User-Authentication-System-with-NodeJs

*Akash patro*

*4th year cse student*

***Deployed Links:***

<https://hsecureauth.herokuapp.com/> => Default Route

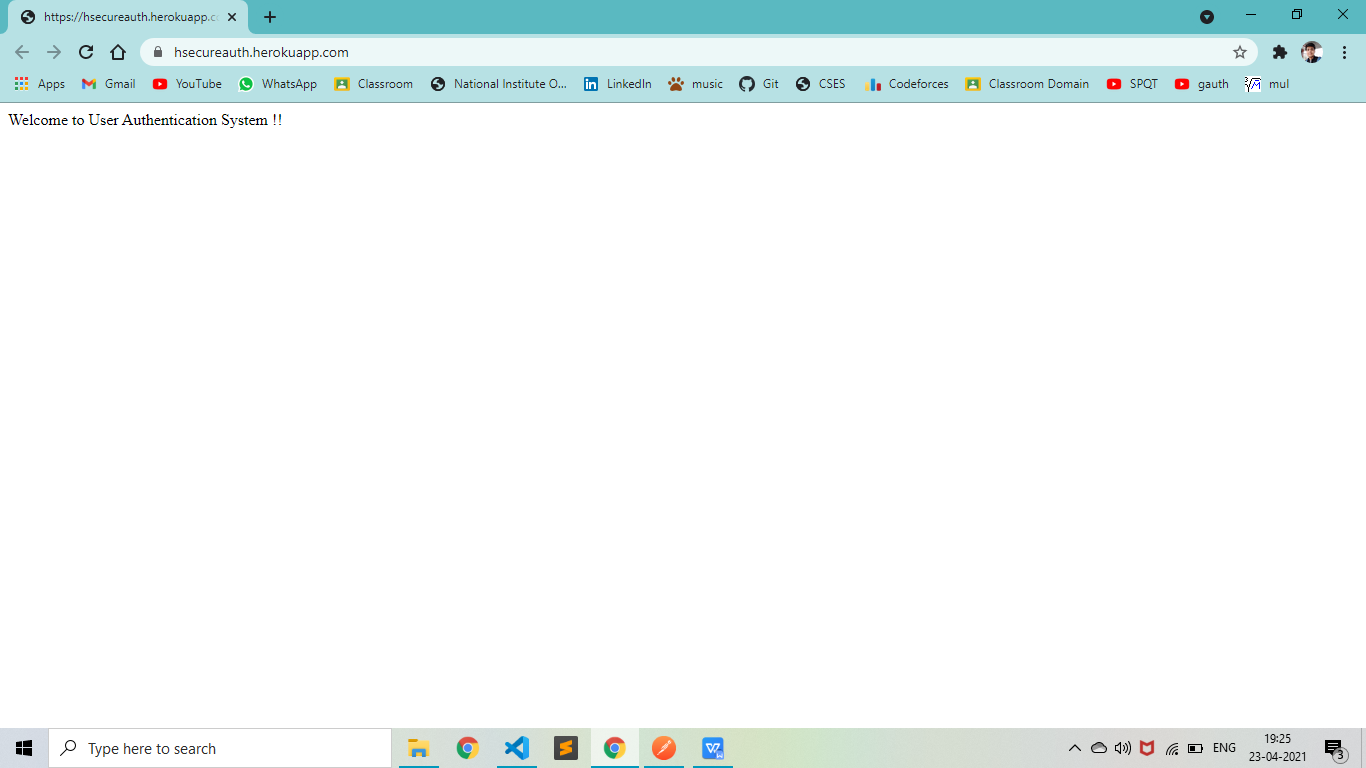
<https://hsecureauth.herokuapp.com/user/signup> => POST

<https://hsecureauth.herokuapp.com/user/login> => POST

<https://hsecureauth.herokuapp.com/user/logout> =>GET

<https://hsecureauth.herokuapp.com/user/dump> =>GET

<https://hsecureauth.herokuapp.com/home/index> => GET



1. ***. Register a User :-***

API: <https://hsecureauth.herokuapp.com/user/signup> => POST

Input in ***req.body*** in ***json*** format: -

{

    "firstname": "<firstname>",

    "lastname": "<lastname>",

    "email": "<email>",

    "password": "<password>"

}

This api will first check for any similar email id already registered.After that it will create a new user object which will store all these details.

