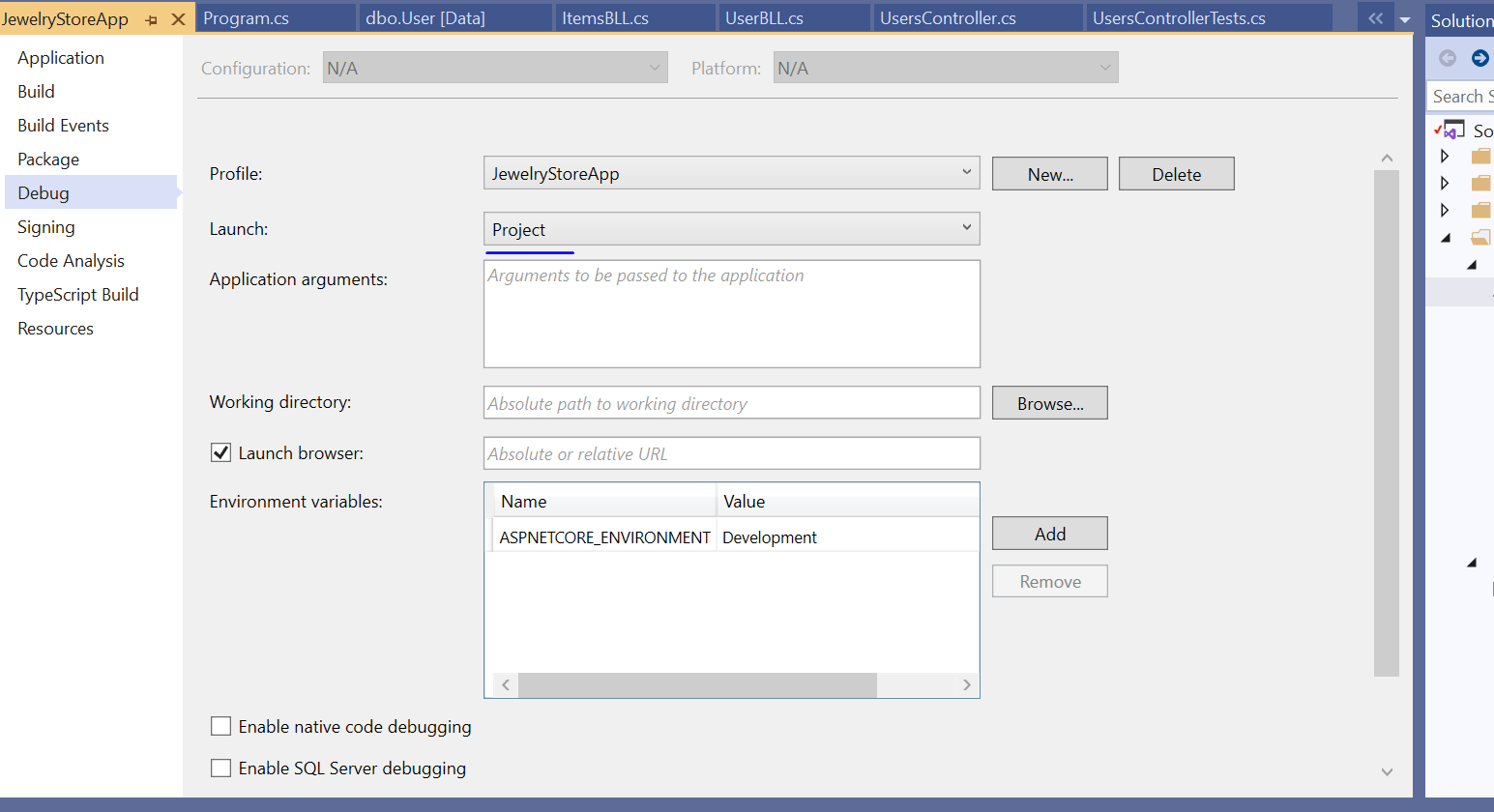
1. The “JewelryStoreApp” web app solution under the folder “JewelryStoreApp.Web.App” is to be set up as the main startup project. Please note the following :-

