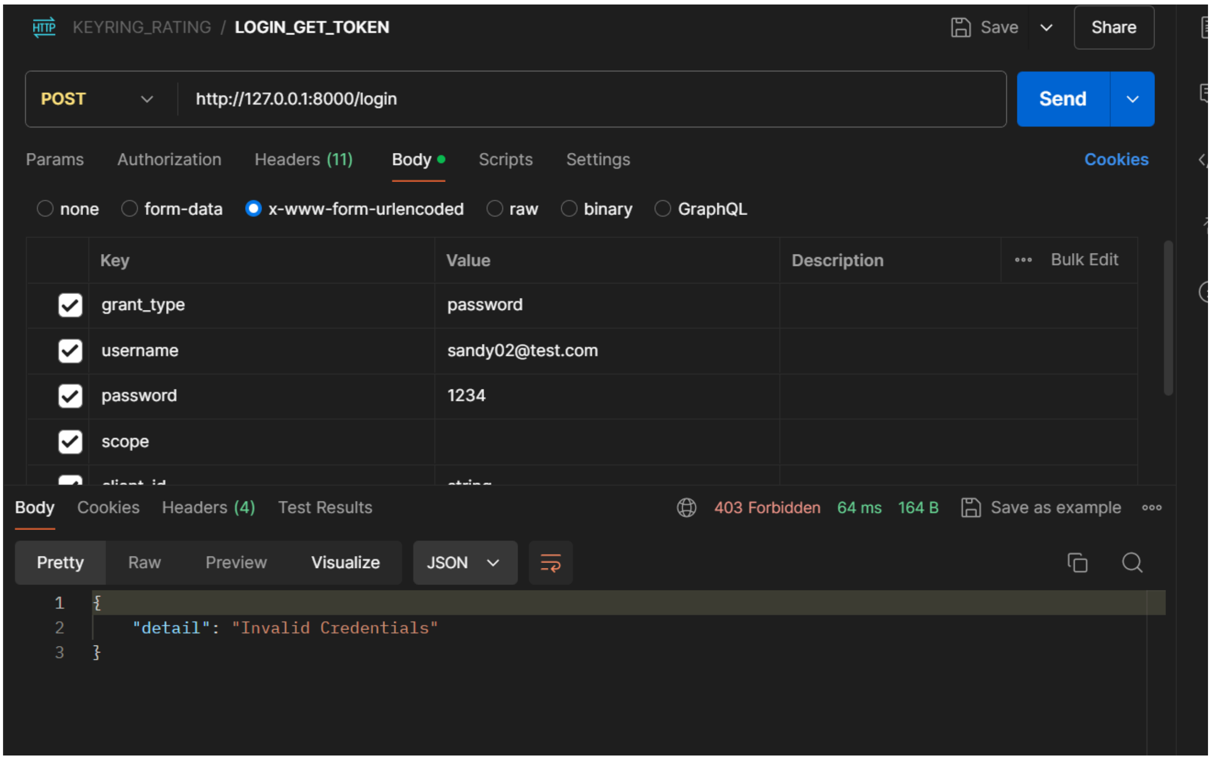
USER TABLE SAVED IN schemas.py

```
users_table = [
    {"email":"sandy01@test.com","passwd":"pass","paying":False},
    {"email":"sandy02@test.com","passwd":"pass","paying":False},
    {"email":"sandy03@test.com","passwd":"1234","paying":True},
    {"email":"sandy04@test.com","passwd":"pass","paying":False},
    {"email":"sandy05@test.com","passwd":"pass","paying":False}
    ]
```

TEST01 -
Logging in with sandy02@test.com as the non-paying user
Wrong Password
Result: PASSED



