Lab 11 -OWASP Top

Installing crAPI

You first have to install a crAPI instance on your local system. You can either install it directly on the system you will be testing it from. Or you can create a new VM to be your target system, separate from your attacker system (this is what I prefer to do).

If this is the way you want to go along, you need to create a fresh VM using your favorite hypervisor software (I use VirtualBox). Install a linux system on this new VM (I use Ubuntu for my target VMs).

Next, you want to install Docker (instructions here). Also install Docker Compose.

Once you're done, you may check your installation by running the hello-world image:
sudo docker run hello-world

If all is good, you can now install crAPI:

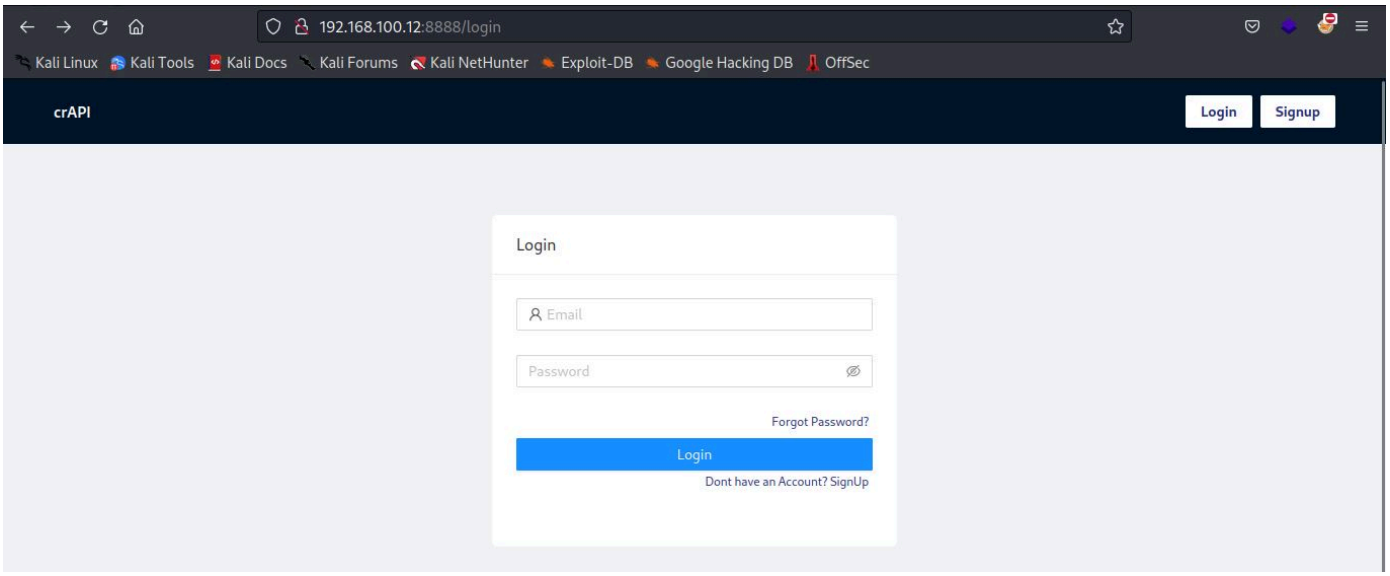
```
curl -o docker-compose.yml
https://raw.githubusercontent.com/OWASP/crAPI/main/deploy/docker/docker-
compose.yml

sudo docker compose pull

sudo docker compose -f docker-compose.yml --compatibility up -d
```

Now wait for all the containers to load (can take a little while).

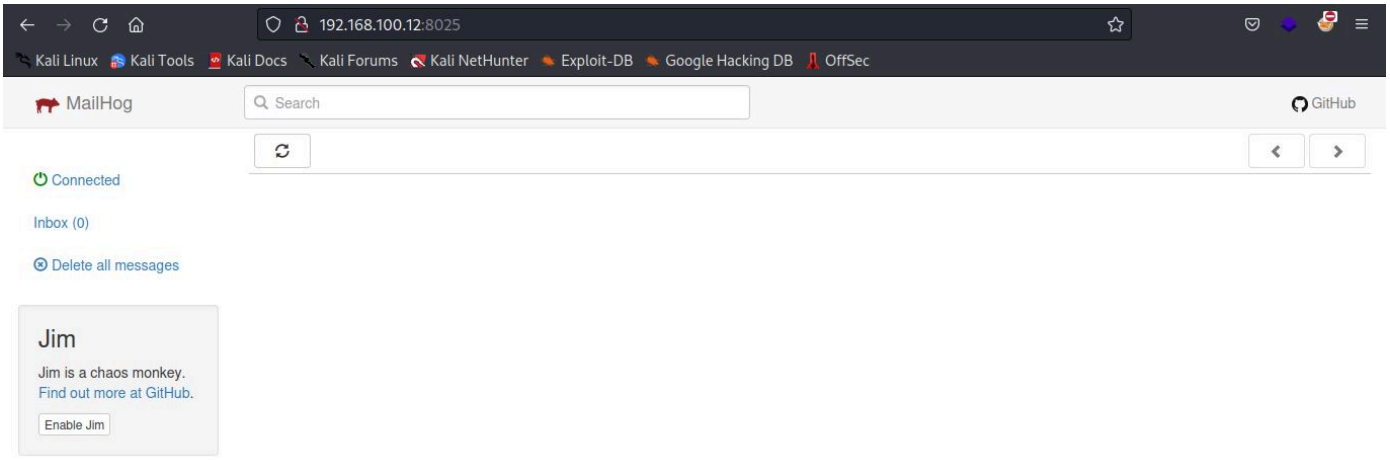
To check if crAPI is running, just type:

```
$ sudo docker ps -a
```

- Access the crAPI web in the linux machine with URL `http://IP_Server:8888` in the browser



- Access mailhog in the engine times linux with different tabs in the browser with URL `http://IP_Server:8025`



Troubleshooting

Incase of any misconfiguration you can run these commands according to your need.

To start all the containers in docker run

```
sudo docker start $(sudo docker ps -aq)
```

to stop all the containers in docker run

```
sudo docker stop $(sudo docker ps -aq)
```

to delete all the containers in docker run

```
sudo docker rm $(sudo docker ps -aq)
```

Configure Burpsuit for testing

The **Burp Suite** is an integrated platform for performing security testing on web applications. You can use it as a proxy to intercept your browser sessions to any website. This can be useful for testing

against web applications, discovering vulnerabilities in websites, and maybe even making some money with a bug bounty.

Burp has a free version called Community Version. To download this version go to the following site:

Download Burp Suite Community Edition – PortSwigger

FoxyProxy

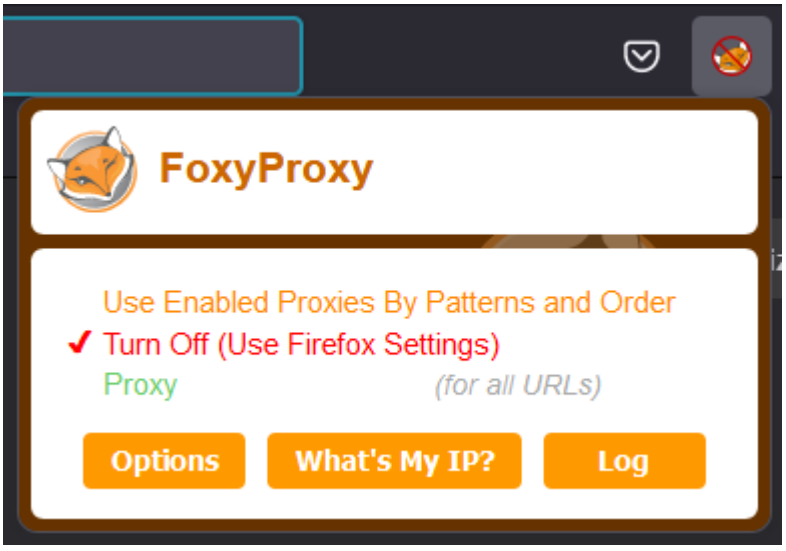
FoxyProxy is an advanced proxy management tool that completely replaces Firefox's limited proxy features. For a simpler tool and less advanced configuration options, FoxyProxy Basic can be used.

You can use FoxyProxy in conjunction with Burp Suite to facilitate proxy activation using Burp.

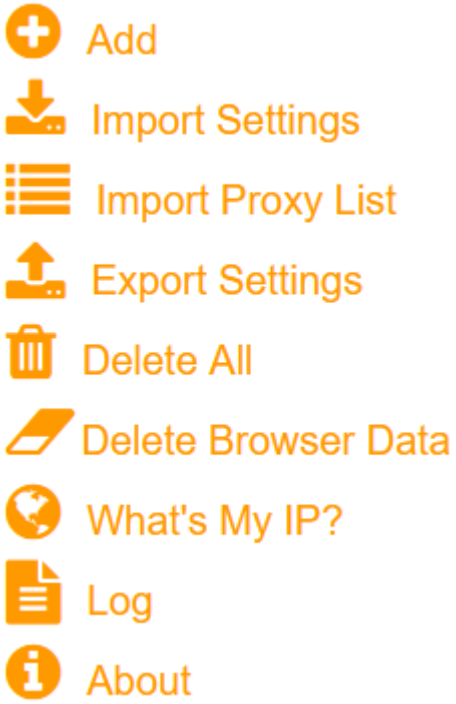
After downloading, do the standard installation and we are ready to install FoxyProxy.

https://addons.mozilla.org/pt-BR/firefox/addon/foxyproxy-standard/

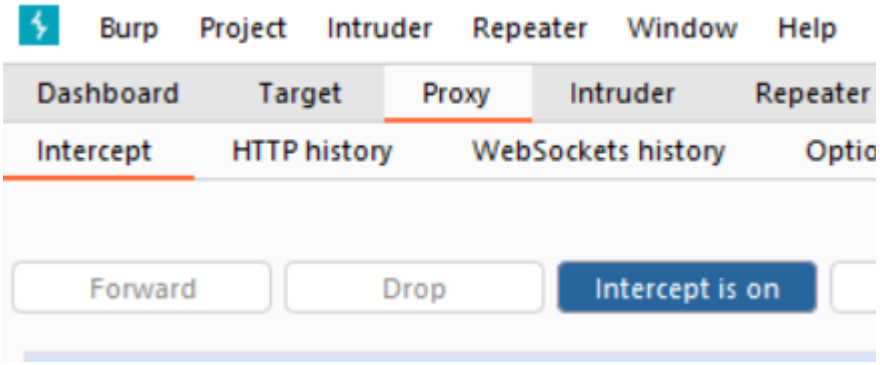
After installing the add-on, you will see it in the top right corner of Firefox like the image below:



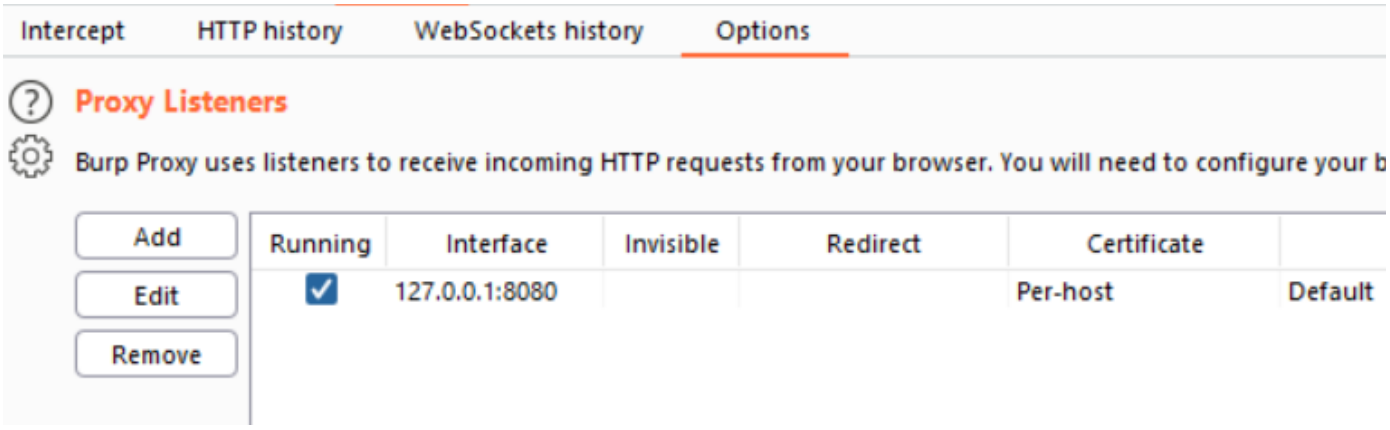
By clicking on options, we are taken to the configuration page and we will add the Burp address by clicking on Add.



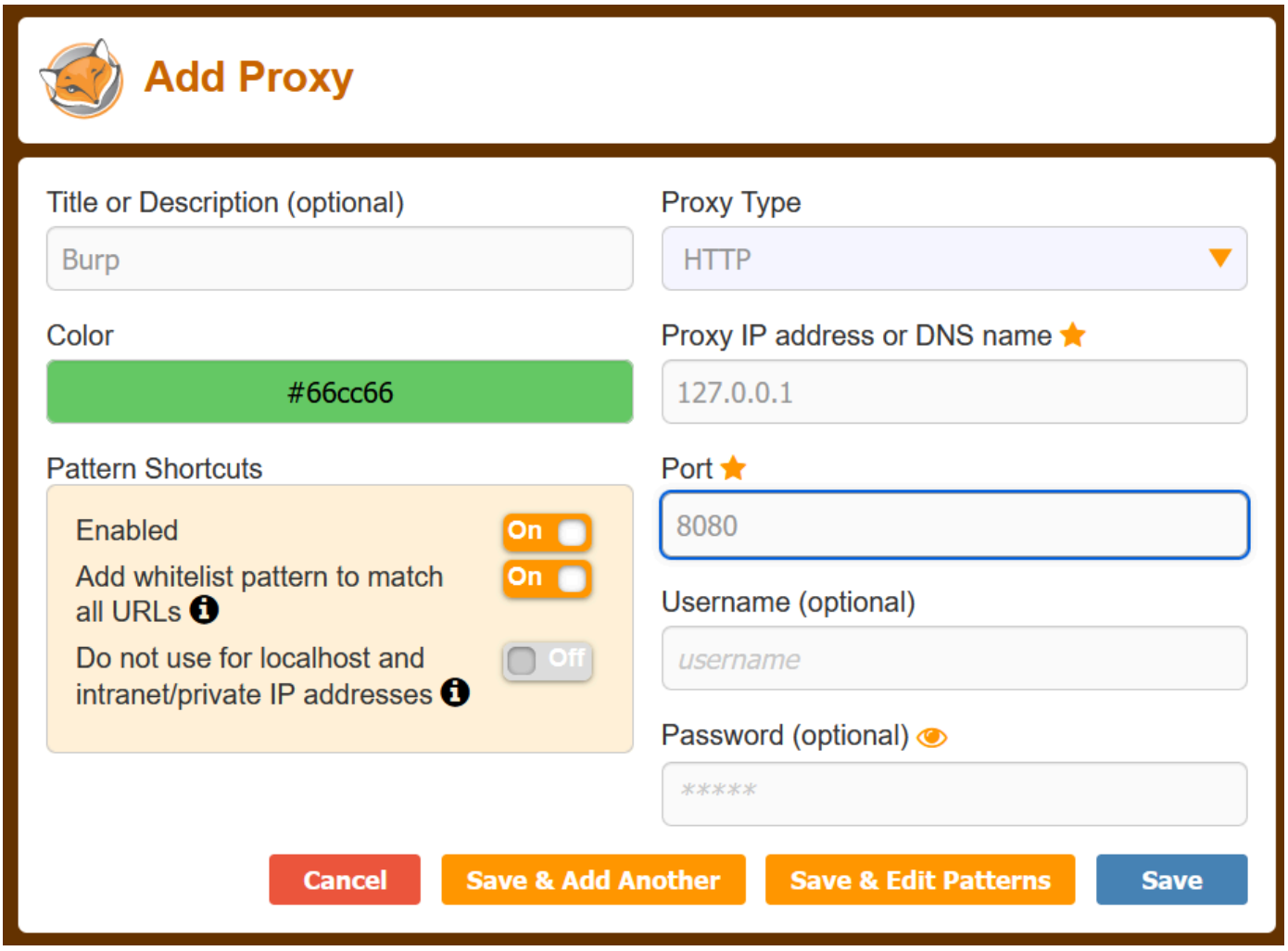
To validate the address in Burp Suite, let's open it and go to the **Proxy** tab:



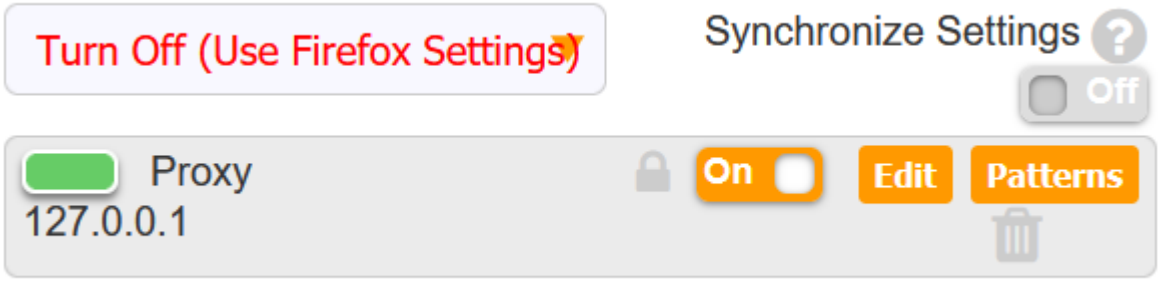
Then click **Options** and we can see the **Burp Proxy Listener**:



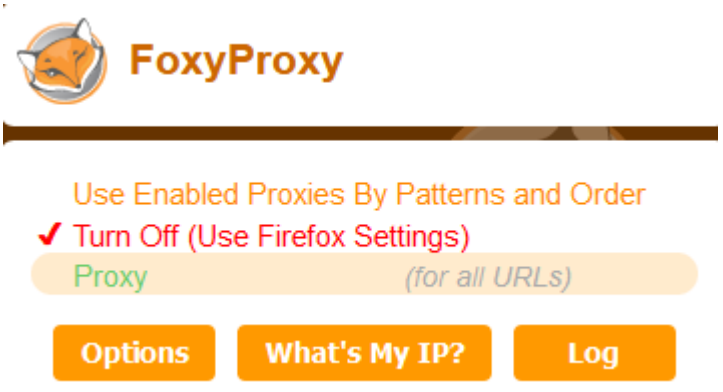
Let's copy this address to use in FoxyProxy, where we will have the result as below:



Once saved, we will have the configuration listed as below:



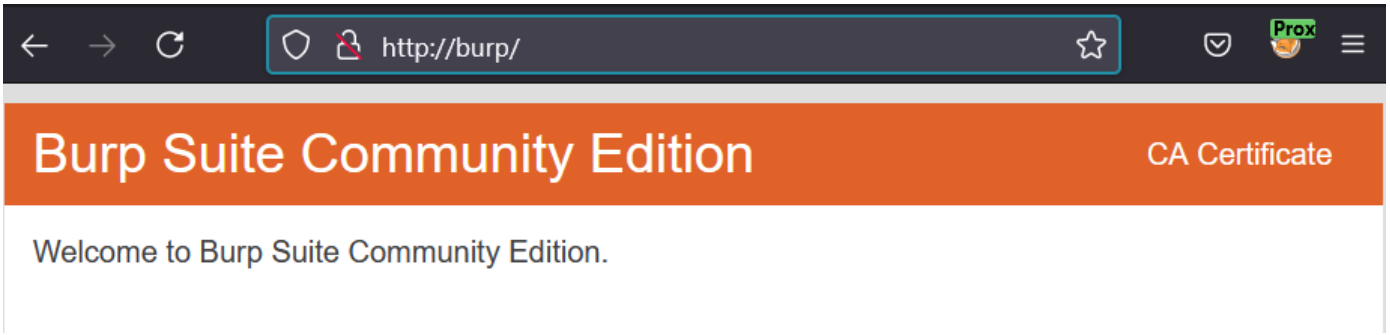
The best thing about FoxyProxy is that it is very easy to use. After the configuration we made above, just click on the green option below "**Proxy**" and it will use Burp as a proxy:



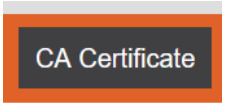
Configuring Burp Suite Certificate in Firefox

So that we can perform the proxy without certificate errors, let's import the burp certificate into the Firefox settings.

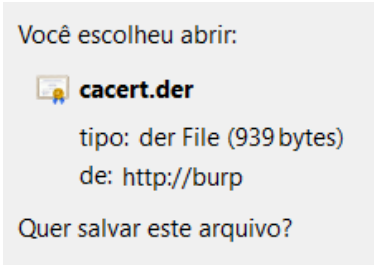
To do this, type in the browser: http://burp/.



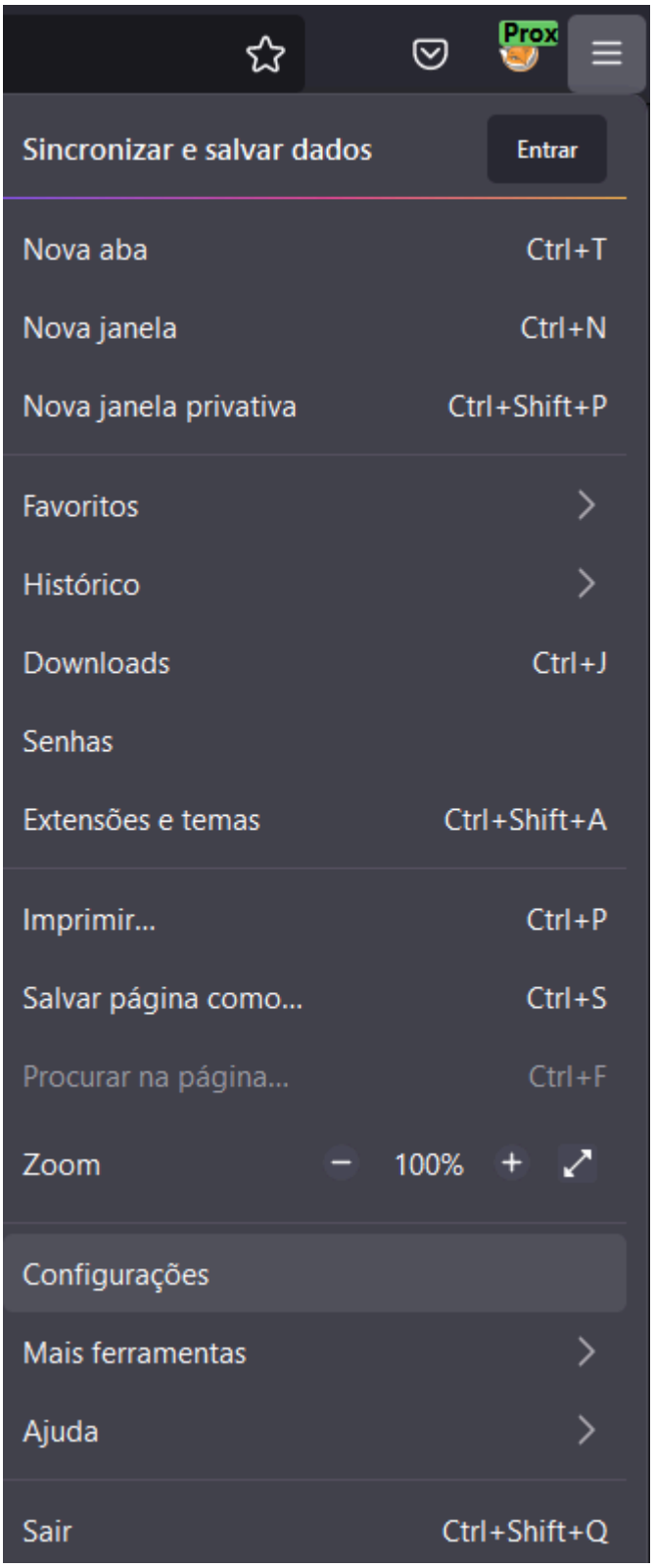
Let's click on **CA Certificate** in the upper right corner:



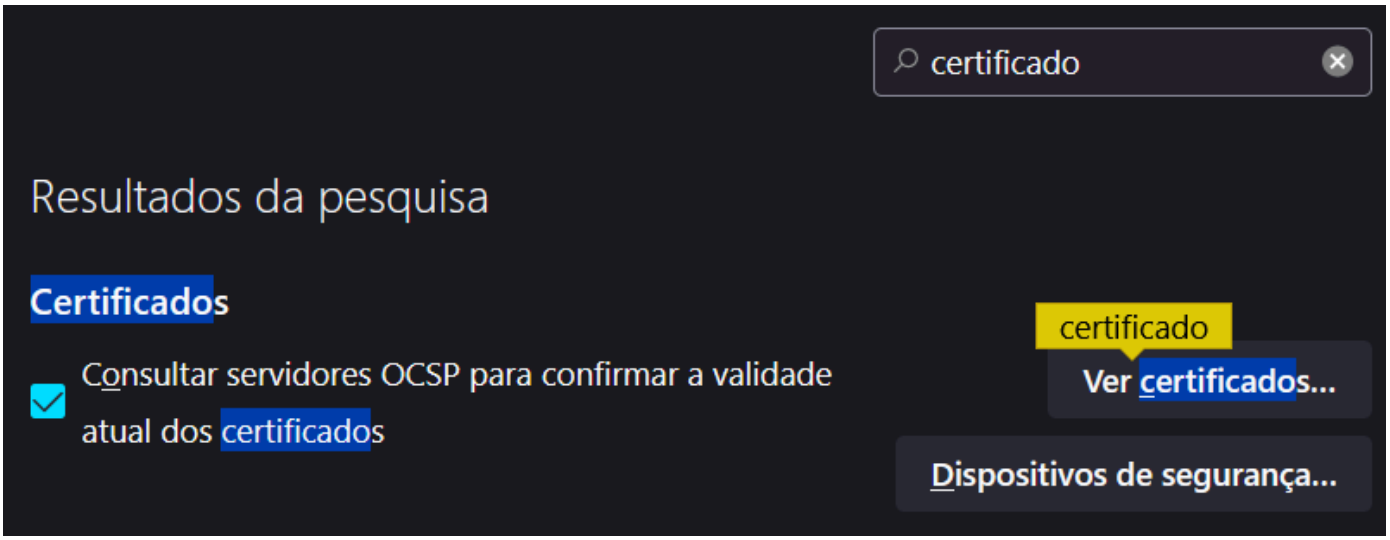
Now just save the certificate:



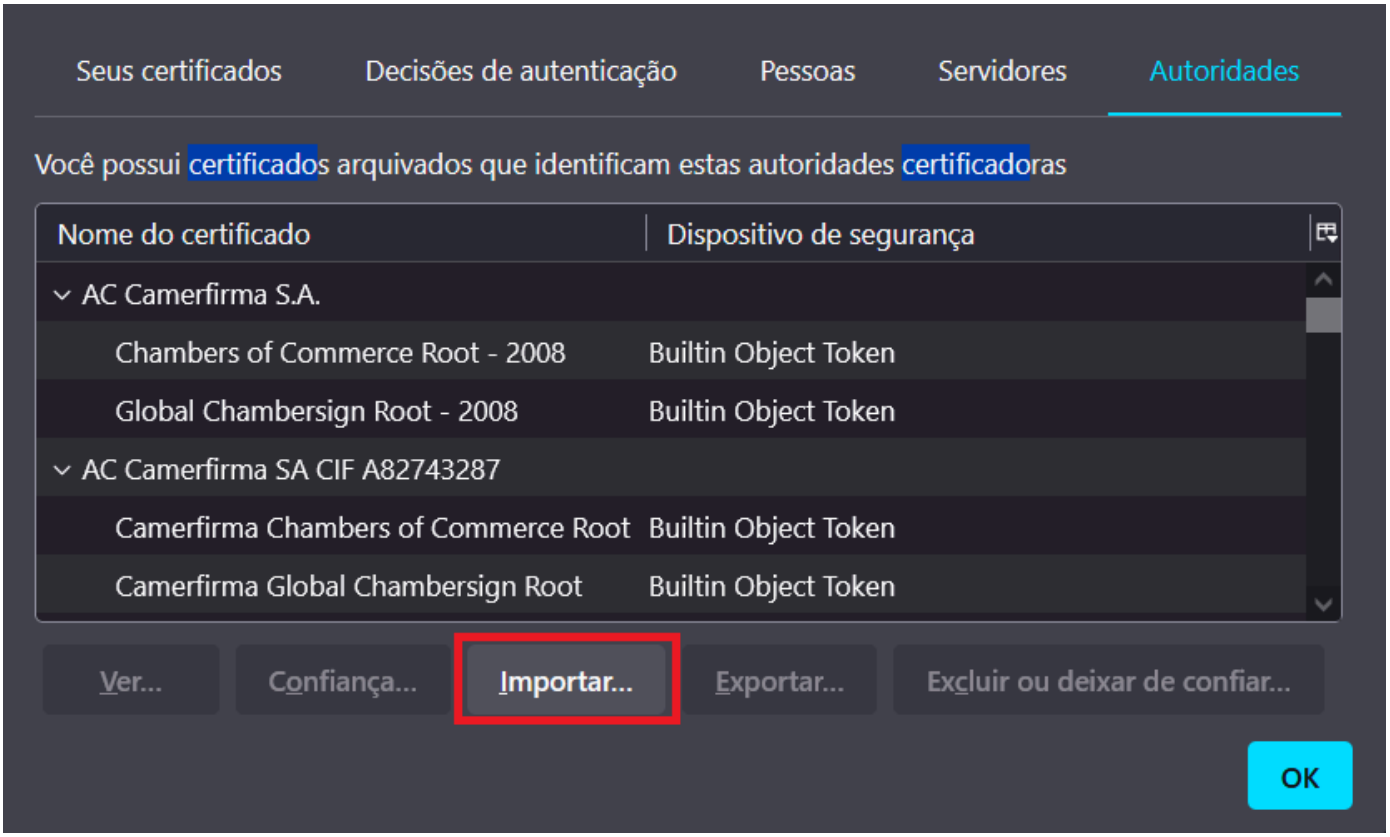
Let's open the Firefox settings:

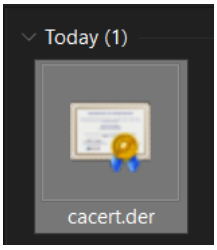


Let's search for **Certificates** and click on **View certificates**:



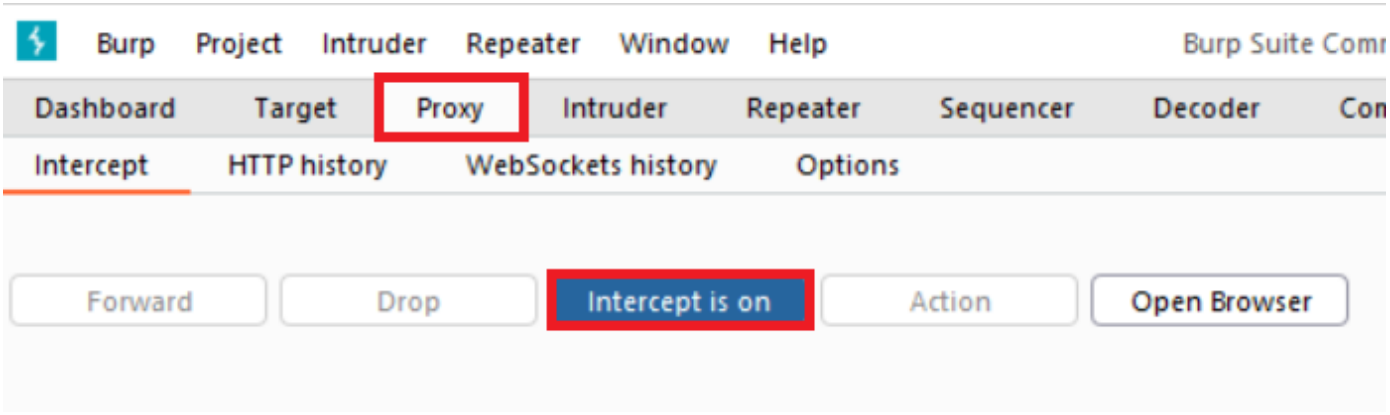
Let's click on **Import** and select the downloaded certificate:



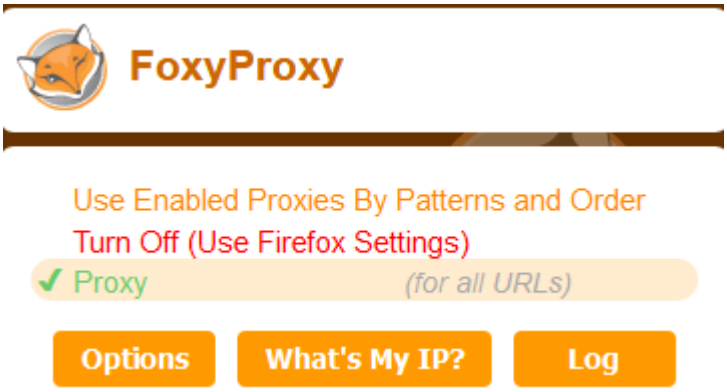


Using Burp as a Proxy

Now let's open our Burp Suite, go to the Proxy tab and check if the "**Intercept is on**" option is enabled:

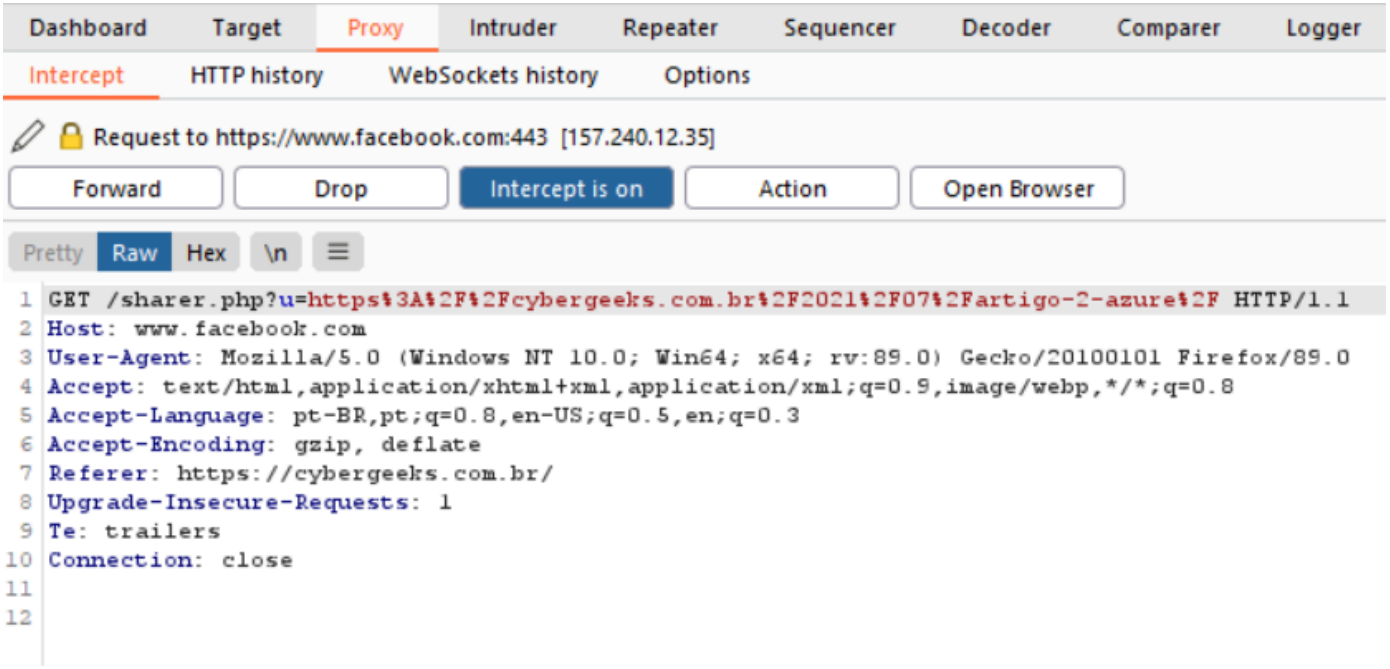


Now let's open Firefox and select the option we configured earlier so that it uses a browsing proxy:



Then visit any address and check that the site is not open because the interception mode is on and you need to accept to continue on Burp. If you click **Forward**, it will forward the request to the next step:

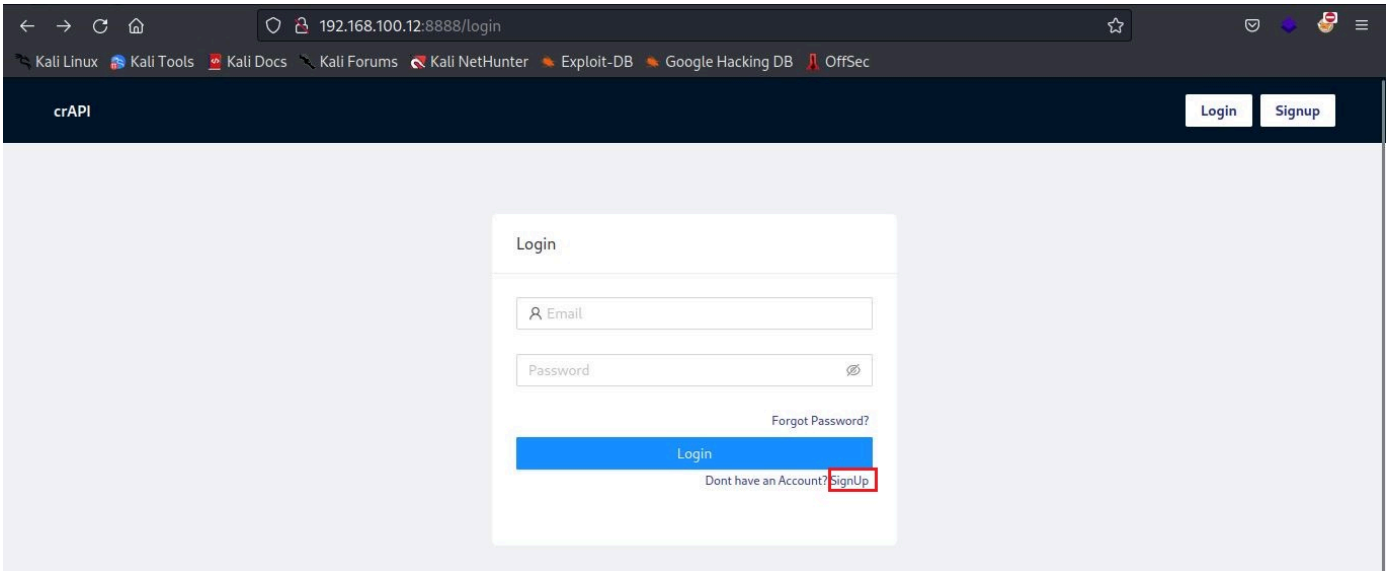
See that I'm going to click on the plugin to share an article on Facebook here on the site and we can see this in the Burp being intercepted:



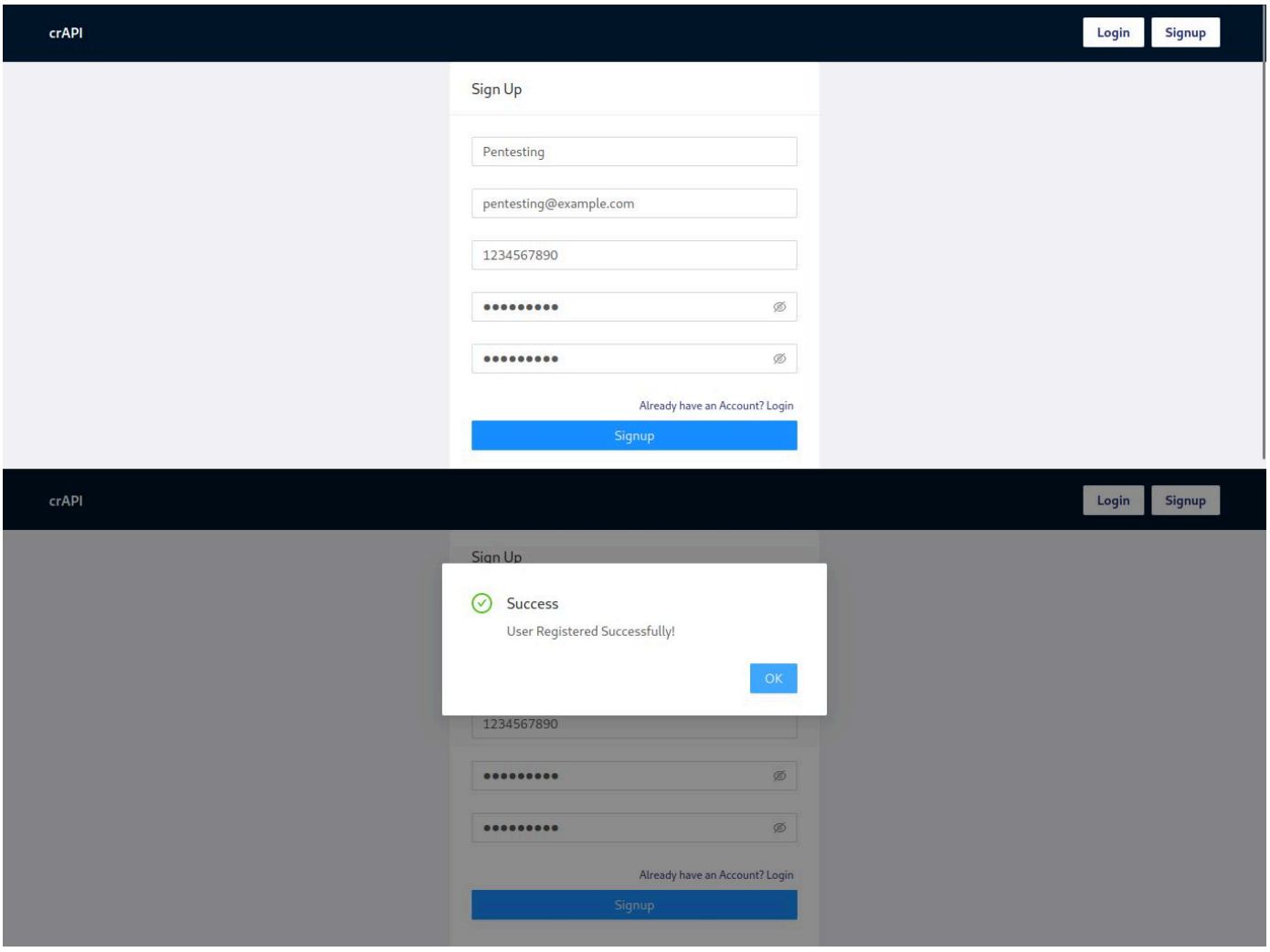
Same steps can be followed for OWASP ZAP and both tools operate on `localhost` and port `8080`. OWASP ZAP certificate can be download from `localhost:8080` after running OWASP ZAP.

First Steps

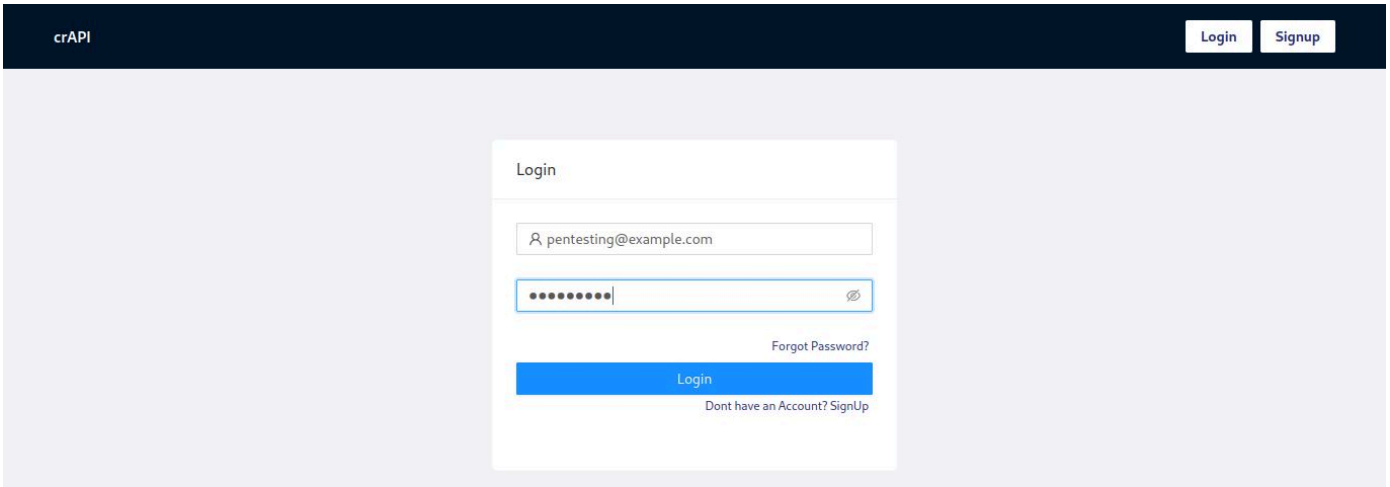
- Before proceeding furhter Create a new account by pressing **the Sign Up** link to access the web crAPI



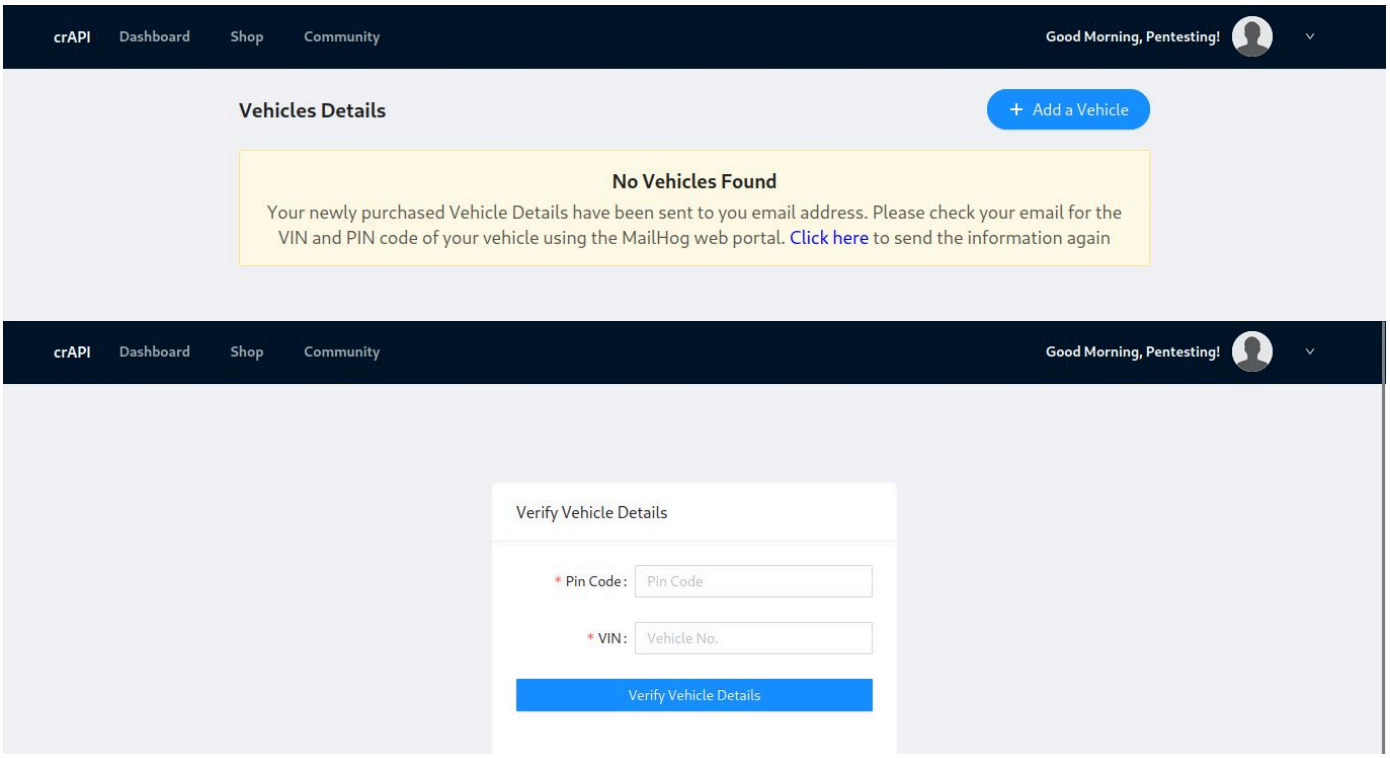
- If you have press the **Signup** button then press **the OK** button. Note you should use **@example.com** domain while signing up only then will you recieve and email in the local **mailhog** server.