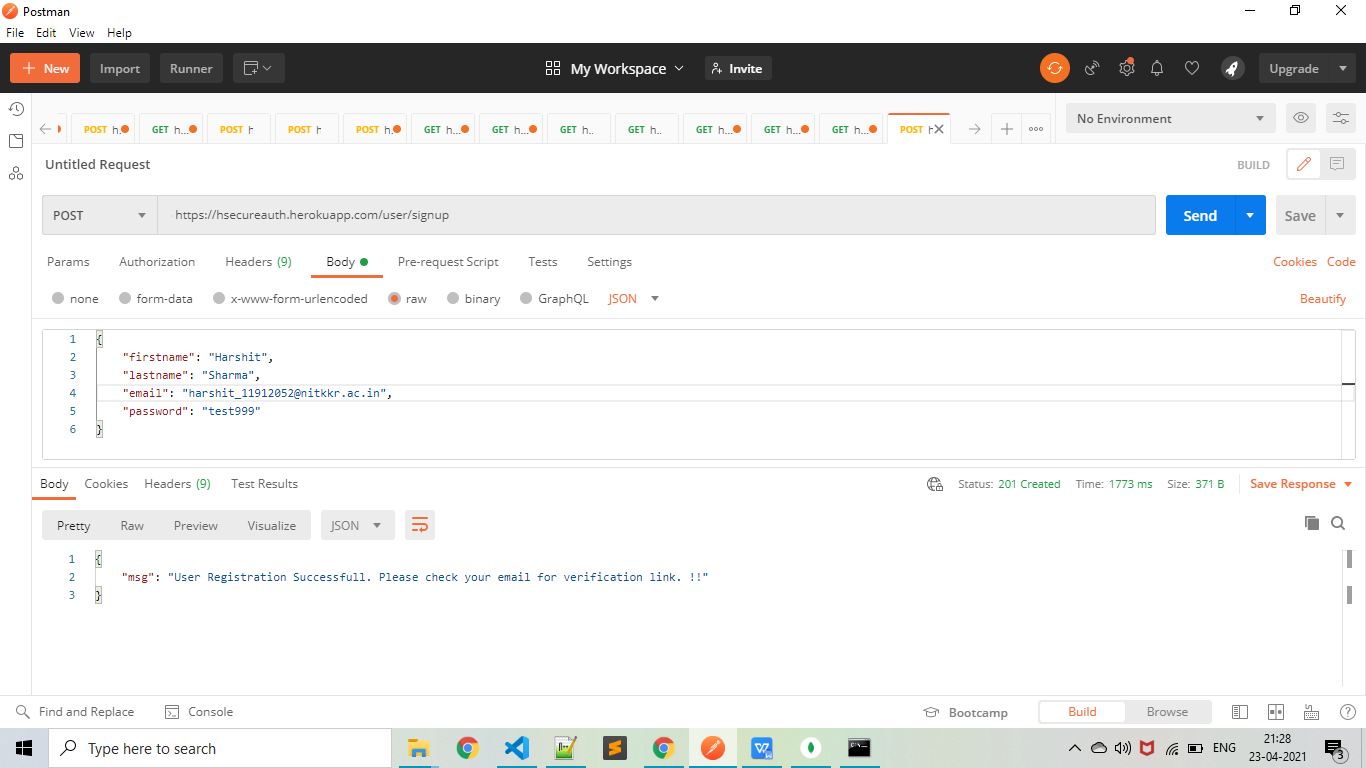
The password will be first converted into a encrypted format using bcrypt (hash and salt) and then added to the user object.

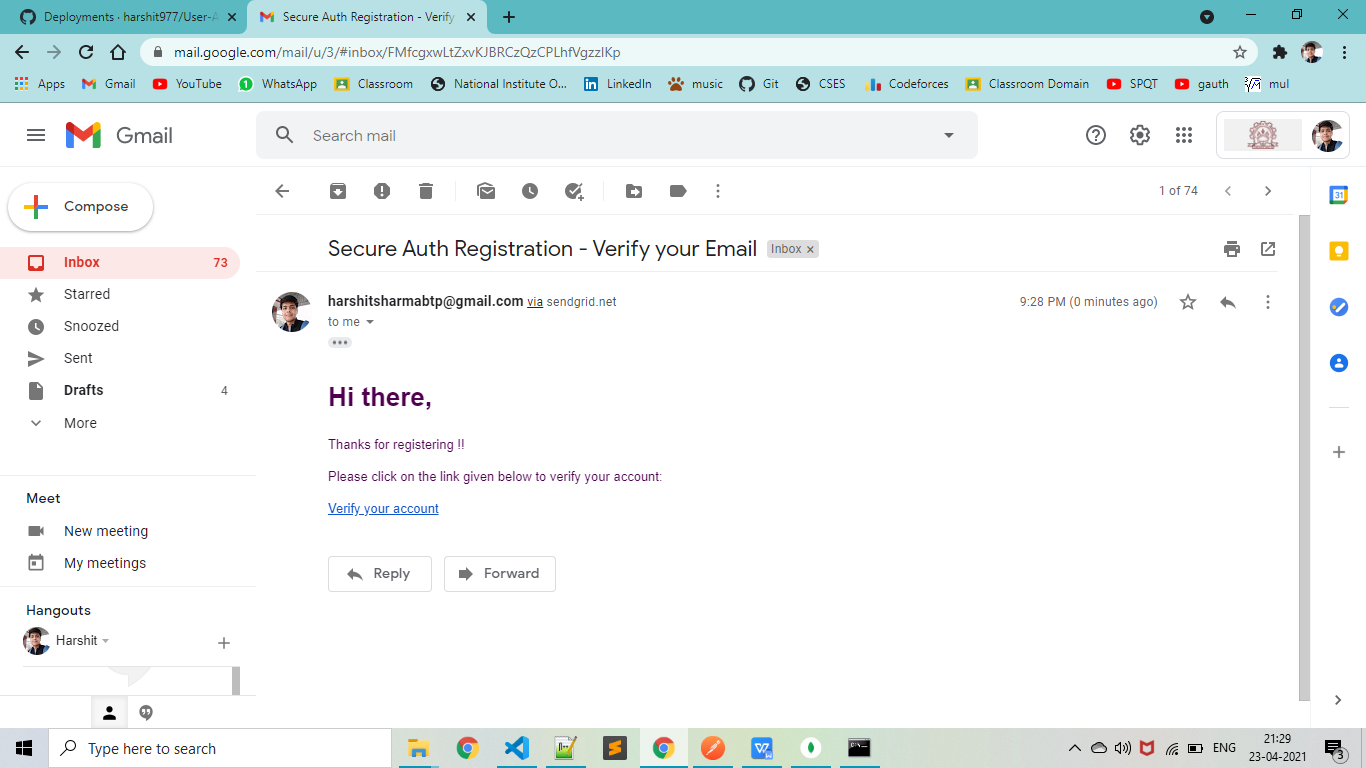
The api will send an email to the given email address to verify the user via ***Sendgrid*** . When the user will click on the link on his/her email, the verify email api will be called :