POST http://127.0.0.1:8000/login

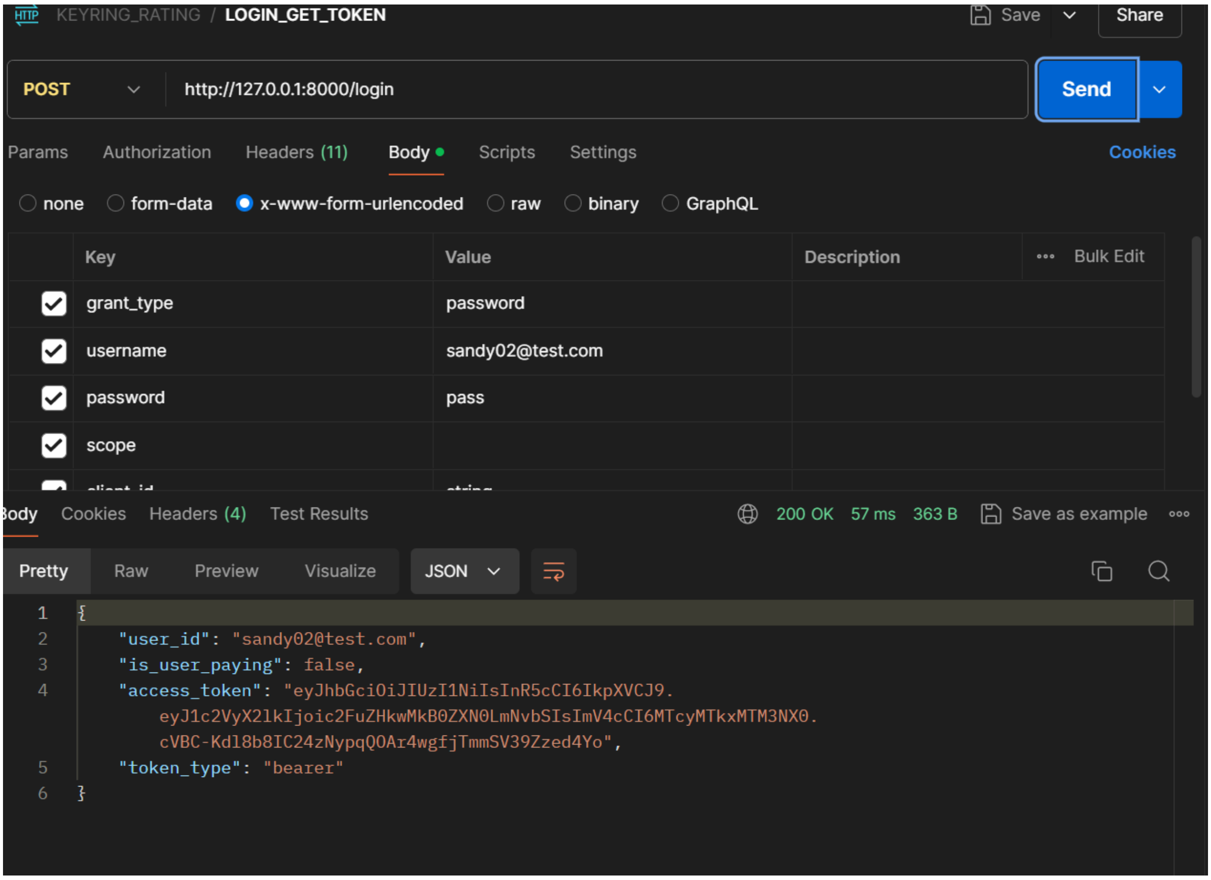
● x-www-form-urlencoded

	Key	Value	Description
☑	grant_type	password	
☑	username	sandy02@test.com	
☑	password	1234	
☑	scope		
☑	client_id	string	

403 Forbidden 64 ms 164 B

```
{
    "detail": "Invalid Credentials"
}
```

TEST02 -
Logging in with sandy02@test.com as the non-paying user
Correct Password