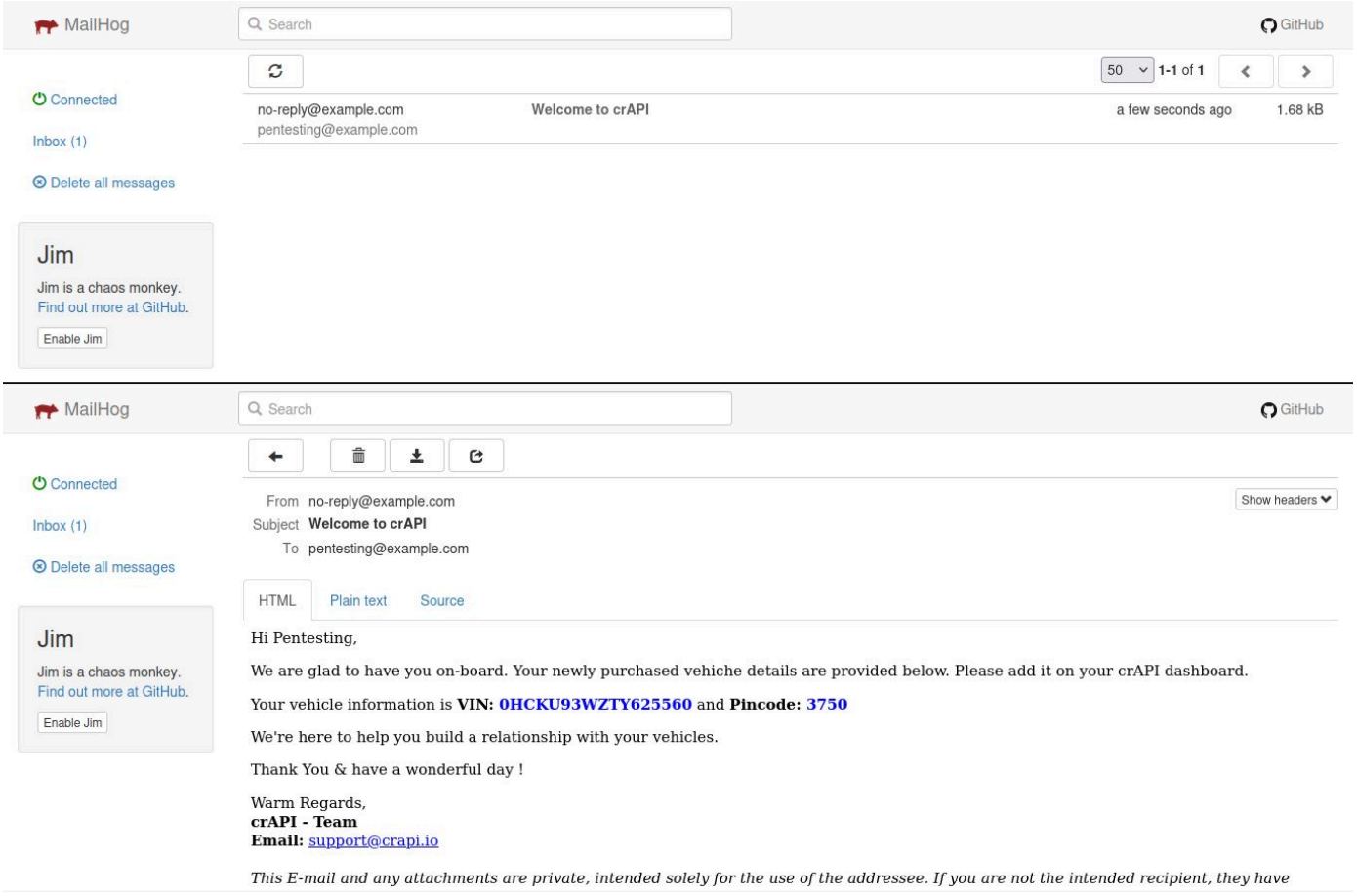
- Now login with a new account that has been created



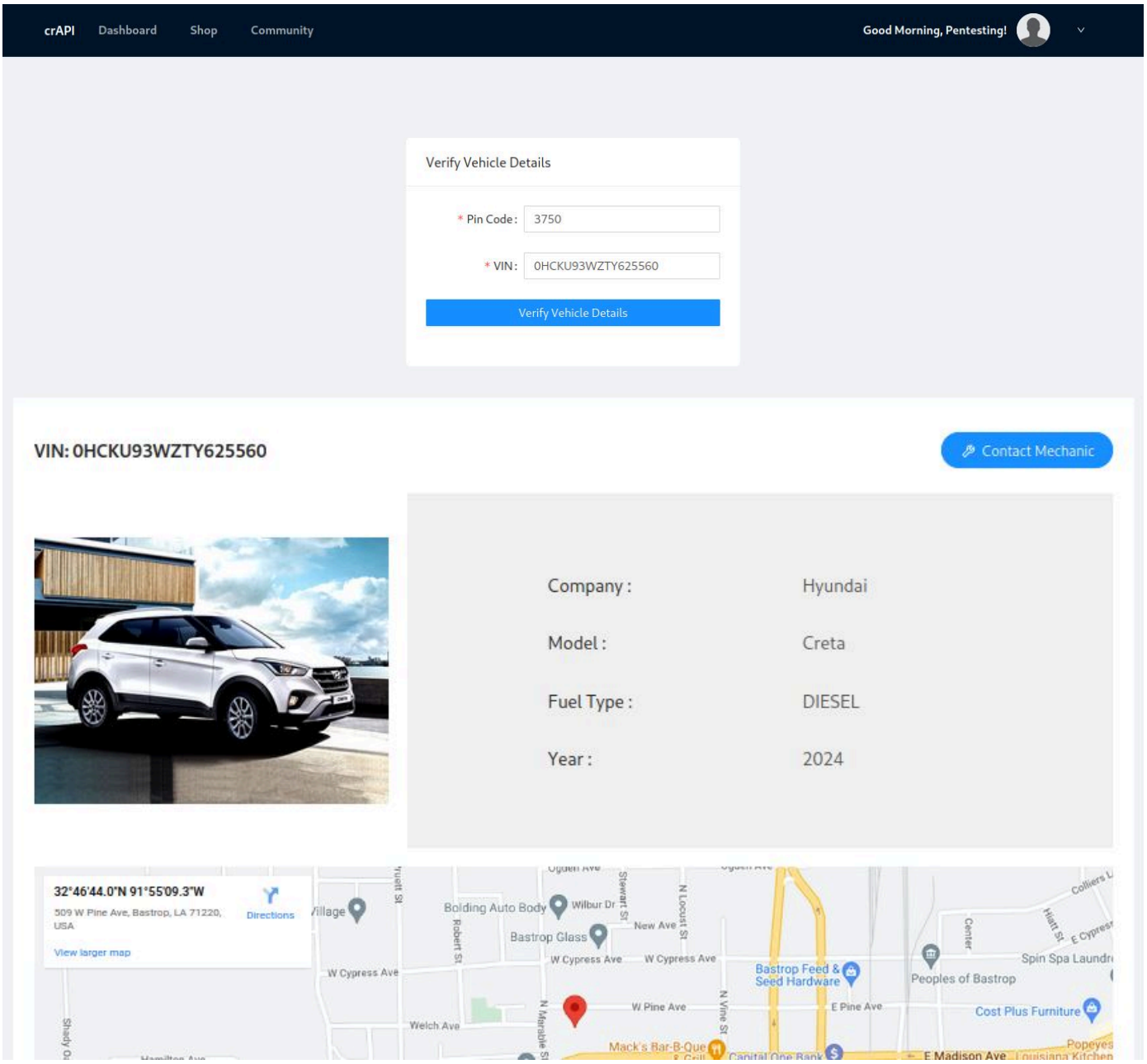
- After successfully logging in press the + **Add a Vehicle** button and we will be asked to enter the PIN Code and PIN



- PIN Code and PIN will be automatically sent to the mailhog, open the mailhog then there will be a new email there that contains the PIN Code and PIN



- Enter the PIN Code and PIN to the crAPI web page, then press **the Verify Vehicle Details** button. If successful will come out the details of the vehicle data and the installation process is complete

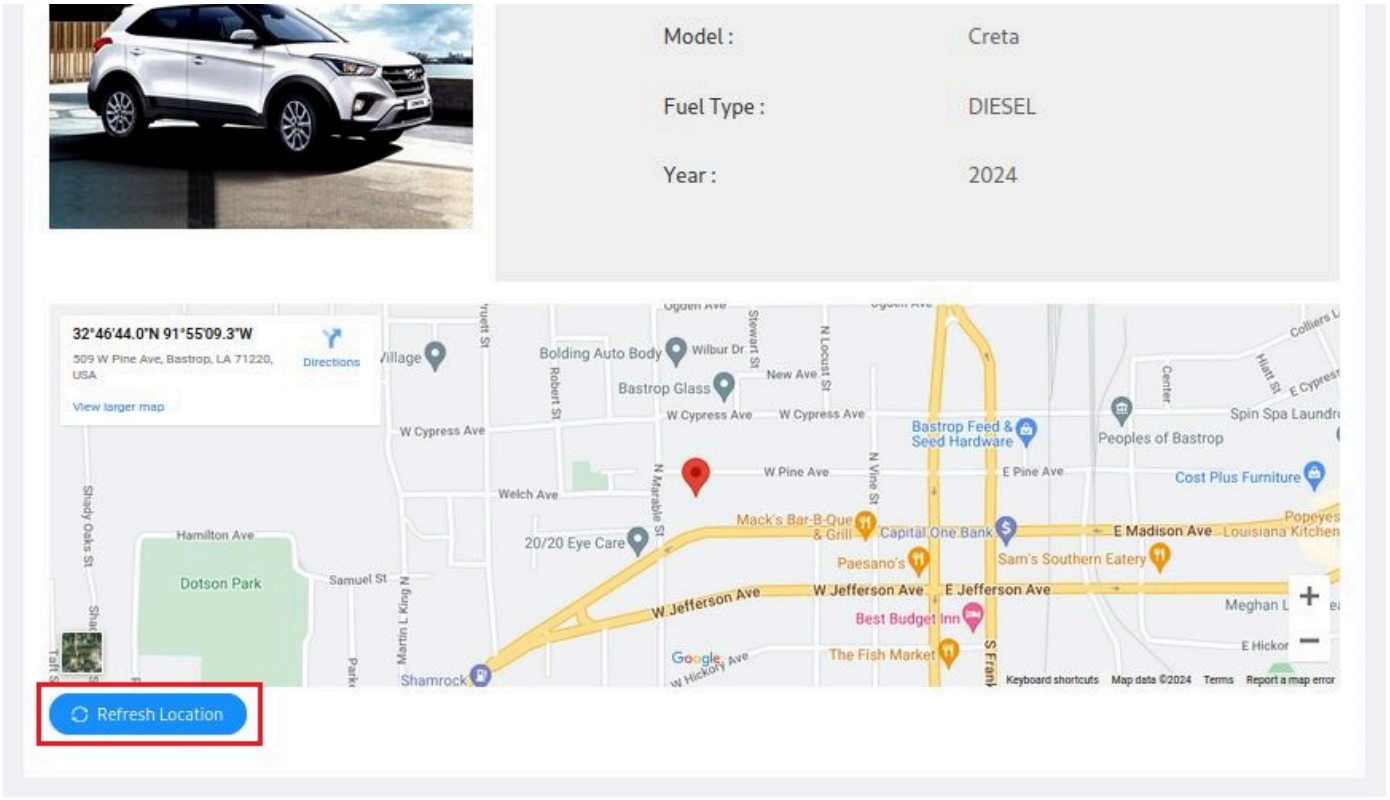


1. Broken Object Level Authorization

Attackants can exploit the vulnerable endpoint API to the Broken Object Level Authorization (BOLA)/IDOR (Insecure Direct Object Reference) by manipulating the object ID sent in the request. The object ID can be anything, ranging from sedential integers, UUID, or strings. Invalid access to other users' objects may result in the disclosure of the data to the unauthorized party, data loss, or manipulation of the data

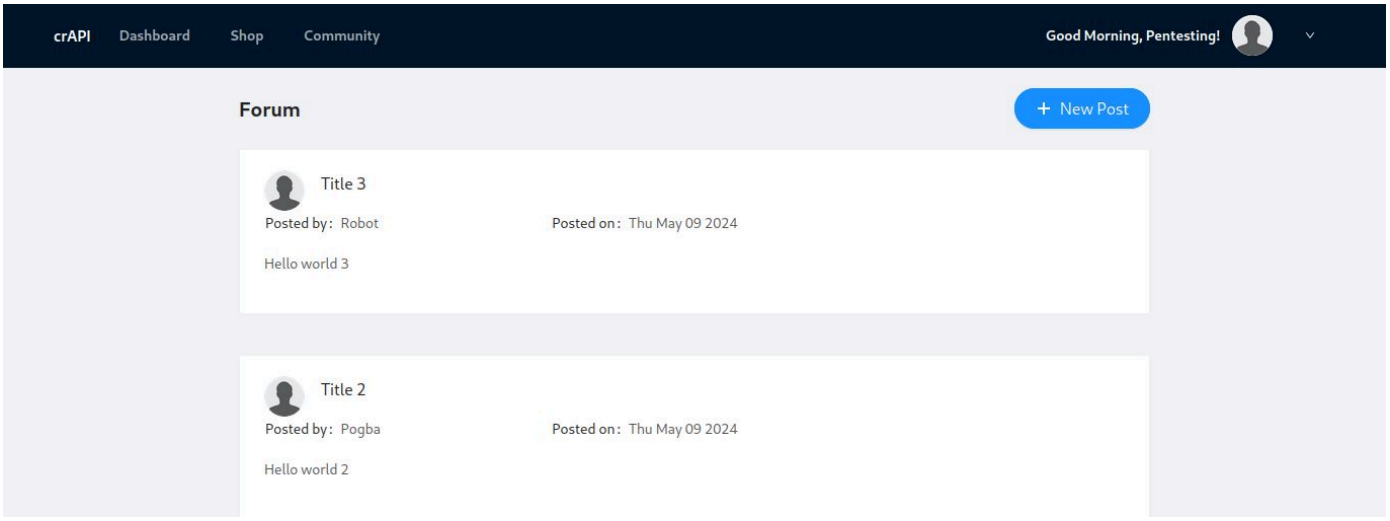
Challenge 1 - Access details of a user's vehicle

- Open the Burp Suite tool and then press **the Refresh Location** button on the dashboard page

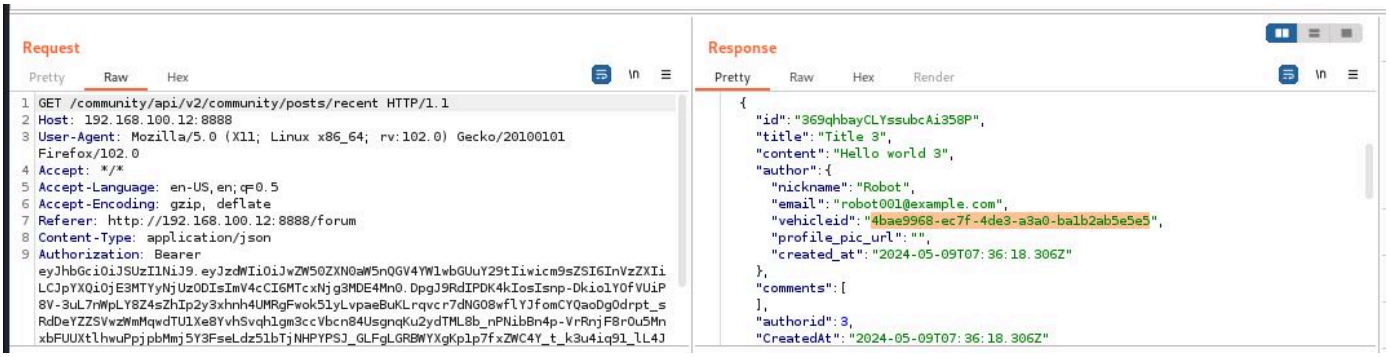


- In the Burp Suite tool will be detected endpoint API which contains vehicle information based on the vehicle ID

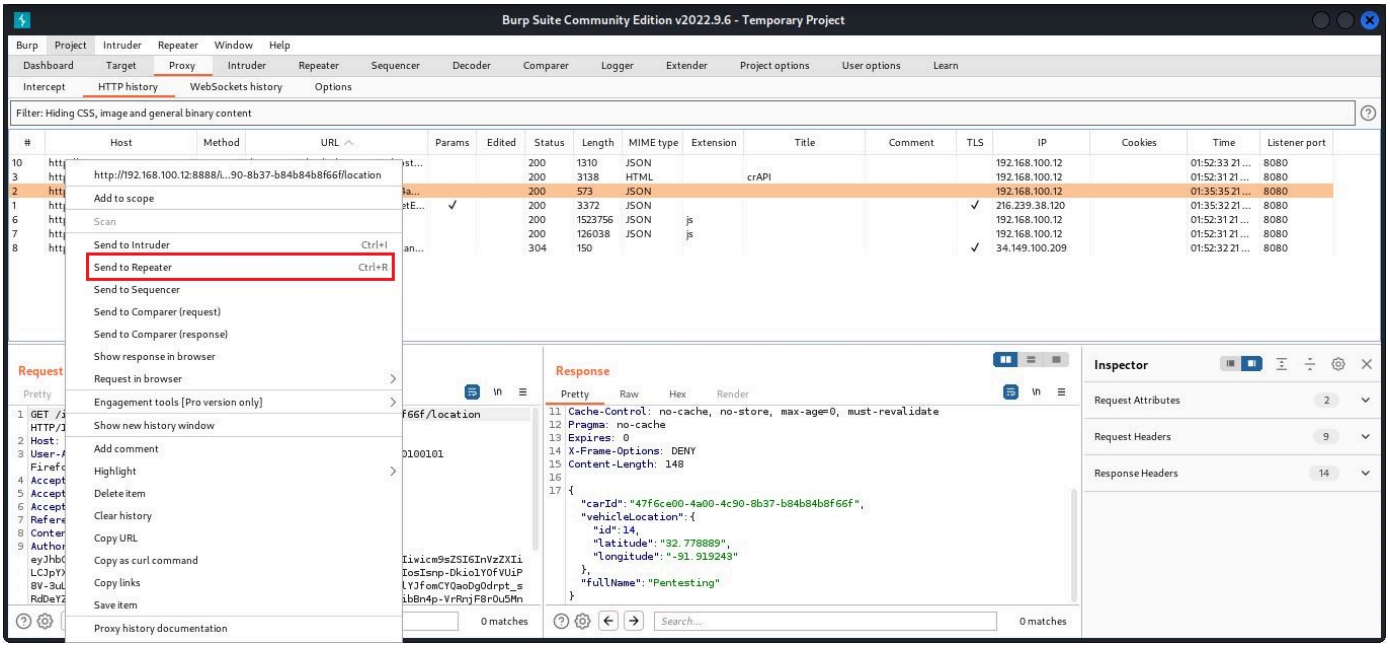
- Click **the Community** menu to open the Community page



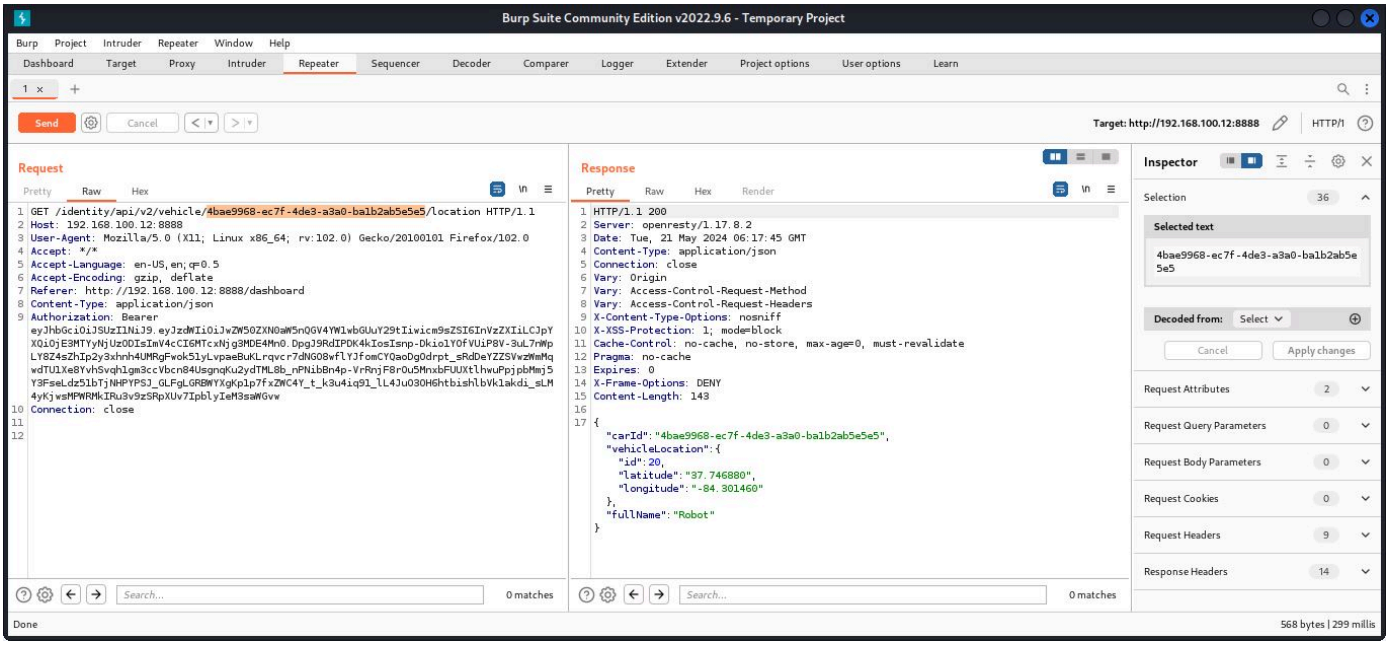
- On the Community page found an endpoint API that contains another user vehicle ID



- Now we send an endpoint API request that contains vehicle information based on the vehicle ID to the Repeater tab

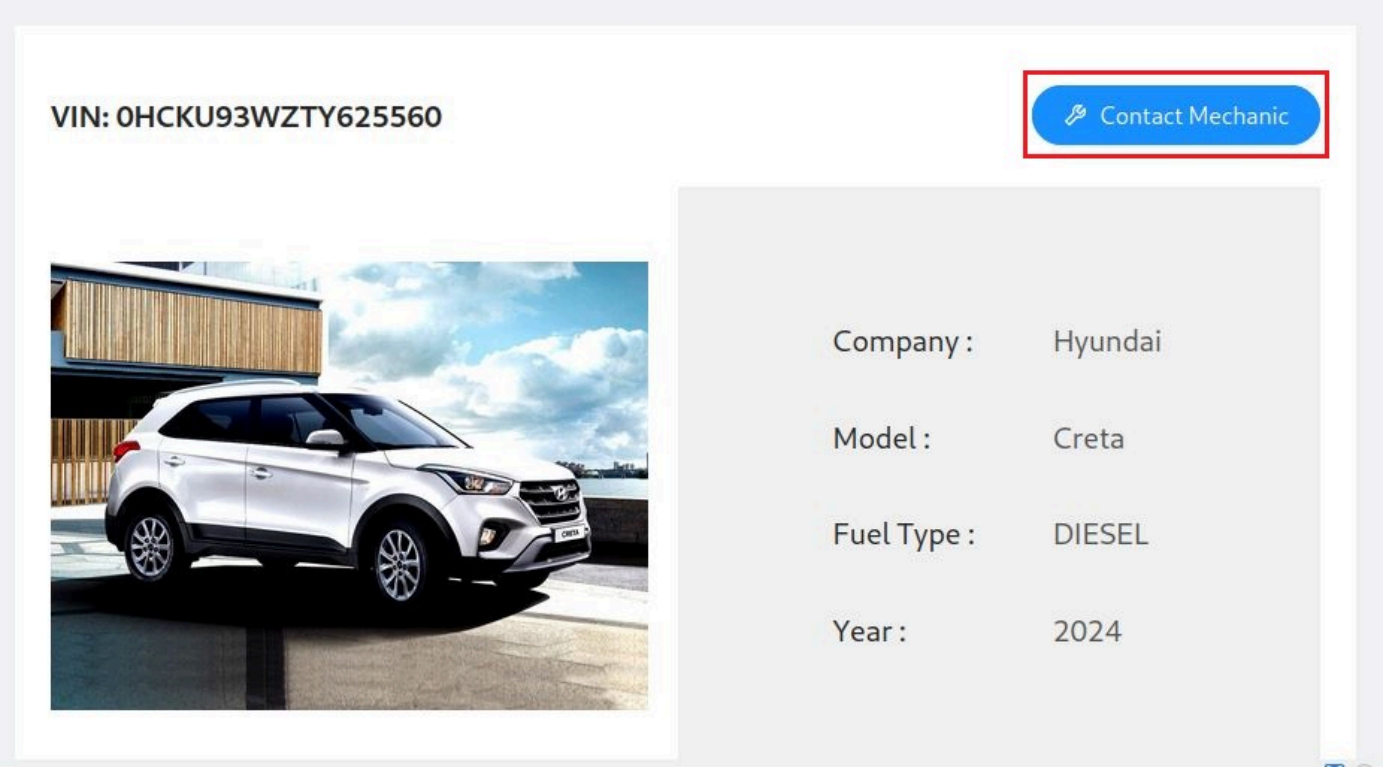


- Then replace the vehicle ID with the vehicle ID found on the Community page and then press **the Send** button then we managed to get vehicle information from other users

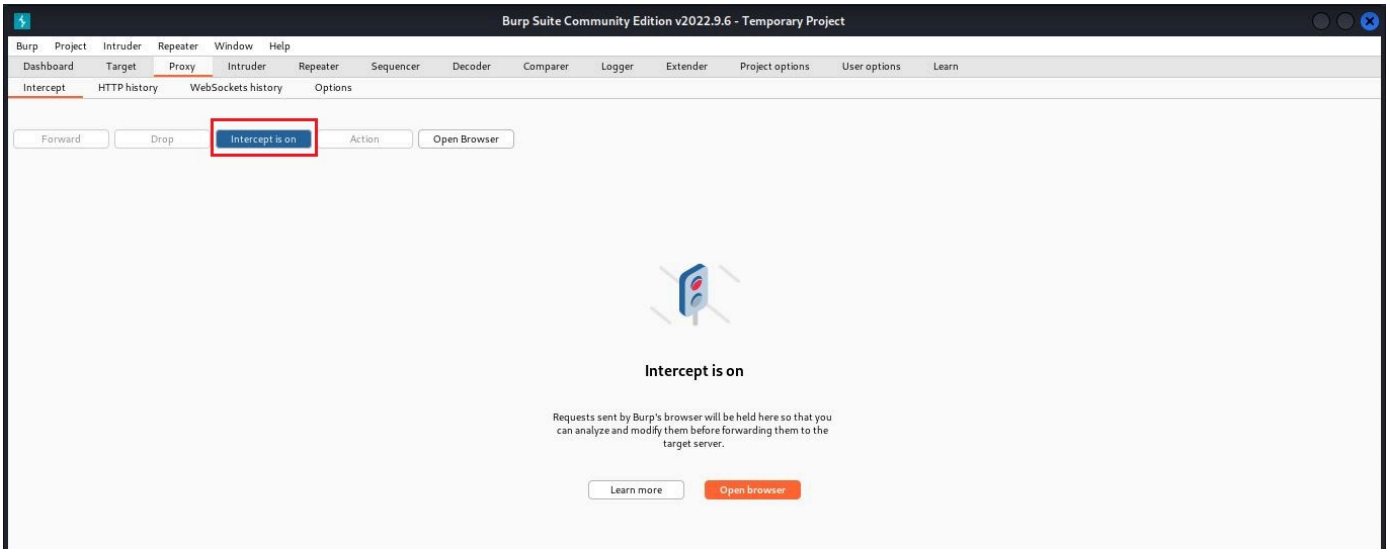


Challenge 2 - Access mechanic reports of other users

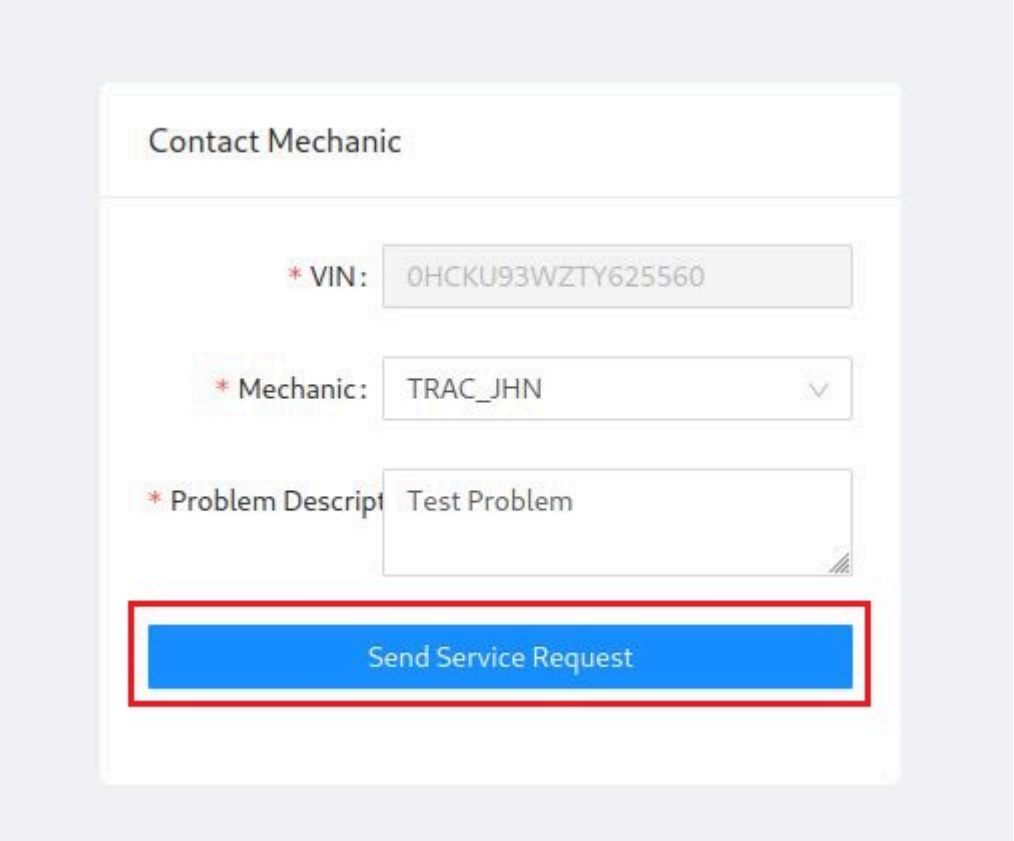
- Back to the dashboard page, press **the Contact Mechanic** button



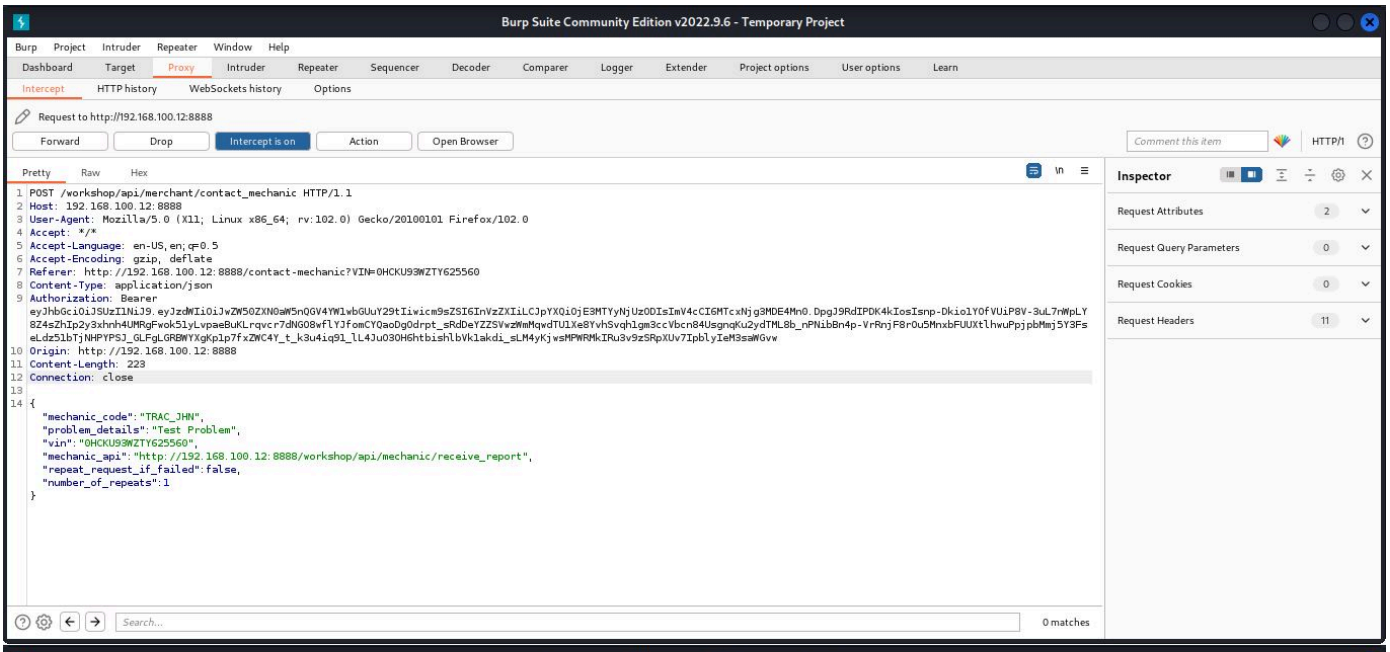
- Turn on the Intercept on the Burp Suite

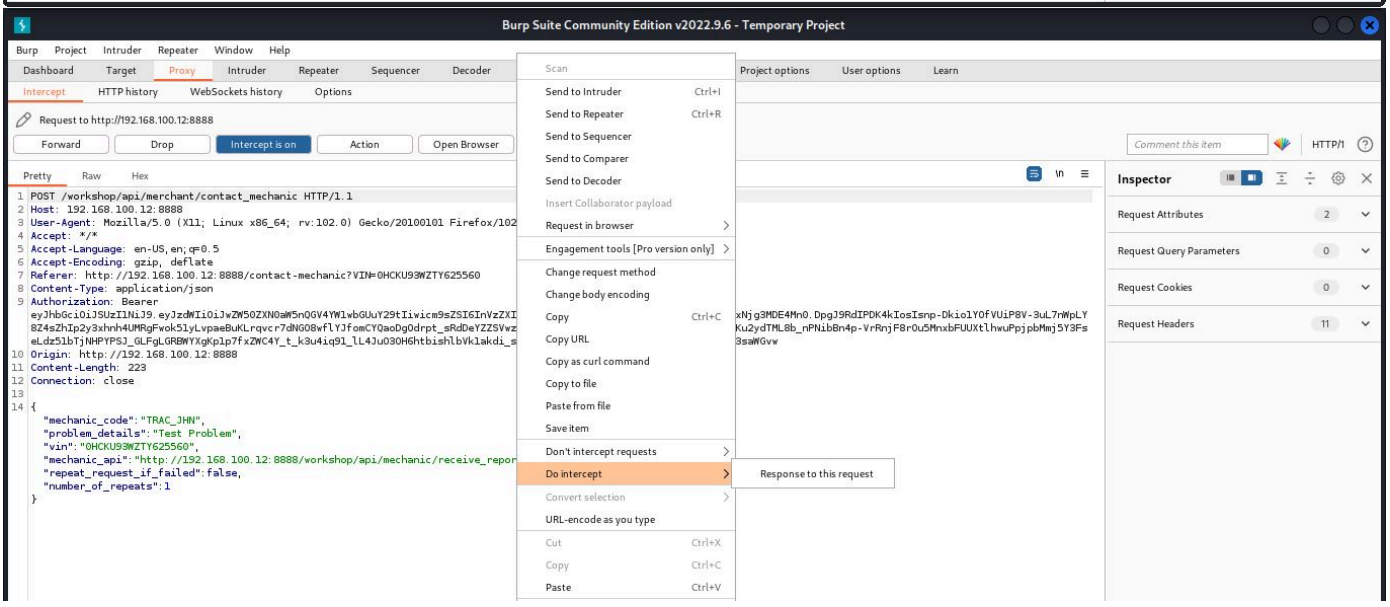


- Fill out the form and press **the Send Service Requests** button

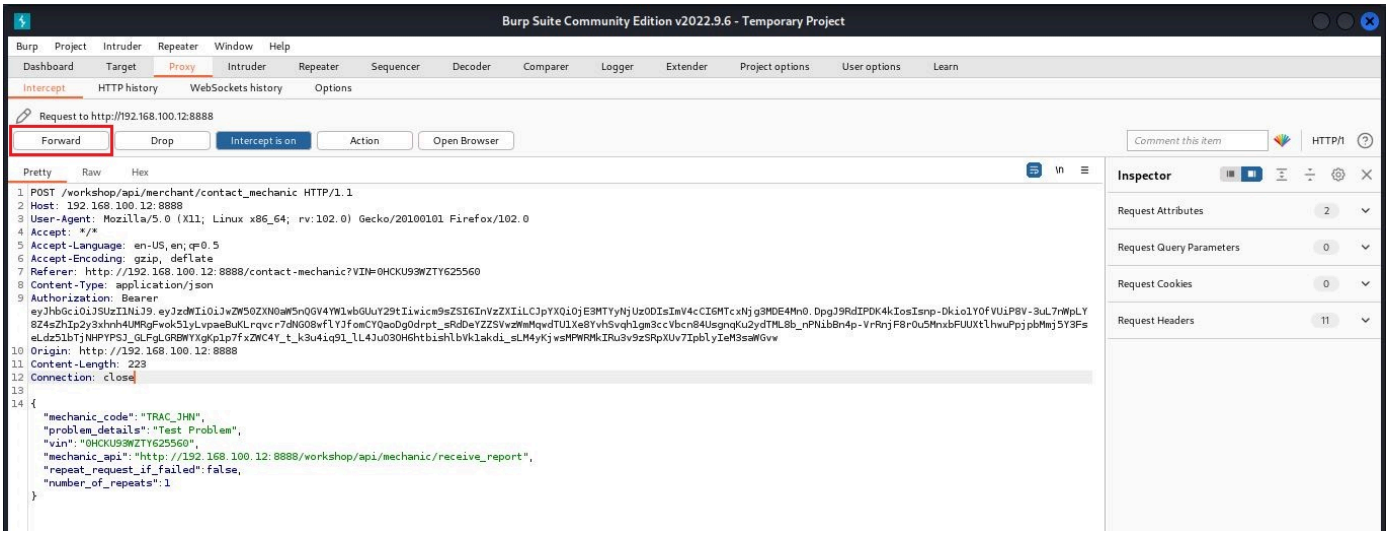


- Once the Burp Suite opens, right-click and select **Do intercept > Response to this request**

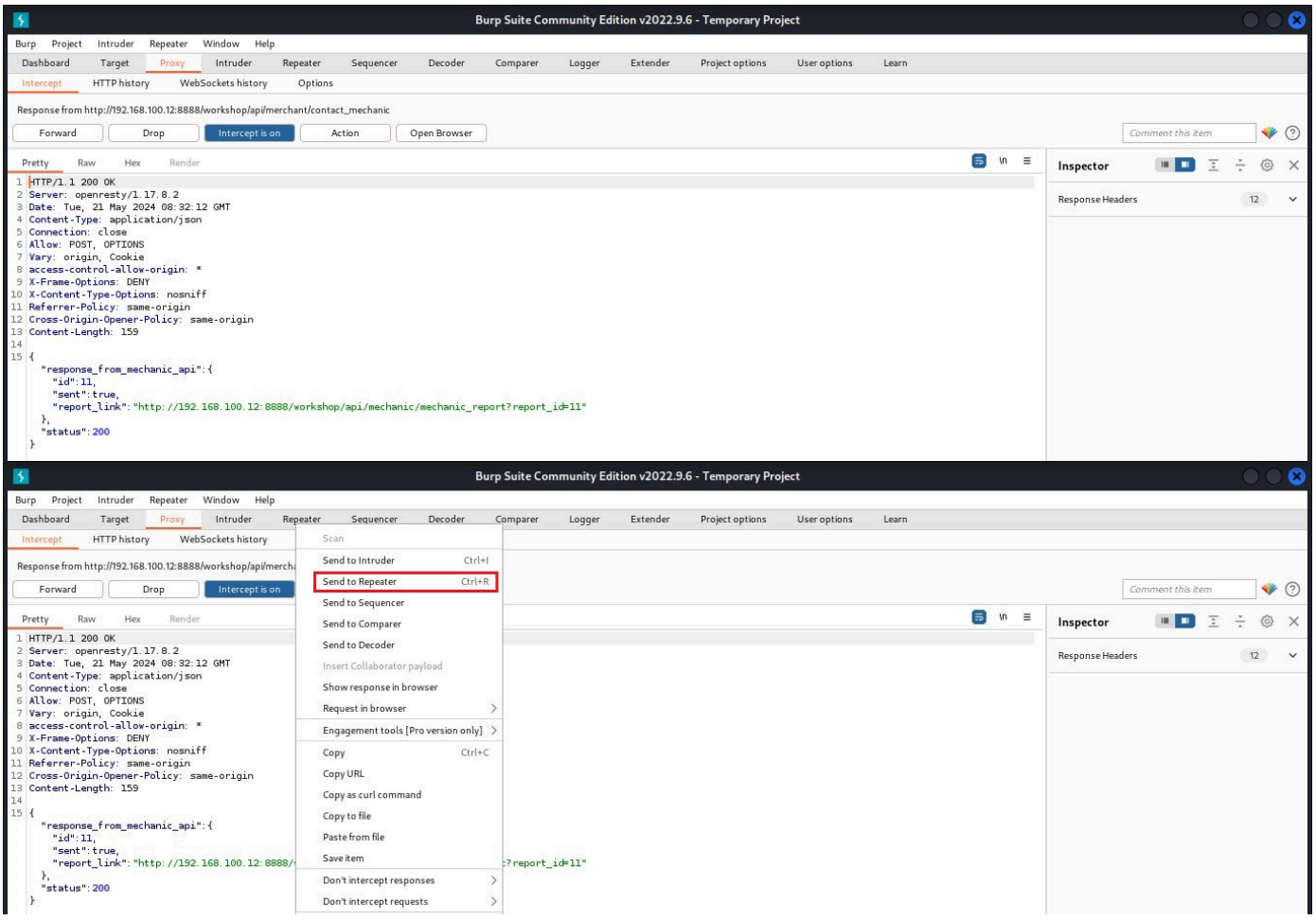




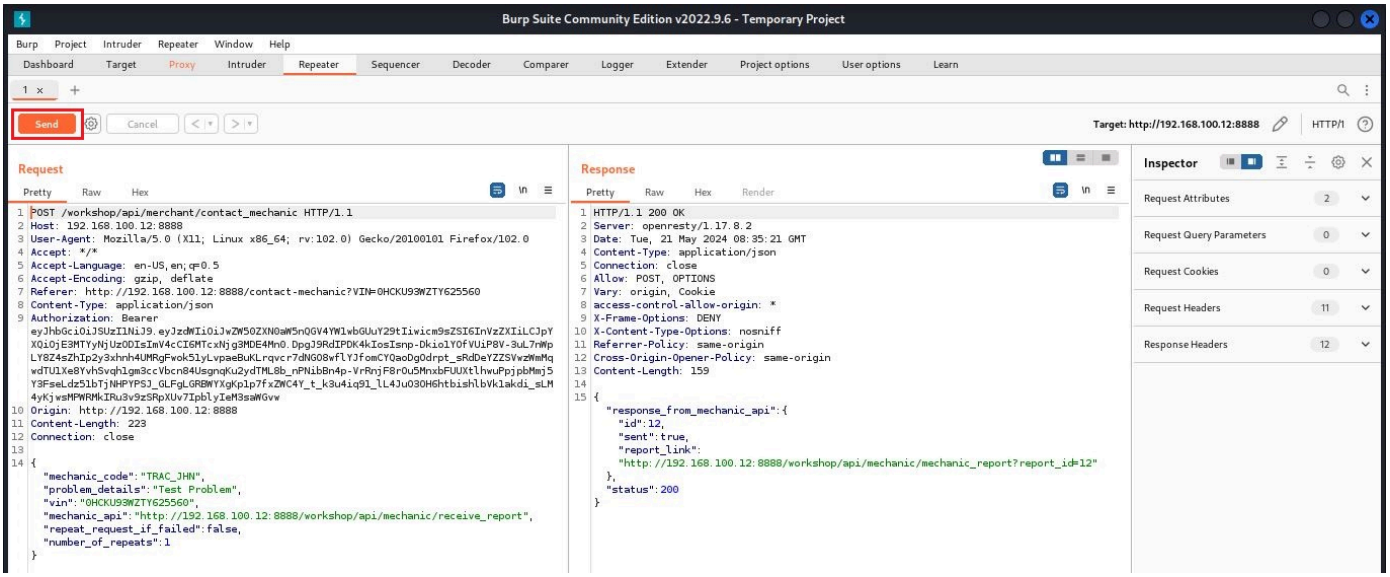
- Next press **the Forward** button



- If you get the following response, right-click and then select **Send to Repeater**