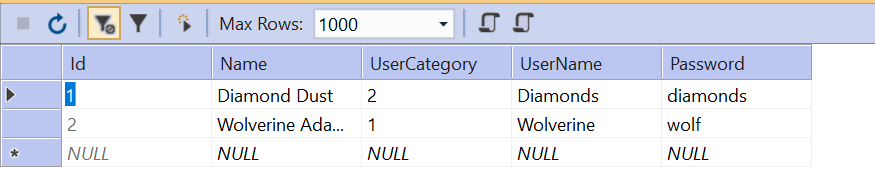
* “Http” has been used instead of “Https” because of my SSL associated settings missing in my system. **The site listens at port number 4242.**
* There are apis for get variants, create, update and delete. However, I have shown the use cases for the apis that are most relevant to the problem statement
* Unit tests have been written for as many scenarios I could think of (both users and items + I am new to writing them). I have tried to cover whatever scenarios I could think of. The files are under “JwelleryStore.Tests” solution under the “JewelryStoreApp.Web.App” folder
* Run the application in the “Project” launch mode as shown below. Logging has been enabled to print logs on to the console window.



1. Seed the data using the following steps :-

* Go to appsettings.json file and set the “Seed” entry to “true” to start the migration and populate the tables [Do not forget to enable the entity framework migrations before doing this step]
* The database created in localDB is named as “JewelryStoreDb” . There will be a total of 3 tables, namely Item, User and UserDiscount.
* The “Item” table has 2 metals “Gold” and “Silver”.
* The “Users” table has 2 users. One is a regular user and another is a privileged user.
* The “UserDiscount” table is a relationship table that contains a user mapping, an item mapping and a discount. The assumption is that One user can be given different discounts for different items [Privileged users only].

Below is the snapshot of the User table. One can use this to enter the “Username” and “Password” in the test cases given in the further steps**. A UserCategory of “1” is mapped to a Privileged User and a value of “2” is mapped to a “Regular User”. For e.g. “Wolverine” is a privileged user whereas “Diamonds” is a regular user.**



1. The second step is to authenticate the user[using usercontroller] and obtain a JWT token in order to authorize the user to have access to items[itemscontroller]. This can be done using the following url :-

**UsersController**

**Use Case 1:-**

**Input Url** :- **[POST Request to authenticate the user and obtain a JWT token in order to access items]. In this case, I am logging in as a “Privileged User”. For a snapshot of a Postman login request, refer Use Case number 4.**

**http://localhost:4242/Users/login**

**Body :-**

{

"userName": "wolverine",

"password": "wolf"

}

Output:-

**{**

**"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bmlxdWVfbmFtZSI6IndvbHZlcmluZSIsIlVzZXJUeXBlIjoiUHJpdmlsZWdlZCIsIm5iZiI6MTYxNjcwMDQ3OCwiZXhwIjoxNjE3MzA1Mjc4LCJpYXQiOjE2MTY3MDA0NzgsImlzcyI6Ikpld2VscnlTdG9yZS5Ub3BJc3N1ZXIiLCJhdWQiOiJKZXdlbHJ5U3RvcmUuVG9wSXNzdWVyIn0.YQe8HwkDFhdU\_H-X0zQ\_1cMg43\_7P2Md4oZP-MiSEiI"**