POST http://127.0.0.1:8000/login

○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

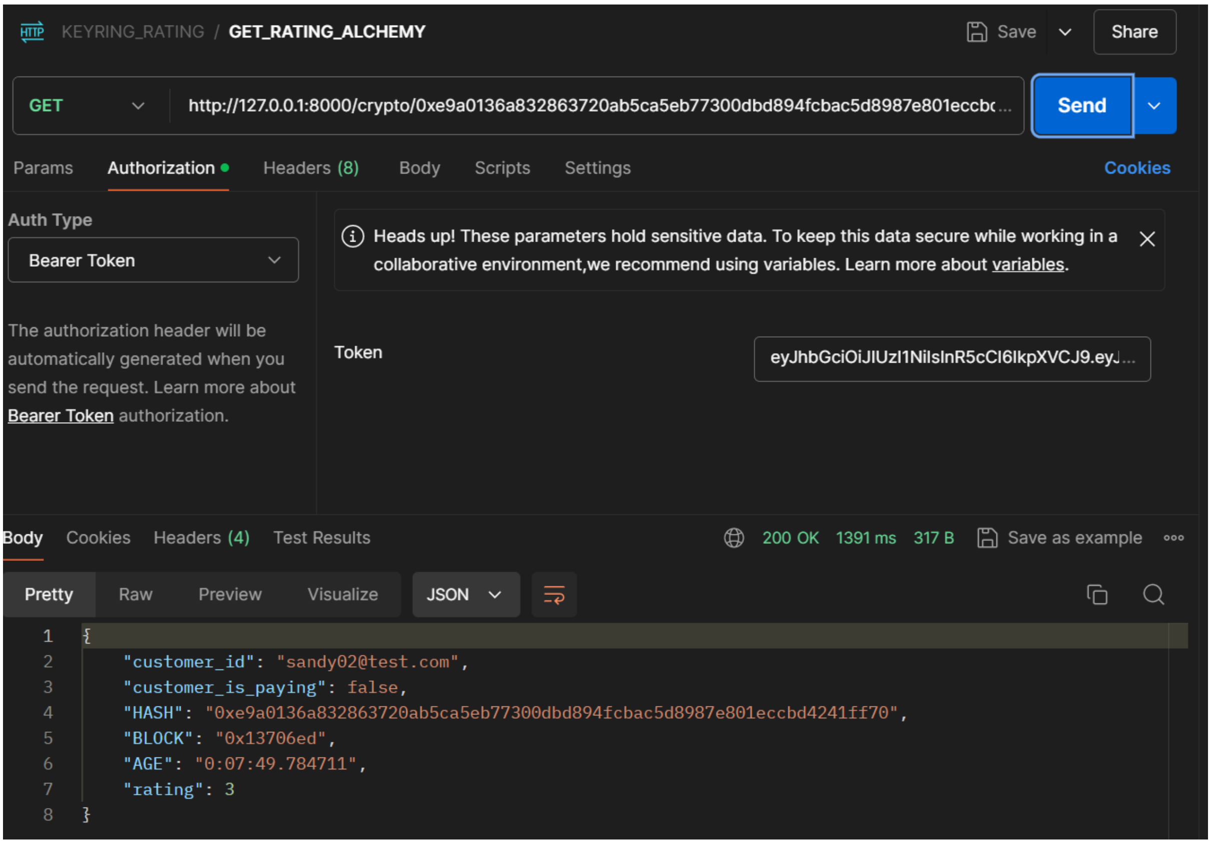
	Key	Value	Description
☑	grant_type	password	
☑	username	sandy02@test.com	
☑	password	pass	
☑	scope		
☑	client_id	string	

200 OK 57 ms 363 B

```
{
    "user_id": "sandy02@test.com",
    "is_user_paying": false,
    "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJ1c2VyX2lkIjoic2FuZHkwMkB0ZXN0LmNvbSIsImV4cCI6MTcyMTkxMTM3NX0.
        cVBC-Kdl8b8IC24zNypqQOAr4wgfjTmmSV39Zzed4Yo",
    "token_type": "bearer"
}
```

Output - Received JWT Token
Result: PASSED
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyX2lkIjoic2FuZHkwMkB0ZXN0LmNvbSIsI
mV4cCI6MTcyMTkxMTM3NX0.cVBC-Kdl8b8IC24zNypqQOAr4wgfjTmmSV39Zzed4Yo

TEST03 -
Using the token and getting a rating.
Getting data from etherscan. Getting an old transaction
TRANSACTION HASH =
0xe9a0136a832863720ab5ca5eb77300dbd894fcbac5d8987e801eccbd4241ff70



GET http://127.0.0.1:8000/crypto/0xe9a0136a832863720ab5ca5eb77300dbd894fcbac5d8987e801eccbc...

Auth Type: Bearer Token

The authorization header will be automatically generated when you send the request. Learn more about Bearer Token authorization.

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment,we recommend using variables. Learn more about variables.