- Move to the Repeater tab and press **the Send** button



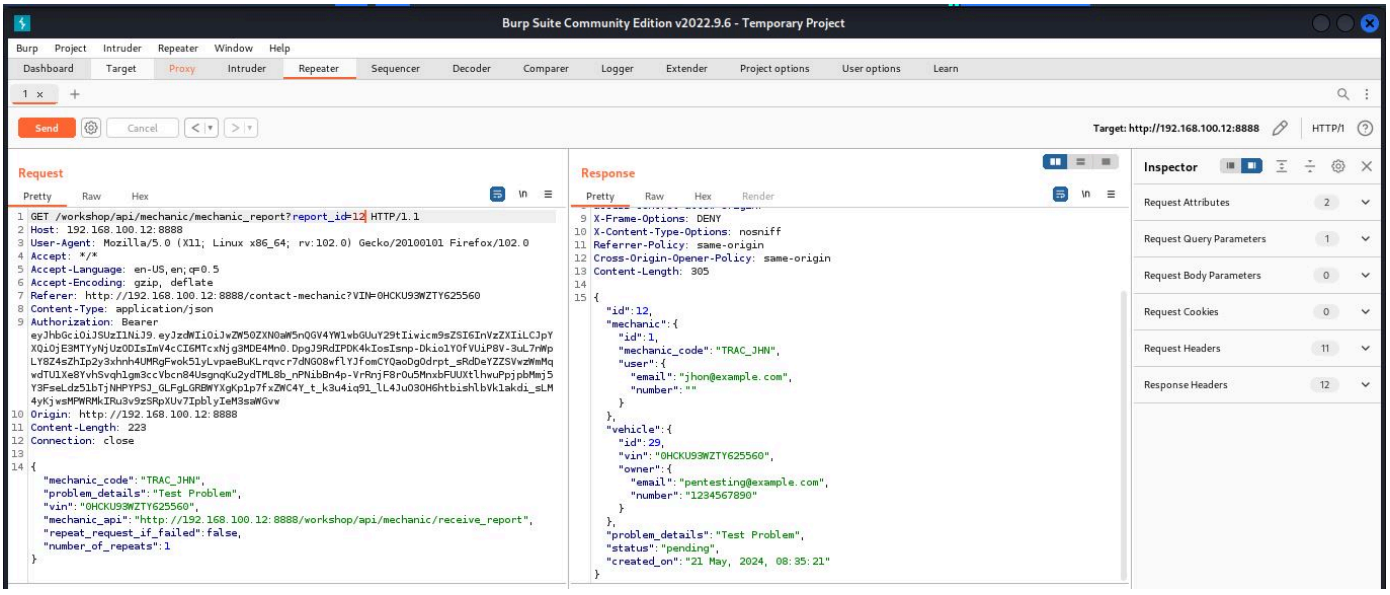
- Change the request `POST` Be a `GET` and change the endpoint API to `/workshop/api/mechanic/mechanic_report?report_id=?` to access the report details

Request

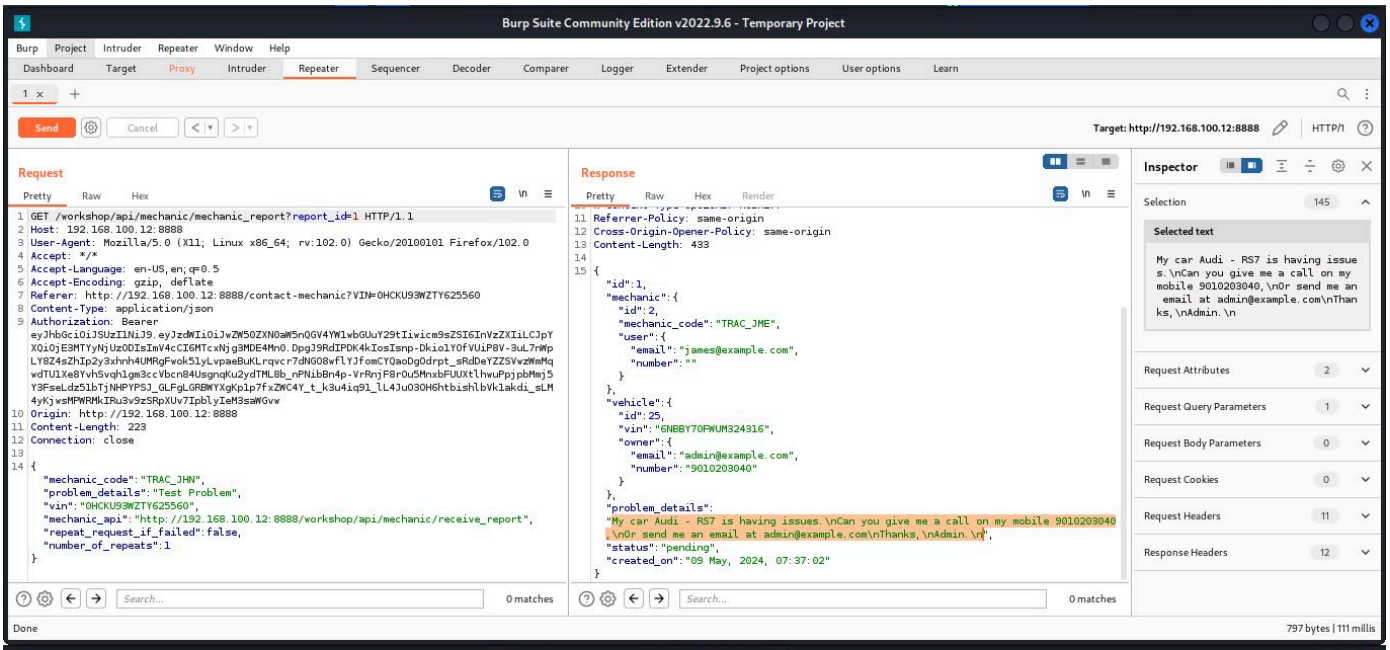
Pretty Raw Hex

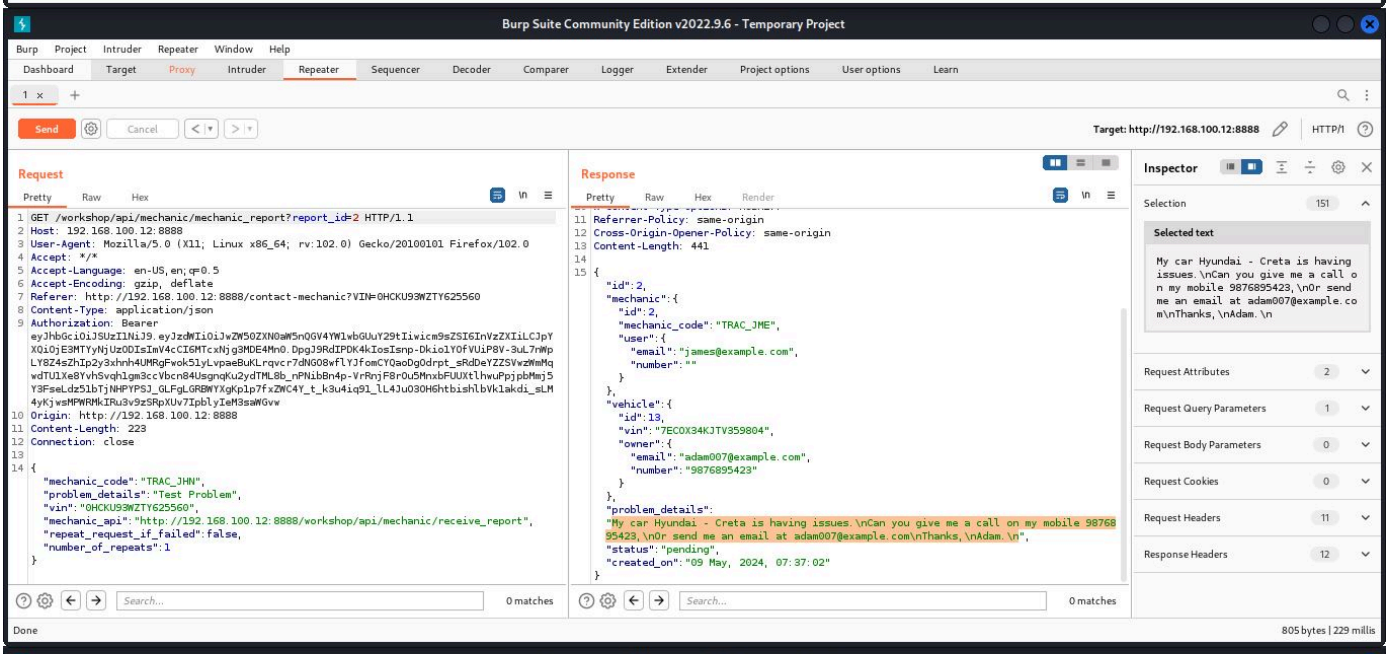
```
1 GET /workshop/api/mechanic/mechanic_report?report_id=12 HTTP/1.1
2 Host: 192.168.100.12:8888
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.100.12:8888/contact-mechanic?VIN=0HCKU93WZTY625560
8 Content-Type: application/json
9 Authorization: Bearer
  eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJwZW50ZXN0aW5nQGV4YW1wbGUuY29tIiwicm9sZSI6InVzZXIiLCJpY
  XQiOjE3MTYyNjUzODIsImV4cCI6MTcxNjg3MDE4Mn0.DpgJ9RdIPDK4kIosIsnp-Dkio1YOfVUiP8V-3uL7nWp
  LY8Z4sZhIp2y3xhnh4UMRgFwok51yLvpaeBuKLrqvcr7dNGO8wflYJfomCYQaoDg0drpt_sRdDeYZZSVwzWmMq
  wdTUlXe8YvhSvqh1gm3ccVbcn84UsgnqKu2ydTML8b_nPNibBn4p-VrRnjF8r0u5MnxbFUUXtlhwuPpjpbMmj5
  Y3FseLdz51bTjNHPYPSJ_GLFgLGRBWYXgKp1p7fxZWC4Y_t_k3u4iq91_lL4Ju03OH6htbishlbVk1akdi_sLM
  4yKjwsMPWRMkIRu3v9zSRpXUv7IpblyIeM3saWGvw
10 Origin: http://192.168.100.12:8888
11 Content-Length: 223
12 Connection: close
13
14 {
     "mechanic_code":"TRAC_JHN",
     "problem_details":"Test Problem",
     "vin":"0HCKU93WZTY625560",
     "mechanic_api":"http://192.168.100.12:8888/workshop/api/mechanic/receive_report",
     "repeat_request_if_failed":false,
     "number_of_repeats":1
   }
```

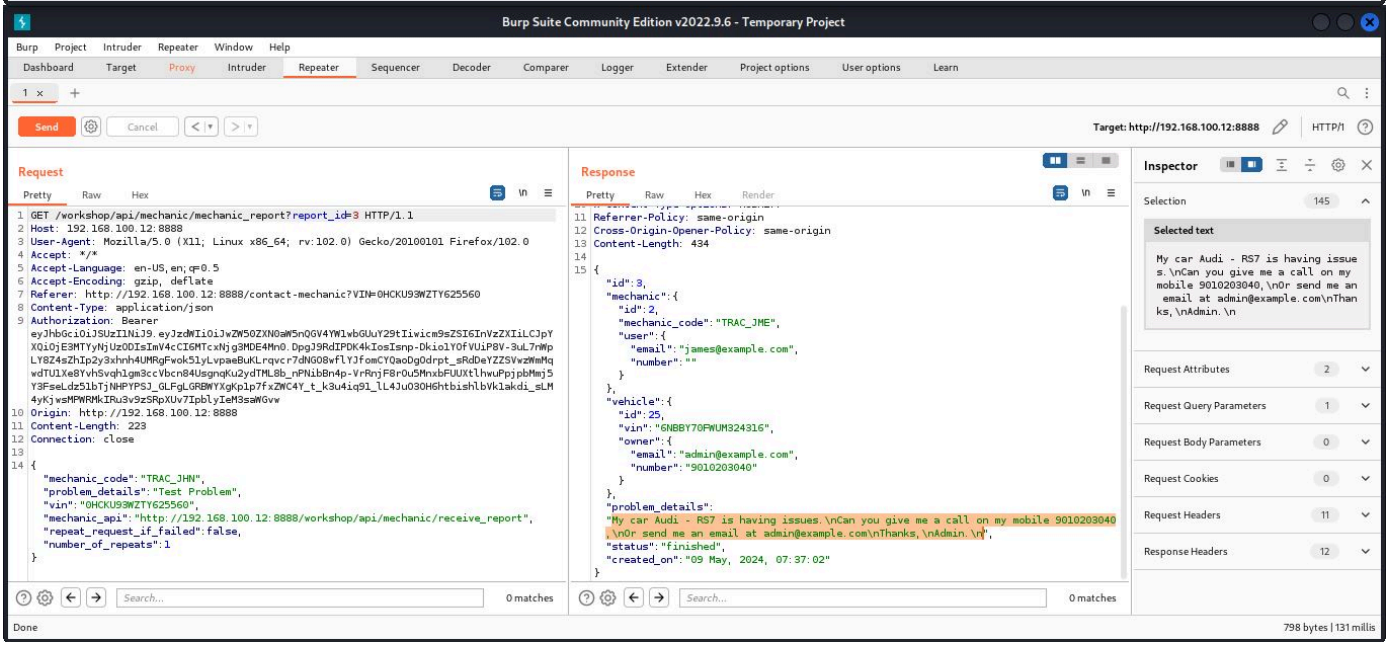
- If you have pressed **the Send** button then we managed to access the details of the report that we have sent before.



- From this it is seen that the parameters `report_id` using integers, thus allowing us to change it with other integers such as numbers 1, 2, and 3 to access reports sent by other users.





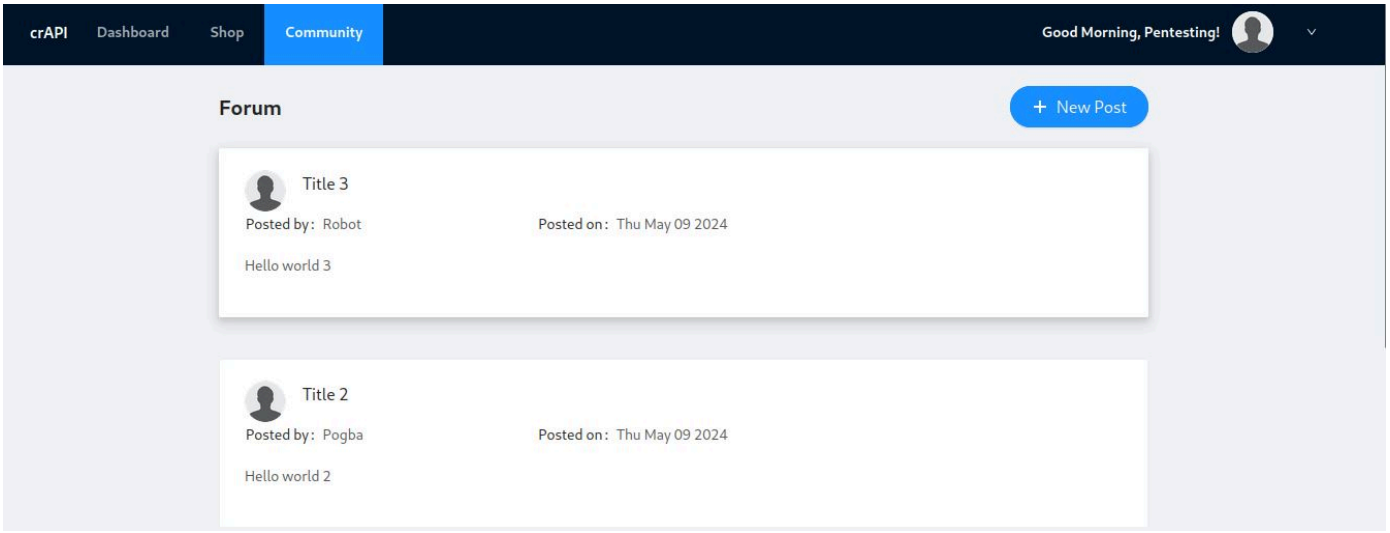


2. Broken User Authentication

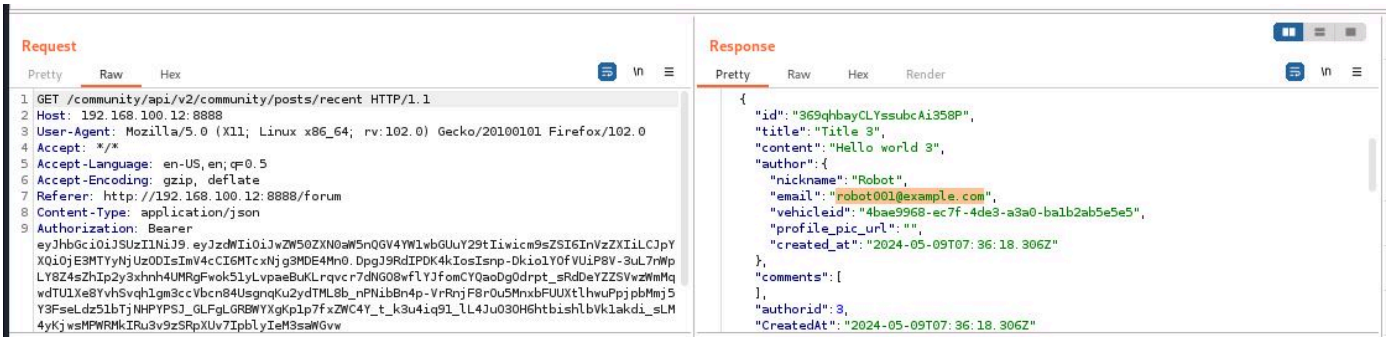
App developers' misunderstandings regarding authentication limitations cause attackers to be able to gain complete control over other users' accounts on the system, such as reading their personal data and performing sensitive actions on their behalf. The system cannot distinguish which actions are from the attacker and which one is from a legitimate user.

Challenge 3 - Reset the password of a different user

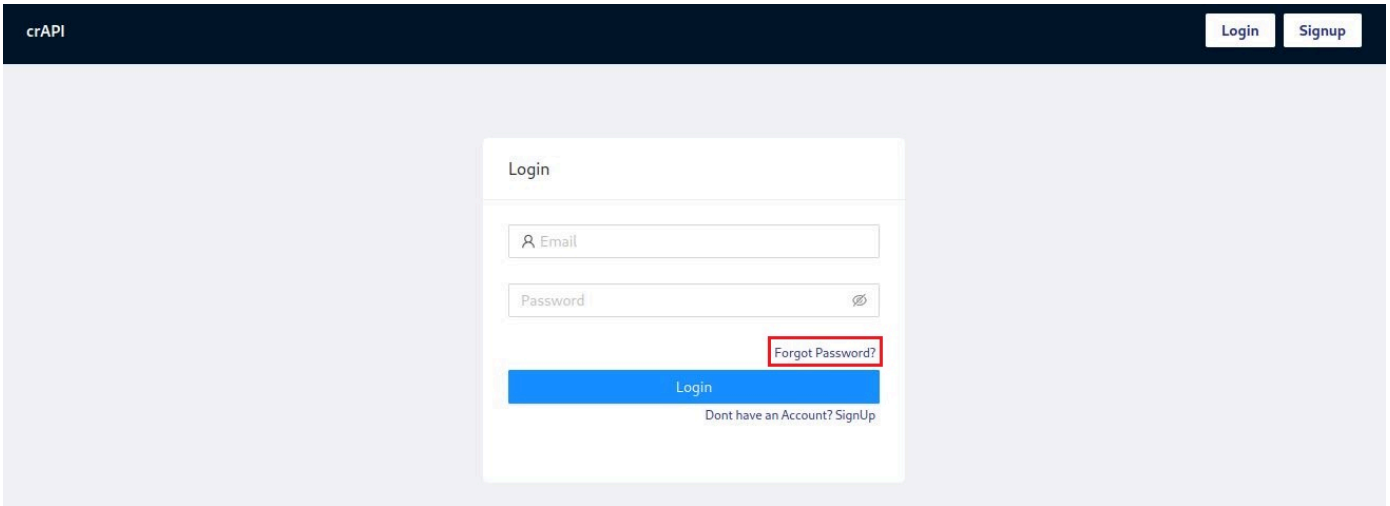
- Turn on the Burp Suite tool and go to **the Community** page



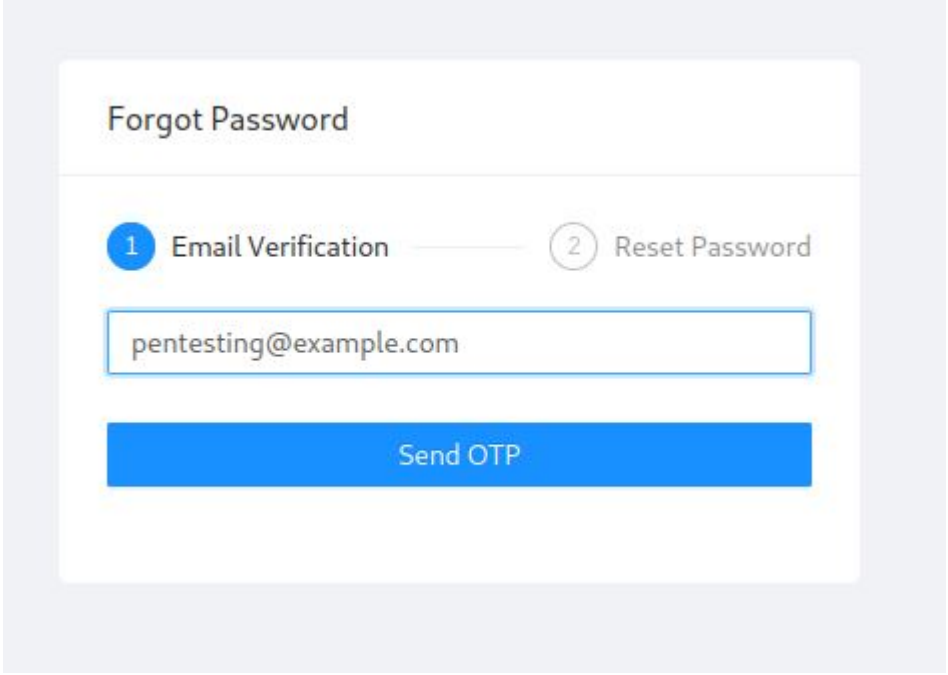
- In the Burp Suite tool it is detected that the page requests to the endpoint API `/community/api/v2/community/posts/recent` whose response contains email from other users that we can use to do a **password reset** request

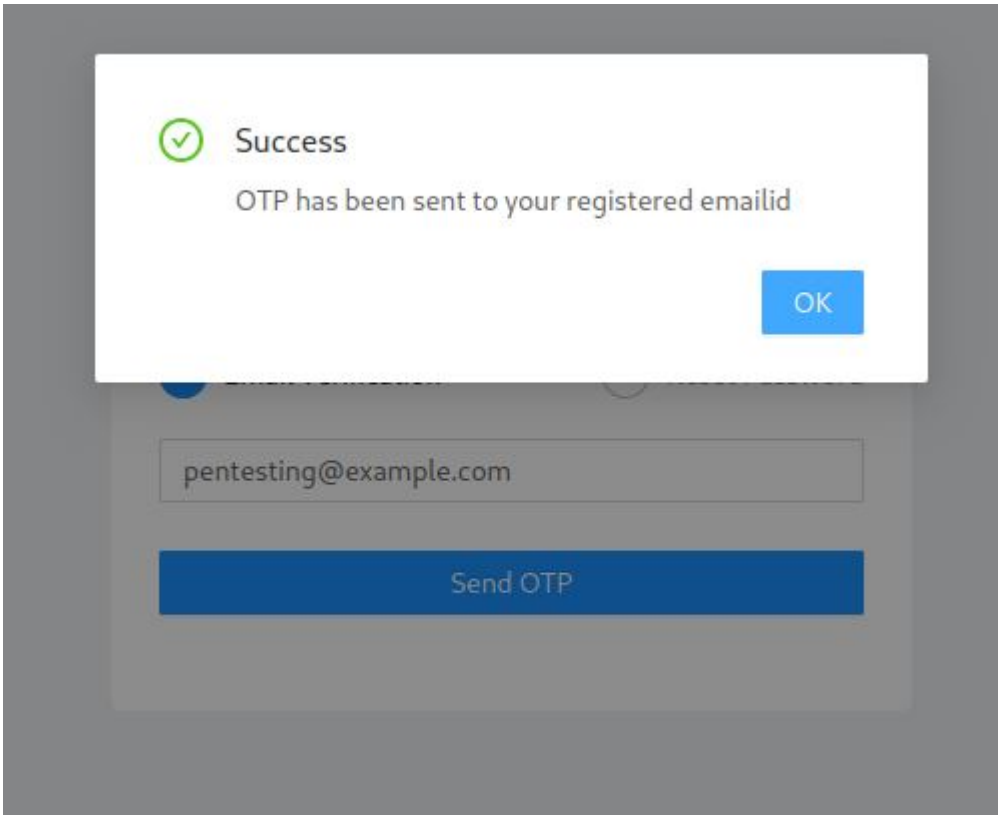


- Now we log over from the account we use then we try **the Forgot Password** feature

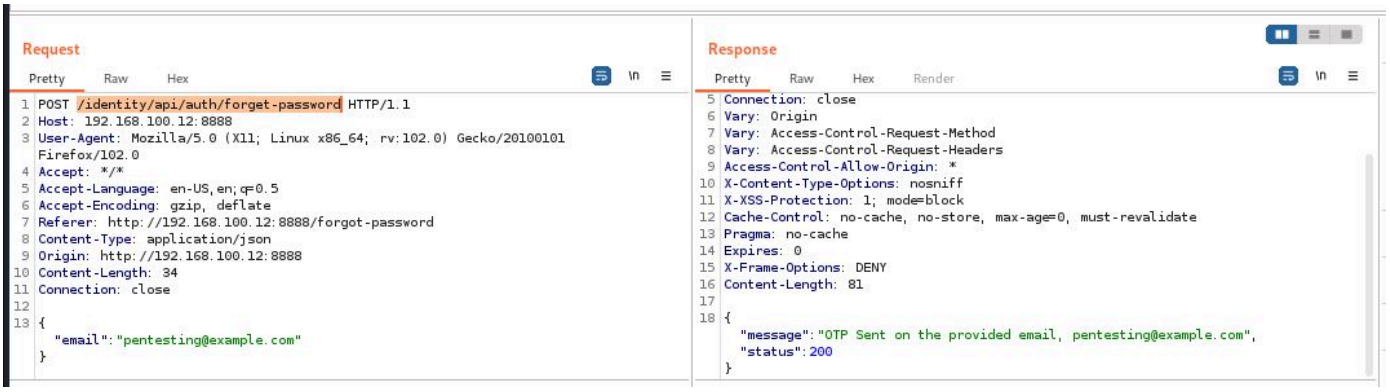


- Then fill in the email that we use to access the crAPI web and press **the Send OTP** button

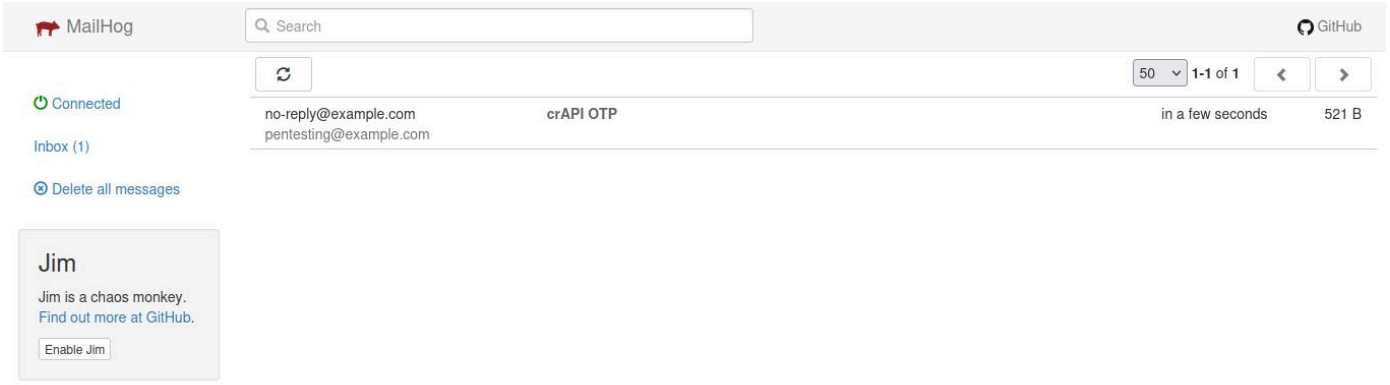




- On the Burp Suite tool detected endpoint API `/identity/api/auth/forget-password` to send an OTP code to the email

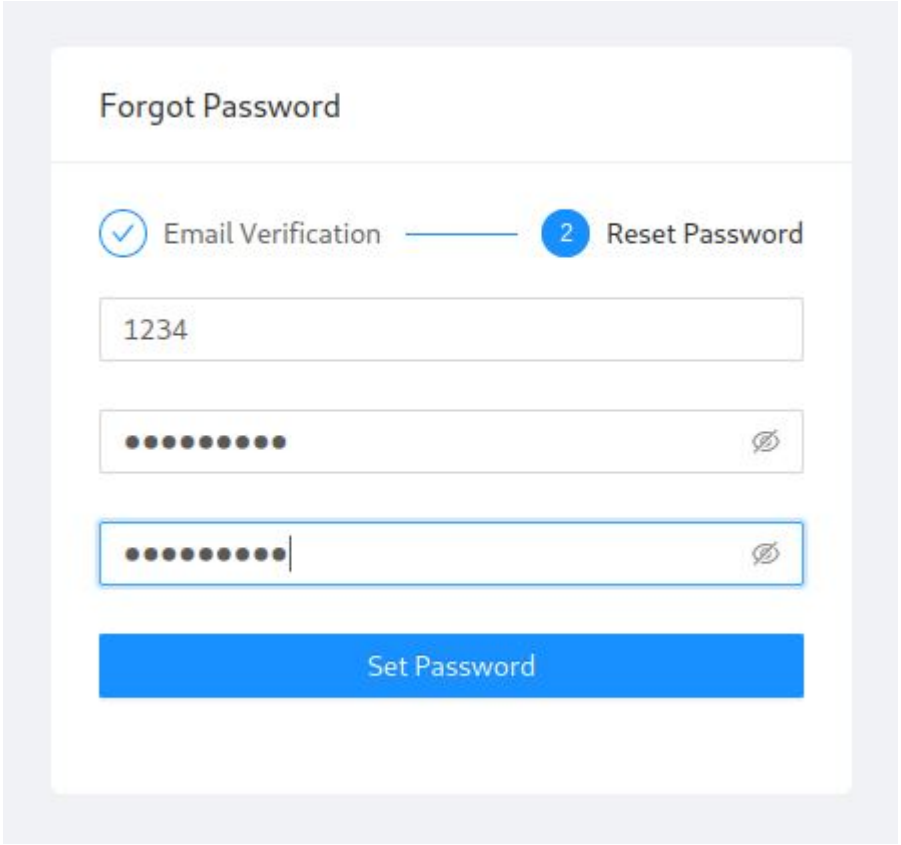


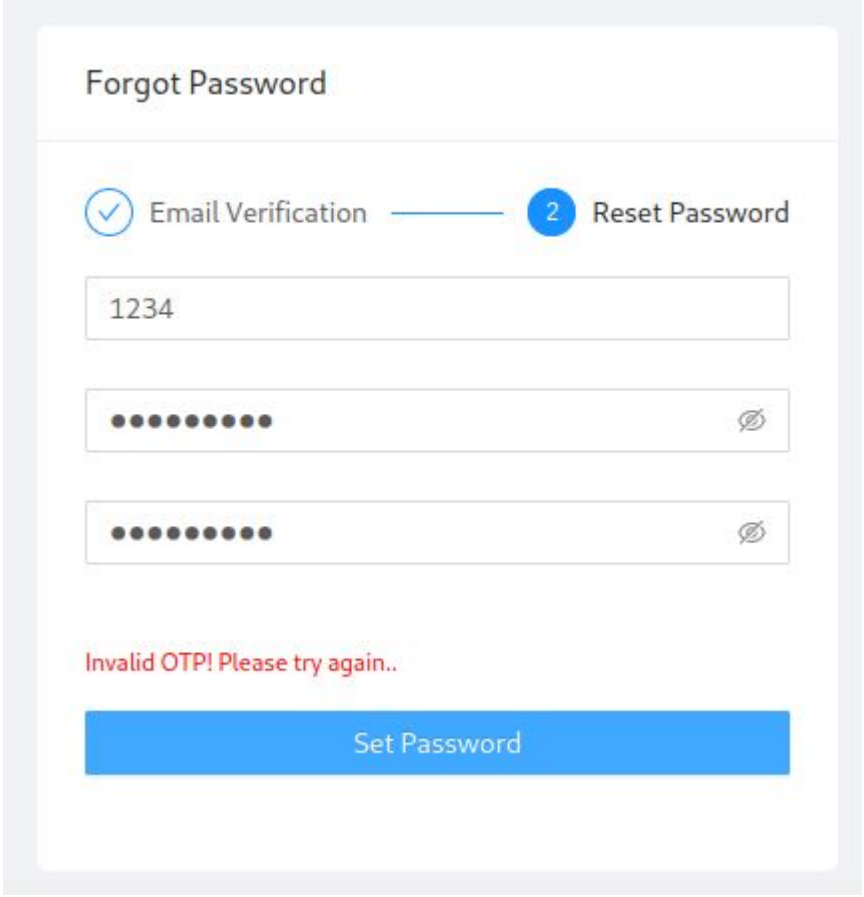
- Open the mailhog in a new tab with URL `http://IP_Server:8025`, here we receive an email containing the OTP code



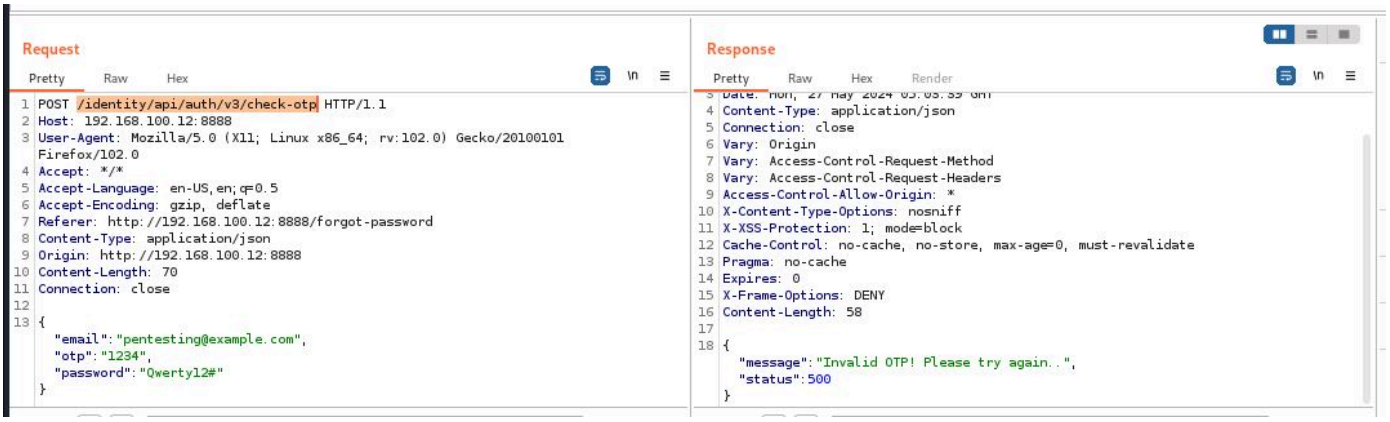
- The OTP code is a number consisting of 4 digits

- Now we try to enter the wrong OTP code into the forgotten password page and enter a new password and then press **the Set Password** button then we will get a message `Invalid OTP`

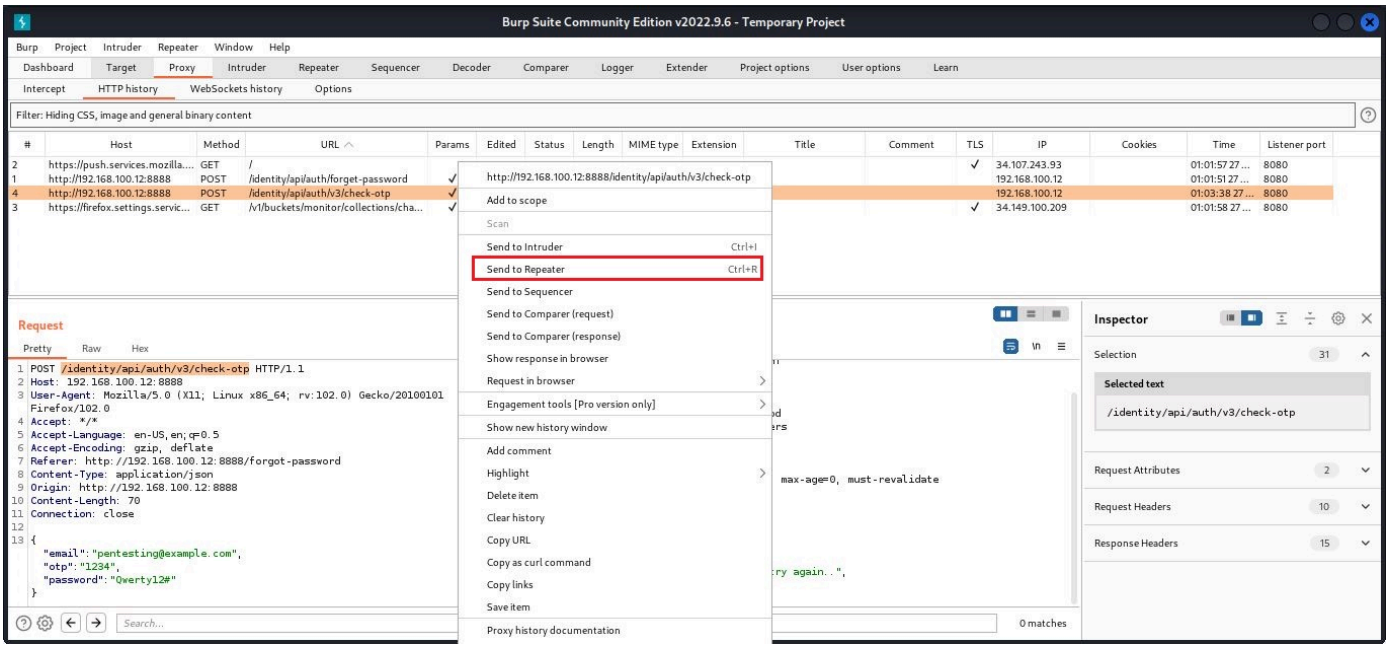




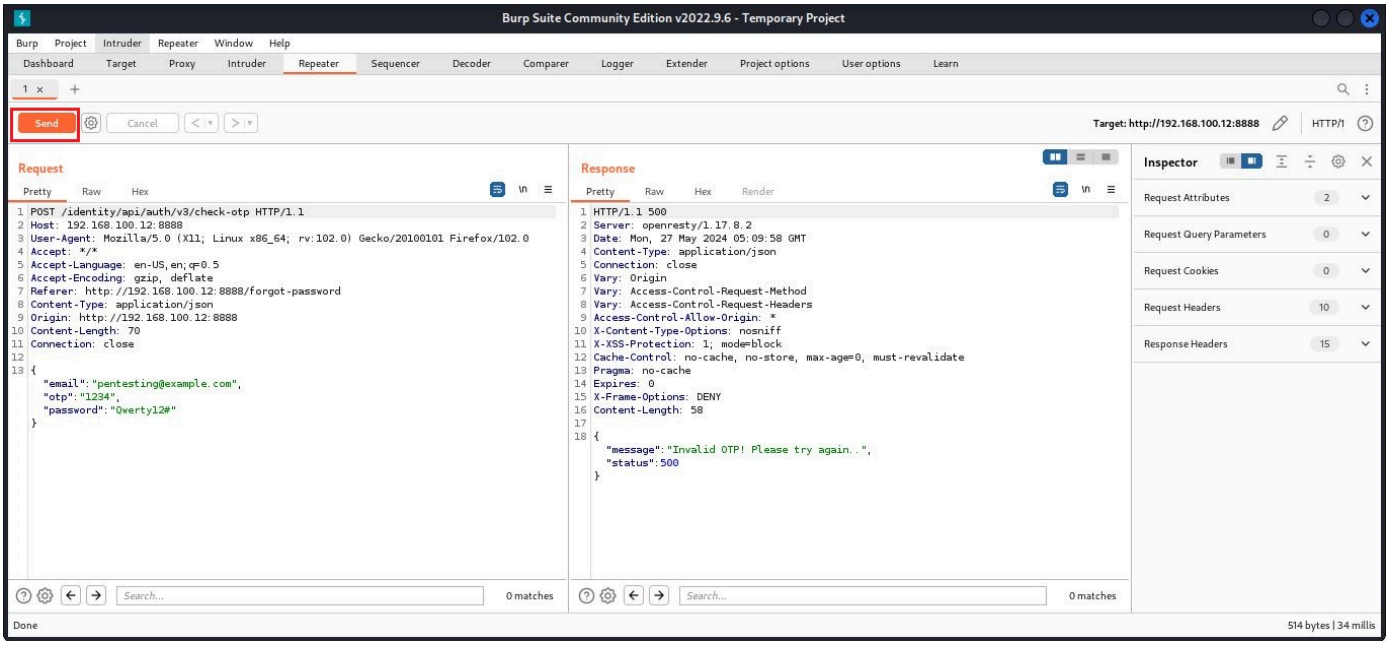
- On the Burp Suite tool detected endpoint API `/identity/api/auth/v3/check-otp` which is used to check the OTP code and reset the password



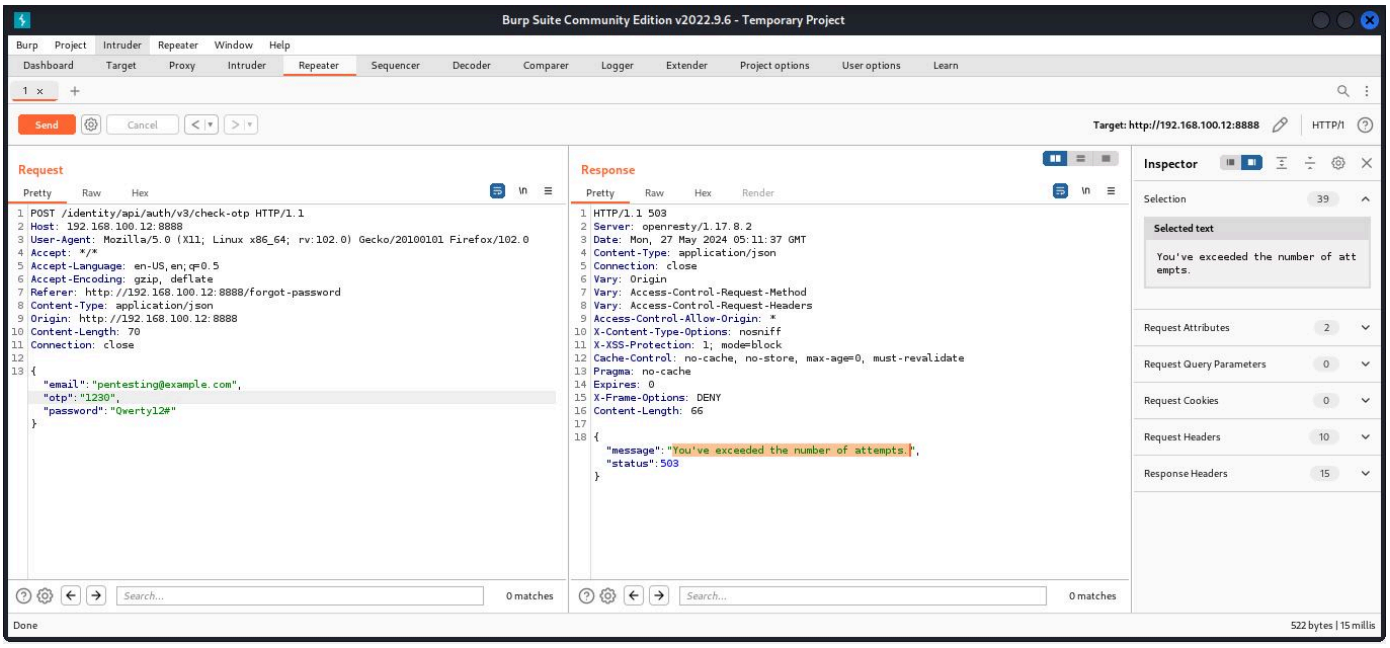
- Right-click the request and select **Send to Repeater**



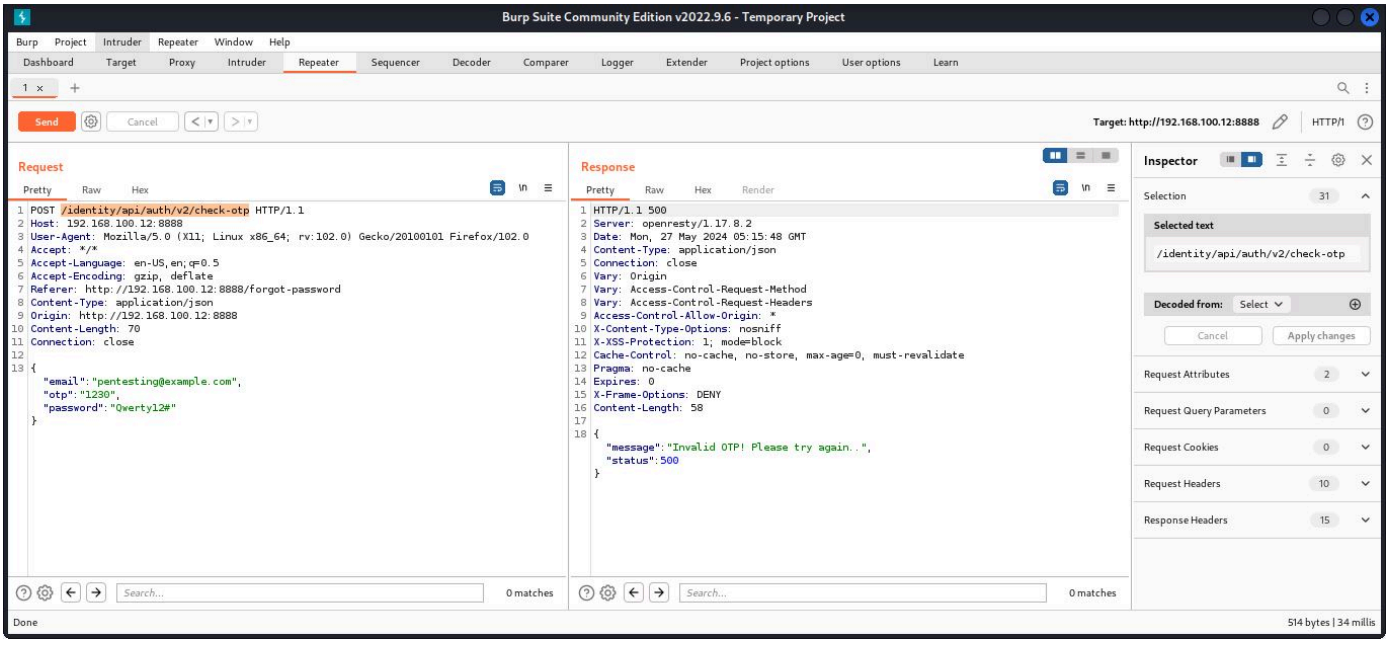
- Move to the Repeater tab and press **the Send** button



- Here we can try to enter the wrong OTP code repeatedly to ascertain whether there are restrictions on requests or not. After several times we got the message that our request reached the maximum limit