**}**

**ItemsController**

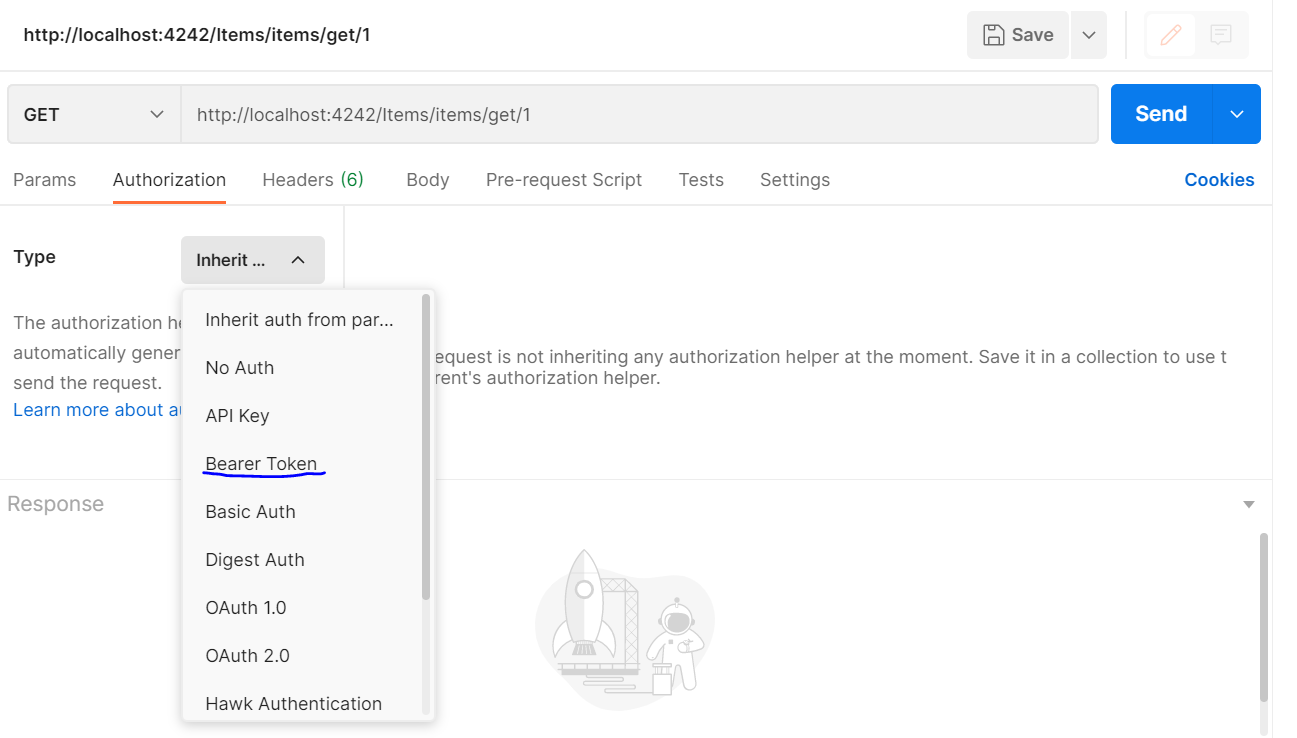
Once the token has been received as a response, a request has to made to the “ItemsController” to access the APIs(authorization). That can be done using the following steps :-

