Bones Error Handling has been implemented).

Logging (Not Implemented)

Commenting of Code (Not Implemented)

Better Role Management/Restriction :- Current Role Restrictions is on one call and rudimentary.

A better path would have been a check on Delete, Edit and Create calls that would allow only ADMINS to create ADMIN users or normal ones while regular users could not create.

A similar check on Delete and Edit calls would allow users to either delete or edit their own entries while only allowing ADMINS to delete everyones.