- Once we try to change the endpoint from `v3` Be a `v2` we find that this endpoint does not have a specified request limit so that it allows us to brute force OTP code and change the password of other users that we find on **the Community** page



- Here we will create a program using python language to do brute force OTP code and reset the password on another user account. First we will send a request to the endpoint `/identity/api/auth/forget-password` to send an OTP code to the email. After that reset the password by doing the OTP code trial from `0000` Until `9999` according to the OTP code format received in the email via endpoint `/identity/api/auth/v2/check-otp`. Here is the program code that we use, save the script below with the name `brute_otp_crapi.py`

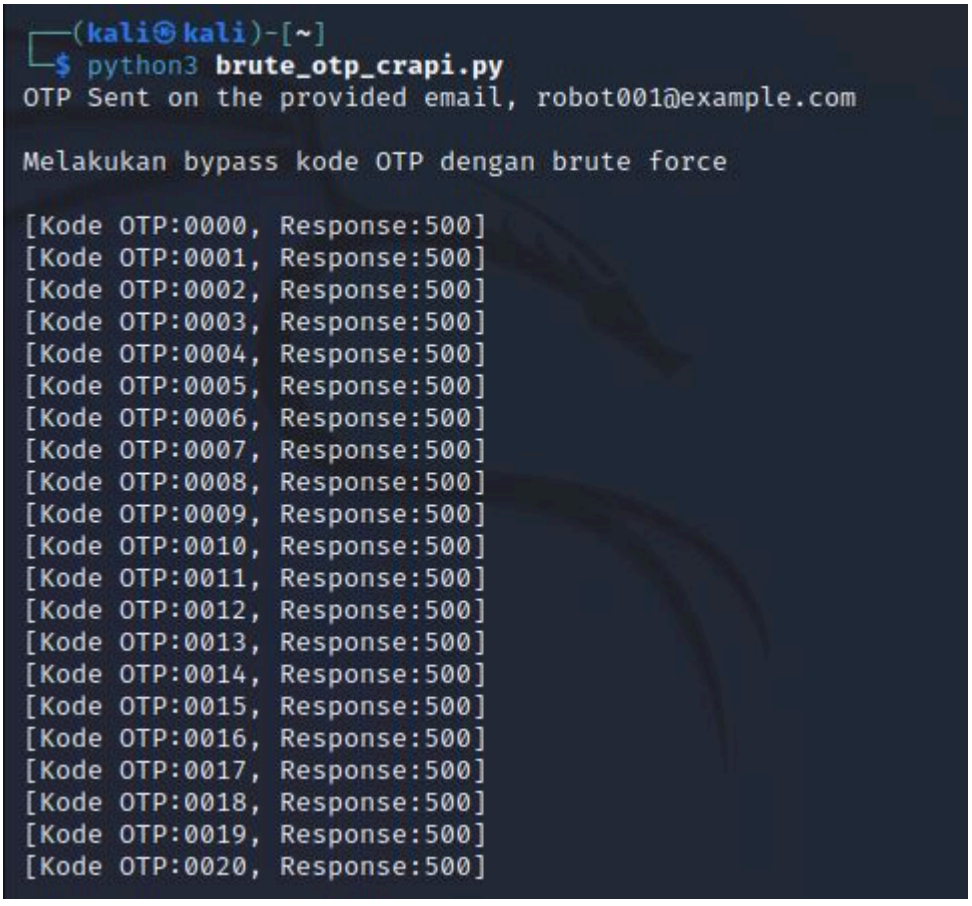
```
import requests
import time

start = time.time()

url = 'http://IP_Server:8888' #ubah url sesuai url server anda
```

```
email = 'email' #ubah dengan email yang anda temukan

headers = {
    'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
Firefox/102.0',
    'Accept': '*/*',
    'Accept-Language': 'en-US,en;q=0.5',
    'Accept-Encoding': 'gzip, deflate',
    'Referer': url+'/forgot-password',
    'Origin': url,
    'Connection': 'close',
    'Content-Type': 'application/json',
    'Cache-Control': 'no-cache',
    'Pragma': 'no-cache',
}

# Mengirim kode OTP ke email
response = requests.post(url+'/identity/api/auth/forget-password',
headers=headers, json={"email": email})

if response.status_code == 200:
    json_response = response.json()
    print(json_response["message"])
    response.close()
    print("\nMelakukan bypass kode OTP dengan brute force\n")

    #bypass kode OTP dengan brute force
    for i in range(10000):
        #padding angka 0
        if i < 1000:
            otp = f"{i:04}"
        else:
            otp = i

        data_reset_password = {
               'email': email,
               'otp': otp,
               'password': 'password', #ubah password sesuai keinginan anda
        }

        r = requests.post(url+'/identity/api/auth/v2/check-otp',
headers=headers, json=data_reset_password)
        print('[Kode OTP:{}, Response:{}]'.format(otp,r.status_code))
```

```
        if r.status_code == 200:
            print("Berhasil melakukan reset password")
            print(r.json())
            r.close()
            break
        elif r.status_code == 503:
            print("Request telah dibatasi")
            print(r.json())
            r.close()
            break

        r.close()


else:
    print("Permintaan tidak dapat dikirim")

end = time.time()
print(f"\nWaktu eksekusi:{(end-start)*10**3:.03f} ms")
```

- Run the program above and wait until the process is complete

```
python3 brute_otp_crapi.py
```



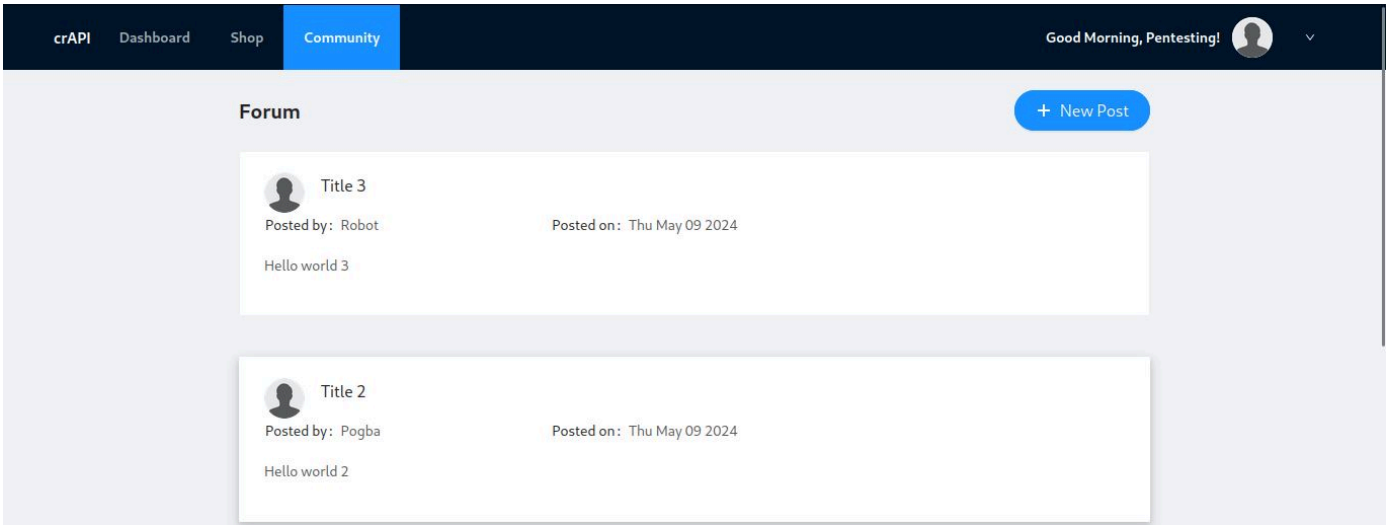
```
[Kode OTP:3163, Response:500]
[Kode OTP:3164, Response:500]
[Kode OTP:3165, Response:200]
Berhasil melakukan reset password
{'message': 'OTP verified', 'status': 200}

Waktu eksekusi:85734.041 ms
```

3. Excessive Data Exposure

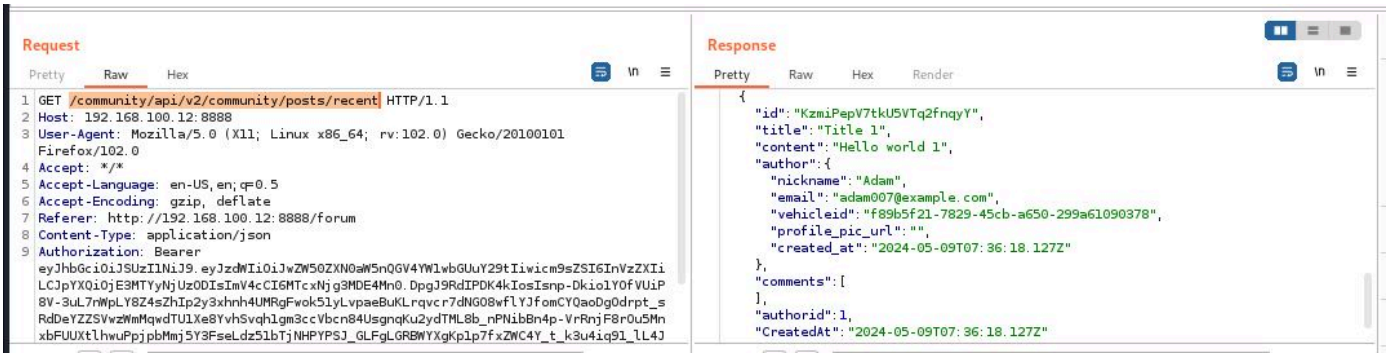
Sometimes API developers don't think about the sensitivity of the data being exposed. Automatic tools usually cannot detect this type of vulnerability because it is difficult to distinguish between legitimate data returned by API and sensitive data that should not be returned without a deep understanding of the application. Insecure endpoint API does not require authentication whatsoever to access data so that the attacker can access the data of other users or official data

Challenge 4 - Find an API endpoint that leaks sensitive information of other users

- Turn on the Burp Suite tool and go to **the Community** page

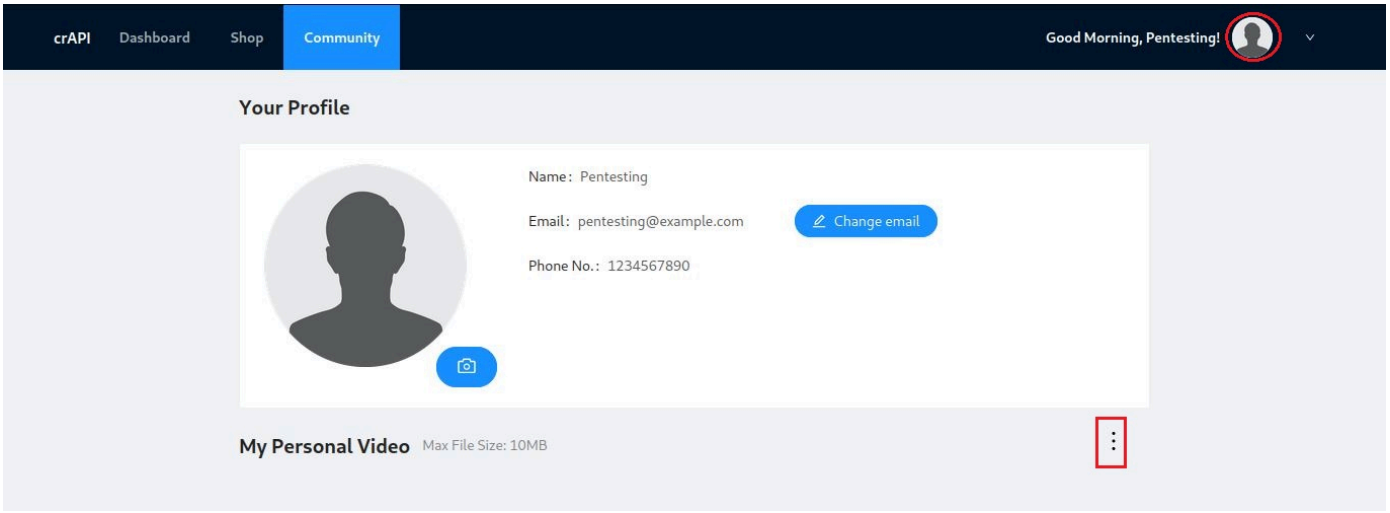


- In the Burp Suite tool it is detected that the page requests to the endpoint API `/community/api/v2/community/posts/recent` whose response contains sensitive data from each user who appears on **the Community** page

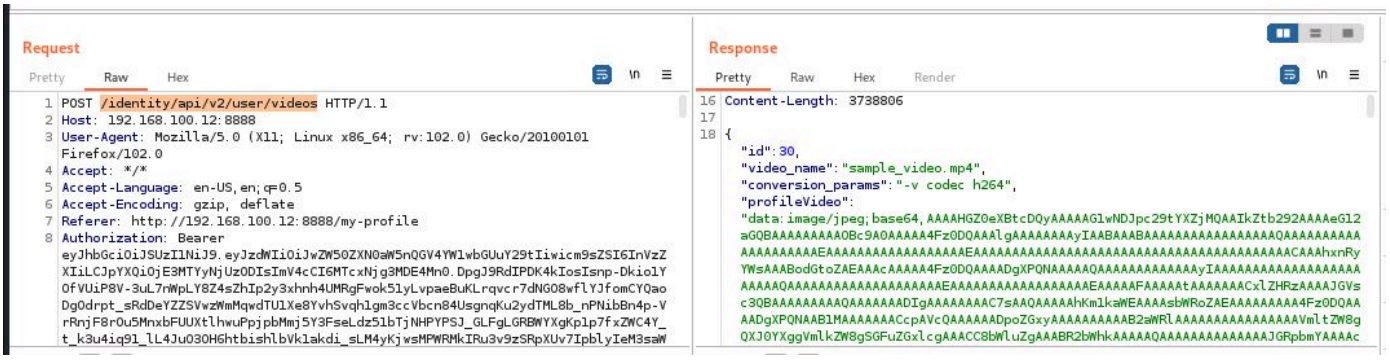


Challenge 5 - Find an API endpoint that leaks an internal property of a video

- Click the profile photo to go to the page `/my-profile` then click the three dots and select Upload Video to upload the video



- After successfully uploading the video to the profile page, on the Burp Suite tool detected the endpoint API `/identity/api/v2/user/videos` which endpoint responds in the form of id and video data that should not be displayed to the user

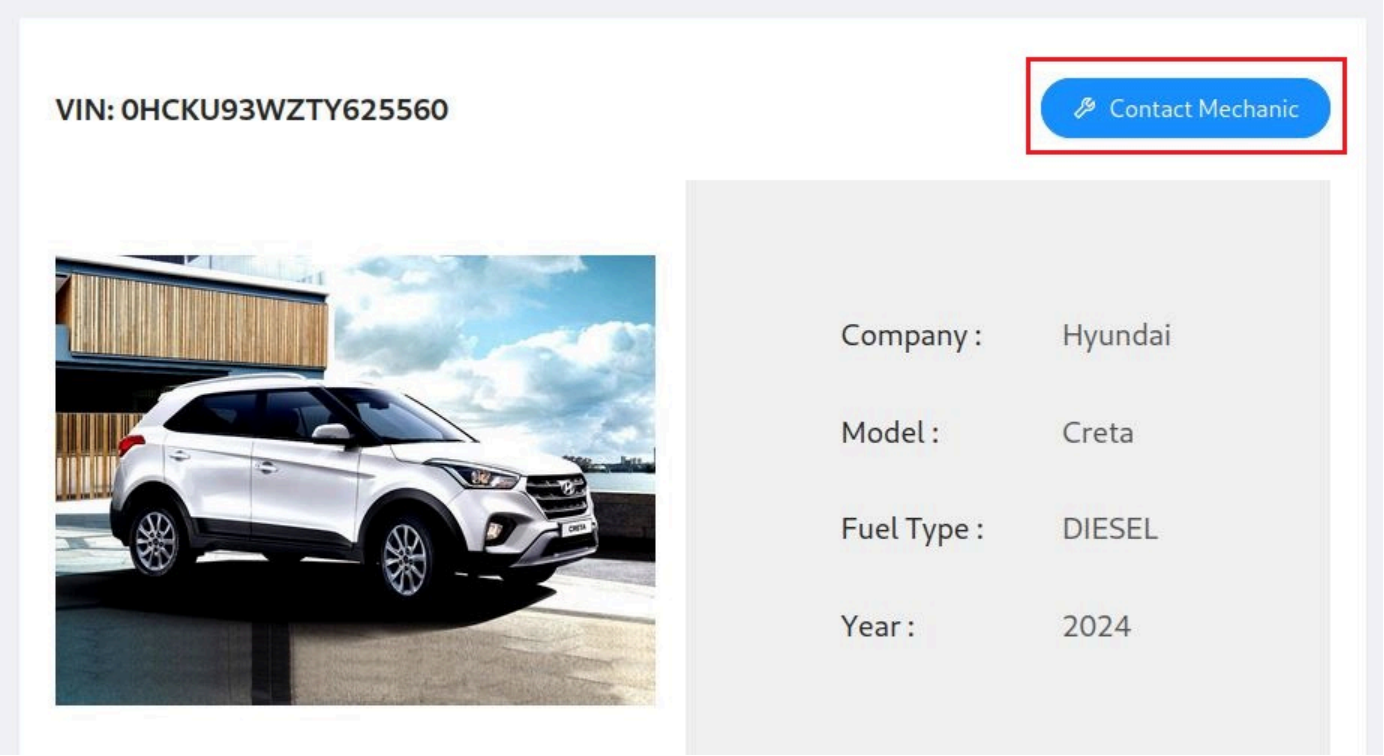


4. Rate Limiting

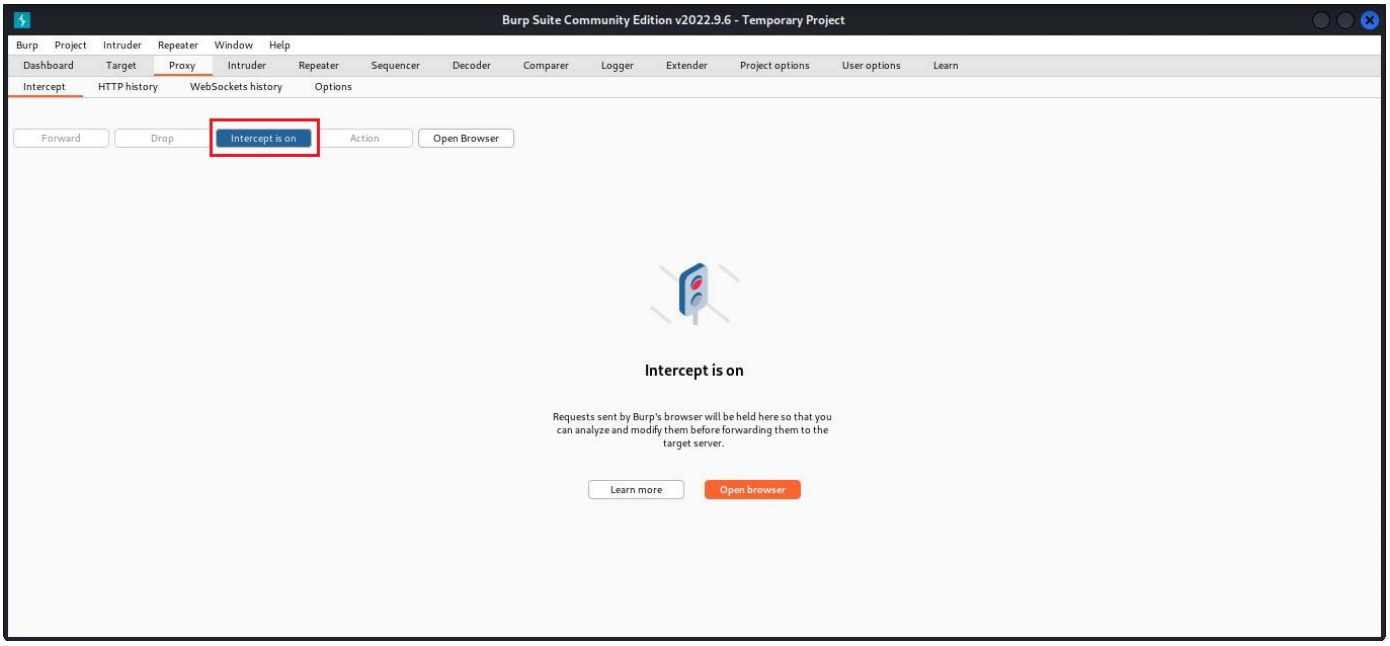
Multiple incurrent requests can be made from a single local computer or by using cloud computing resources. Some APIs do not apply access restrictions. Given that there are no restrictions on access, the brute force attack is a very appropriate choice. On the other hand, the exploits on this security gap can cause the DoS, making the API unresponsive or even unavated by sending spam to the API with hundreds or thousands of requests per second.

Challenge 6 - Perform a layer 7 DoS using 'contact mechanic' feature

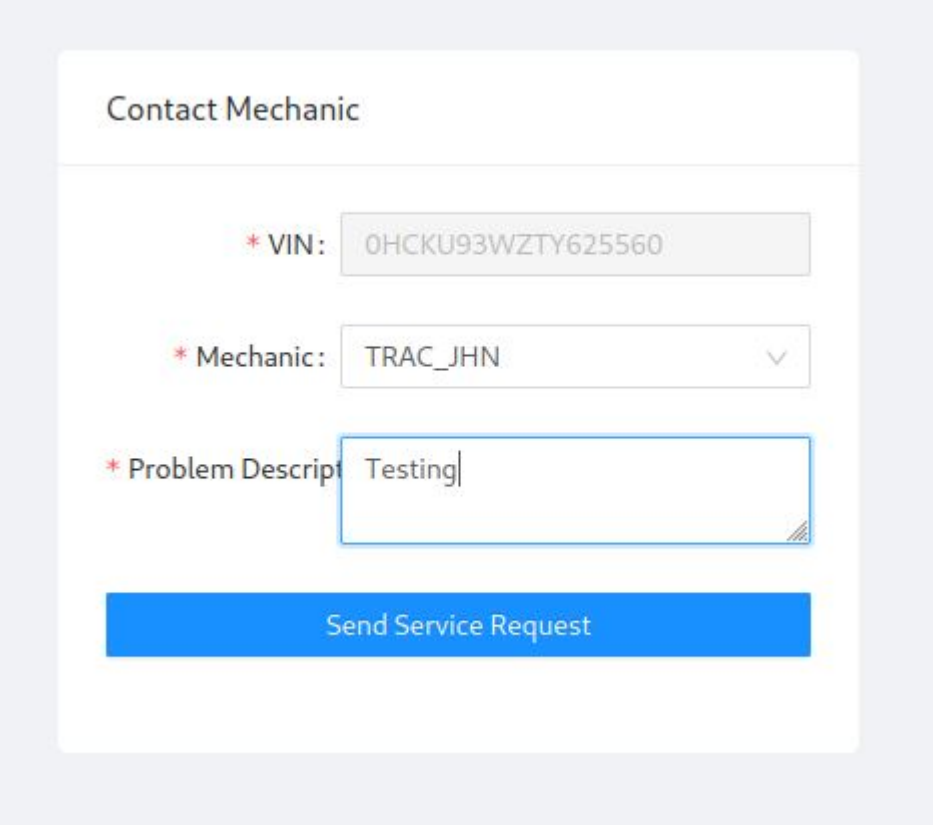
- Go to the Dashboard page and click **the Contact Mechanic** button



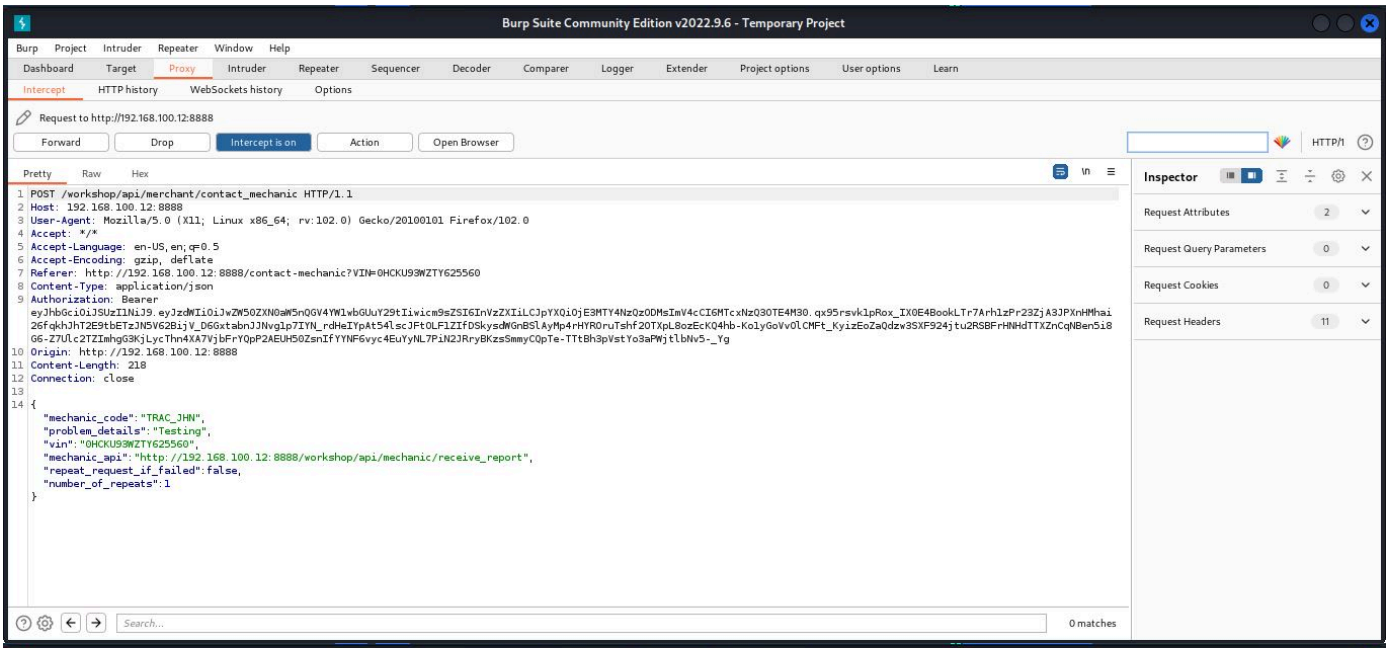
- Open the Burp Suite tool and turn on the intercept

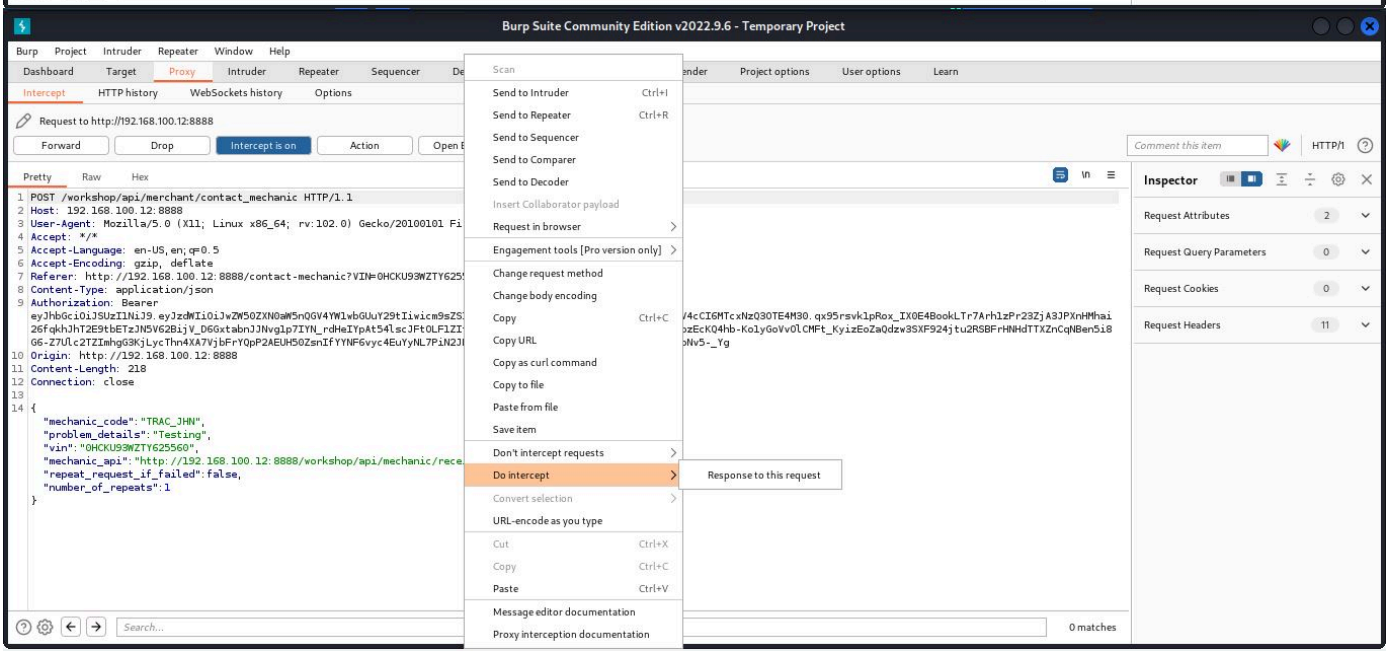


- Fill out the form and click **the Send Service Request** button

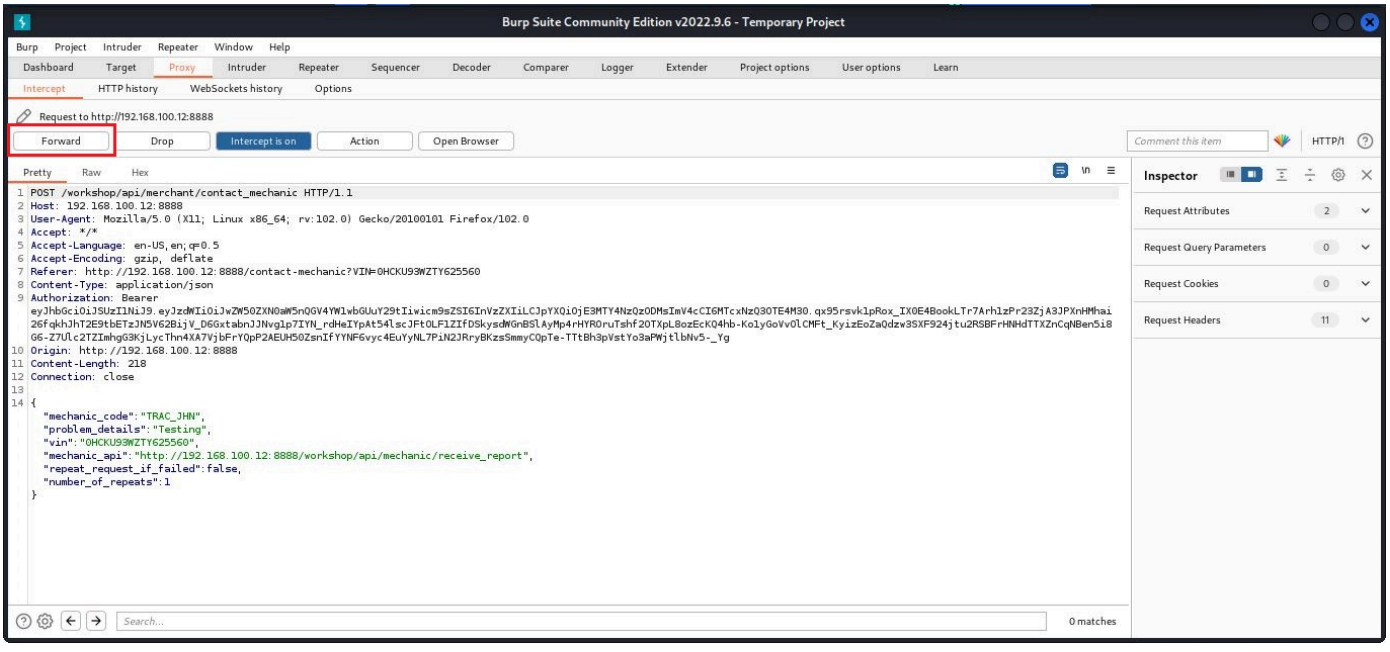


- Once the Burp Suite opens, right-click and select Do intercept > Response to this request

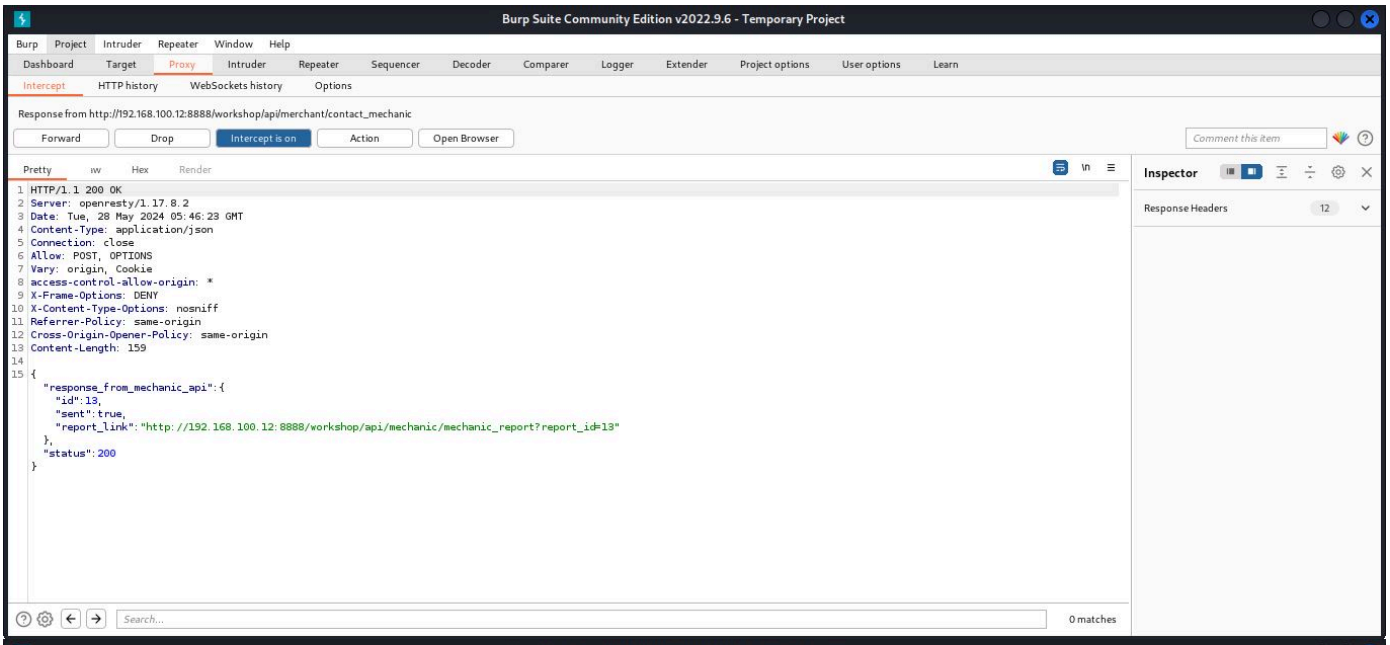


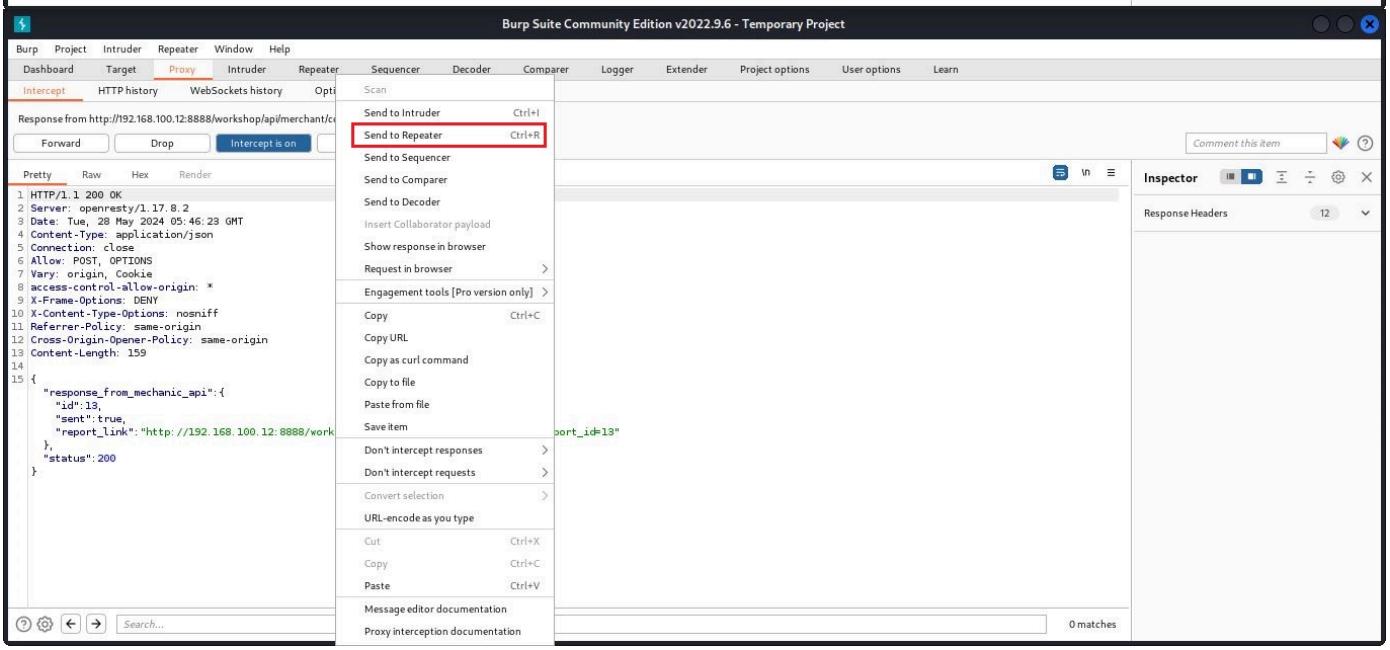


- Then click the Forward button

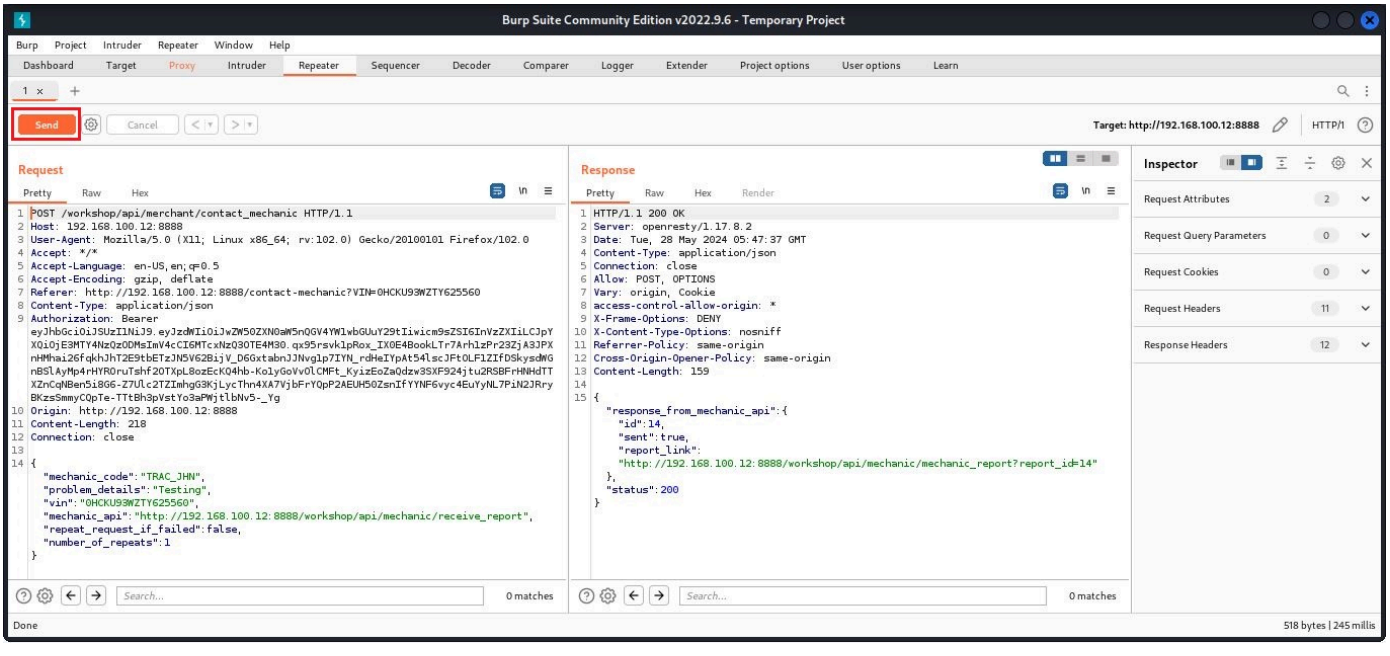


- If you find the following response, right-click and select Send to Repeater

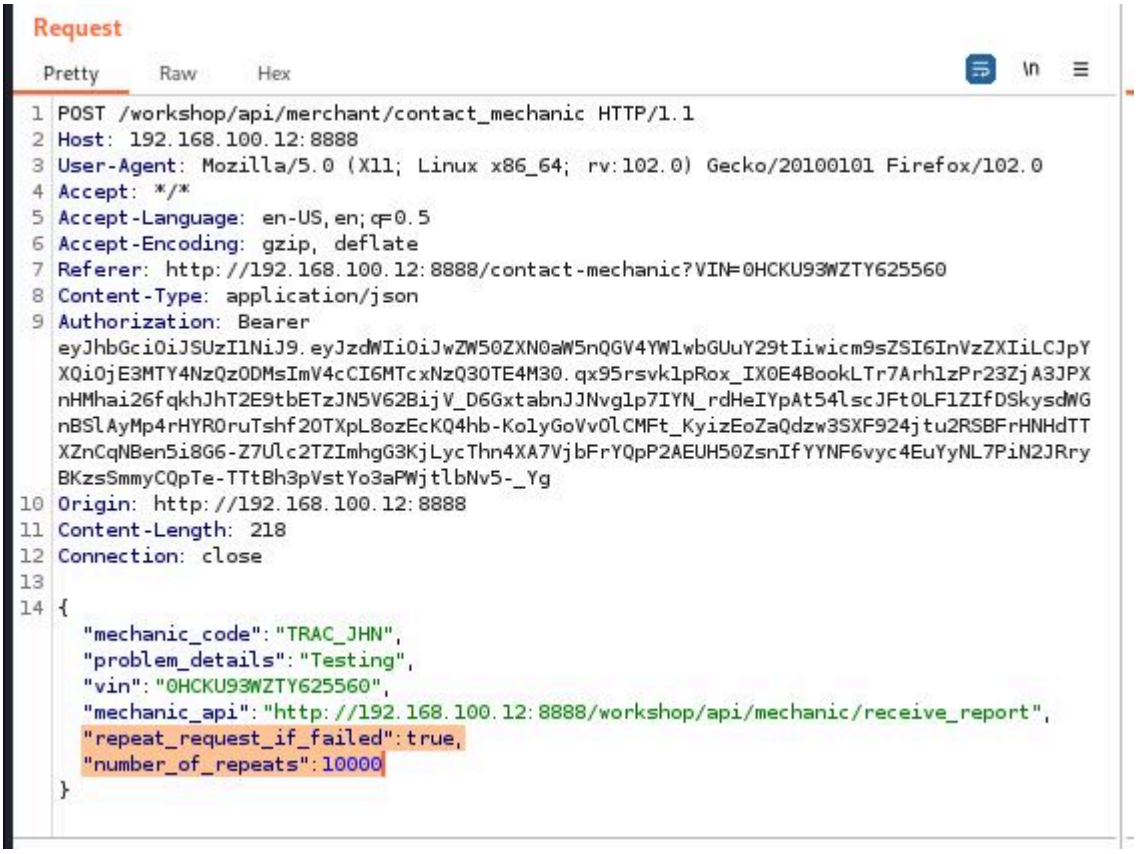




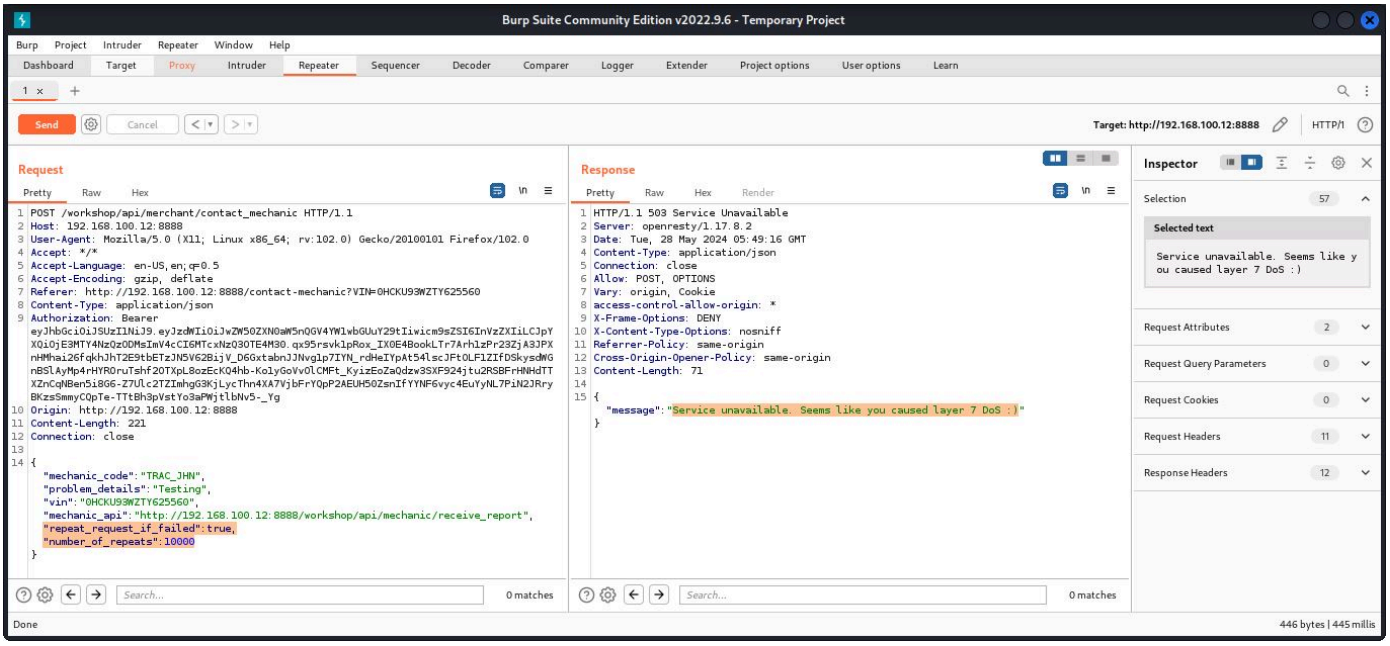
- Move to the Repeater tab and click **the Send** button



- In the request above there are parameters `repeat_request_if_failed` which is worth `false` which means that the request will not be repeated if the delivery fails and the parameters `number_of_repeats` which is worth `1` This means that it is repeated only 1 time. Now we try to change the value of the paramaeter `repeat_request_if_failed` Be a `true` which means that if the request fails to be repeated and the parameters `number_of_repeats` Be a `10000` which means the request will be repeated `10000` times



- If it has been changed click **the Send** button then the DoS activity will be detected

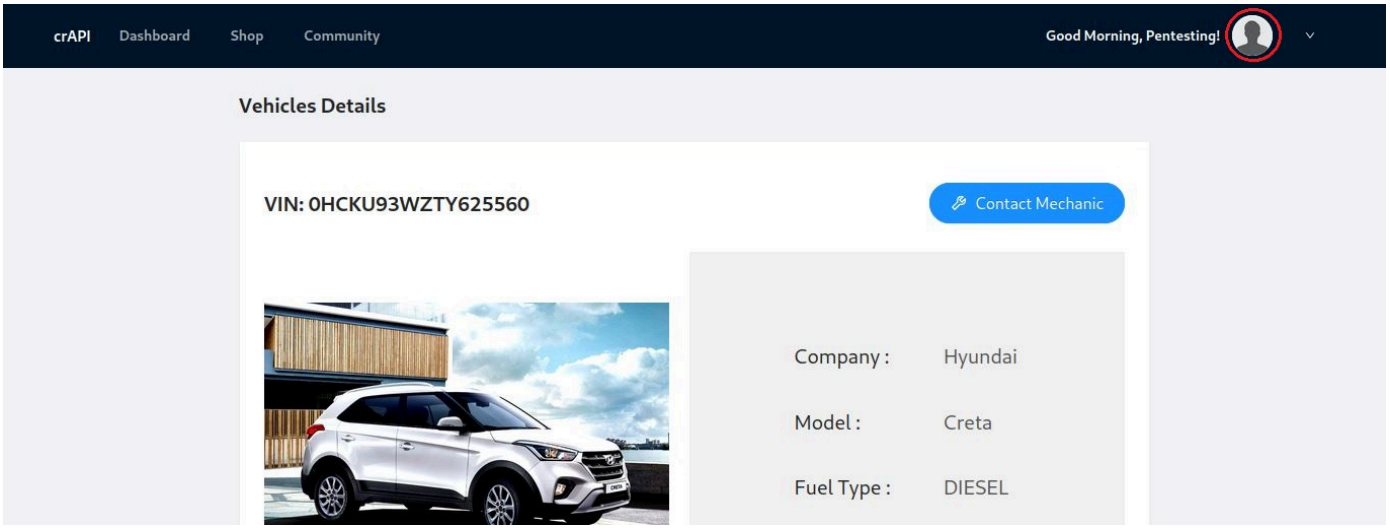


5. Broken Function Level Authorization

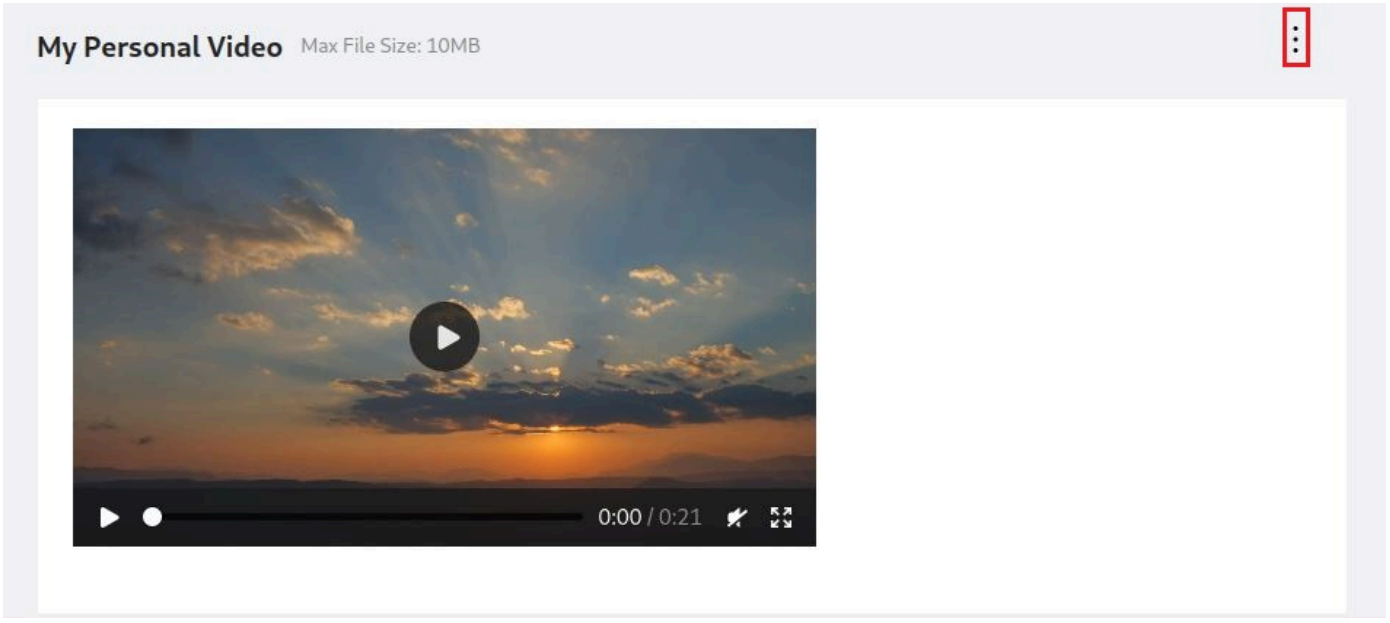
The attacker sends a request to the endpoint API that should not be accessed by ordinary or unauthorized users. Implementing appropriate check methods can be a confusing task because modern applications can contain many types of complex user roles, groups, and hierarchy (e.g. sub-users, or users with more than one role). Administrative functions are the primary targets for this type of attack and can lead to data disclosure, data loss, or data damage. In the end, this can lead to disruption of service.