**Use Case 2 : To Obtain an Item**

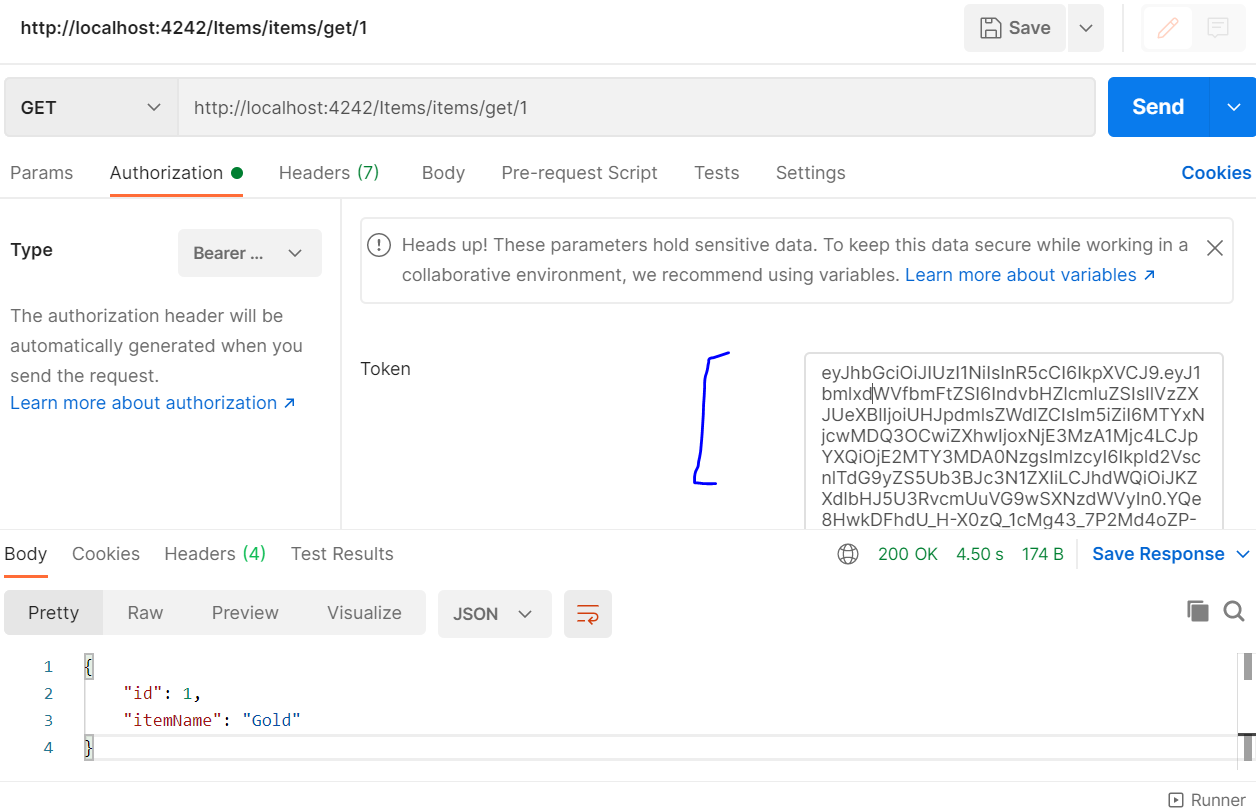
**Input :-**

http://localhost:4242/Items/items/get/1

* Go to postman -> enter the above url -> Go to “Authorization” -> Choose “Bearer Token” as shown below :-



* In the field beside the token, copy and paste the token received in the previous steps shown below and click on “Send” .



**[Note : In some postman versions, we may need to enter the token (in the Authorization Section) in the format -> Bearer<space><token\_value>.**

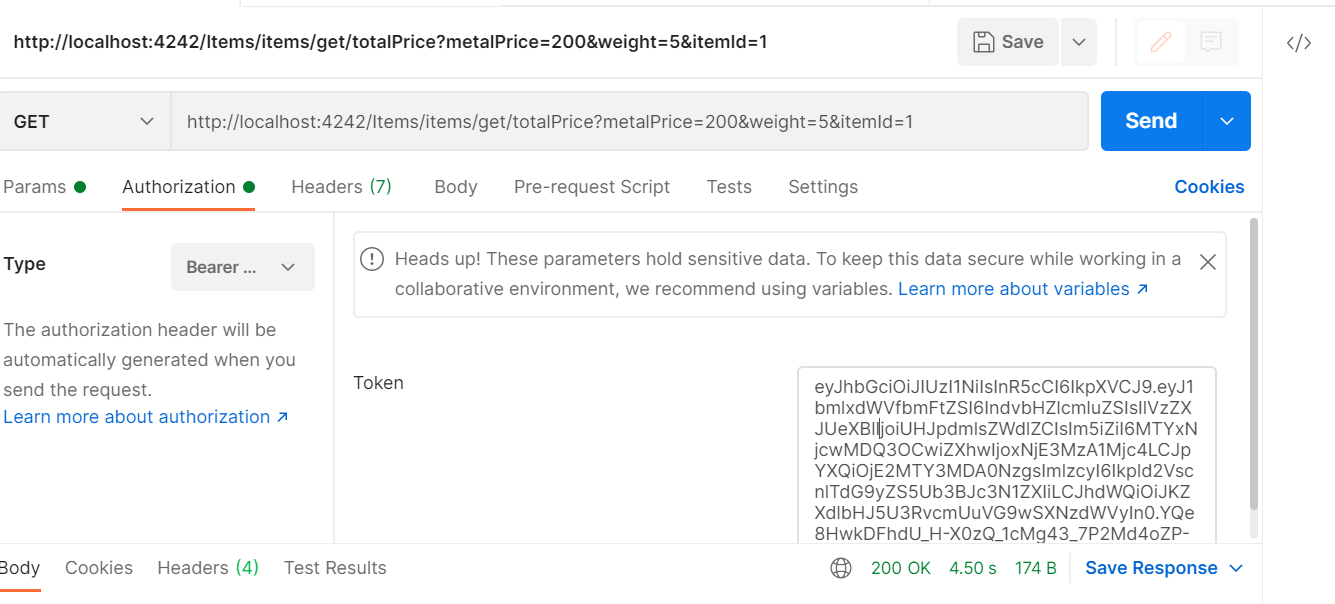
**Example : “Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bmlxdWVfbmFtZSI6IndvbHZlcmluZSIsIlVzZXJUeXBlIjoiUHJpdmlsZWdlZCIsIm5iZiI6MTYxNjcwMDQ3OCwiZXhwIjoxNjE3MzA1Mjc4LCJpYXQiOjE2MTY3MDA0NzgsImlzcyI6Ikpld2VscnlTdG9yZS5Ub3BJc3N1ZXIiLCJhdWQiOiJKZXdlbHJ5U3RvcmUuVG9wSXNzdWVyIn0.YQe8HwkDFhdU\_H-X0zQ\_1cMg43\_7P2Md4oZP-MiSEiI”**