[https://hsecureauth.herokuapp.com/user/verify-email?token=<token>](https://hsecureauth.herokuapp.com/user/verify-email?token=<token> )

Clicking on this api link will set the isVerified field in schema to true.

And, the user object will be saved into the database.





1. ***. Login a User :-***

<https://hsecureauth.herokuapp.com/user/login> => POST

{

     "email": "<email>",

    "password": "<password>"

}

=> Only the verified user (isVerified=true) can log in.

=>this api will verify the email and password credentials from database.

The password is compared by using bcrypt function.

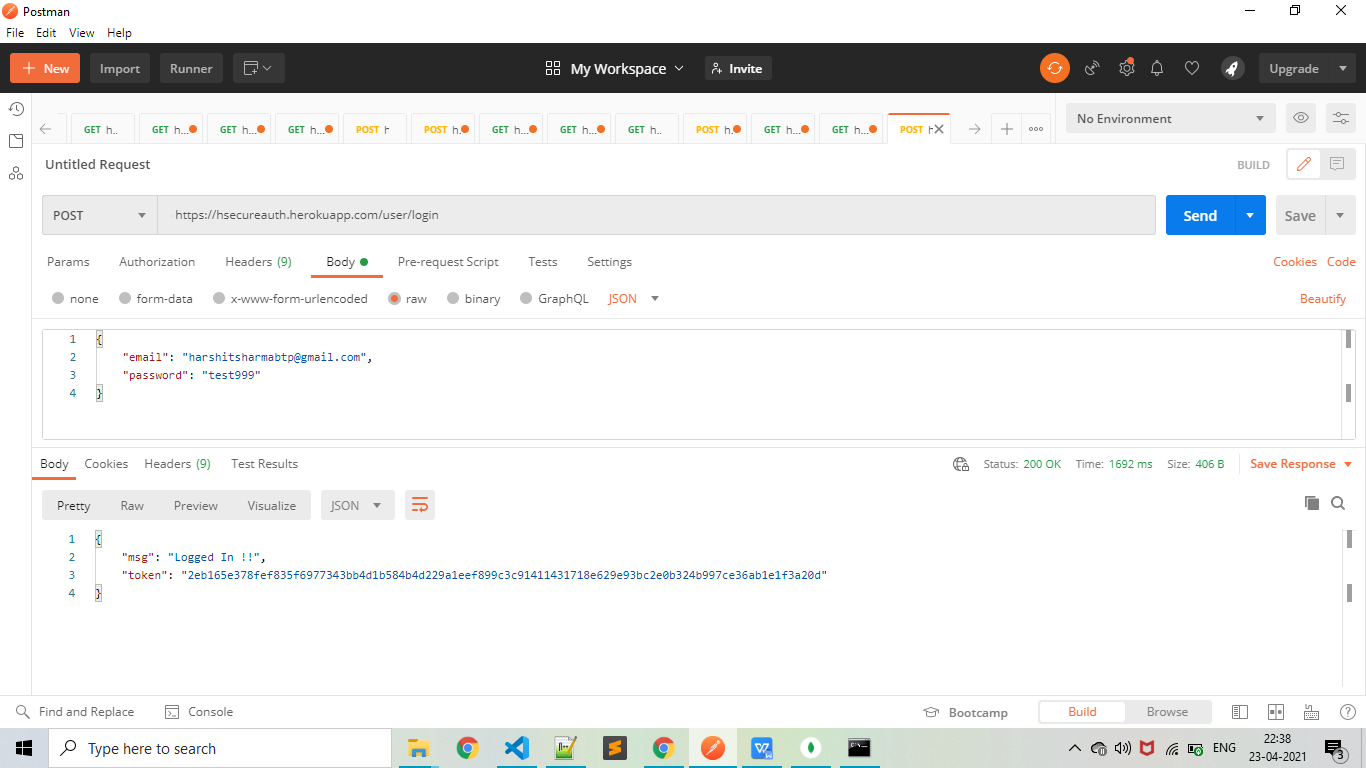
=> on successfull verification =>

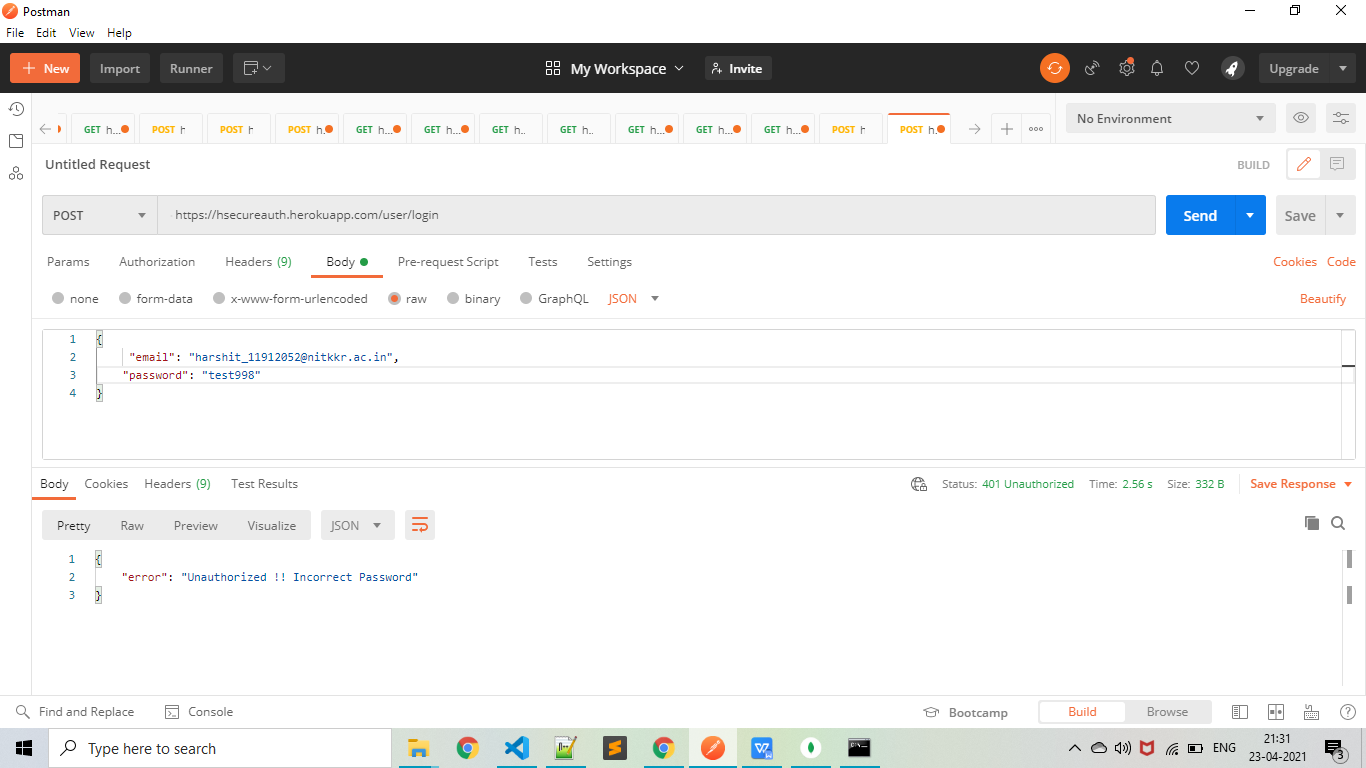
=> A **token** is generated a using ***crypto*** by encrypting (***user.\_id*** and ***expiry time***) String using a ***secret key***.

=> ***This token is stored in tokens array in user object***.

=>  ***this token is sent back to user in response.***

***=> Token Exipry Time : 1hr (can be easily increased or decreased accordingly )***





***( C) verifyUser Middleware: -***

=> this middleware will be required to access protected routes.It will ensure that only logged in user can acces that route.

Protected Route:

<https://hsecureauth.herokuapp.com/home/index> => GET

=> To access this route , the ***login token*** needs to be passed as a **Bearer <token>** in authorization header.