Challenge 7 - Delete a video of another user

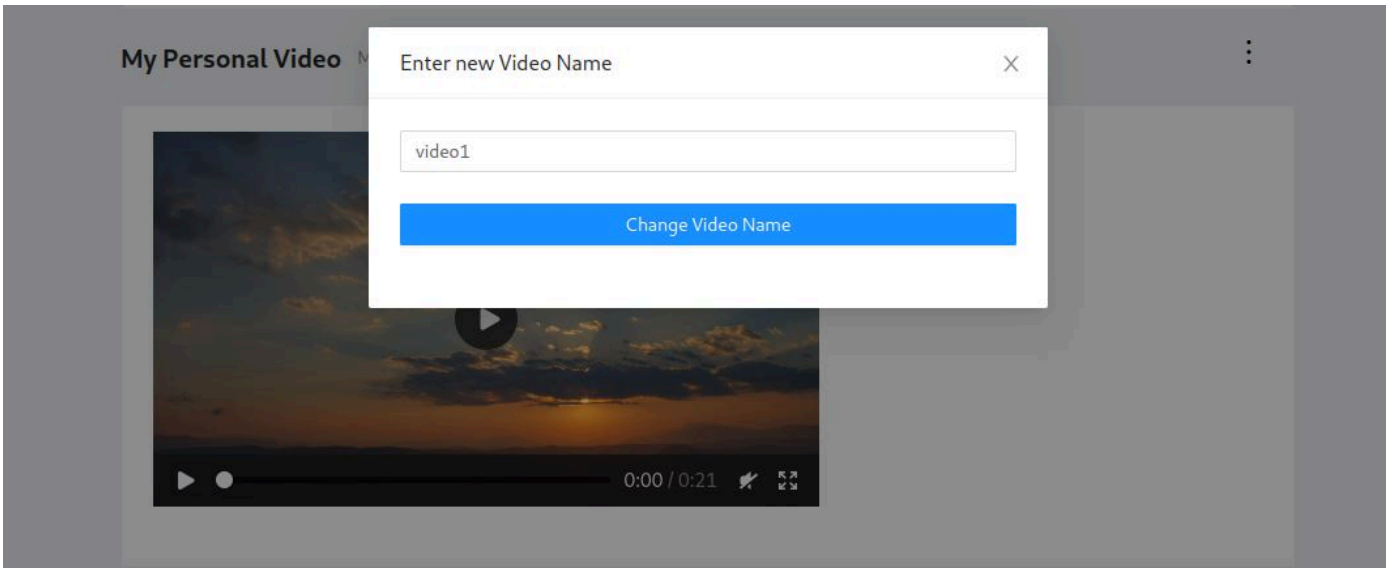
- Turn on the Burp Suite tool and click the profile photo to go to the profile page

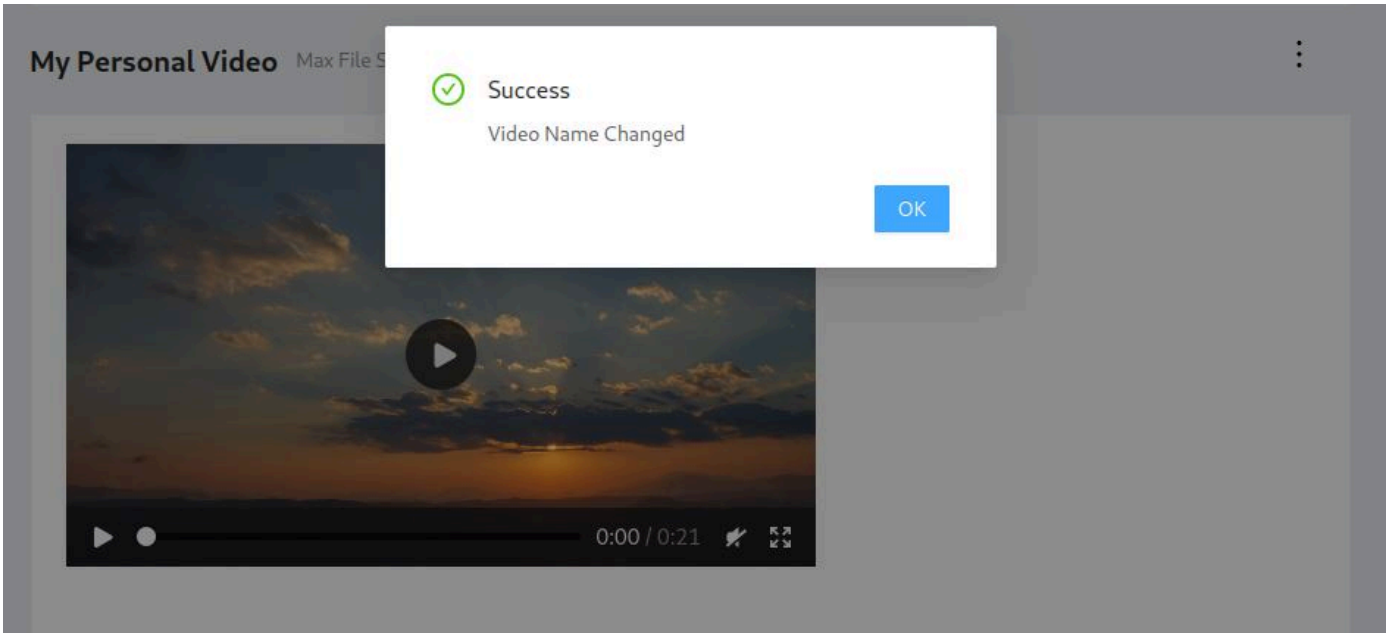


- Scroll down, in the My Personal Video section click the three dots and select **Change Video Name**

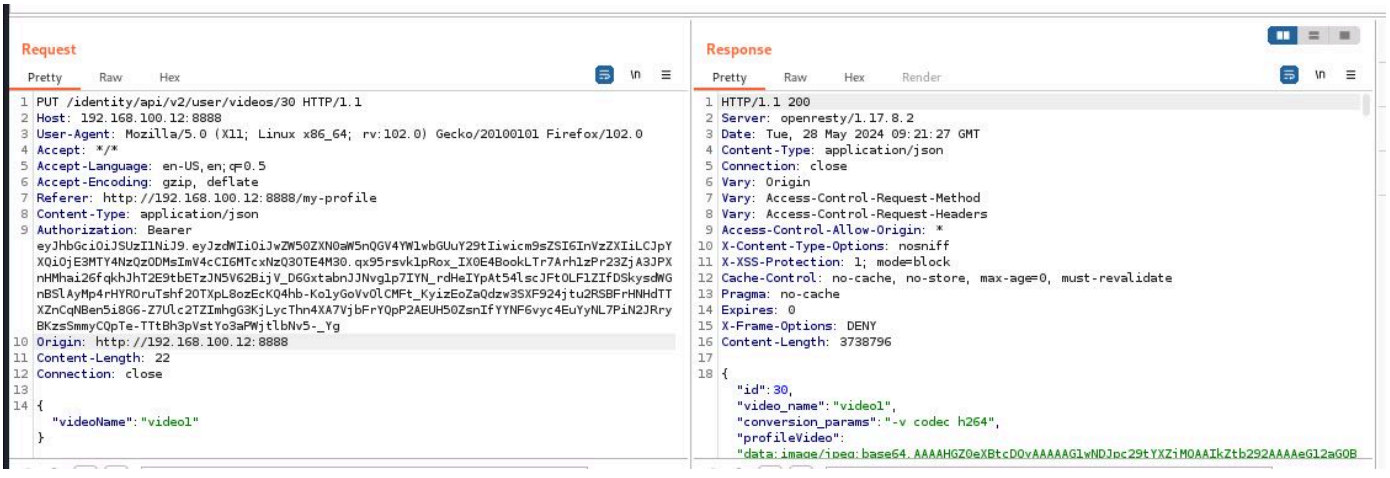


- Change the title of the video according to your wishes and then press **the Change Video Name** button

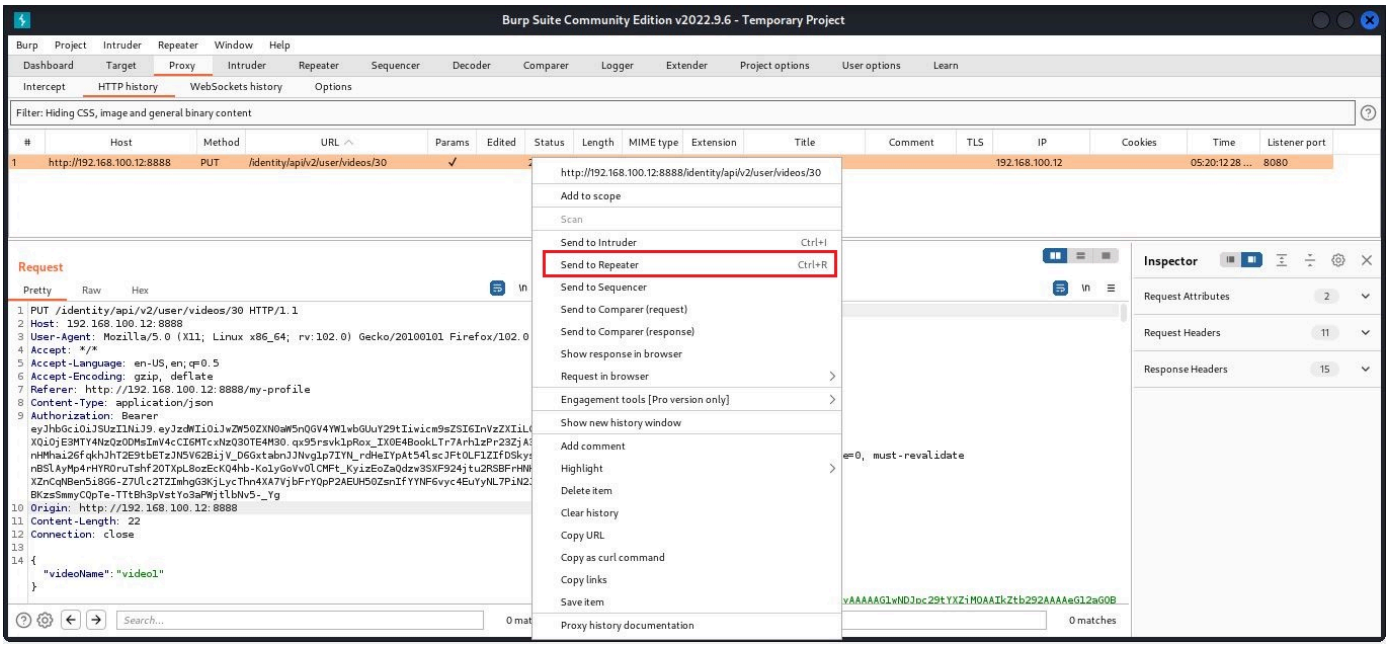




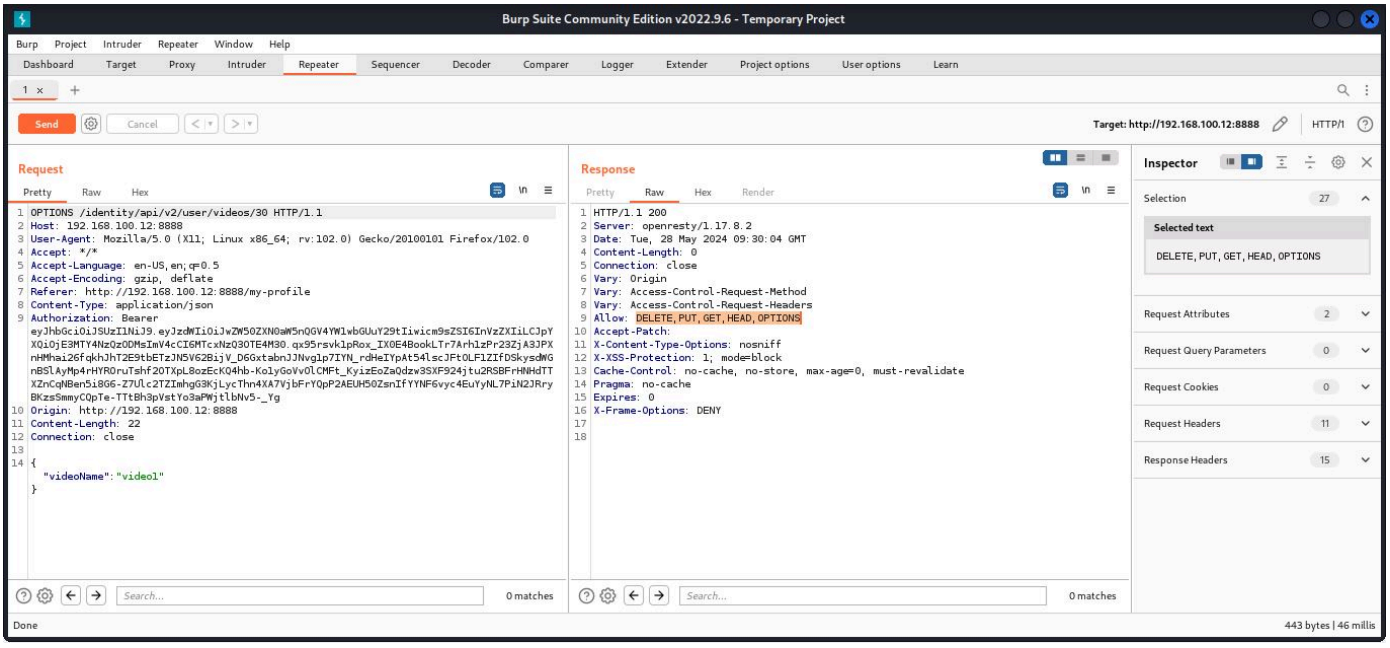
- On Burp Suite will be detected endpoint `/identity/api/v2/user/videos/id` used to edit videos



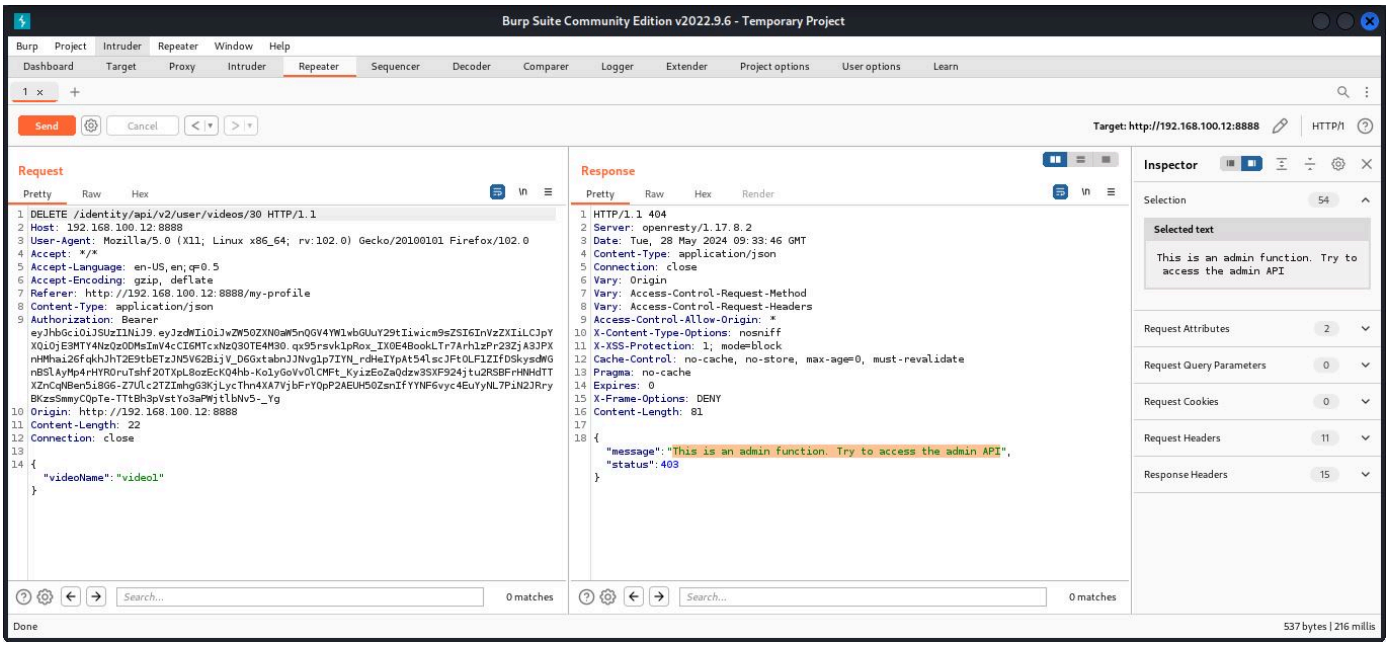
- Right-click the request and select Send to Repeater



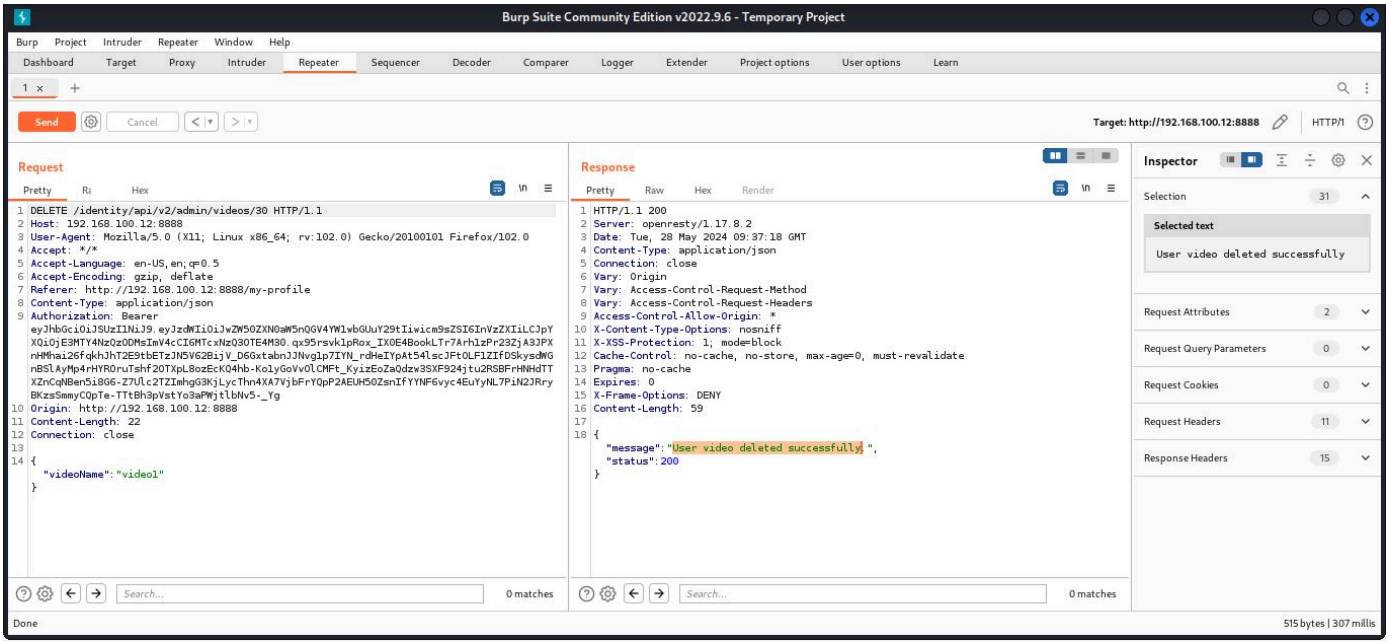
- Moving to the Repeater tab, here we can change the type of request `PUT` Be a `OPTINS` to find out what types of requests can be done in the endpoint. After that click **the Send** button



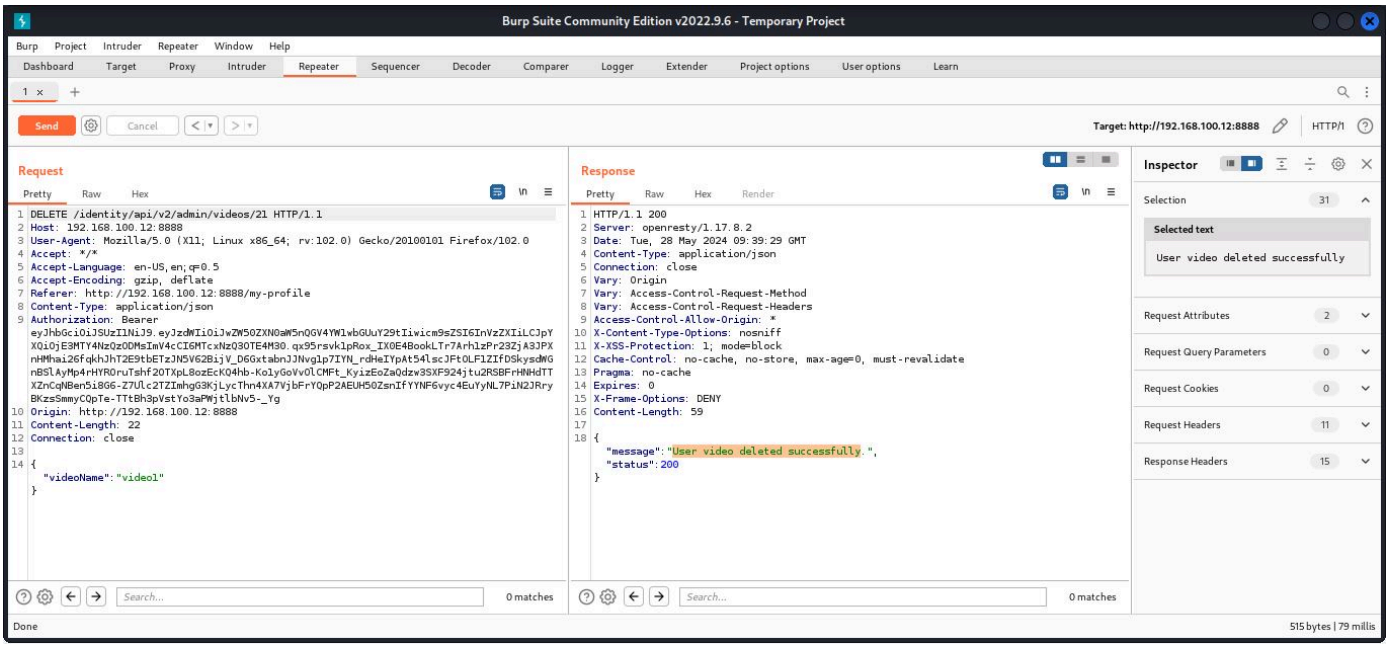
- The endpoint actually allowed the operation `DELETE`, so now we change the request type `OPTINS` Be a `DELETE` and click **the Send** button



- A message appears that we have to access the endpoint user admin to be able to perform the operation `DELETE`. So we need to change the endpoint API from `/identity/api/v2/user/videos/` Be a `/identity/api/v2/admin/videos/` then click **the Send** button then the video in our profile is successfully deleted



- Because the id is an integer, we can remove the personal video from other users just by sorting his id to the back or forward. Here we managed to delete the video with the id `20`

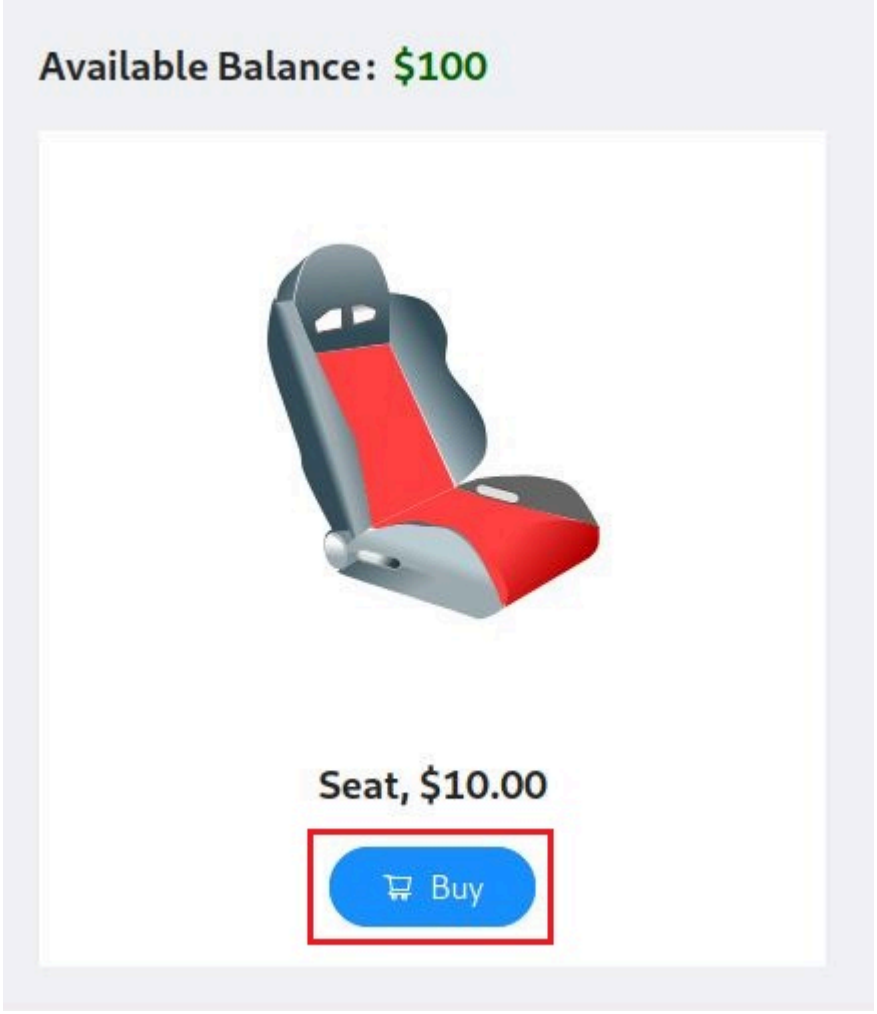


6. Mass Assignment

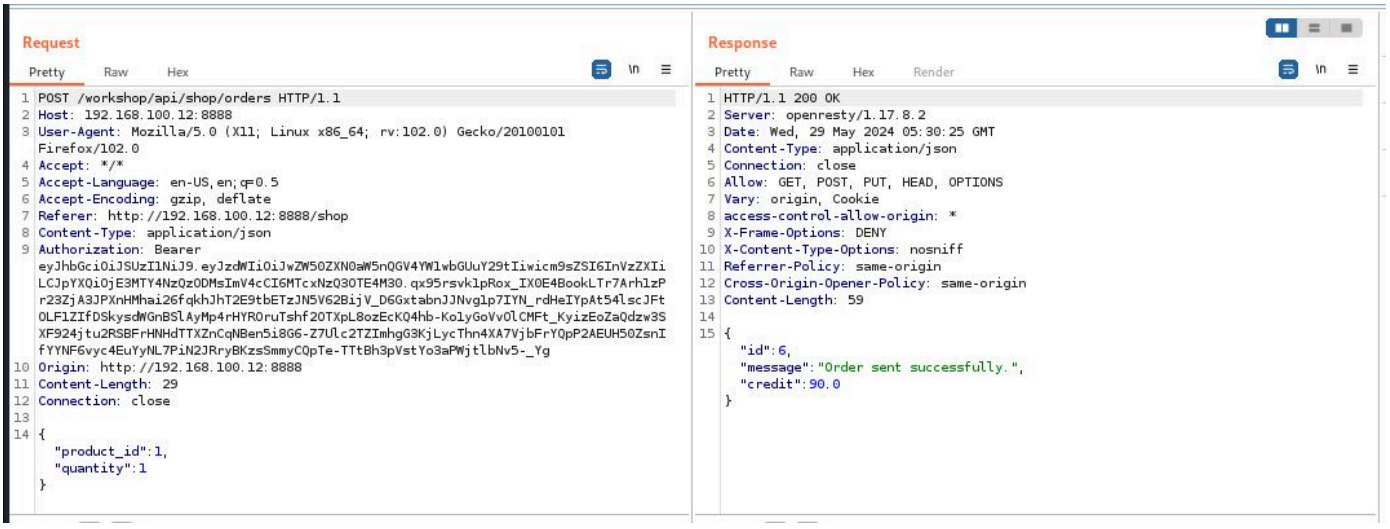
Frameworks sometimes allow developers to tie HTTP request parameters into variables or objects automatically to make the use of the framework easy to use. This can sometimes cause losses. Attackes can sometimes use this methodology to create new parameters that are never intended by developers that in turn create or chisfet new variables in unimported program code.

Challenge 8 - Get an item for free

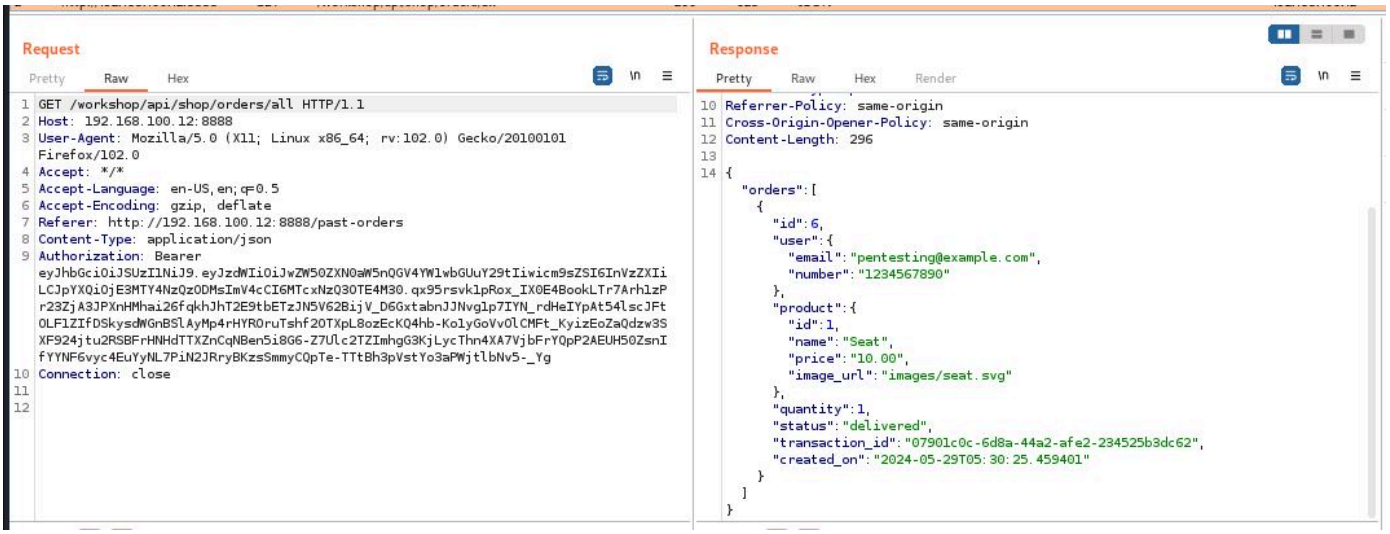
- Turn on the Burp Suite tool and go to **the Shop** page and buy one of the items



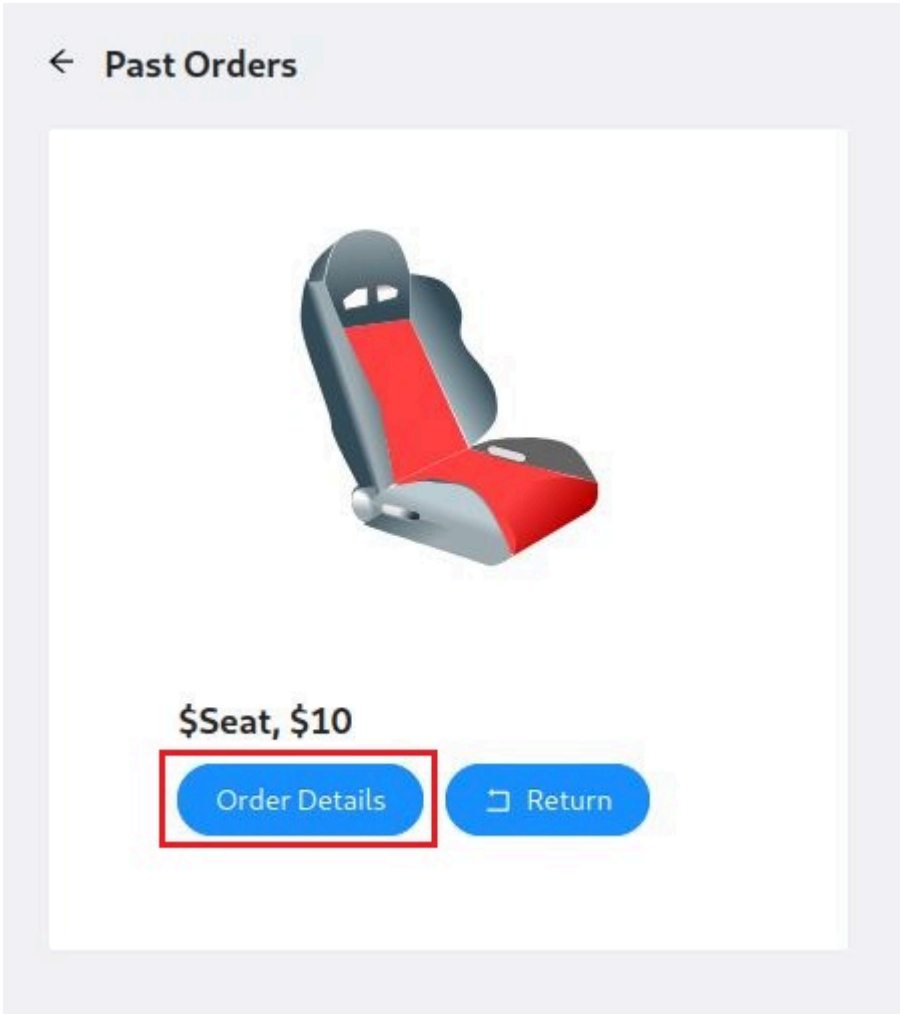
- On the Burp Suite tool detected endpoint `/workshop/api/shop/orders` with type request `POST` used to place an order



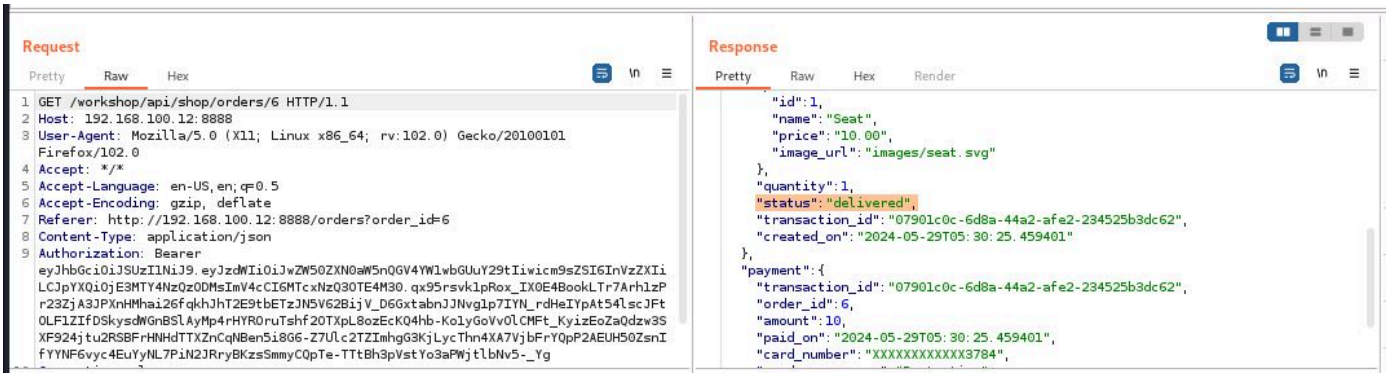
- In addition, the Burp Suite tool is detected endpoint `/workshop/api/shop/orders/all` with type request `GET` which displays the details of the order data



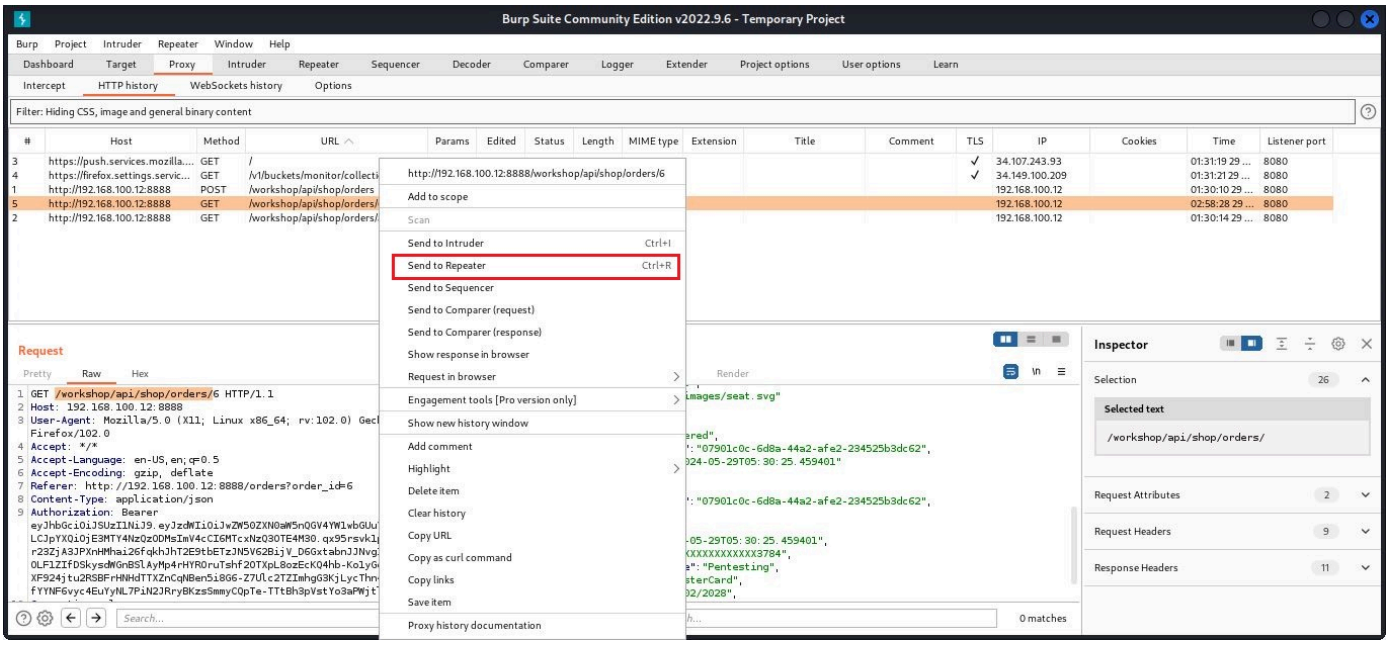
- Now we press **the Order Details** button



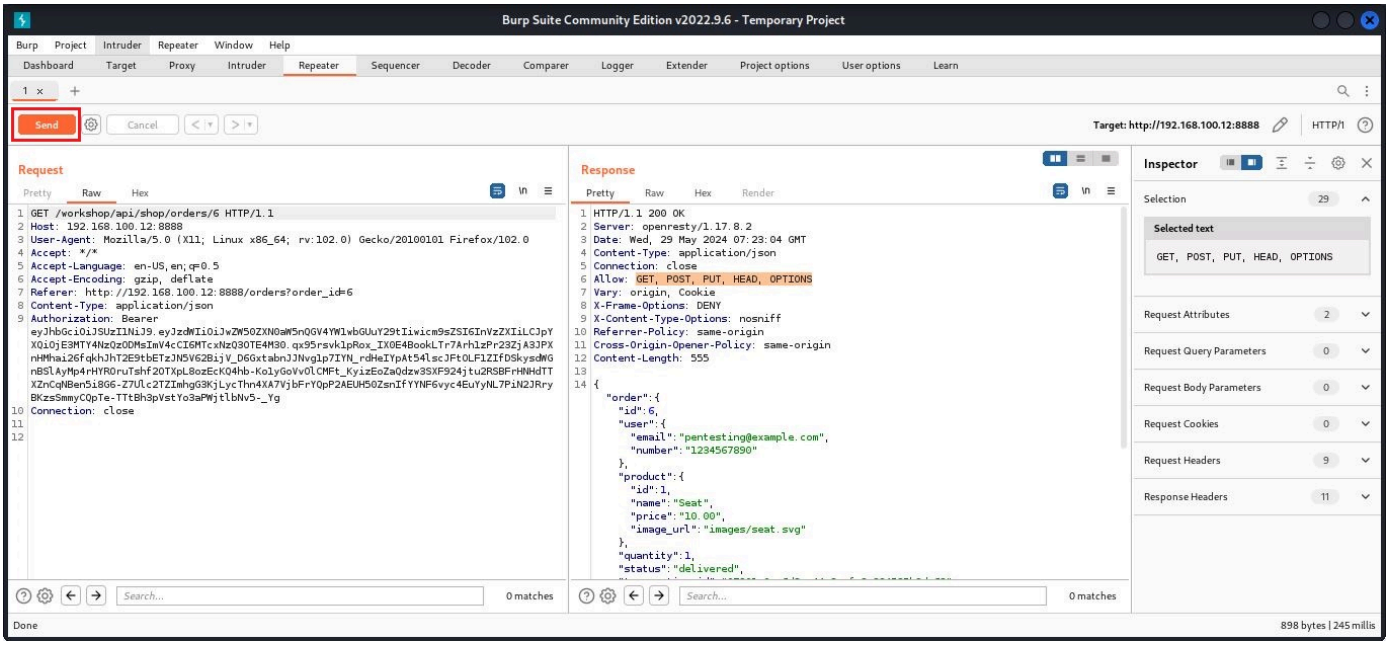
- On the Burp Suite tool detected endpoint `/workshop/api/shop/orders/` with type request `GET` which contains booking and payment details