**Skip the inverted commas (“) while entering this field]**

**Use Case 3 – To Calculate the total price of the item for a Privileged User [ As per the problem statement, a priviledged user currently has a discount of 2% that may change over time]**

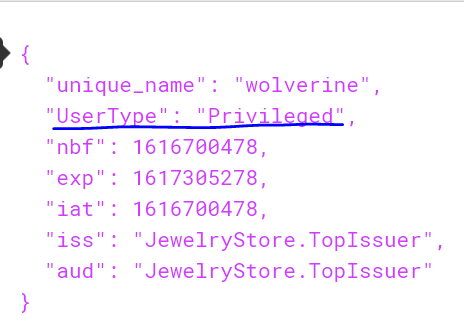
Input Url :-

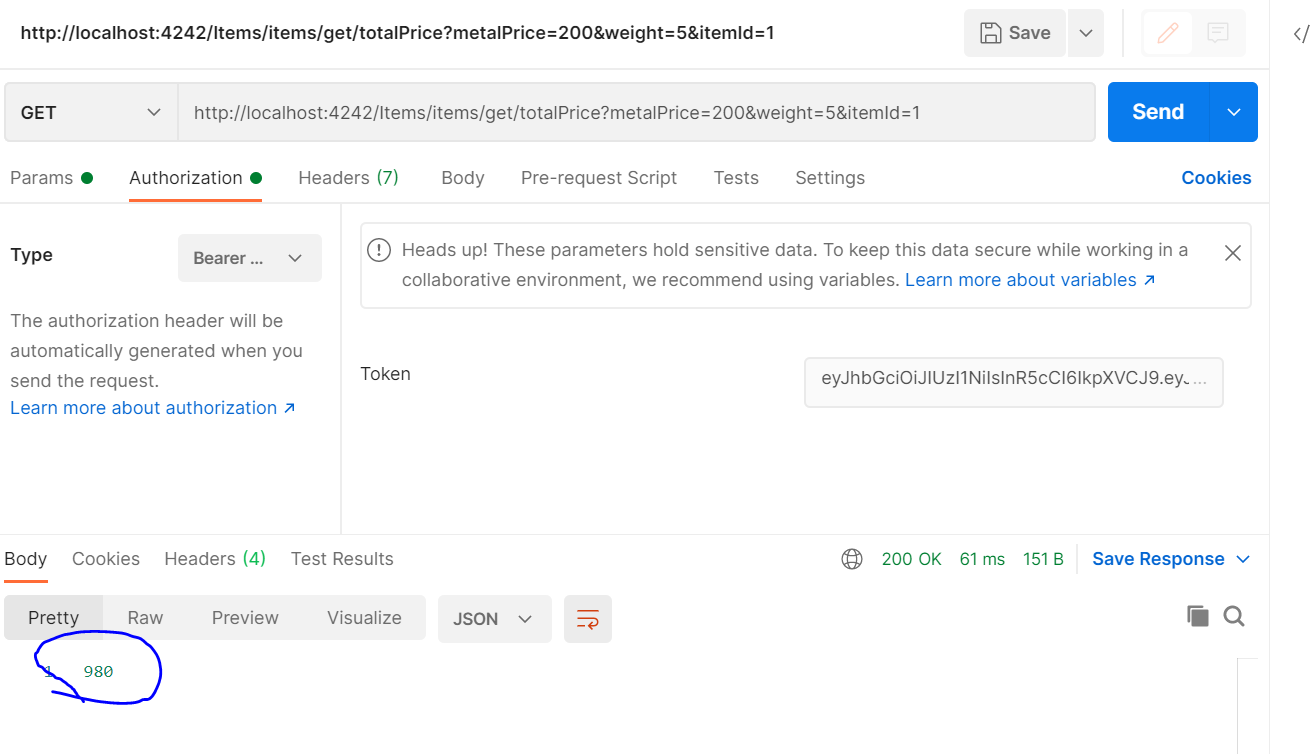
http://localhost:4242/Items/items/get/totalPrice?metalPrice=200&weight=5&itemId=1