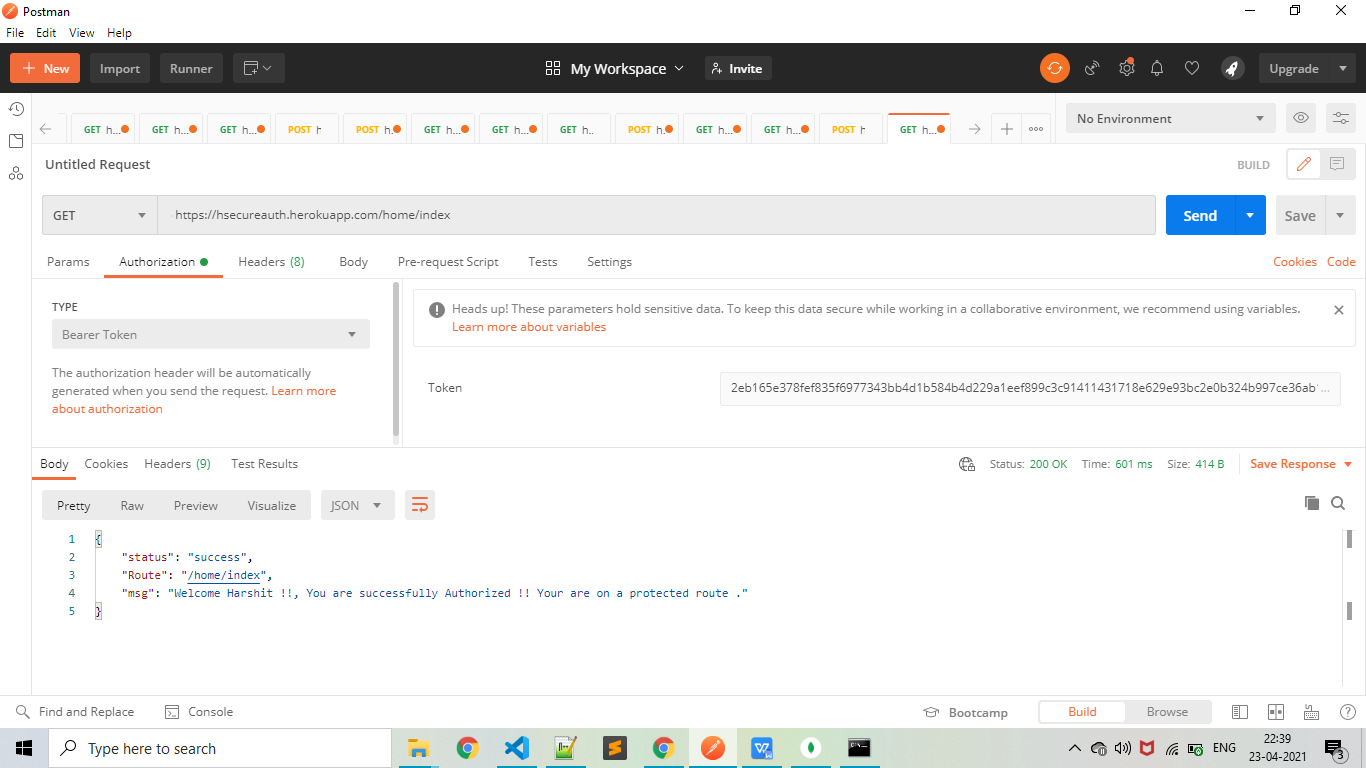
=>this verifyUser middleware will extract ***id*** and ***expiry time*** from this token using ***crypto*** and ***secretkey. (decrypting the encrypted token)***

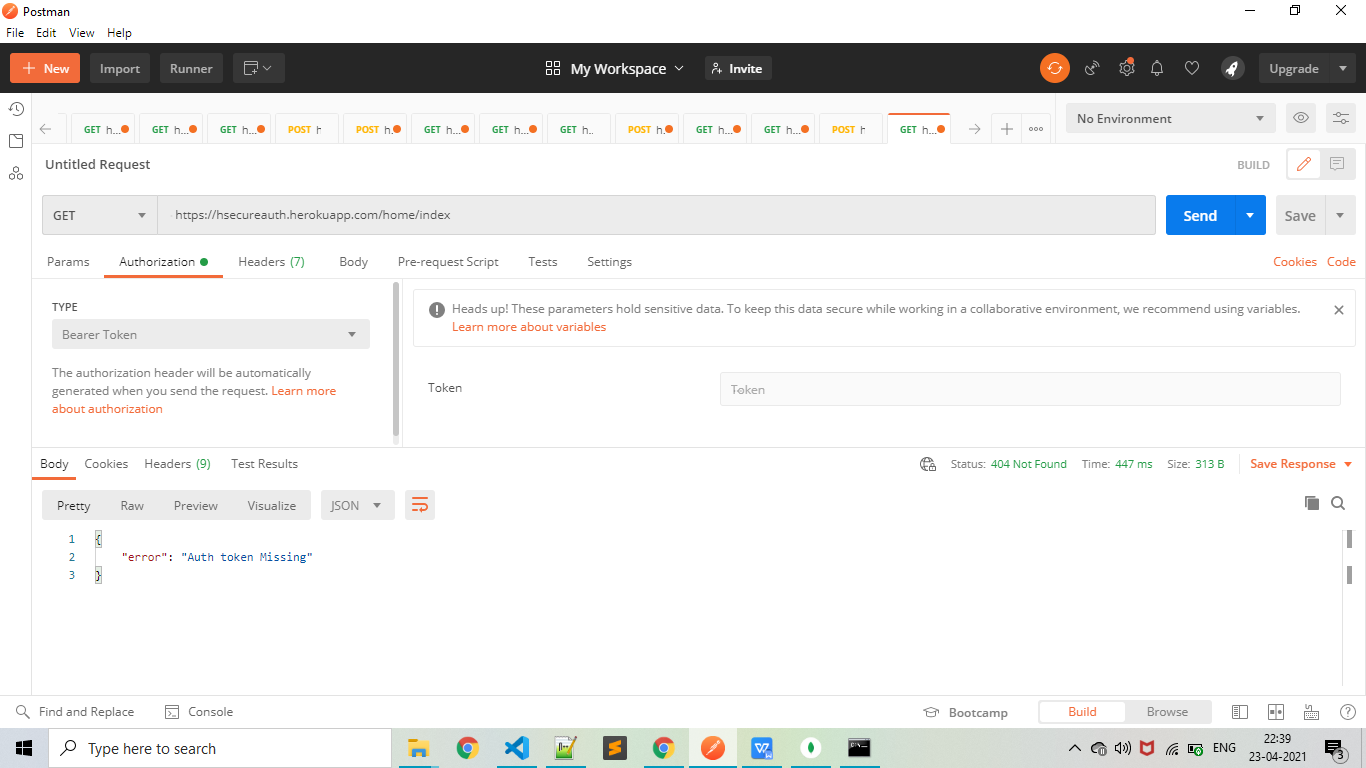
***=>If*** (expiry time<current time) => ***token is expired = > UnAuthorized***