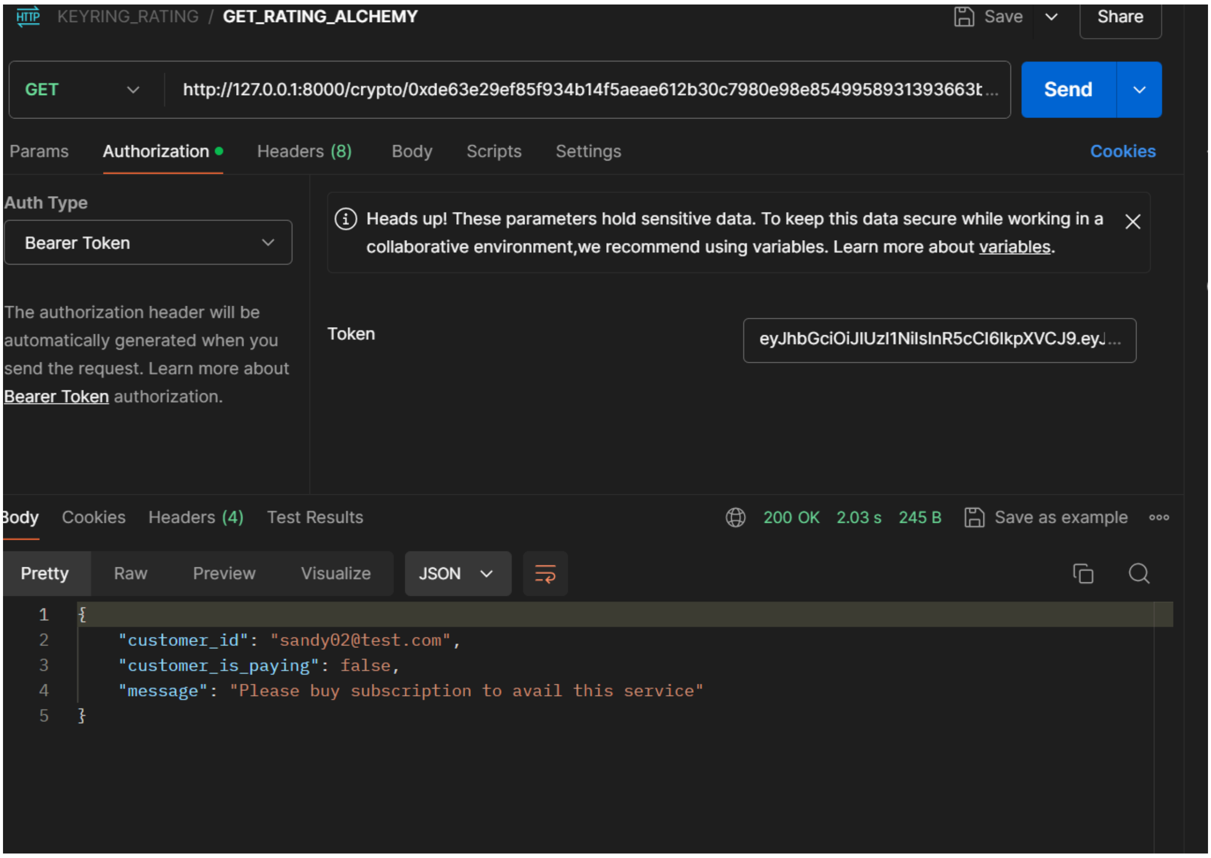
Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ...

200 OK 1391 ms 317 B

```
{
    "customer_id": "sandy02@test.com",
    "customer_is_paying": false,
    "HASH": "0xe9a0136a832863720ab5ca5eb77300dbd894fcbac5d8987e801eccbd4241ff70",
    "BLOCK": "0x13706ed",
    "AGE": "0:07:49.784711",
    "rating": 3
}
```

Received Rating of 3.
Also as a non paying customer the access to API was given as age more than 5 mins.

TEST04 -
Using the token and getting a rating.
Getting data from etherscan. Getting an new transaction (less than one min old)
TRANSACTION HASH =
0xde63e29ef85f934b14f5aeae612b30c7980e98e8549958931393663bb3505d12



http://127.0.0.1:8000/crypto/0xde63e29ef85f934b14f5aeae612b30c7980e98e8549958931393663b...

Auth Type: Bearer Token

The authorization header will be automatically generated when you send the request. Learn more about Bearer Token authorization.

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment,we recommend using variables. Learn more about variables.

Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ...