Output:-

Since there is a two percent discount by default for a privileged user, hence the total price is 980. If one logs as a normal user, then the discount doesn’t apply. The logic is based on the fact that the JWT token has a field(claim) called **“UserType**” to differentiate between a regular user and a privileged user. Based on this claim, we apply the discount at the business logic layer. The contents of the token can be viewed by following the steps in point number **(4)**





**Use Case 4 : [Logging in as a regular user and computation of total price]**

1. **Login:-**

**Input Url :** <http://localhost:4242/Users/login>

**Body :-**

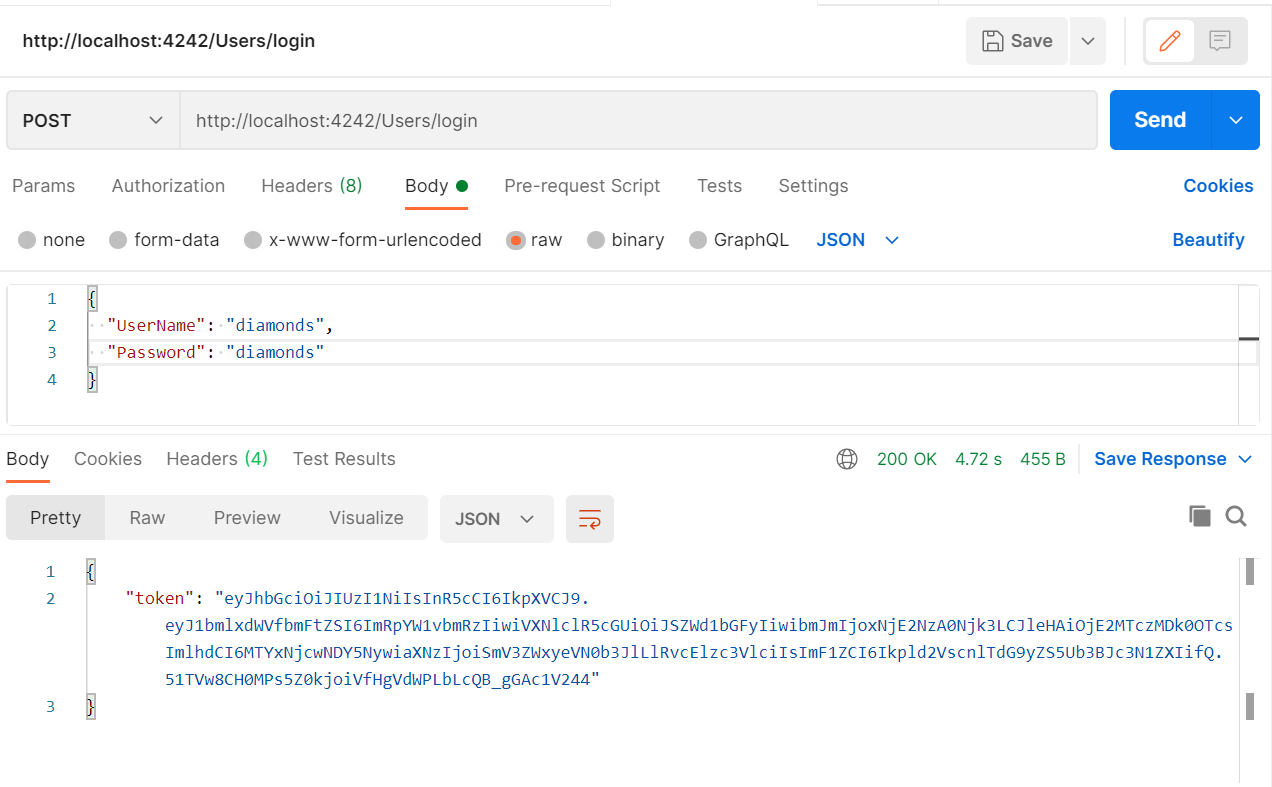
{

  "UserName": "diamonds",

  "Password": "diamonds"