Else, it will search for the user having ***\_id=id and tokens.token=token***

=> if such user is found = > ***Authenticated*** = > next()

=> else ***UnAuthorized***





***( D ) Logout User: -***

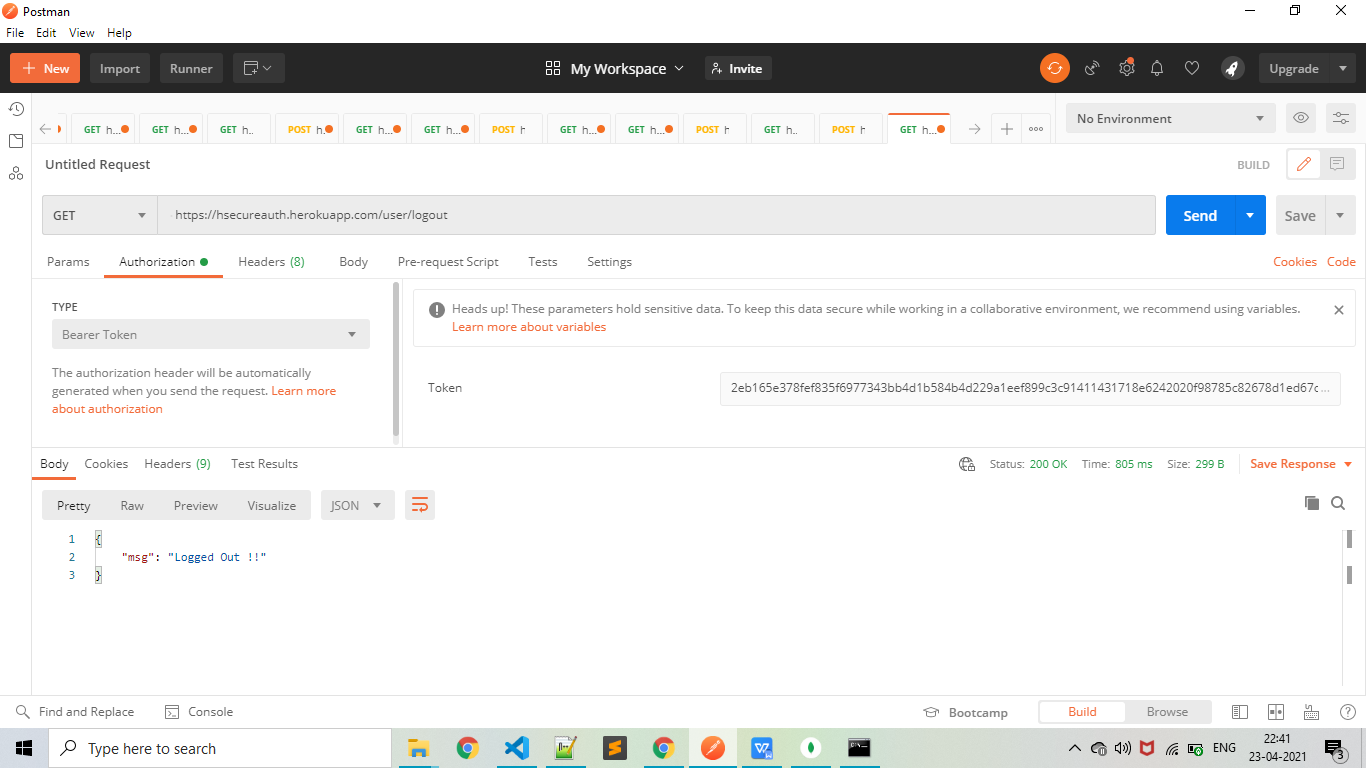
<https://hsecureauth.herokuapp.com/user/logout> =>GET