200 OK 2.03 s 245 B

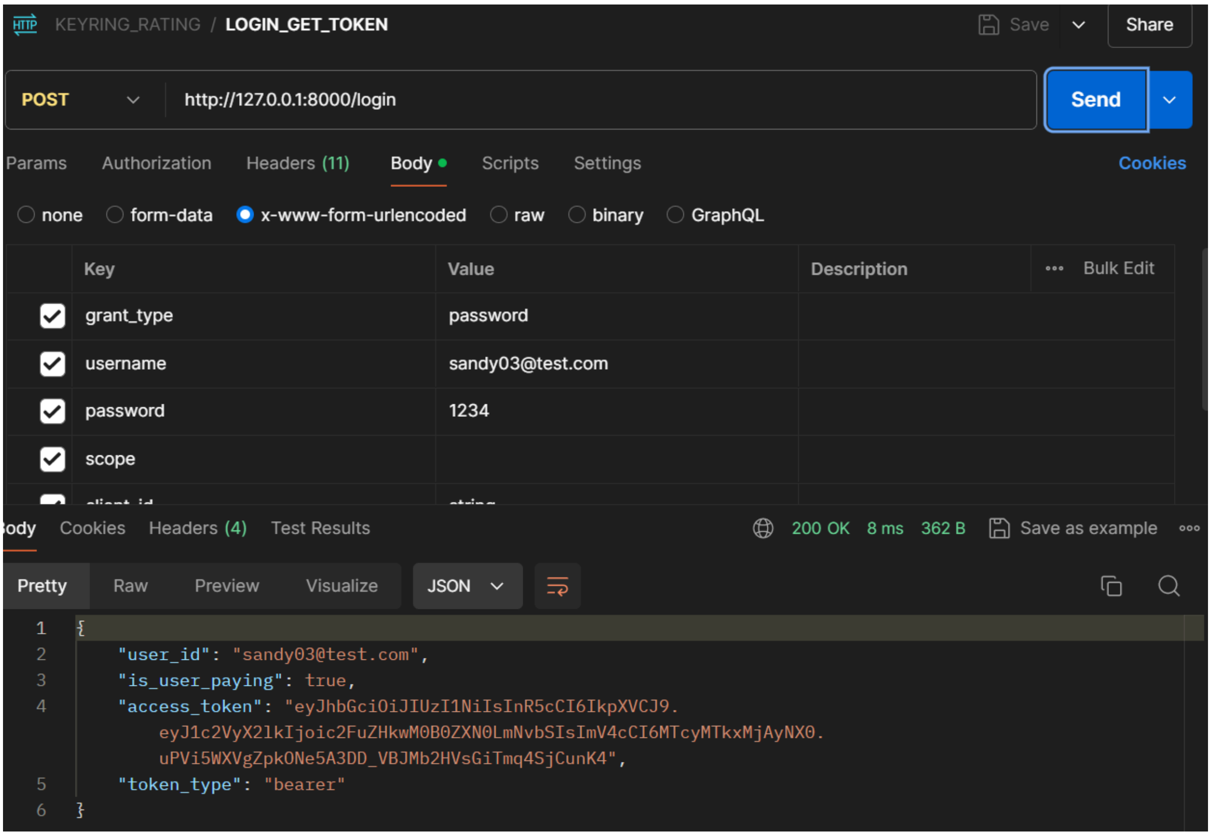
```
{
    "customer_id": "sandy02@test.com",
    "customer_is_paying": false,
    "message": "Please buy subscription to avail this service"
}
```

Received no rating. Also message was sent pay to buy subscription.

TEST05:

```
users_table = [
   {"email":"sandy01@test.com","passwd":"pass","paying":False},
   {"email":"sandy02@test.com","passwd":"pass","paying":False},
   {"email":"sandy03@test.com","passwd":"1234","paying":True},
   {"email":"sandy04@test.com","passwd":"pass","paying":False},
   {"email":"sandy05@test.com","passwd":"pass","paying":False}
   ]
```

Logging in with sandy03@test.com as he is a paying customer
Result: PASSED, token is returns and also paying status is true



POST http://127.0.0.1:8000/login

○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	Key	Value	Description
☑	grant_type	password	
☑	username	sandy03@test.com	
☑	password	1234	
☑	scope		
☑	client_id	string	