}

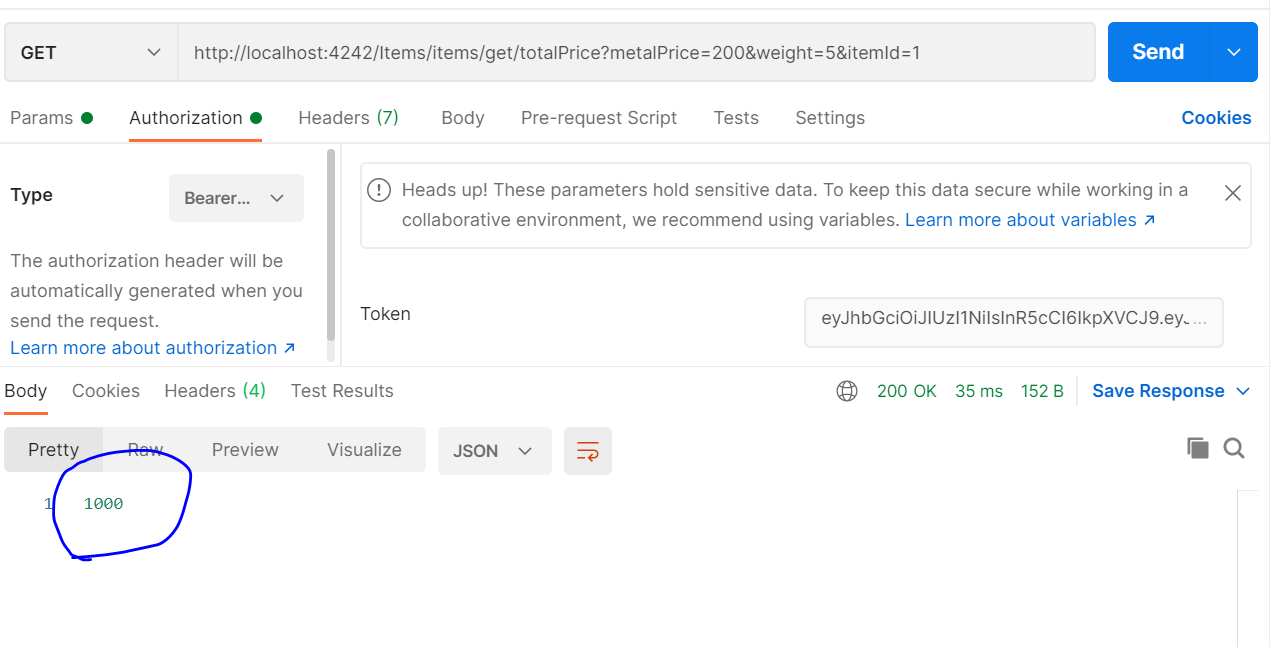
**Output:-**



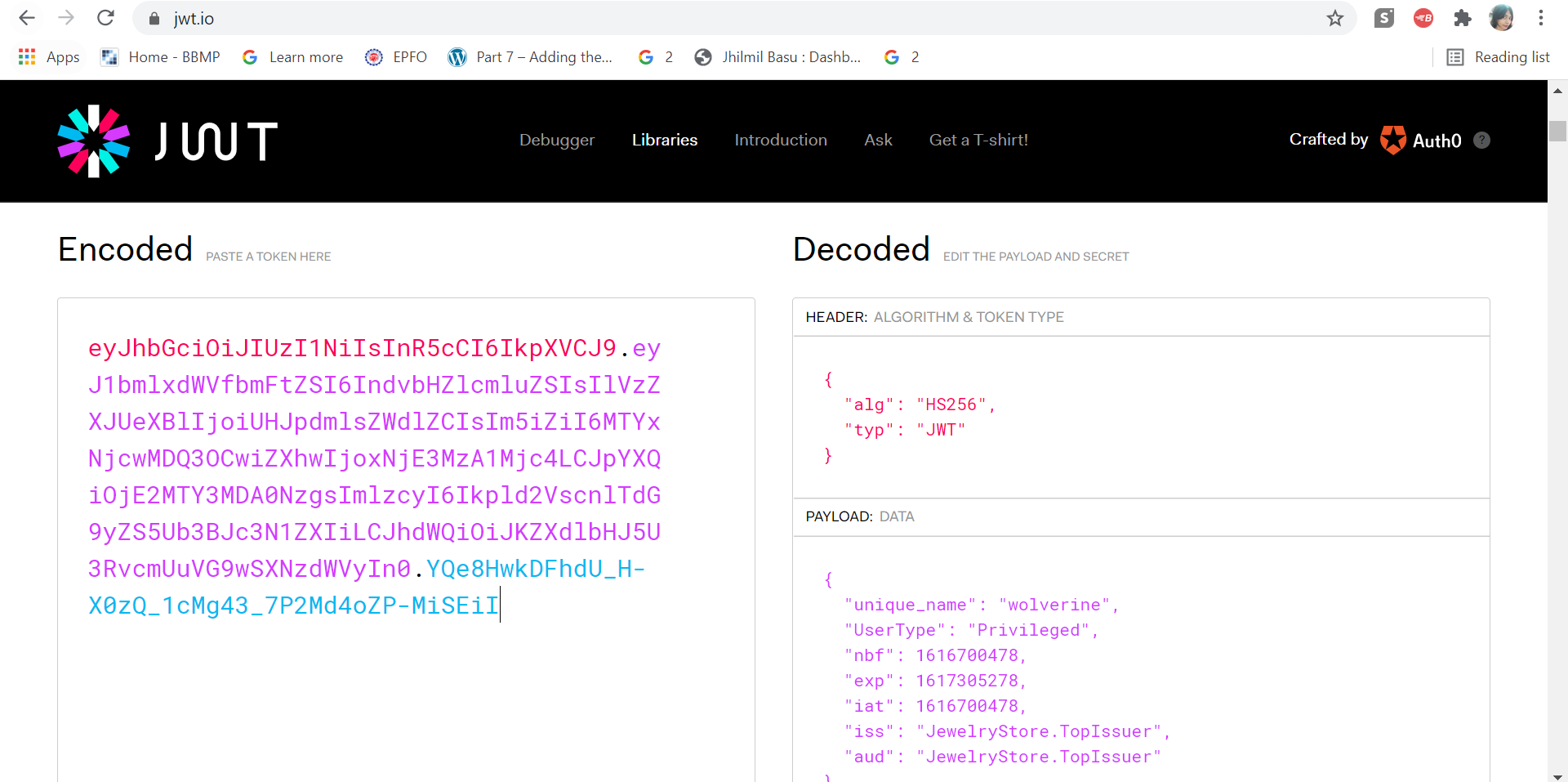
1. **Compute the total price. [Do not forget to put the above token in the “Authorization” section as a bearer token, otherwise an “Unauthorized” (401) status code will be returned]**

**Input Url :** http://localhost:4242/Items/items/get/totalPrice?metalPrice=200&weight=5&itemId=1

**Output: - [As the 2% discount doesn’t apply to a regular user, hence the output is 1000]**



1. If you want to view the token contents, head over to jwt.io and paste the token as shown below :-



* Mouse over the “exp” field under the “Payload” to view the expiration date and time of the token as shown below :-