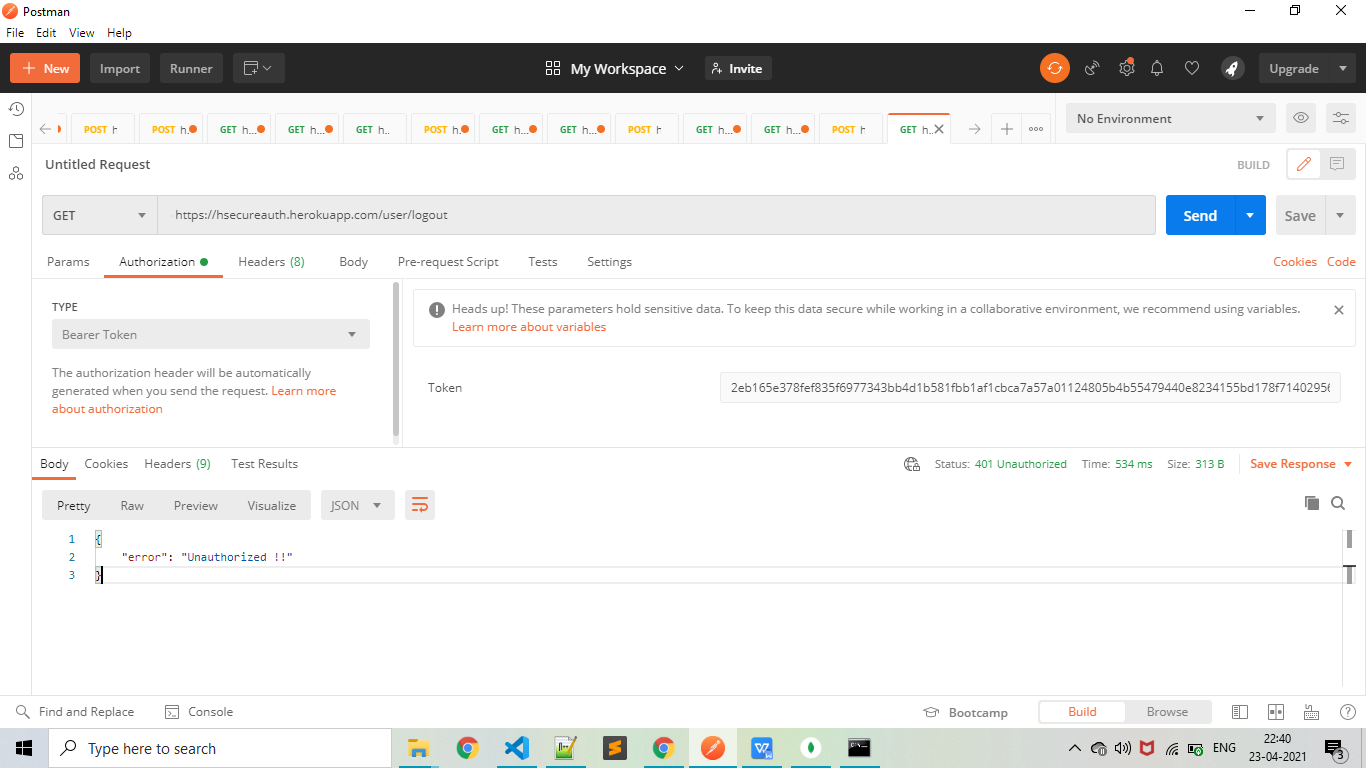
=> this will also use verifyUser Middleware to ensure that only logged in user can log out.

=> if that is succes => then

=> verifyUser Middleware will set ***req.user=user*** and ***req.token=token*** fields in the request object.

=> So, the logout api will **traverse** through the **array of tokens** of **req.user** object and it will remove the **tokens which are expired** as well as the current token(**req.token==req.user.tokens.token**).