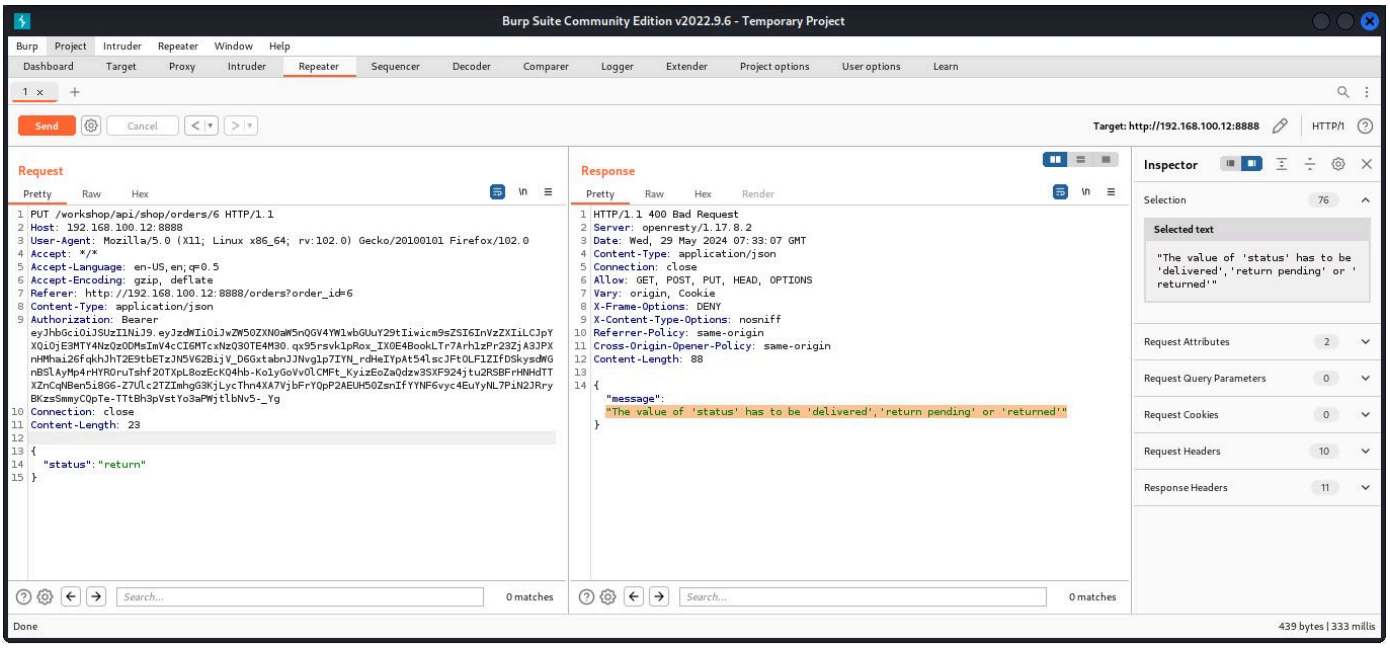
- Right-click the request and select **Send to Repeater**



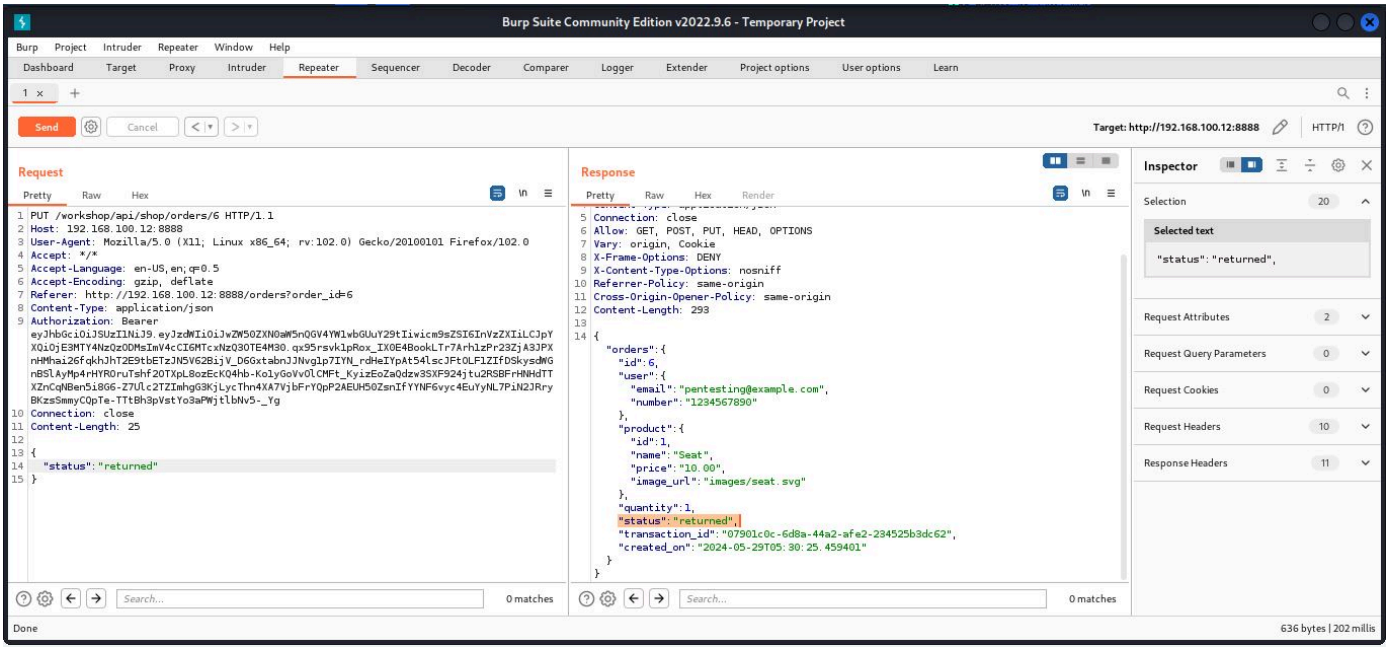
- Move to the Repeater tab and click the **Send** button, in response there is information that this endpoint supports the type of request `GET, POST, PUT, HEAD, OPTIONS`



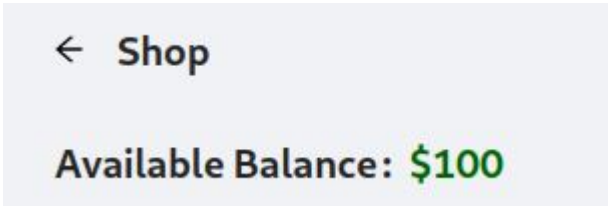
- Here the order status is `delivered` So we'll change the status to `return` Let our balance return. Change the request type from `GET` Be a `PUT` and add parameters `status` with value `return` then press **the Send** button



- From the results above the status `return` not available, the status available only `delivered`, `return pending` and `returned`. So we just change the value to `returned`

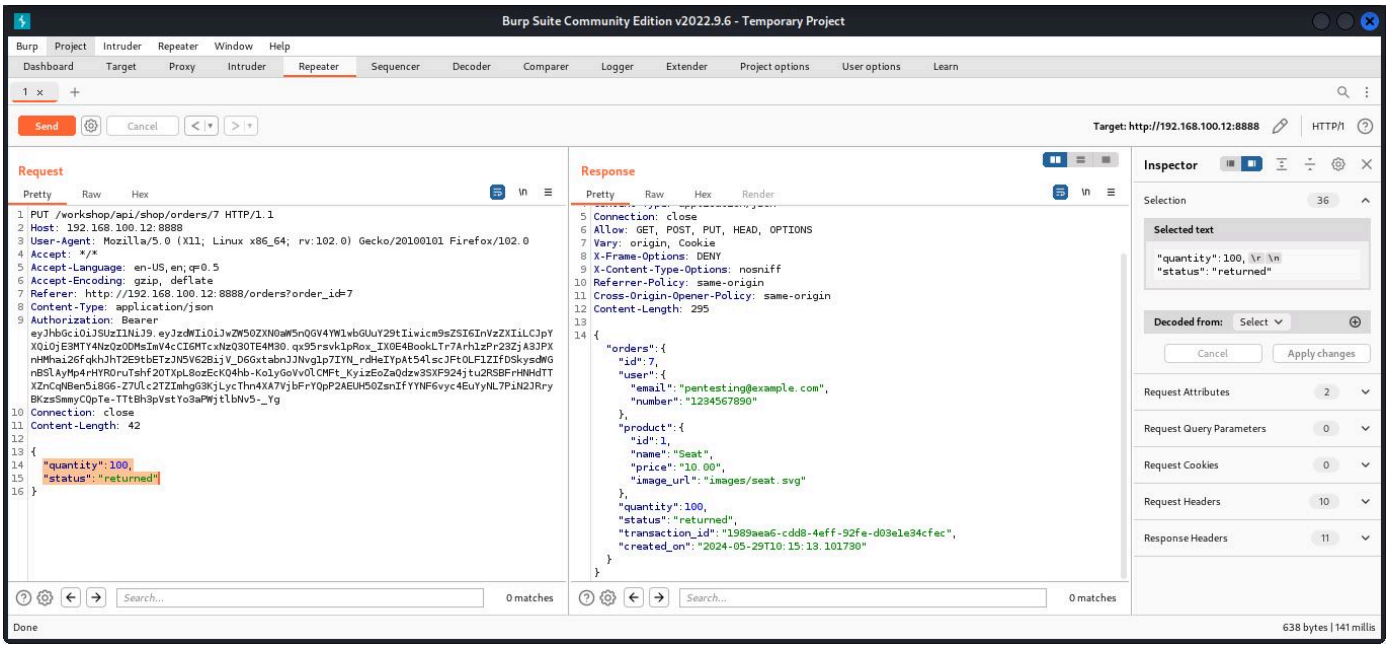


- Now we go back to **the Shop** page and our balance remains as it was

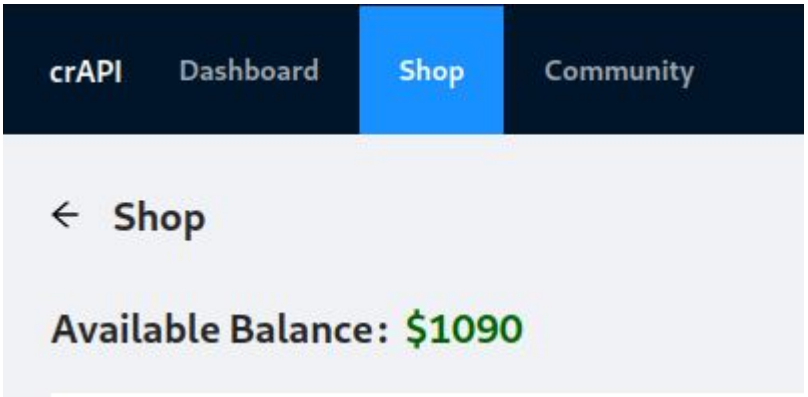


Challenge 9 - Increase your balance by $1,000 or more

- Make another purchase and repeat the above steps until sending the endpoint `/workshop/api/shop/orders/` to the Repeater tab. Now we can simply use the request type `PUT` to change the order data, add parameters `status` with value `returned` so that our balance is refunded as much as the total order and add parameters `quantity` with value `100` so that our balance will be refunded as much as `100` times over then press **the Send** button

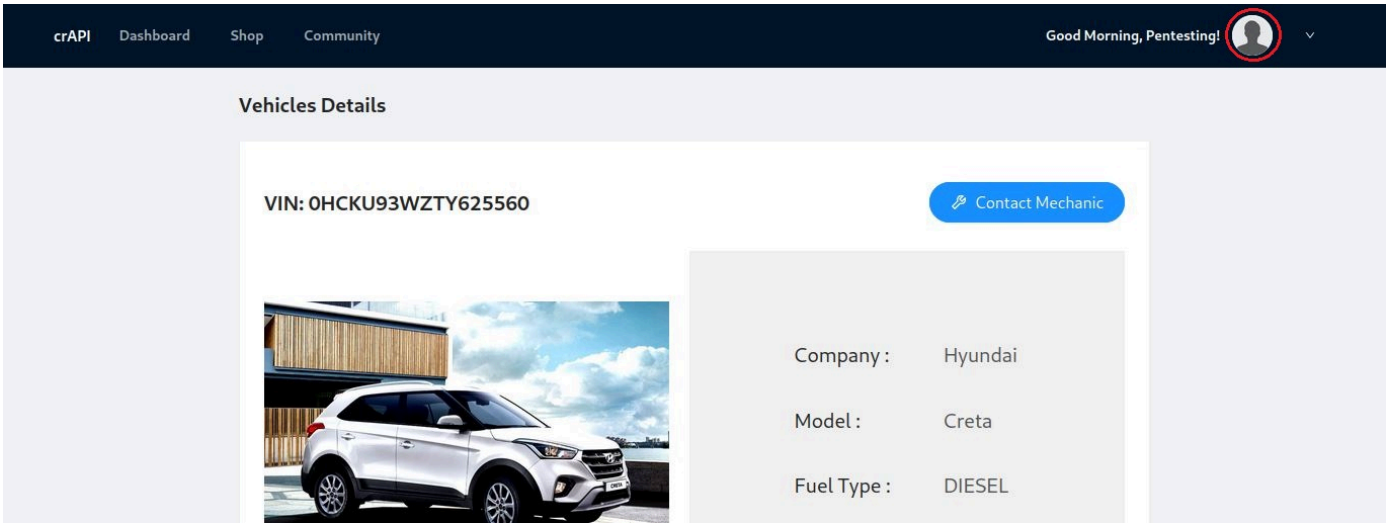


- Now we go back to **the Shop** page then our balance adds to the `90` Be a `1090`

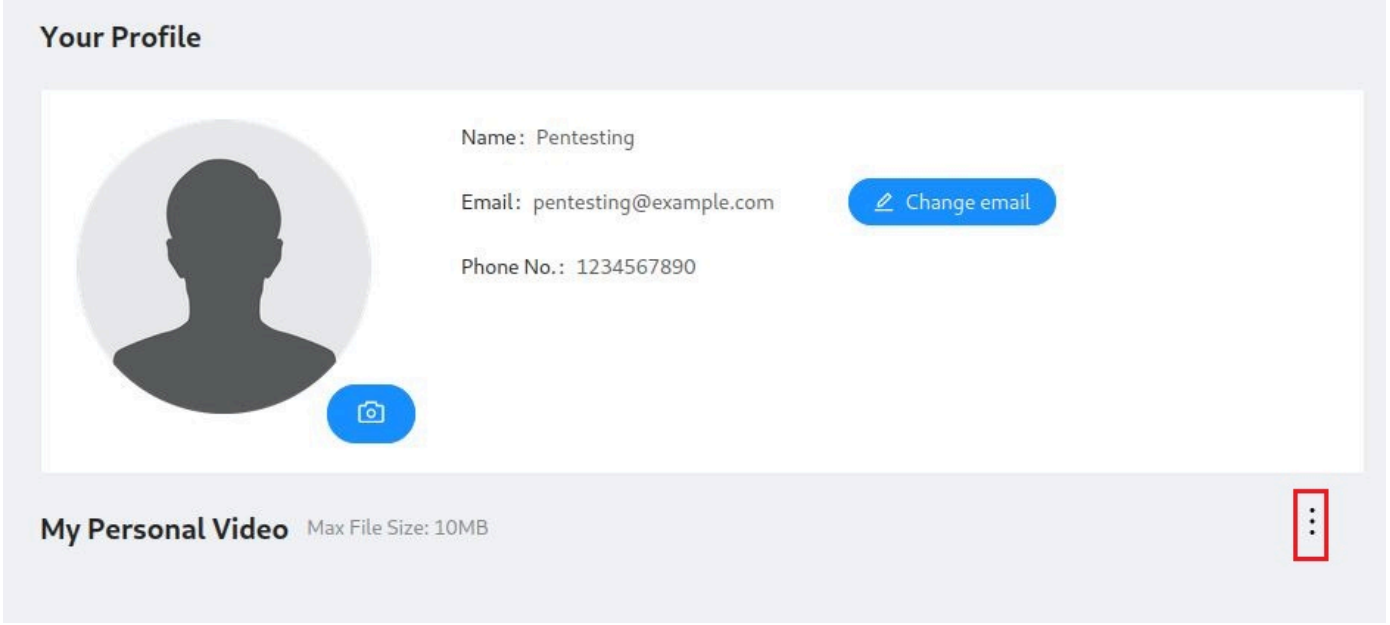


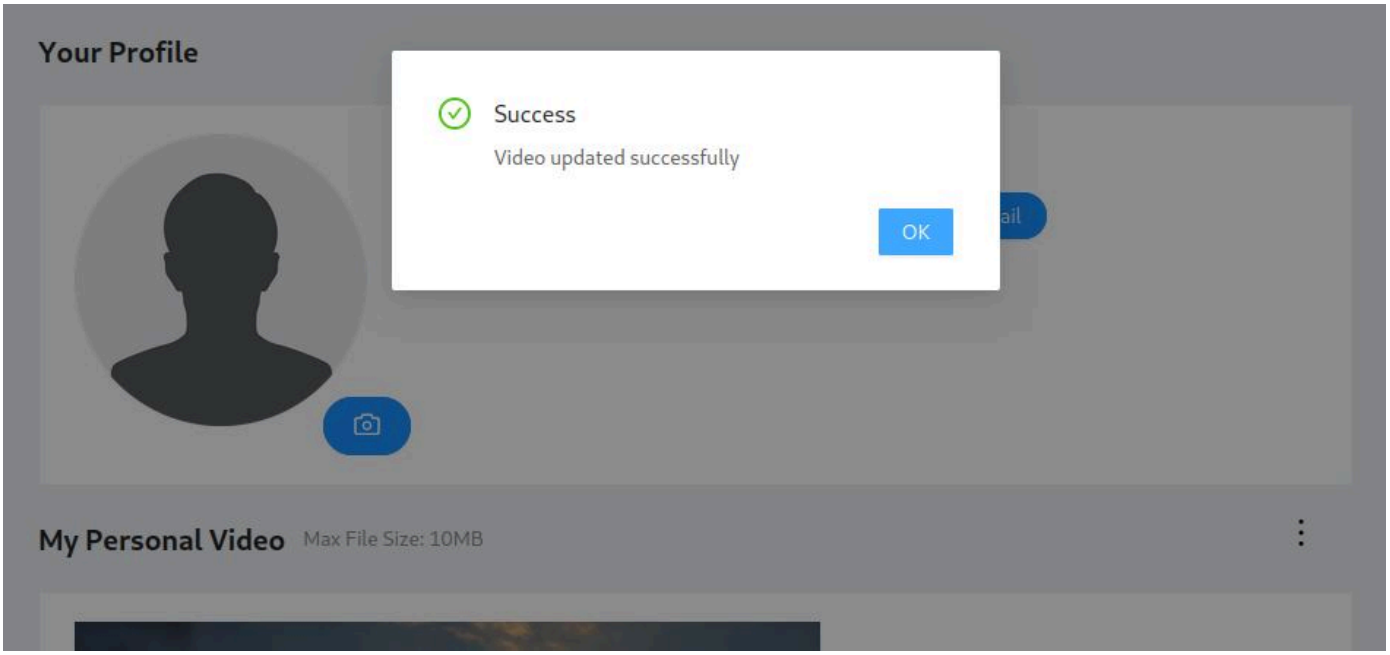
Challenge 10 - Internal video properties update

- Turn on the Burp Suite tool and click the profile photo to go to the profile page

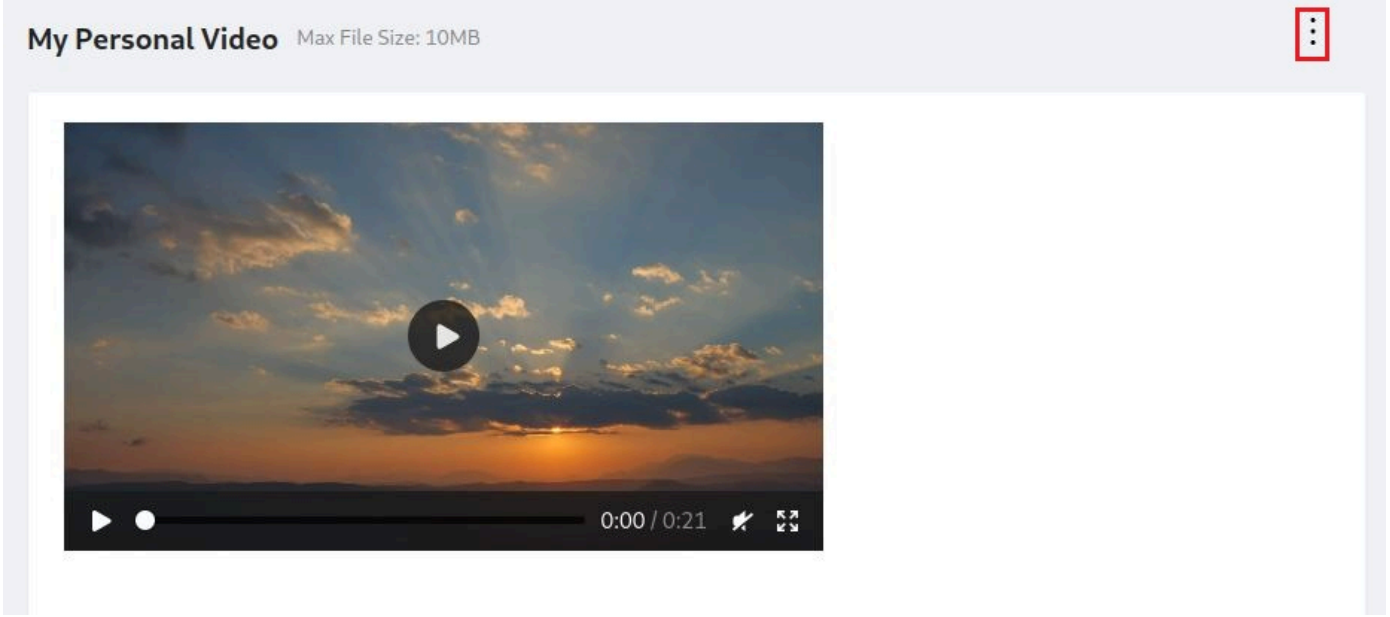


- If your My Personal Video is deleted in Challenge 7 (BFLA). You can upload again by clicking the point of the three and then select Upload Video

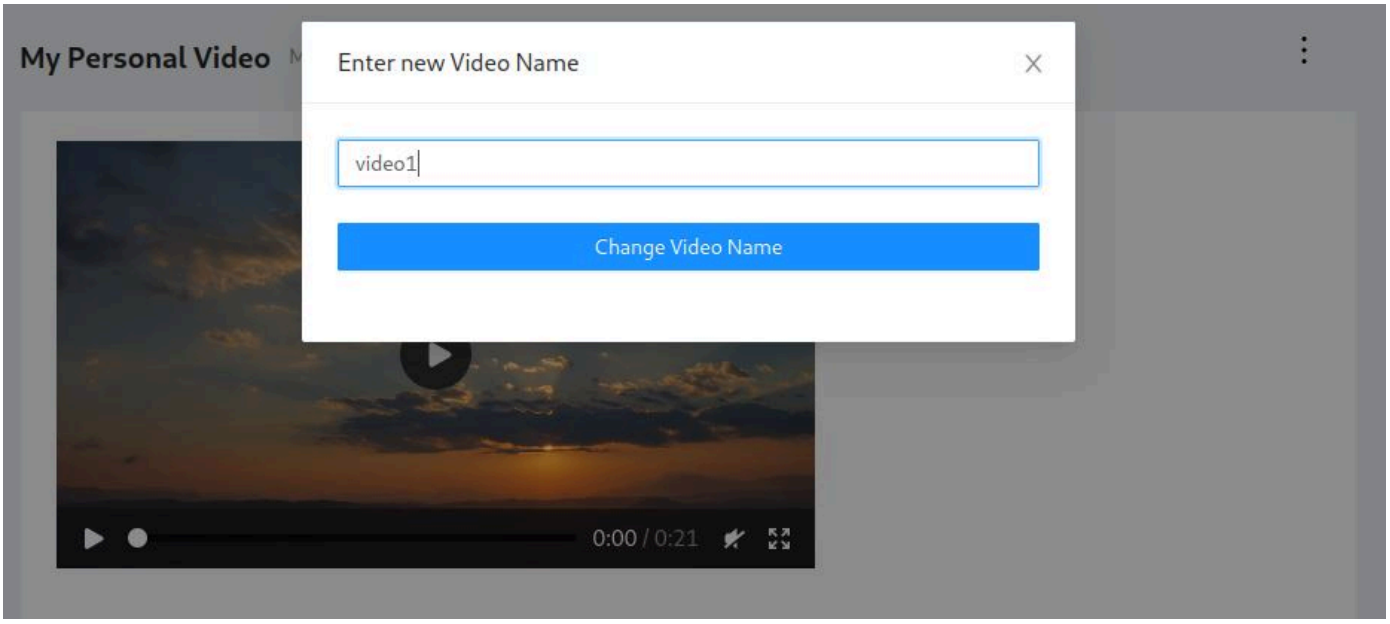


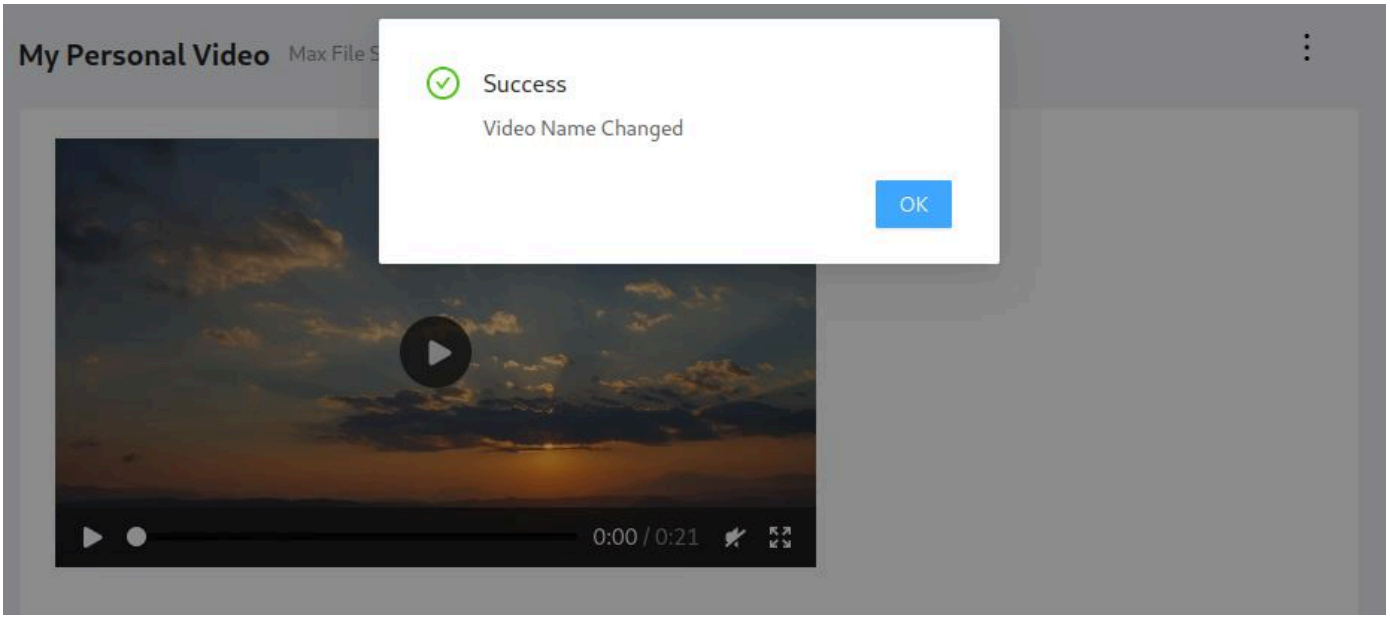


- Click the triple dot sign again and select **Change Video Name**

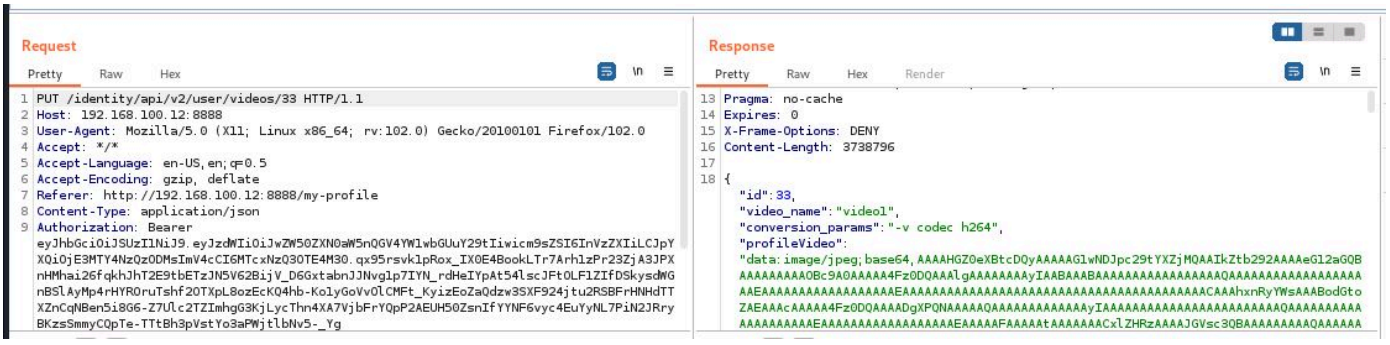


- Change the title of the video according to what you want

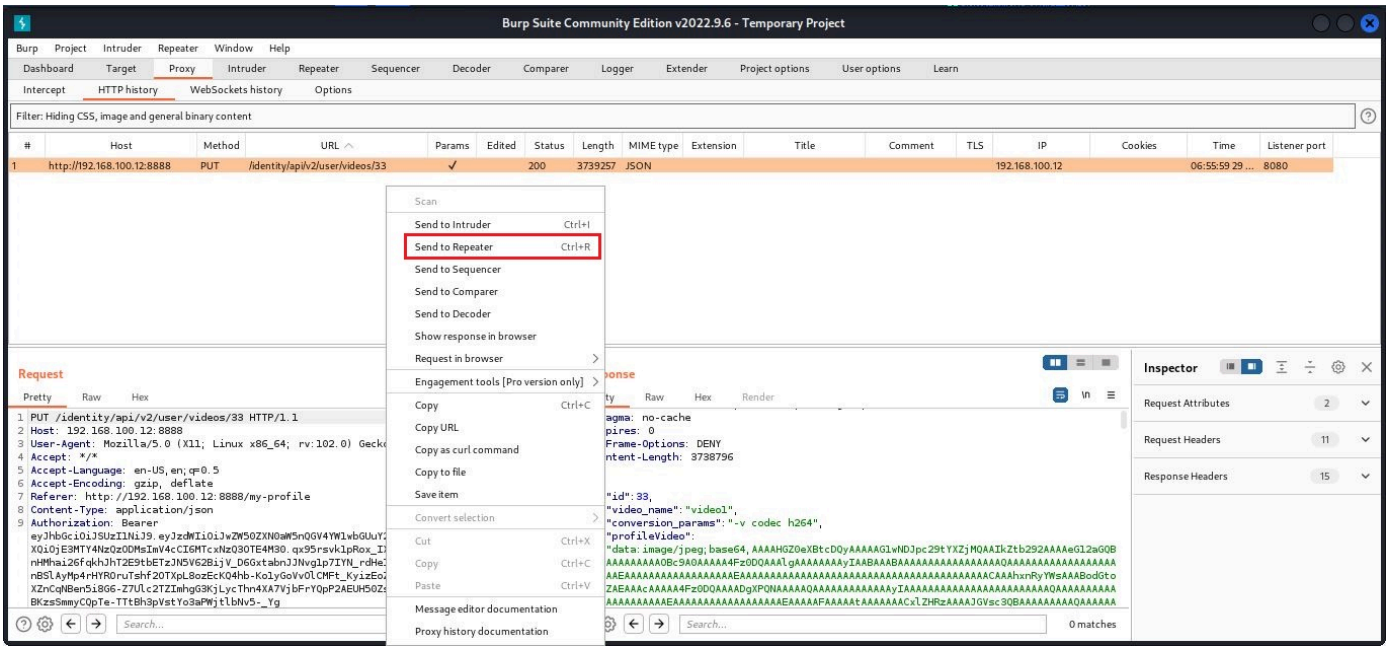




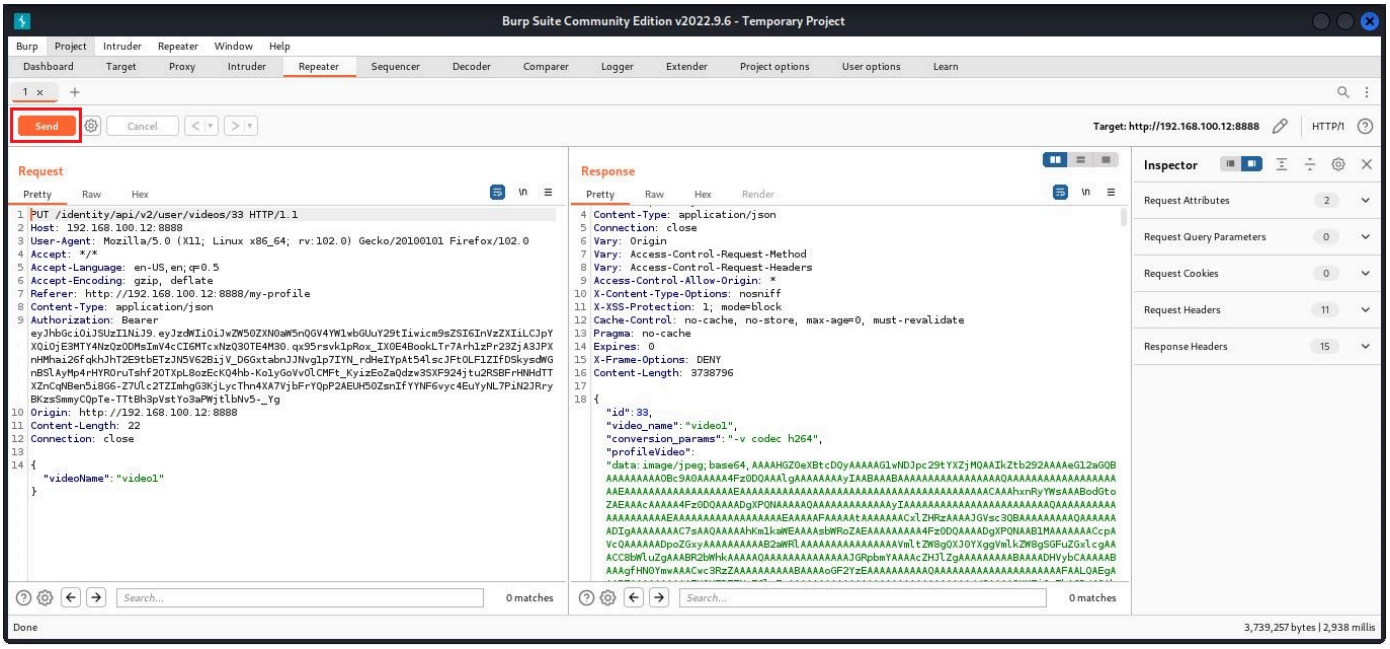
- On Burp Suite will be detected endpoint `/identity/api/v2/user/videos/id` used to edit videos



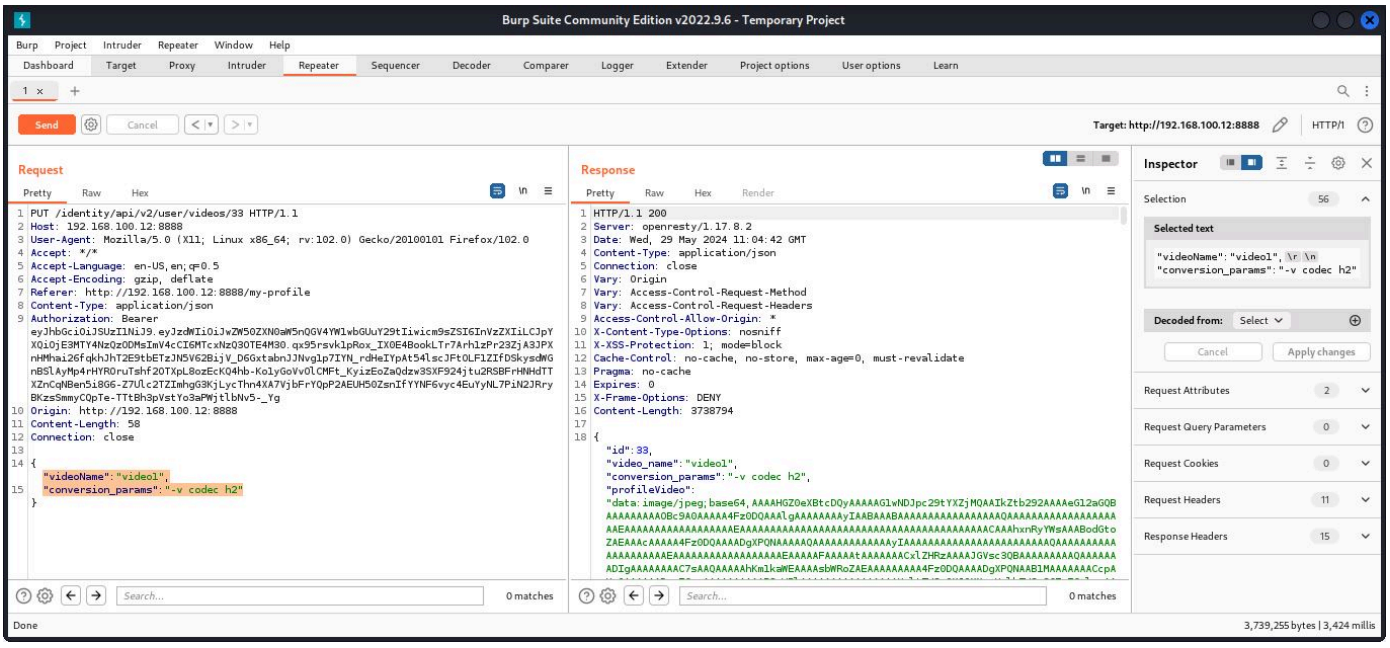
- Right-click the request and select Send to Repeater



- Move to the Repeater tab and click **the Send** button



- Here we can add parameters `conversion_params` with value `-v codec h2` Then click **Send**the Send button. Once the request is sent we managed to change the value `conversion_params`

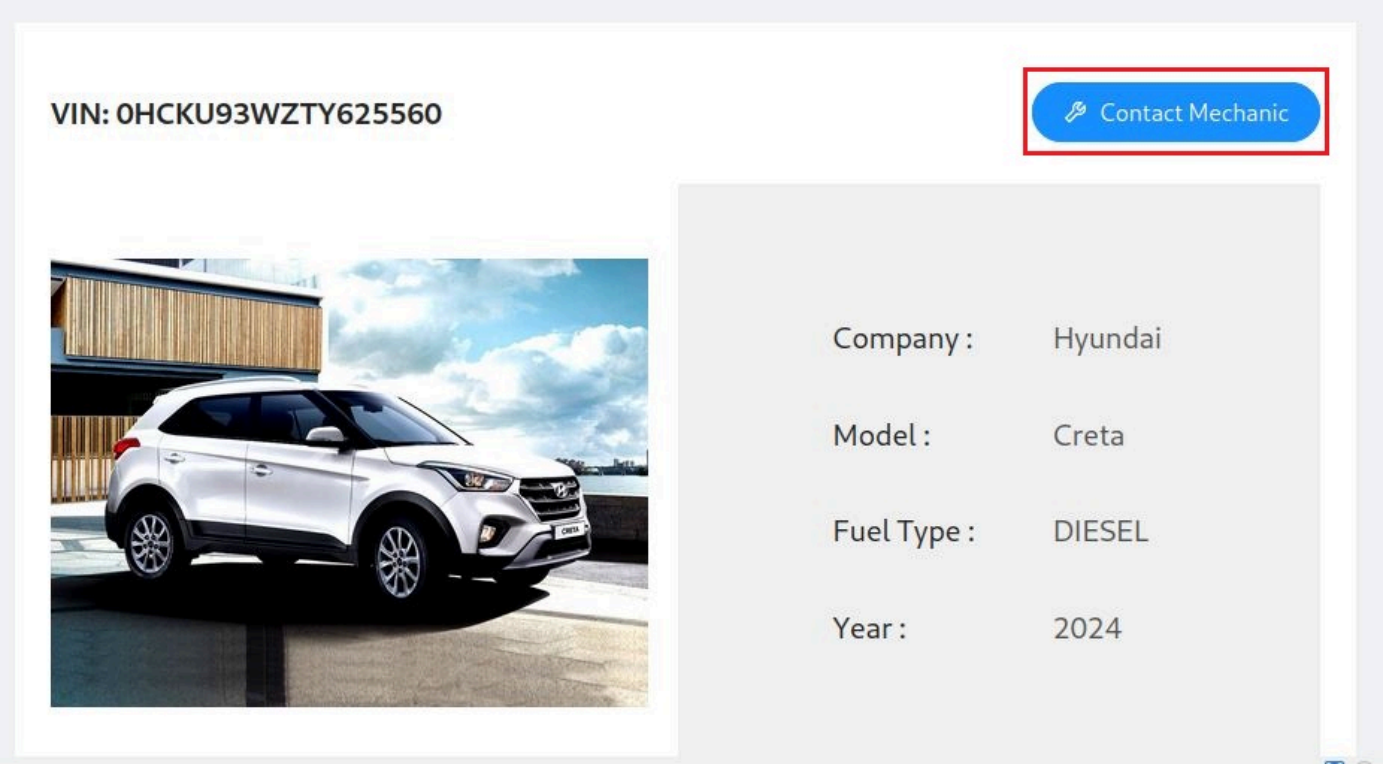


7. Server-Side Request Forgery (SSRF)

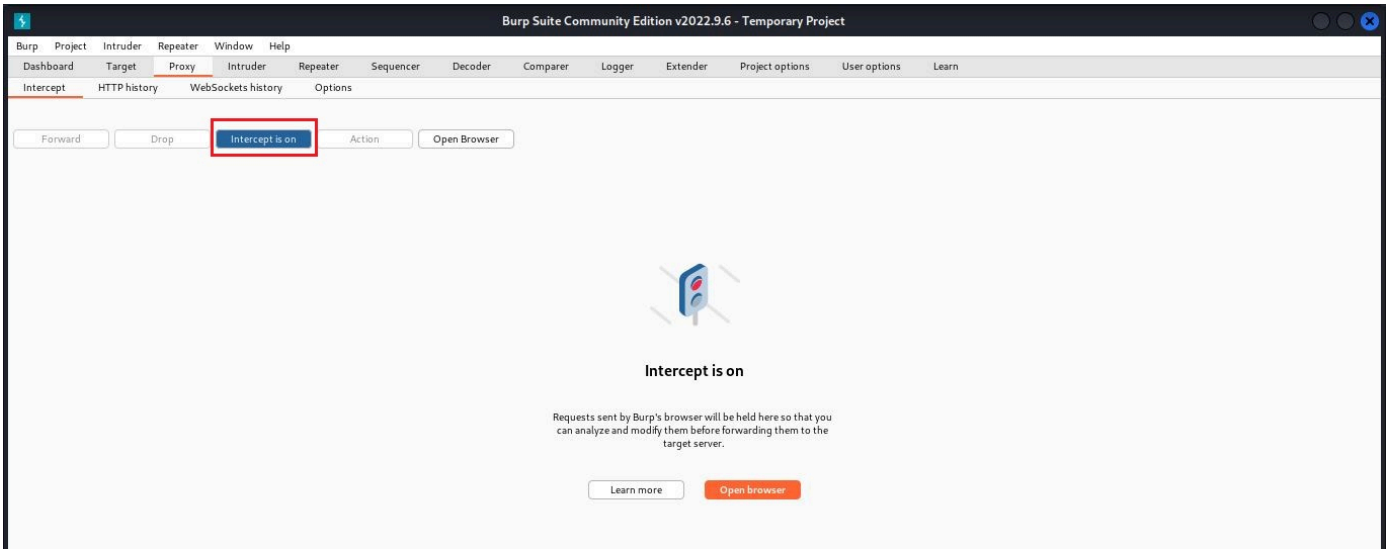
The weakness of SSRF occurs whenever a web application takes remote resources without validating the URL that the user gives. The target application may have a function to import data from a URL, publish data to a URL, or read data from a URL that can be tamed. This allows the attacker to force the application to send requests made to unexpected purposes, even when protected by a firewall, VPN, or other type of network access control (ACL) list.

Challenge 11 - Make crAPI send HTTP call to "www.google.com" and return the HTTP response.