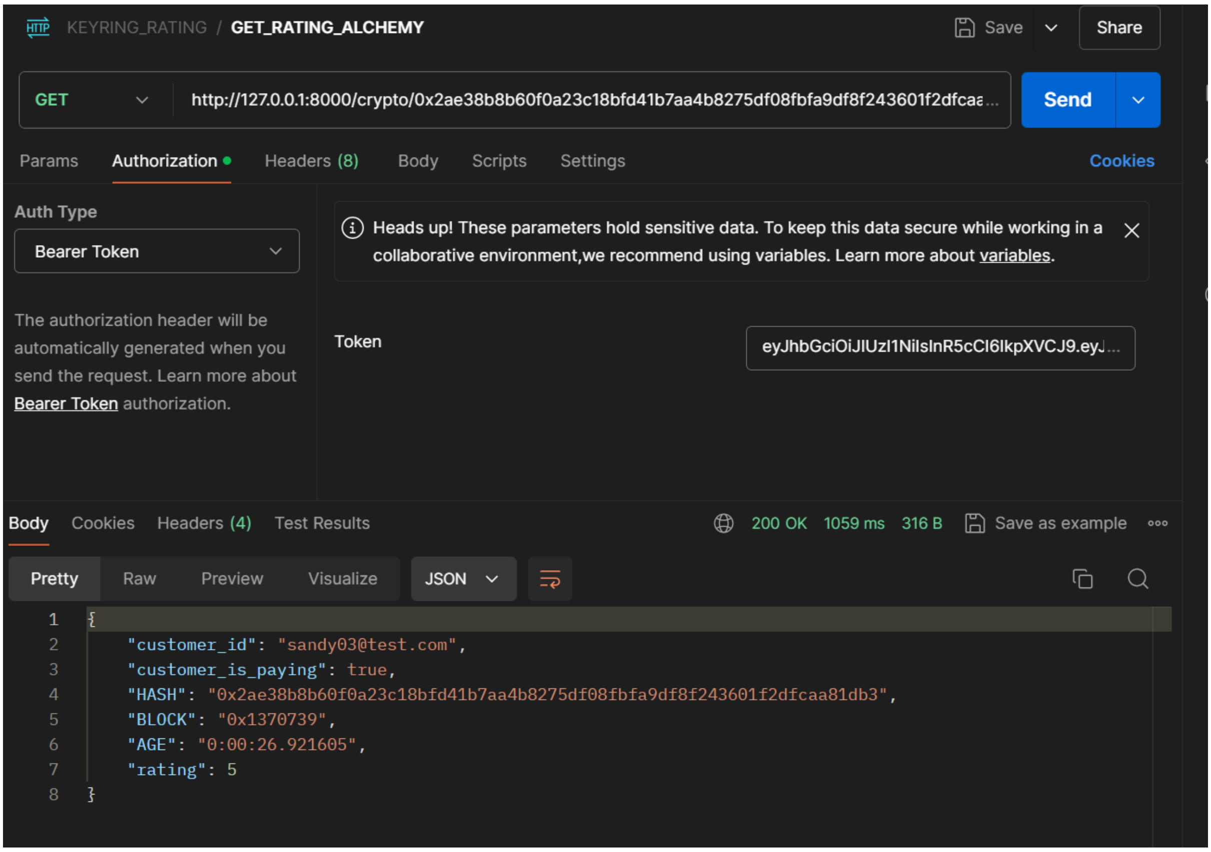
200 OK 8 ms 362 B

```
{
    "user_id": "sandy03@test.com",
    "is_user_paying": true,
    "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJ1c2VyX2lkIjoic2FuZHkwM0B0ZXN0LmNvbSIsImV4cCI6MTcyMTkxMjAyNX0.
        uPVi5WXVgZpkONe5A3DD_VBJMb2HVsGiTmq4SjCunK4",
    "token_type": "bearer"
}
```

TEST06:

Getting a rating for new customer for the paying customer:

As customer is paying rating is returned for a 26 sec old customer



GET http://127.0.0.1:8000/crypto/0x2ae38b8b60f0a23c18bfd41b7aa4b8275df08fbfa9df8f243601f2dfcaa...

Auth Type: Bearer Token

The authorization header will be automatically generated when you send the request. Learn more about Bearer Token authorization.

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment,we recommend using variables. Learn more about variables.

Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ...

200 OK 1059 ms 316 B

```
{
    "customer_id": "sandy03@test.com",
    "customer_is_paying": true,
    "HASH": "0x2ae38b8b60f0a23c18bfd41b7aa4b8275df08fbfa9df8f243601f2dfcaa81db3",
    "BLOCK": "0x1370739",
    "AGE": "0:00:26.921605",
    "rating": 5
}
```