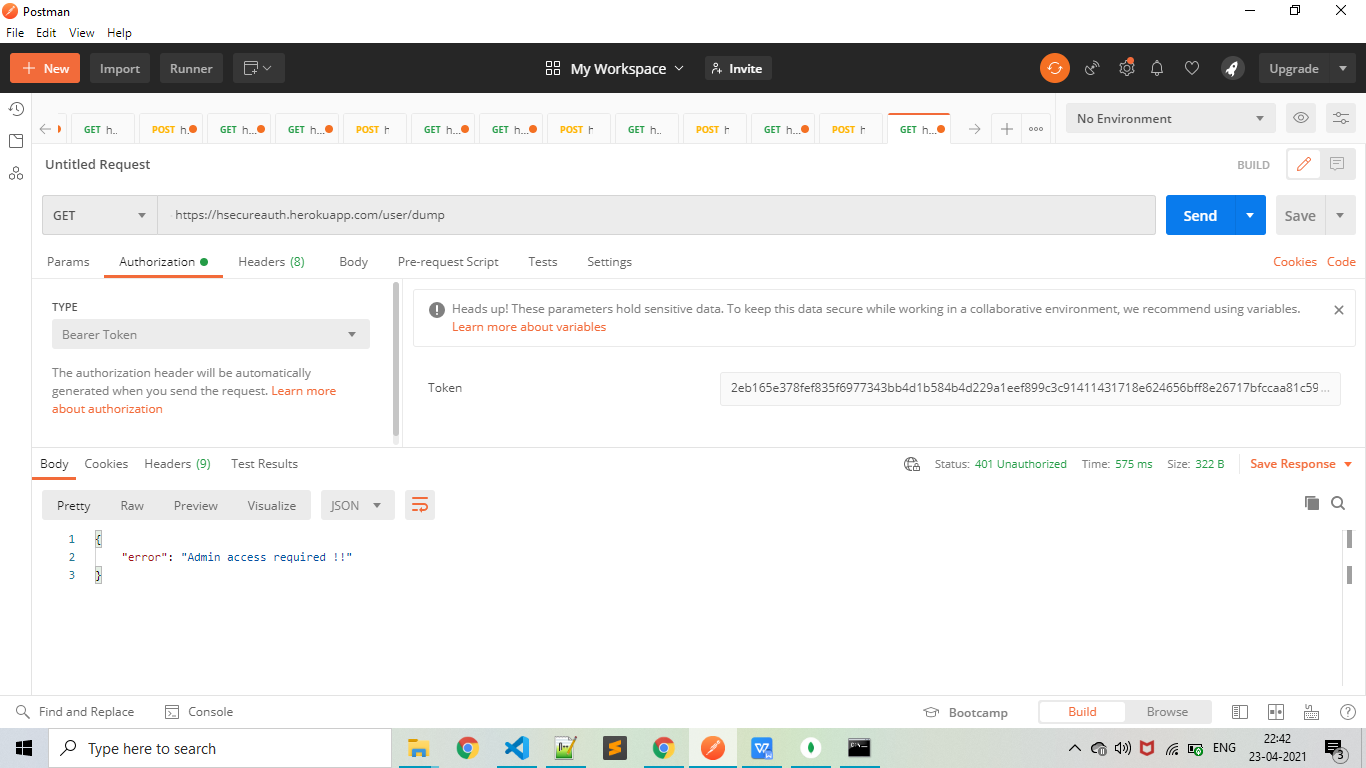
***( E ) dump users: -***

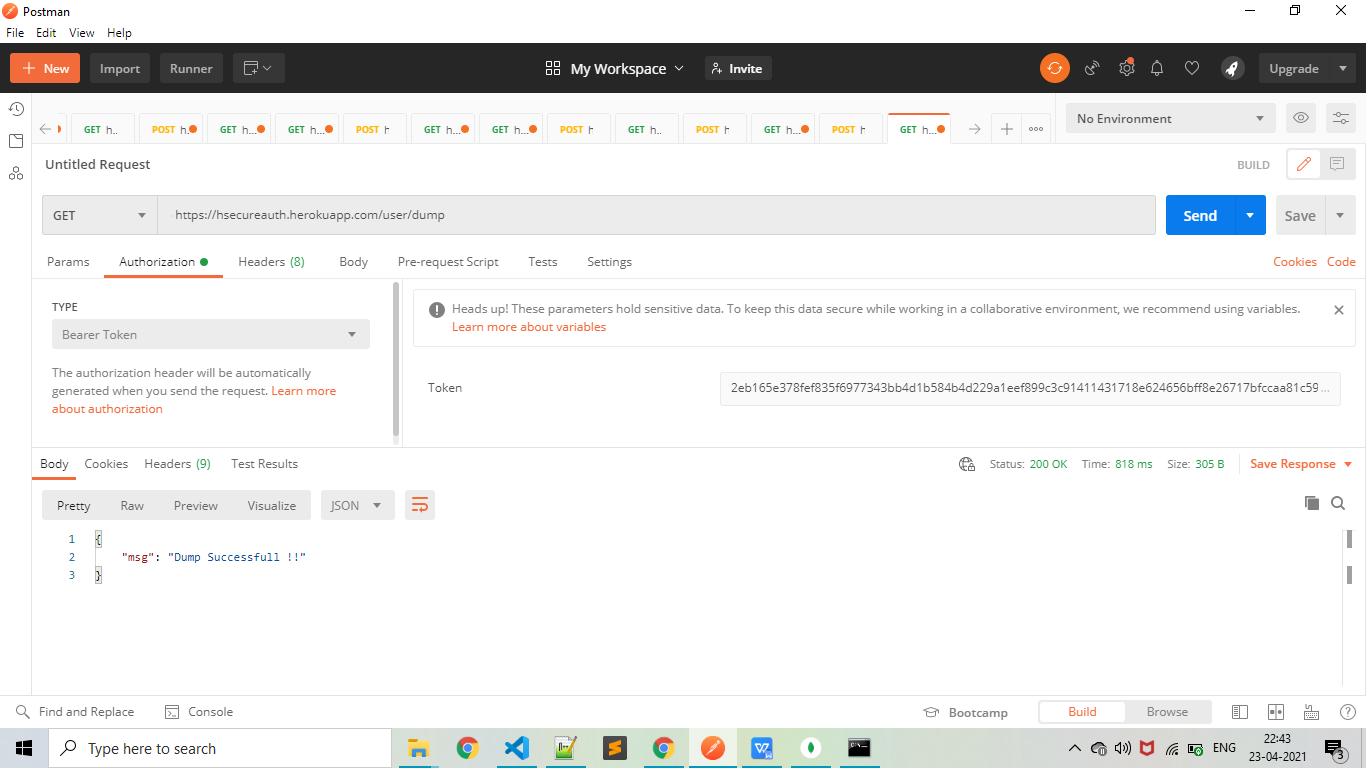
<https://hsecureauth.herokuapp.com/user/dump> =>GET

=> This route can only be accessed by the **admin**( user having admin: true)

=> it will use verifyUser and verifyAdmin middlewares to ensure that only logged in admin can dump the users.

=> ***this api will finally delete all the users except the admin user.***





*Here are the admin credentials for your reference to run & verify this API :-*

{

    "email": "harshitsharmabtp@gmail.com",

    "password": "test999"

}

***Github Repository***

***https://github.com/MeAkash77/User-Authentication-System***