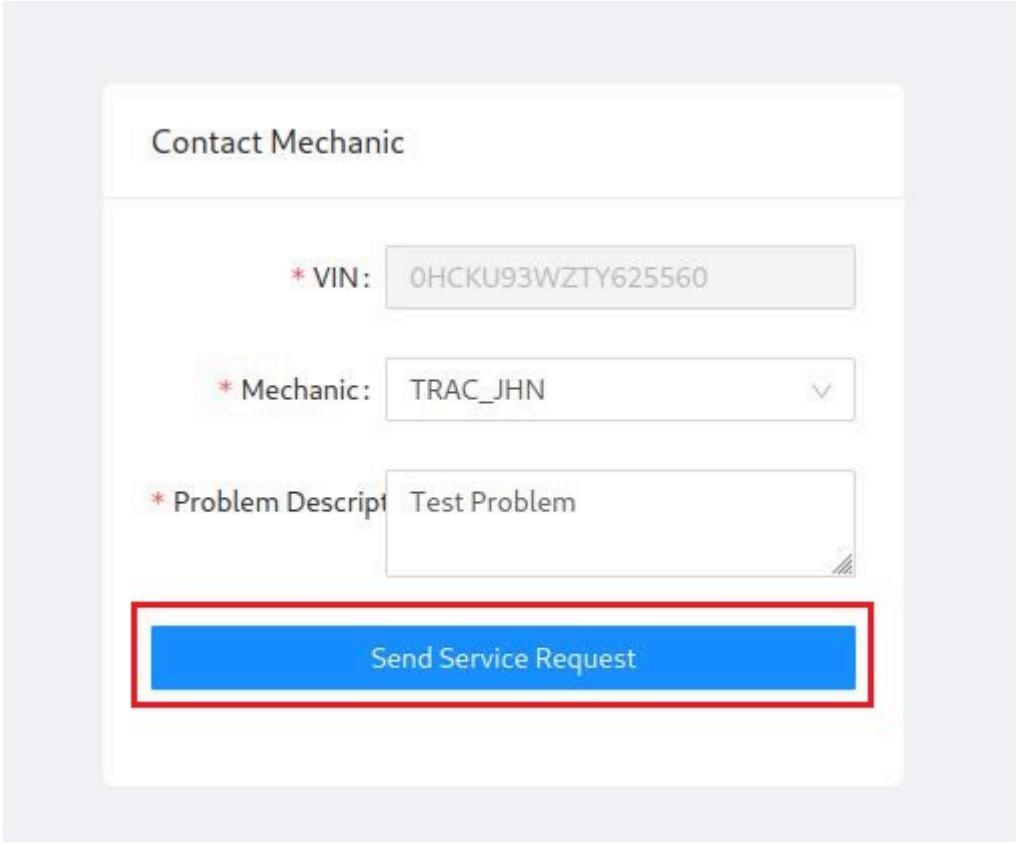
- Back to the dashboard page, press **the Contact Mechanic** button



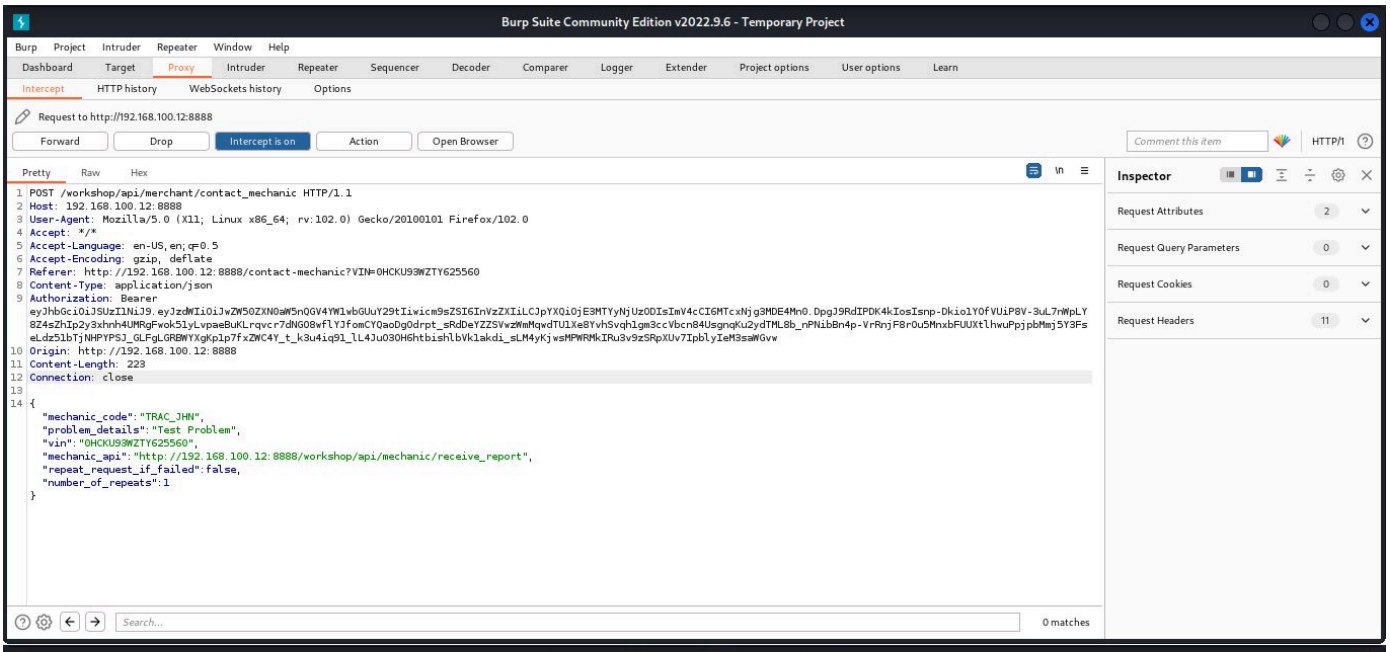
- Turn on the Intercept on the Burp Suite

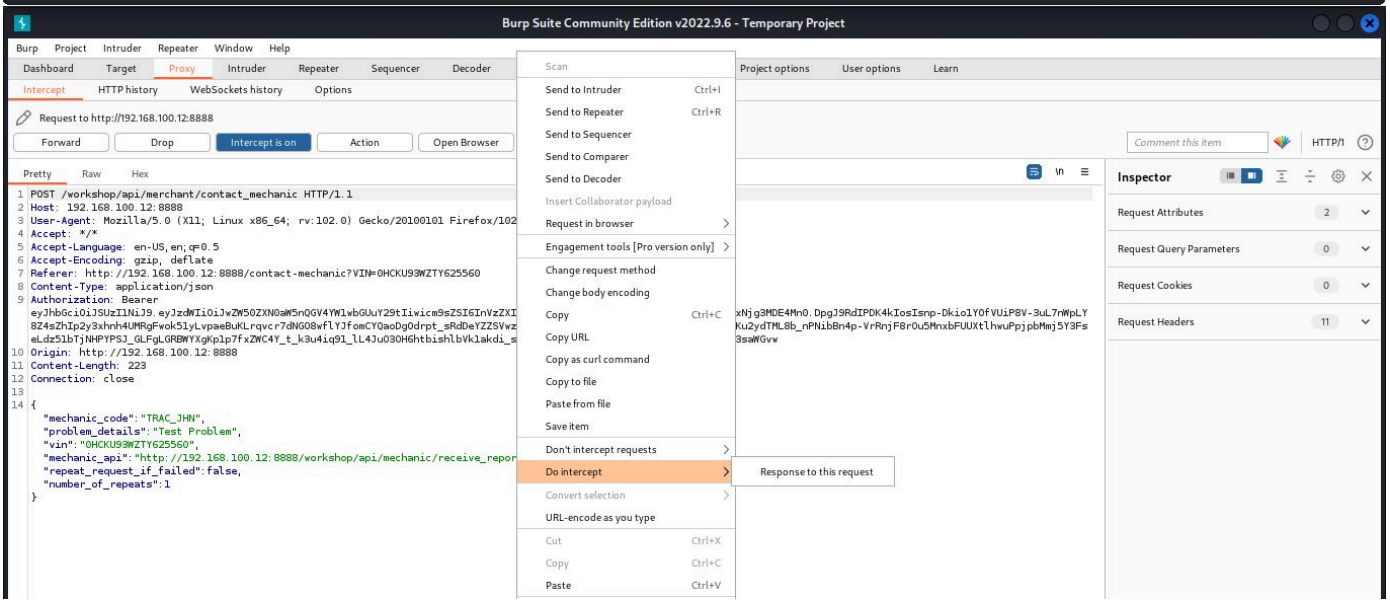


- Fill out the form and press **the Send Service Requests** button

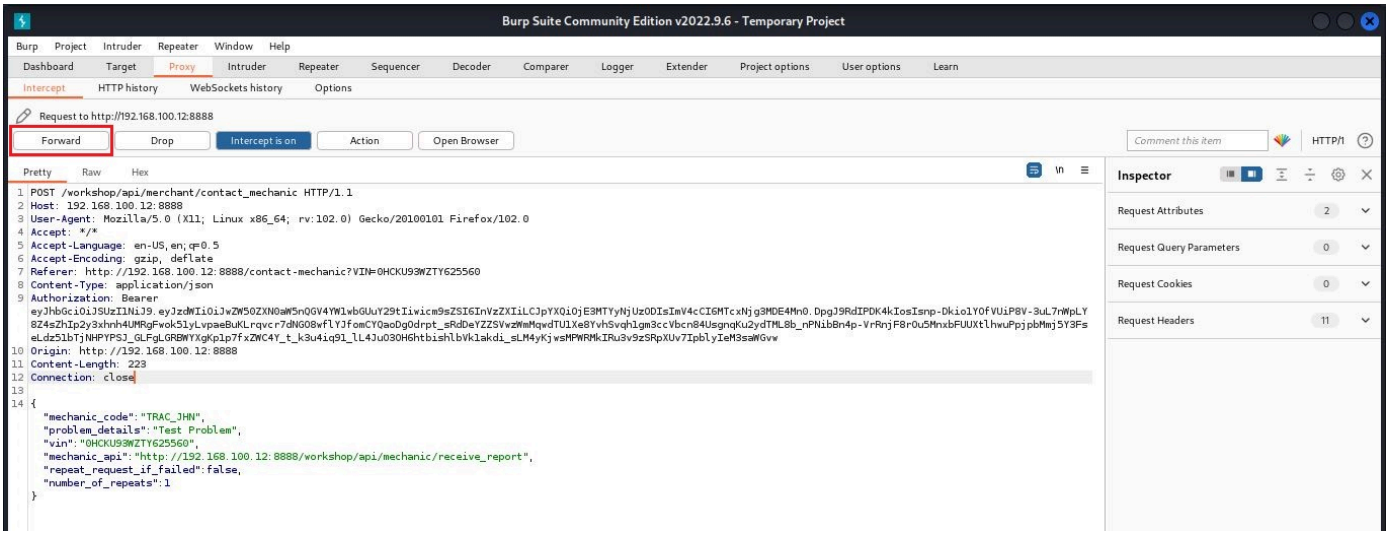


- Once the Burp Suite opens, right-click and select **Do intercept > Response to this request**

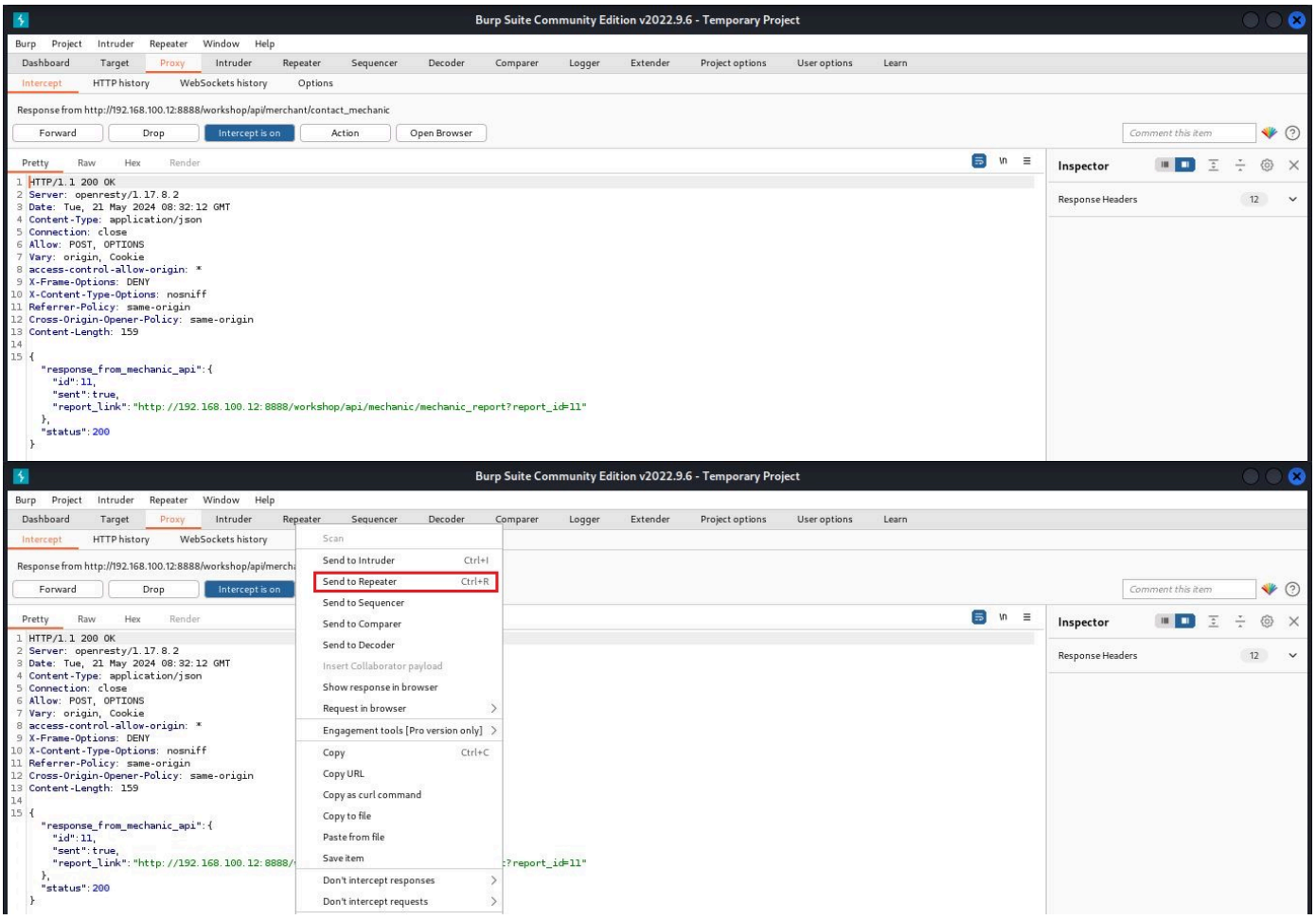




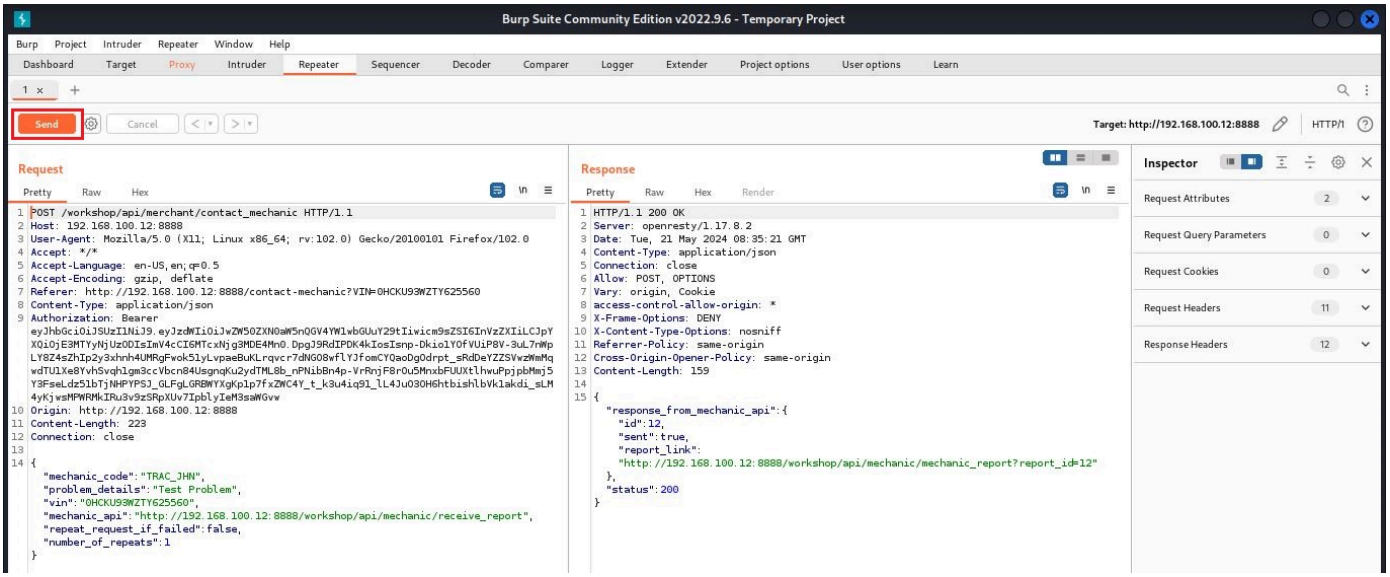
- Next press **the Forward** button



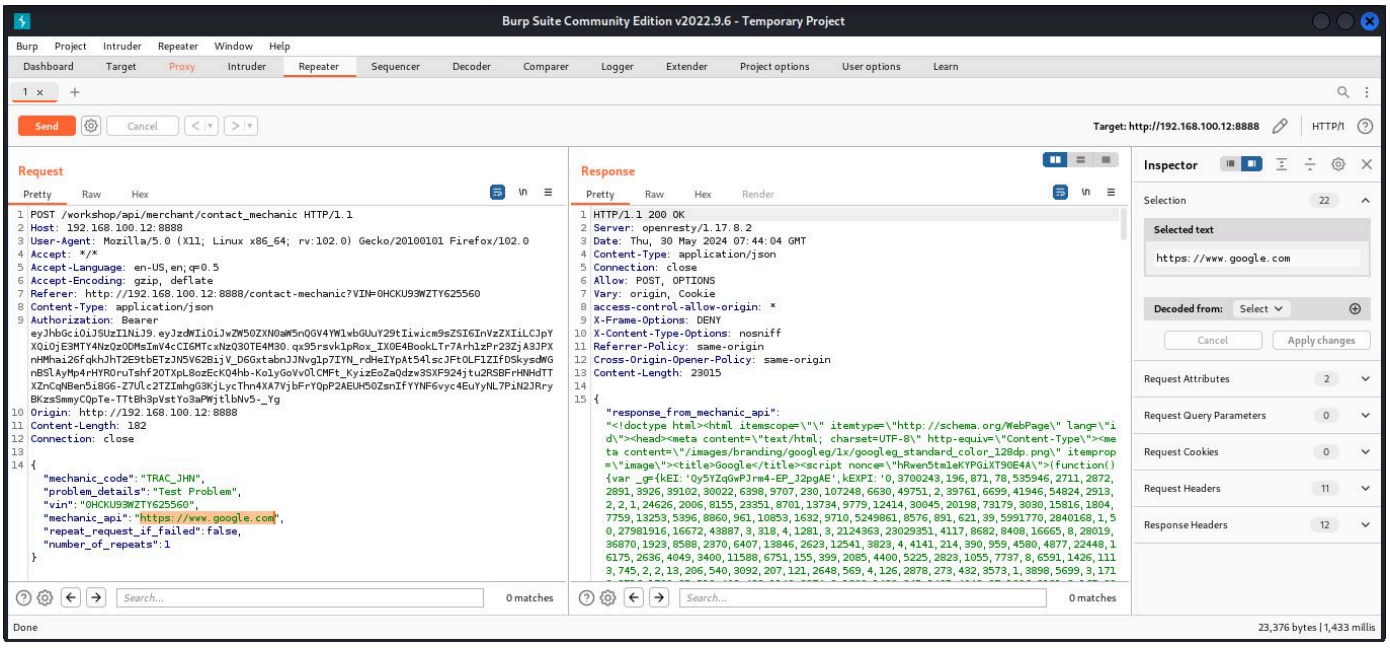
- If you get the following response, right-click and then select **Send to Repeater**



- Move to the Repeater tab and press **the Send** button



- Change the parameter value `mechanic_api` Be a `https://www.google.com` Press **the Send** button**.**

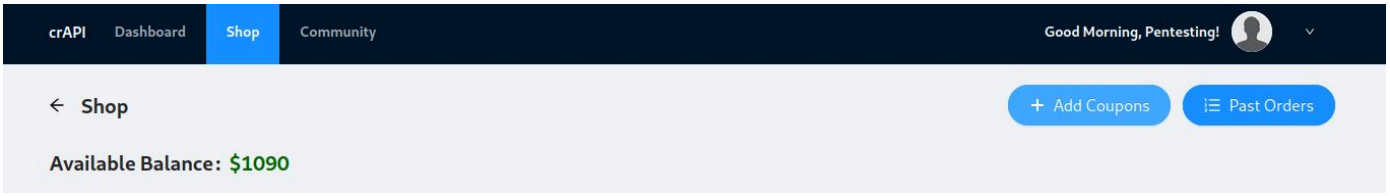


8. NoSQL Injection

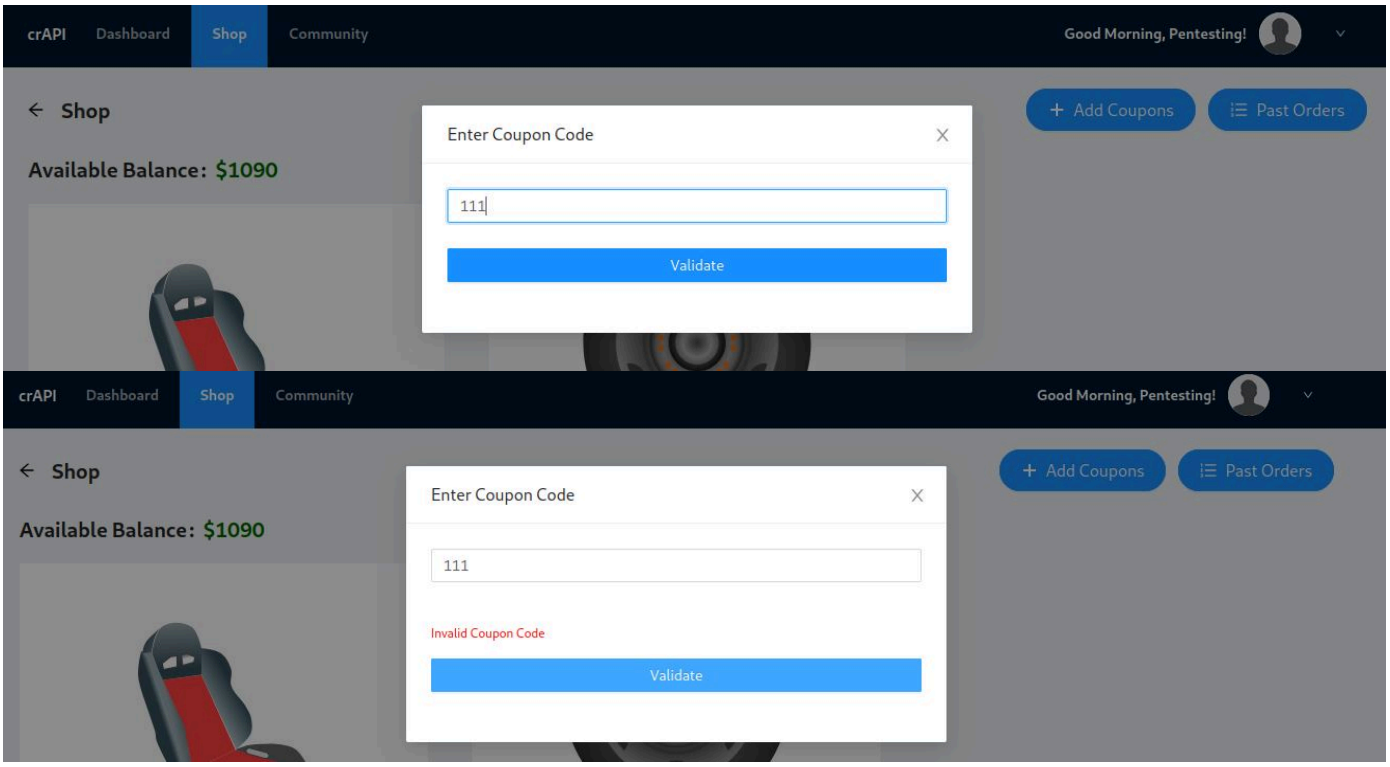
NoSQL Injection attacks can be run in different application areas than regular SQL Injection. If SQL Injection will be run inside the database engine, the NoSQL variant can be run inside the application layer or database layer, depending on the NoSQL API used and the data model. Usually the NoSQL Injection attack will be run when the string of attacks is decomposed, evaluated, or combined into a NoSQL API call.

Challenge 12 - Find a way to get free coupons without knowing the coupon code.

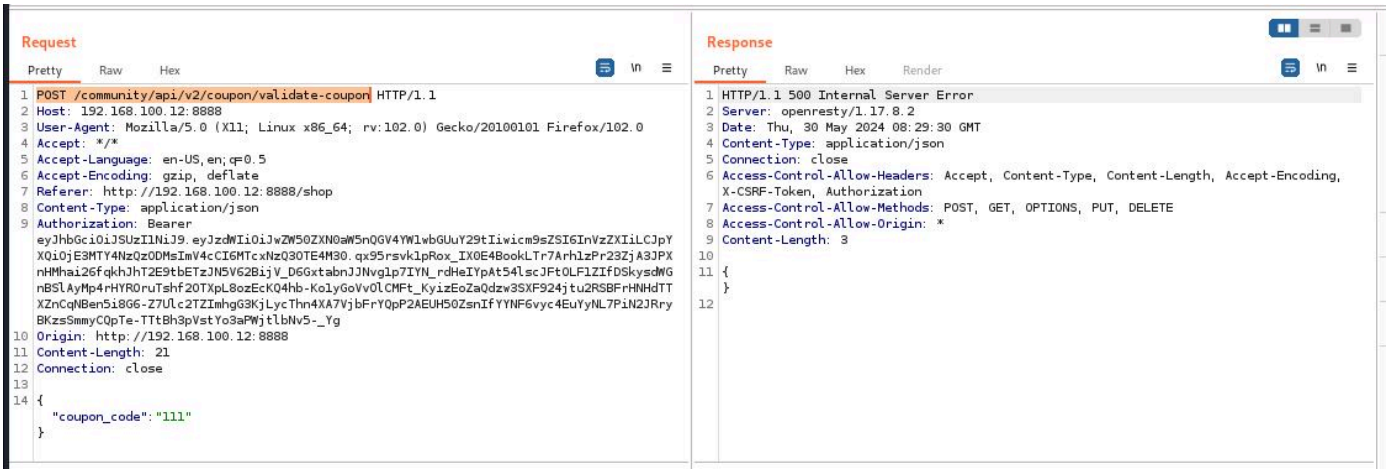
- Turn on the Burp Suite tool and go to **the Shop** page



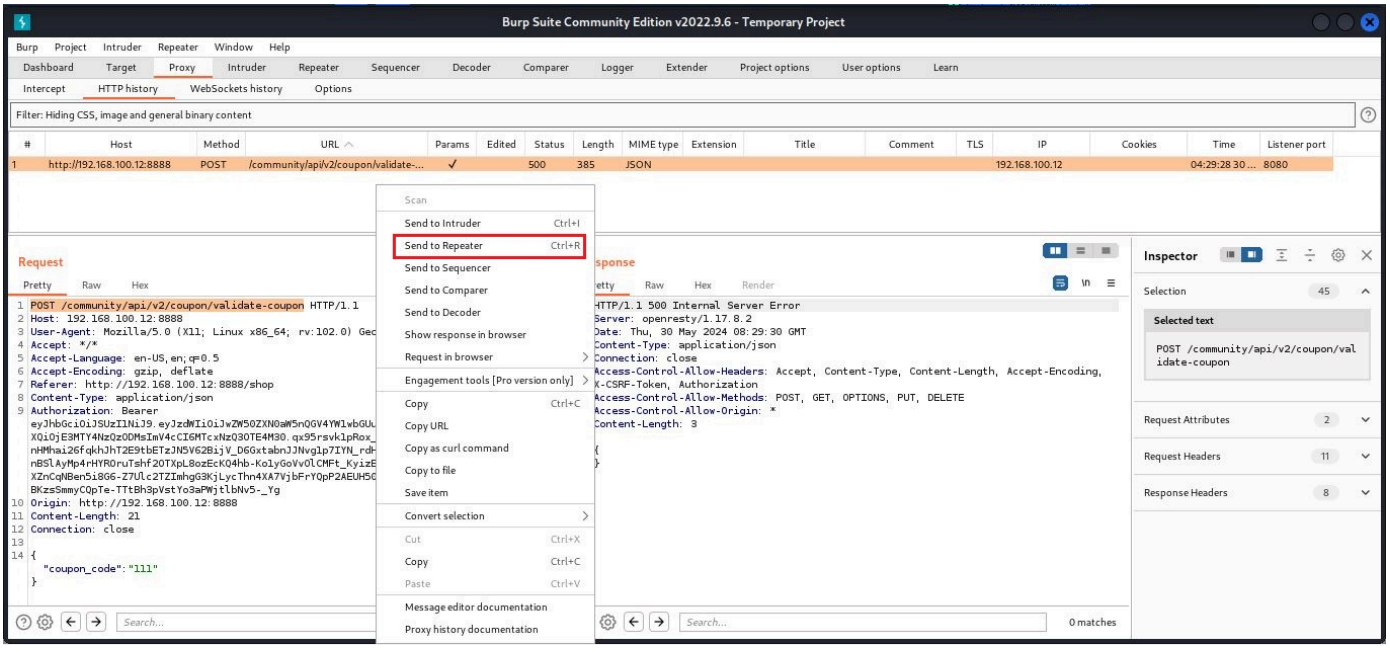
- Click **the Add Coupuns** button then enter any coupon code and press **the Validate** button



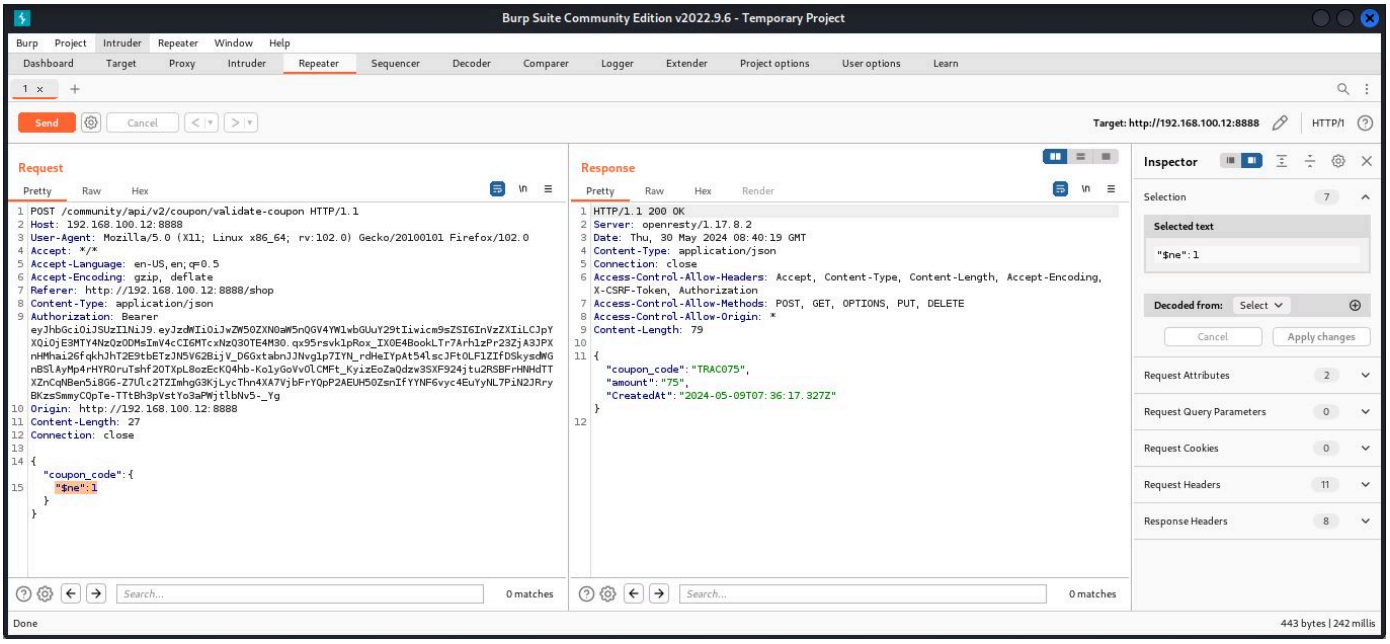
- In the Burp Suite tool will be detected endpoint API `/community/api/v2/coupon/validate-coupon` used for validation of coupon codes



- Right click the request and then select **Send to Repeater**



- On the reference https://github.com/swisskyrepo/PayloadsAllThings/tree/master/NoSQL%20Injection payload NoSQL Injection is not equal (`KaTeX parse error: Expected '}', got 'EOF' at end of input: …he payload `{ "ne": 1 }. Go to the Repeater tab, change the parameter value coupon_code Becoming { "$ne": 1 }Then click **Send**the Send button. After the request is successfully sent we managed to get a valid coupon code from the system that is TRAC075which amount to 75`

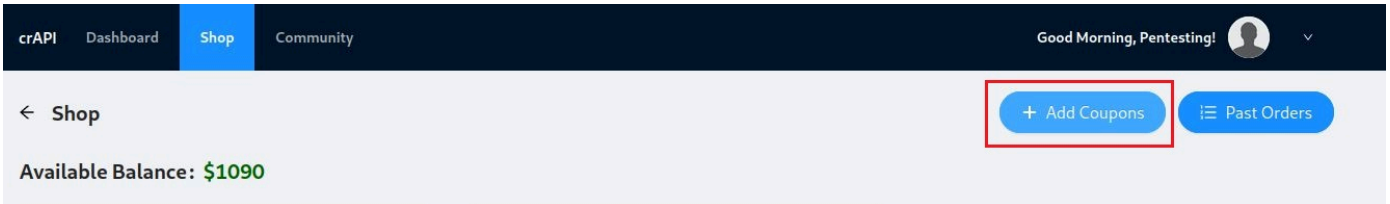


9. SQL Injection

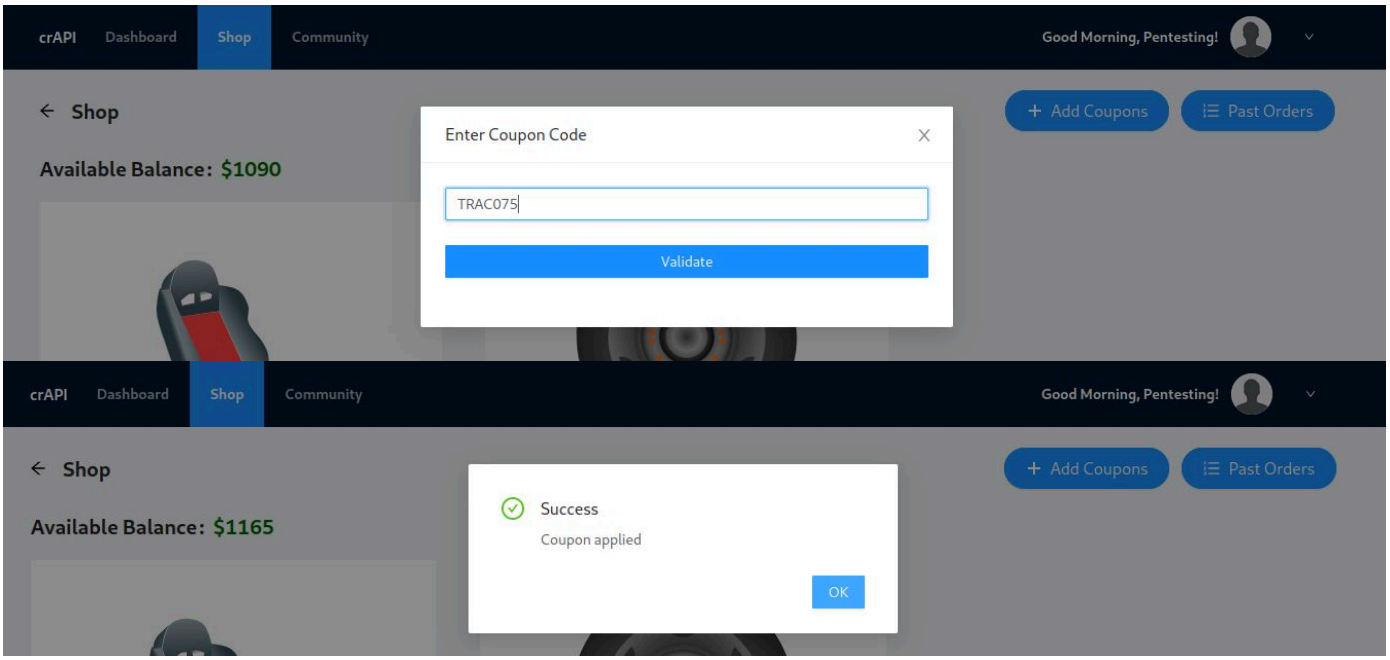
SQL Injection is a type of injection attack, in which the SQL command is inserted into the data input to influence the execution of predefined SQL commands. Successful SQL Injection attacks can read sensitive data from a database, modify database data, or run administrative operations on a database. In general, the way web applications create SQL queries is by involving SQL syntax written by programmers mixed with data provided by users.

Challenge 13 - Find a way to redeem a coupon that you have already claimed by modifying the database

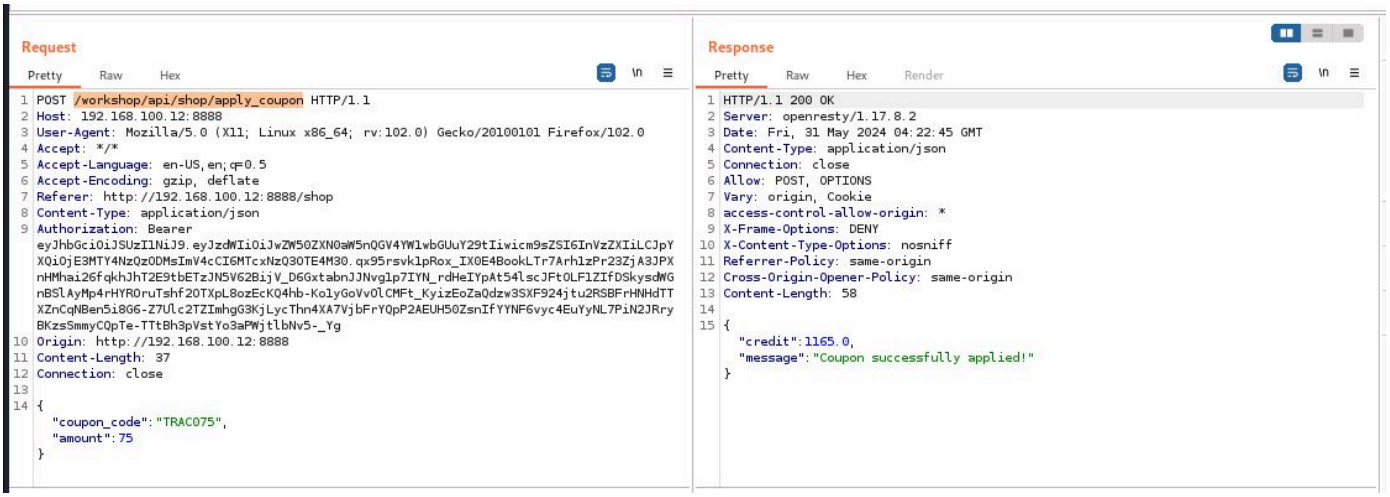
- In Challenge 12 (No SQL Injection) we managed to get the coupon code. Now turn on the Burp Suite tool and go to **the Shop** page



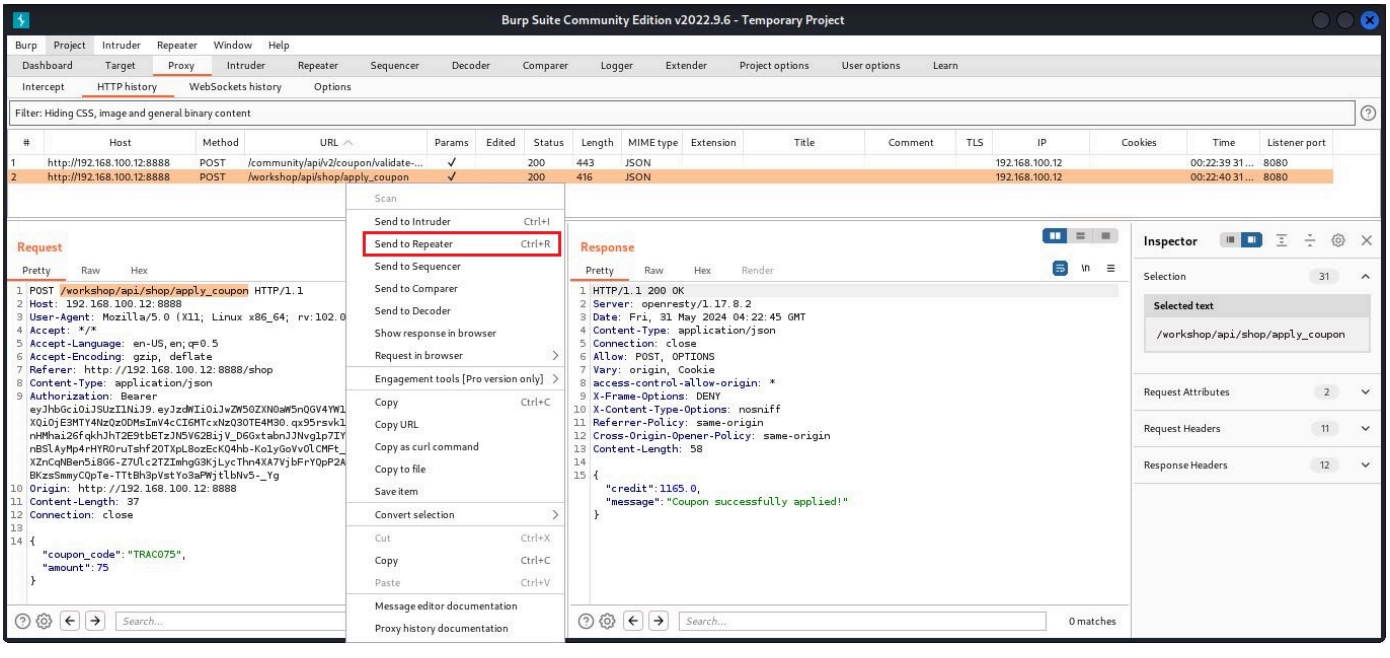
- Enter the coupon code that has been obtained from the previous challenge and press **the Validate** button



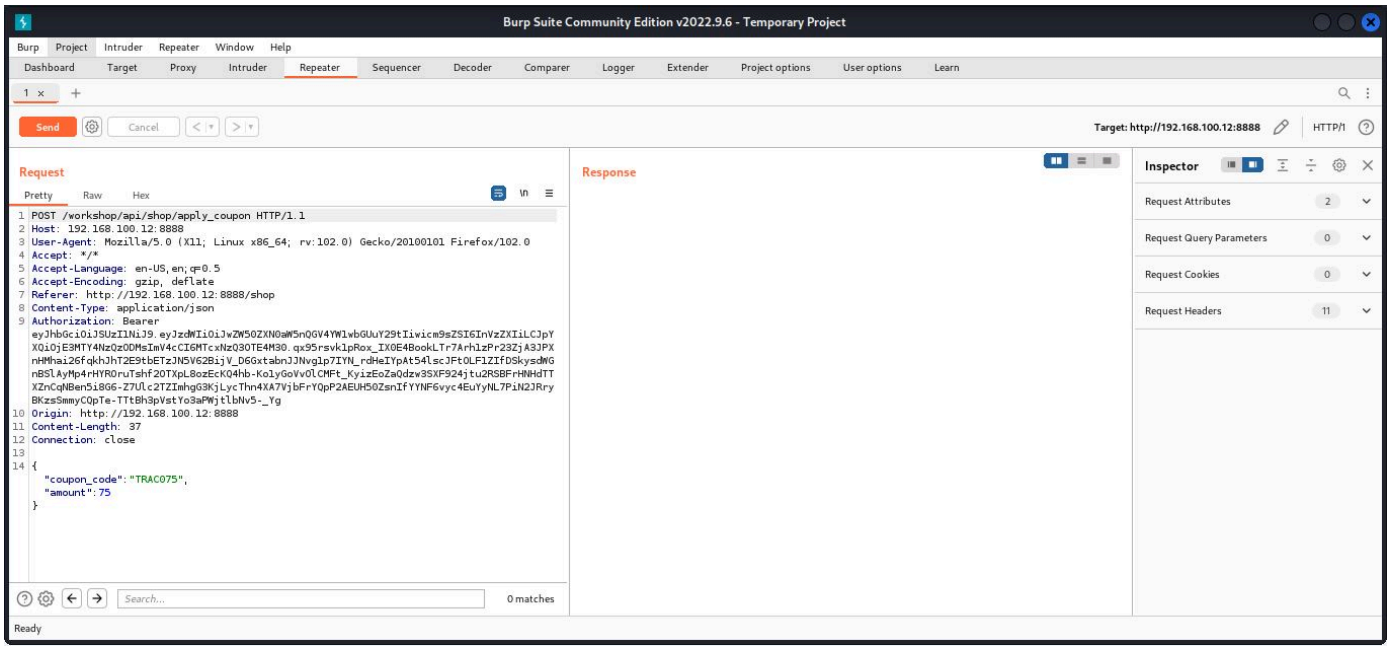
- In the Burp Suite tool will be detected endpoint `/workshop/api/shop/apply_coupon` to apply the coupon code



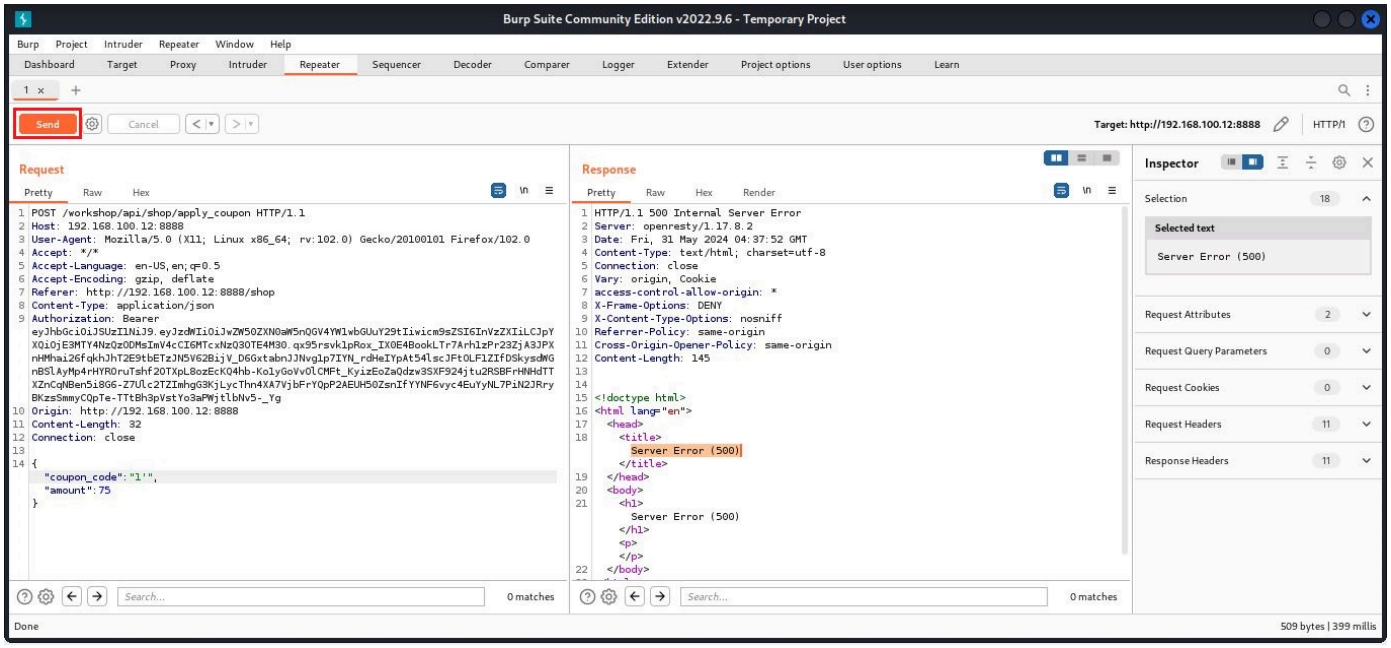
- Right click the request and then select **Send to Repeater**



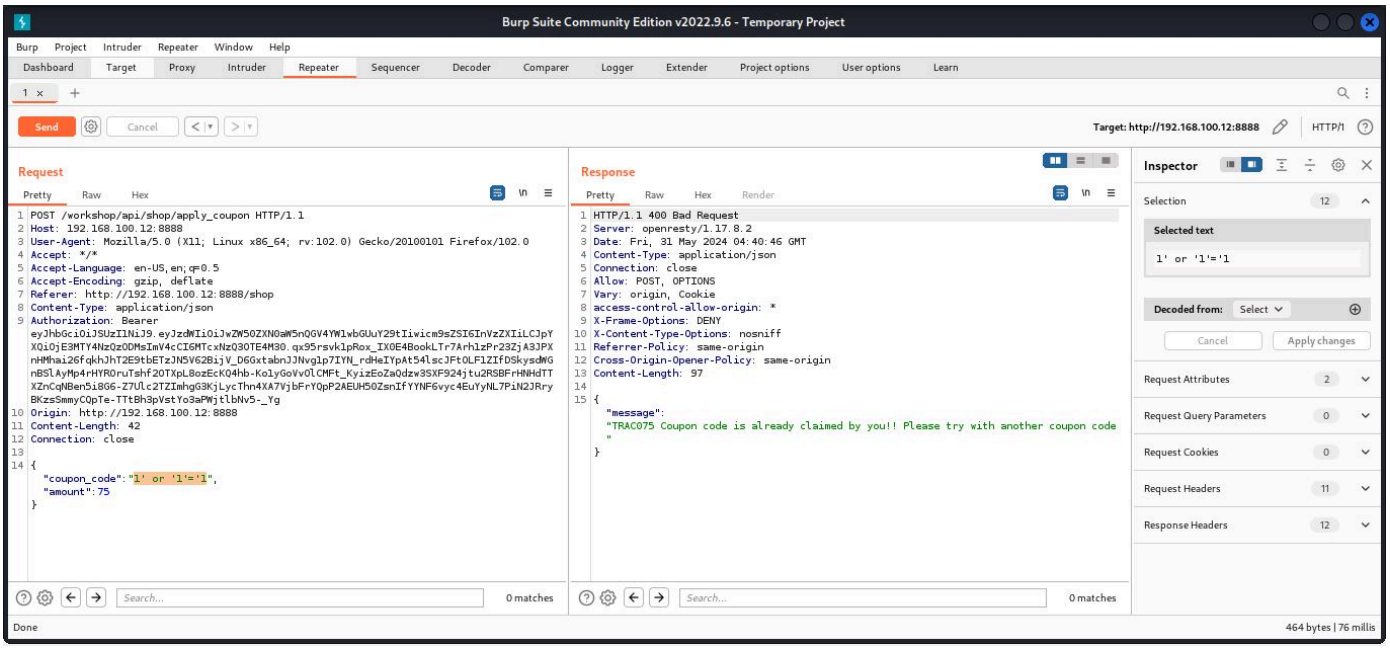
- Move to the Repeater tab



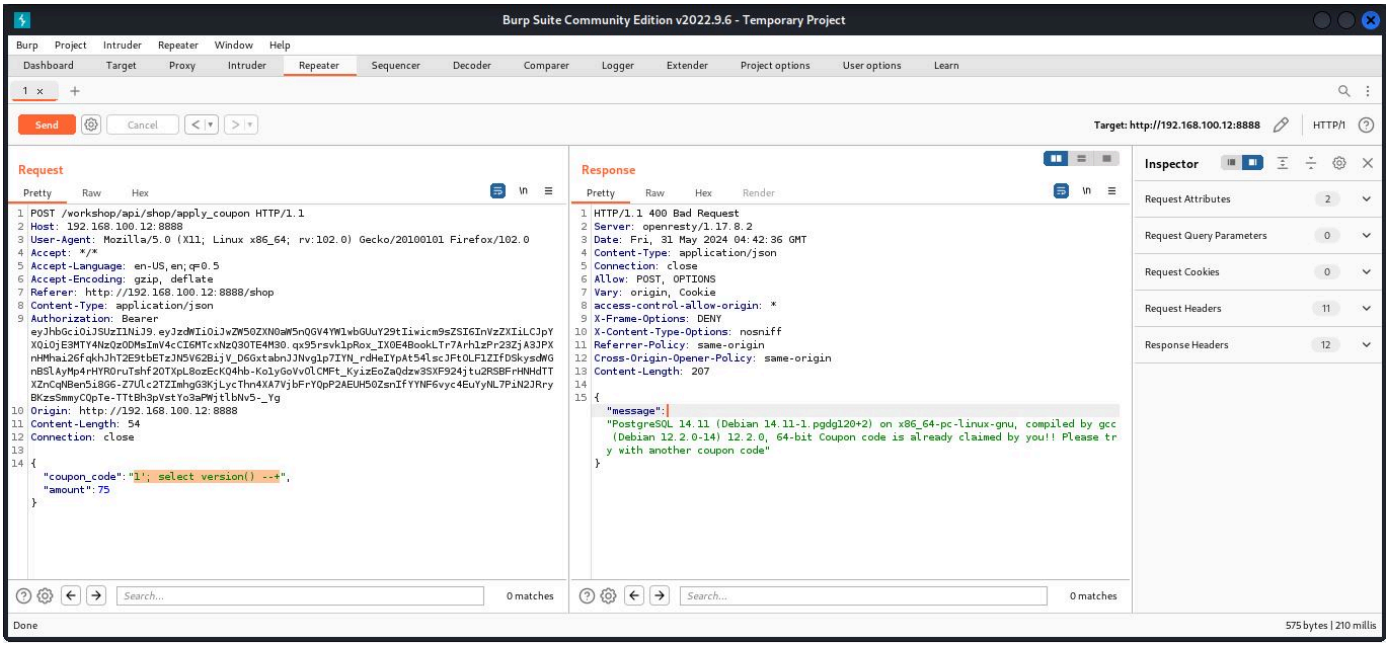
- Now we try to change the parameters `coupon_code` with value `1'` then click **the Send** button then a message appears that the server is experiencing an error



- We can try to change the parameter value `coupon_code` with one of the SQL Injection payloads such as `1' or '1'='1` then click **the Send** button then appear the message that the coupon code has been claimed which indicates that the query is worth `true` even if we don't enter a valid coupon code.



- Now we change the parameter value `coupon_code` with value `1'; select version() --+` Press the Send button**.** After the request was successfully sent we managed to get the database information used

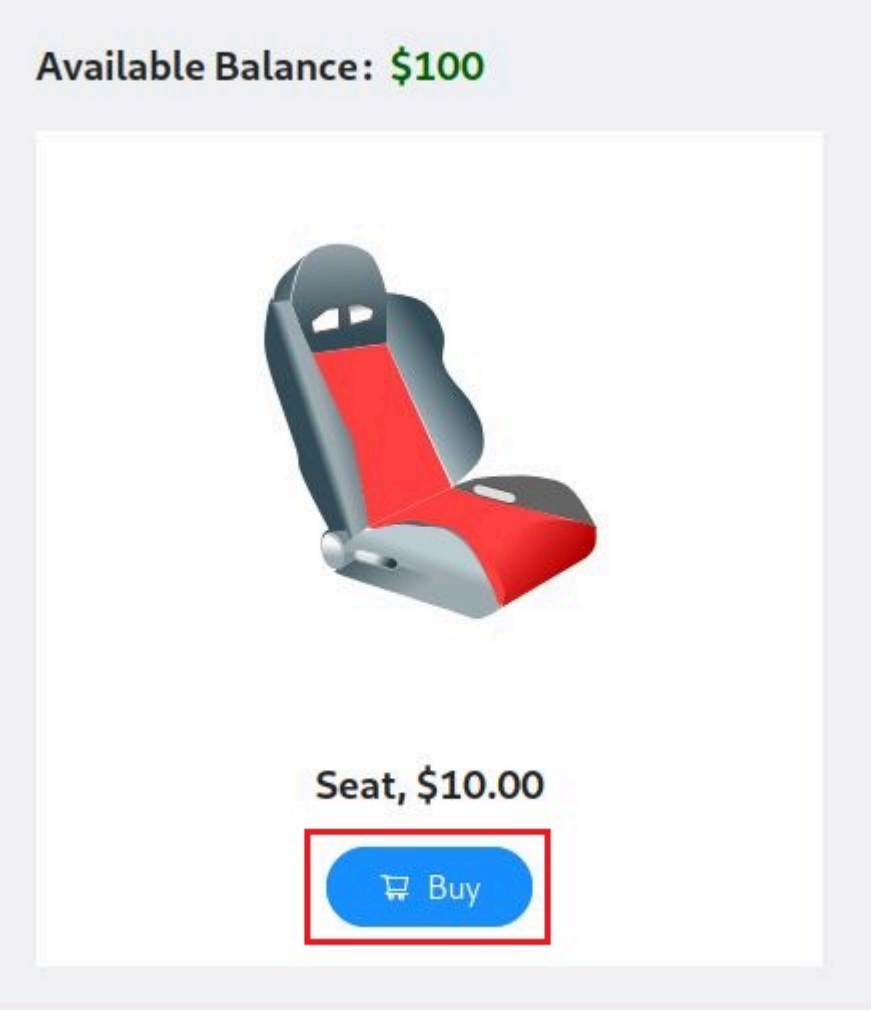


10. Unauthenticated Access

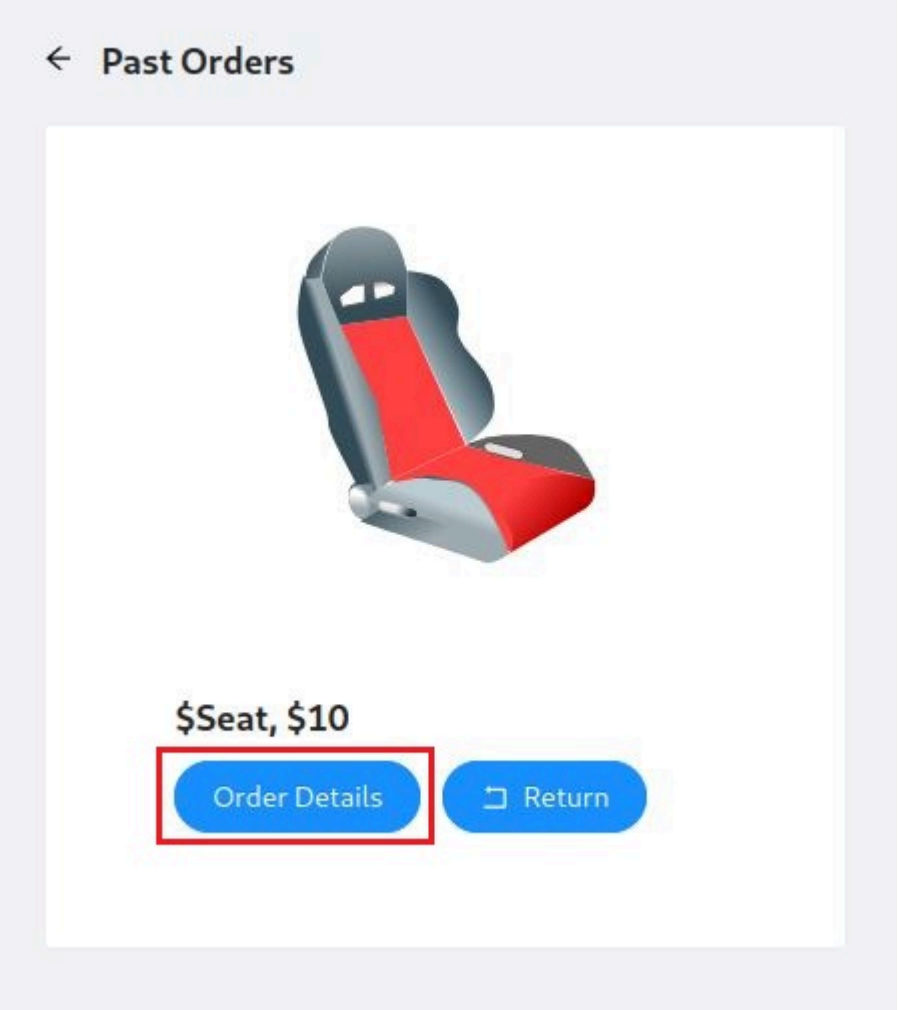
Access without authentication refers to the granting of permission to the user to interact with the system or application without requiring them to authenticated or login in advance so that they can perform any action without providing credentials.

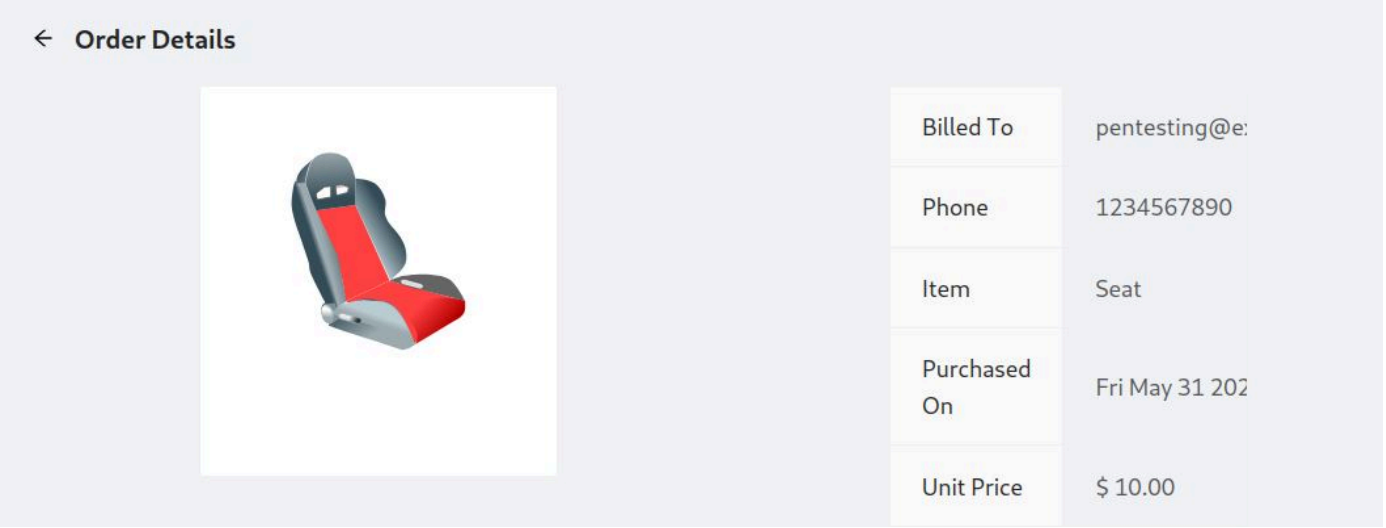
Challenge 14 - Find an endpoint that does not perform authentication checks for a user.

- Turn on the Burp Suite tool and go to **the Shop** page then buy one of the items

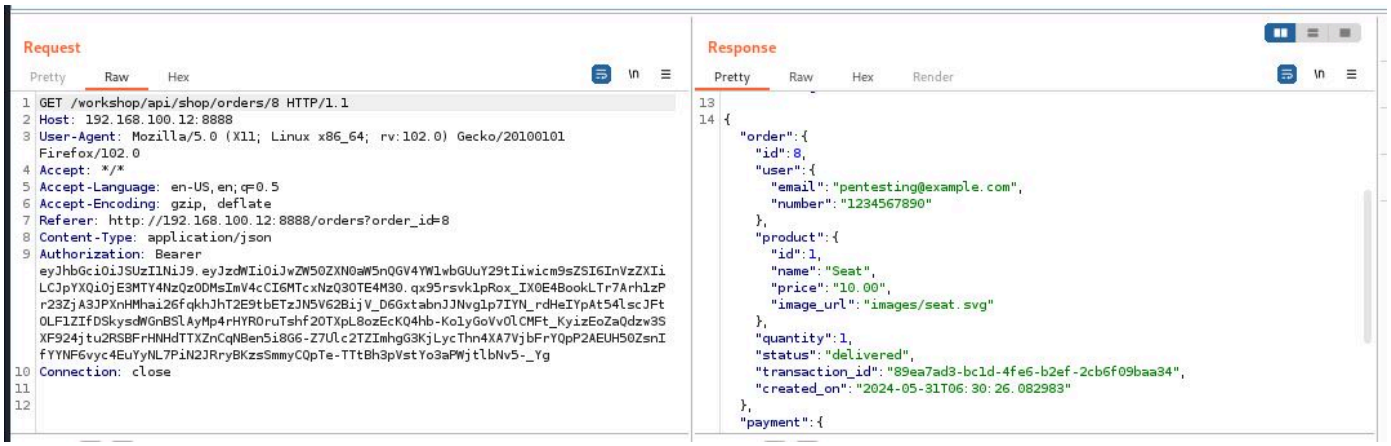


- Then click **the Order Details** button

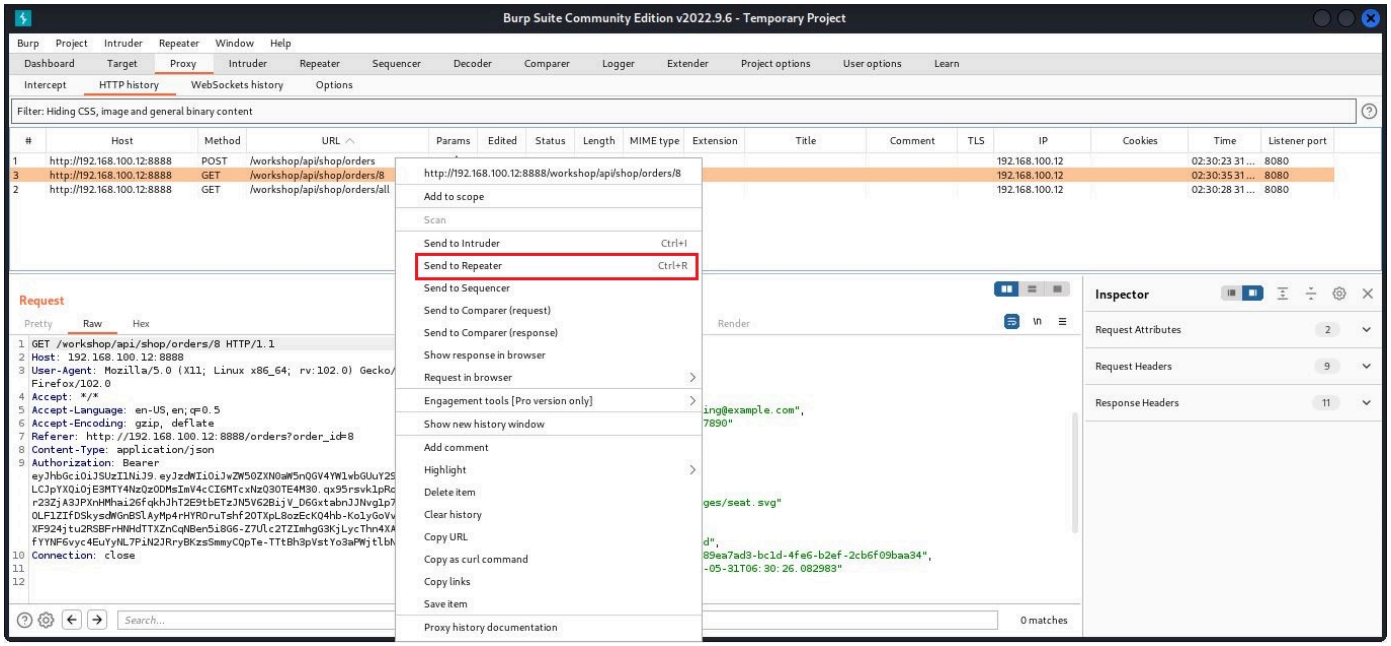




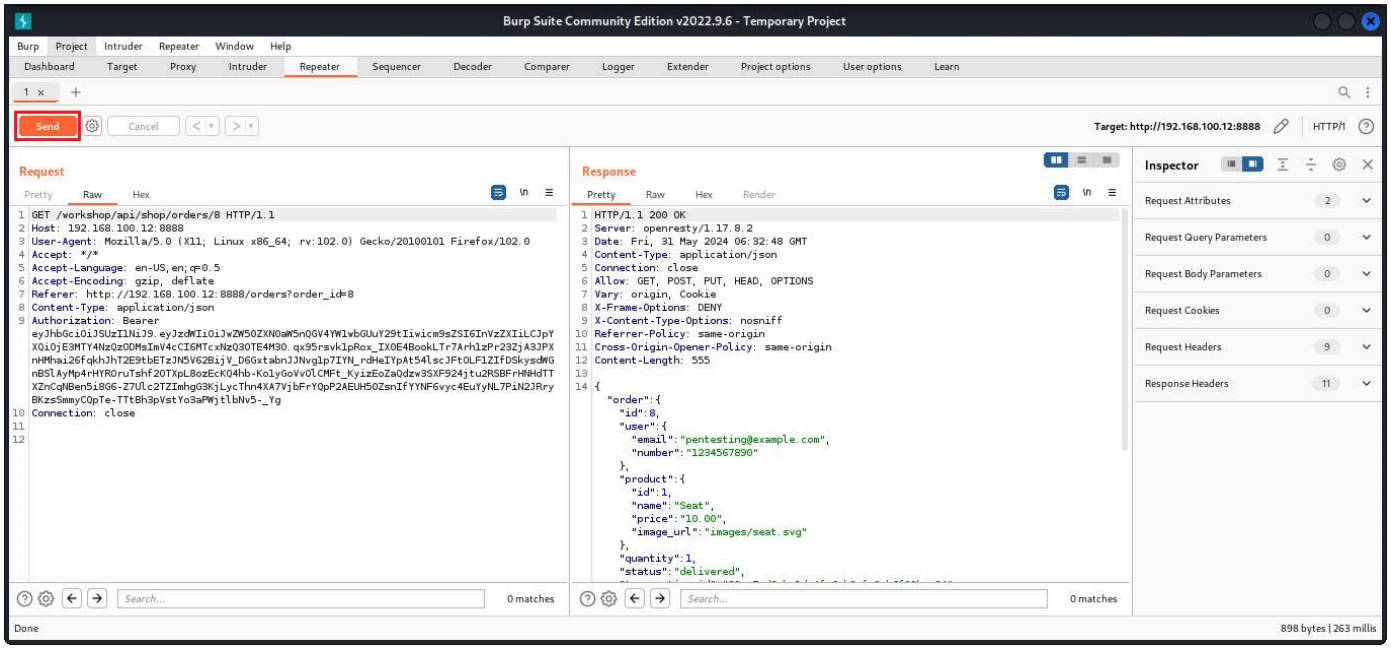
- On the Burp Suite tool detected endpoint `/workshop/api/shop/orders/` with type request `GET` which contains booking and payment details



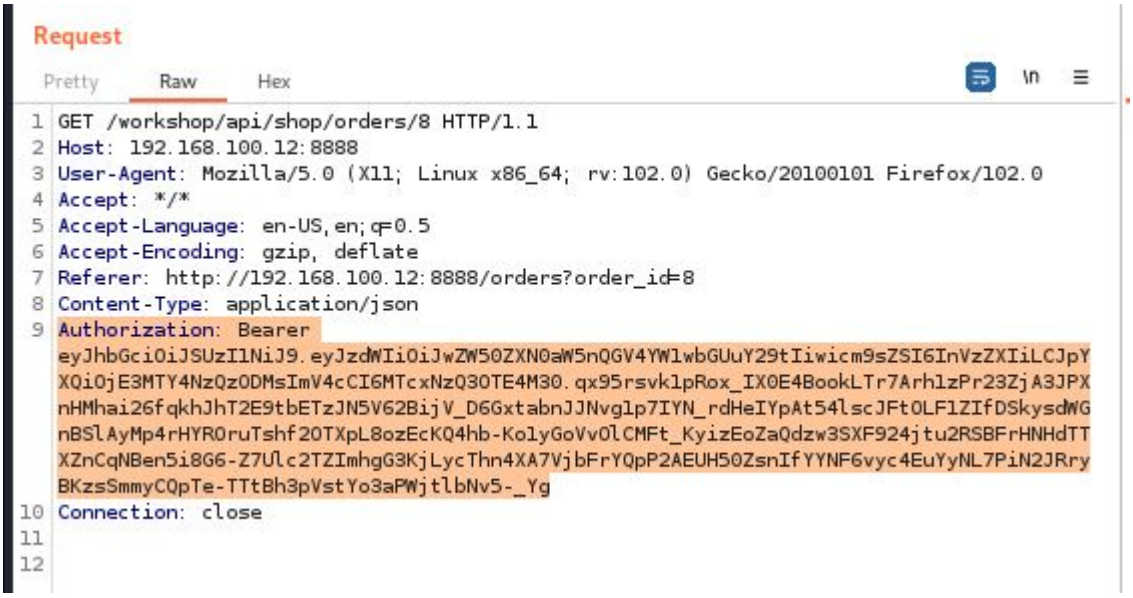
- Right-click the request and select **Send to Repeater**



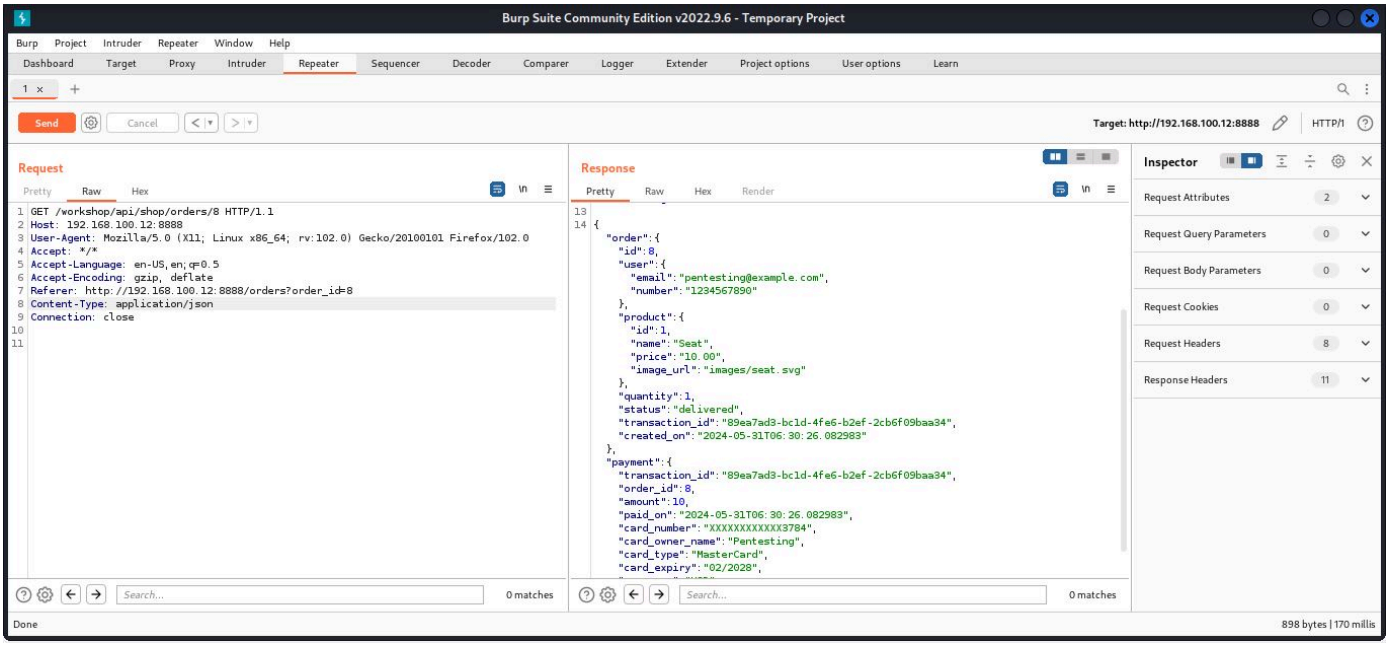
- Move to the Repeater tab and click **the Send** button



- Now we try to remove the authentication header



- If you have pressed **Send**the Send button. After the request is successfully sent we can still access the order data



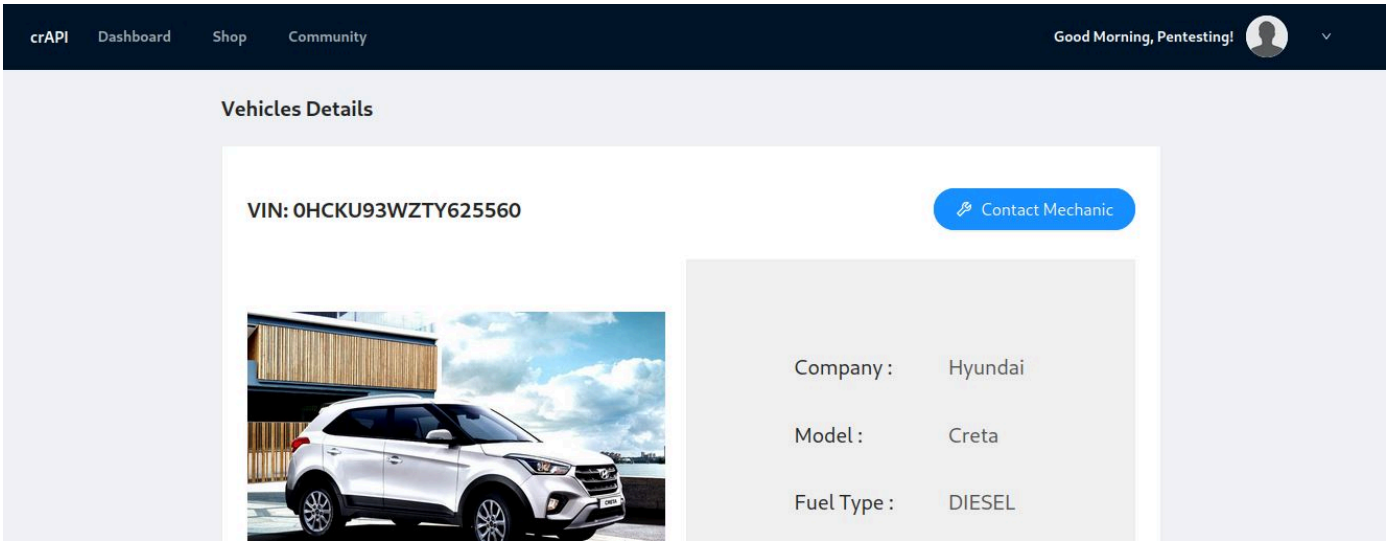
11. JWT Vulnerabilities

JSON Web Token (JWT) is a cryptographically signed JSON token, which is intended to share claims between systems. These tokens are often used as authentication tokens or sessions, particularly in

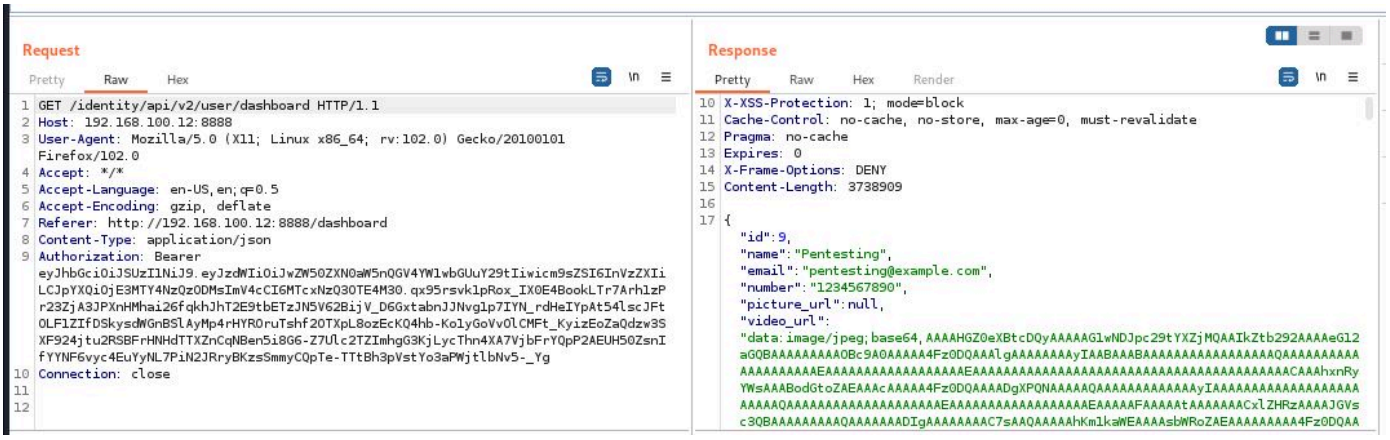
REST APIs. Because it is used for authentication, vulnerabilities can easily result in the application being fully compromised.

Challenge 15 - Find a way to forge valid JWT Tokens

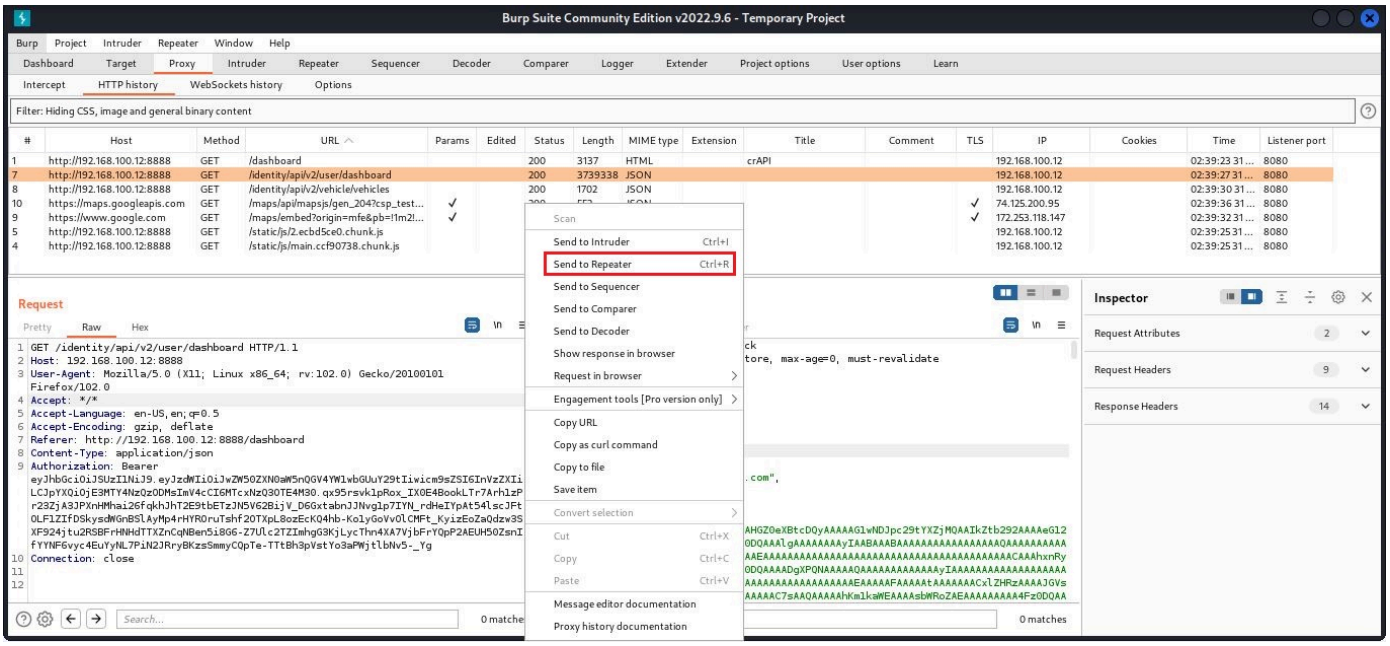
- Turn on the Burp Suite tool and go to **the Dashboard** page



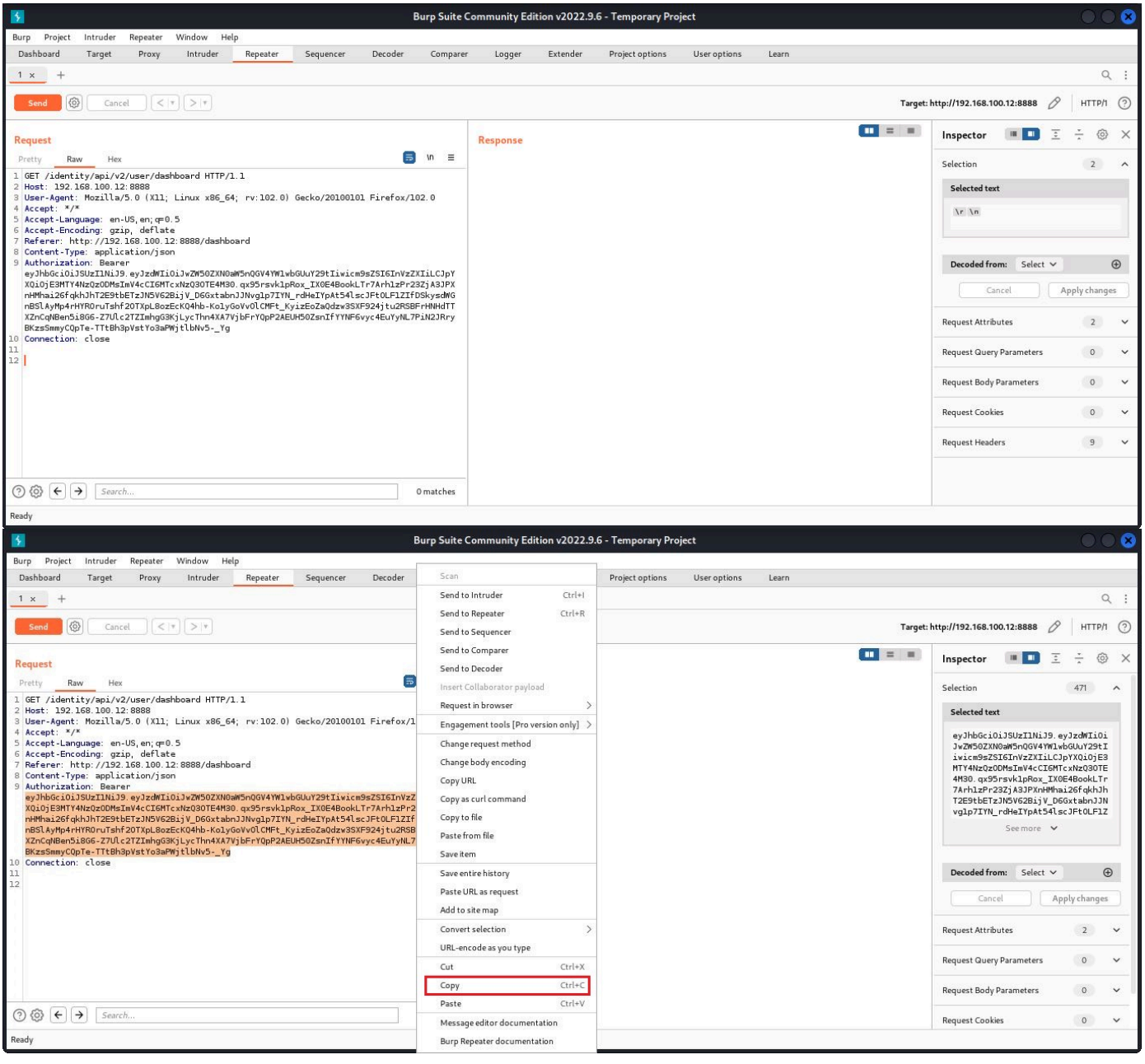
- On the Burp Suite tool detected endpoint `/identity/api/v2/user/dashboard` who accessing user data



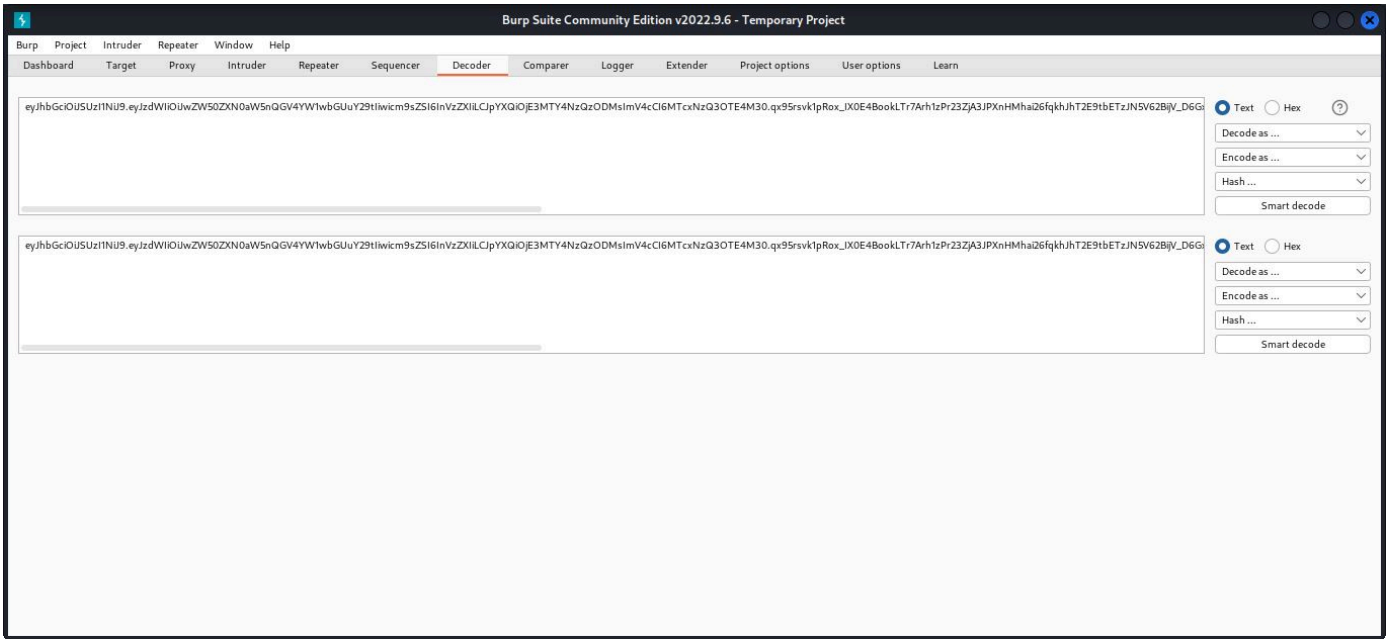
- Right-click the request and select **Send to Repeater**



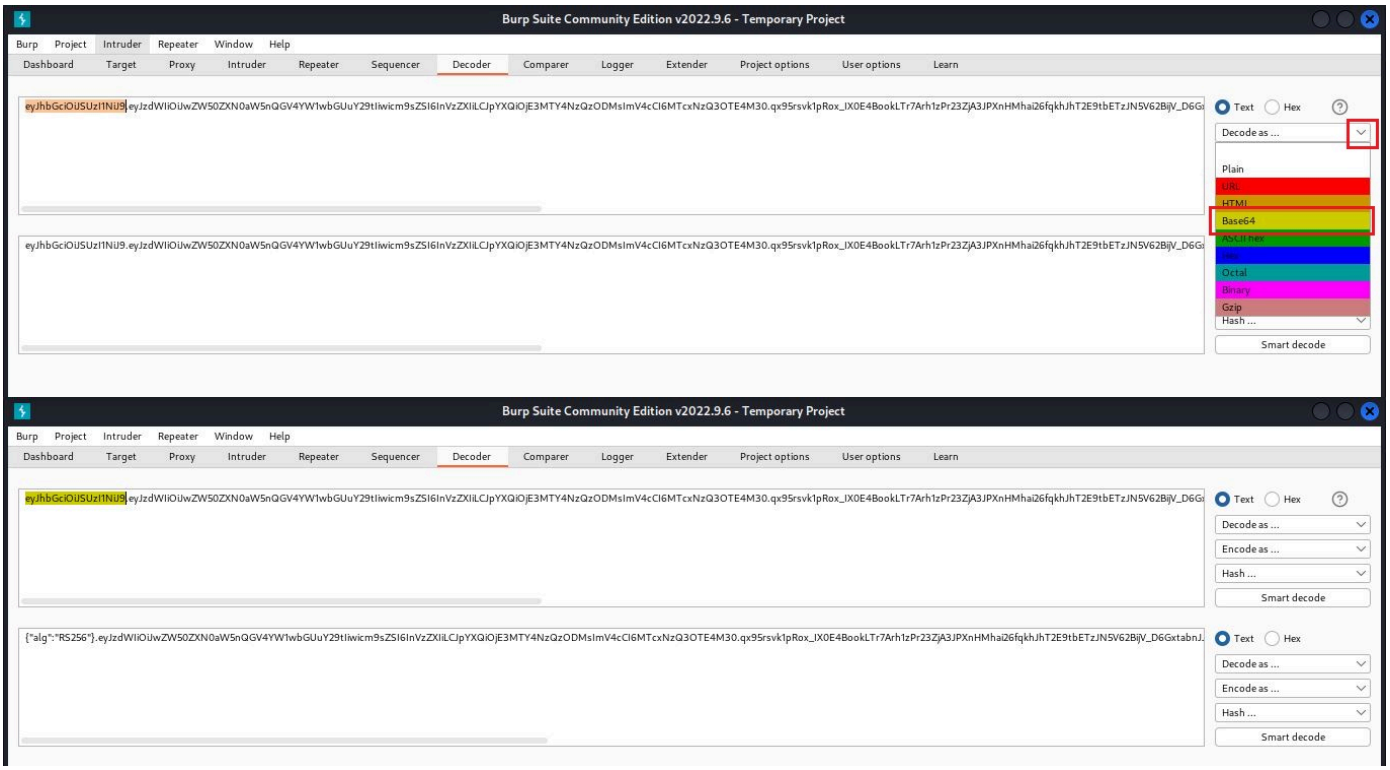
- Move to the Repeater tab, block JWT Token and then right-click and select **Copy**



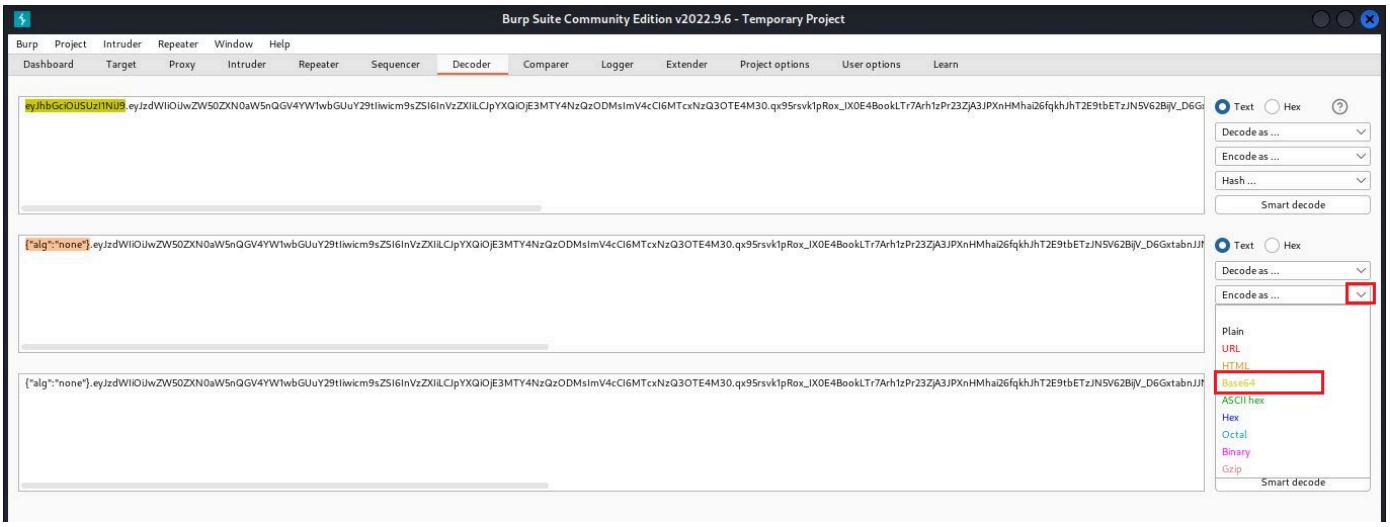
- Move to Decoder tab and paste JWT Token in the field that has been provided. Please note that the JWT Token format is `Header.Payload.Signature`



- Block the JWT token header and then select Decode as Base64

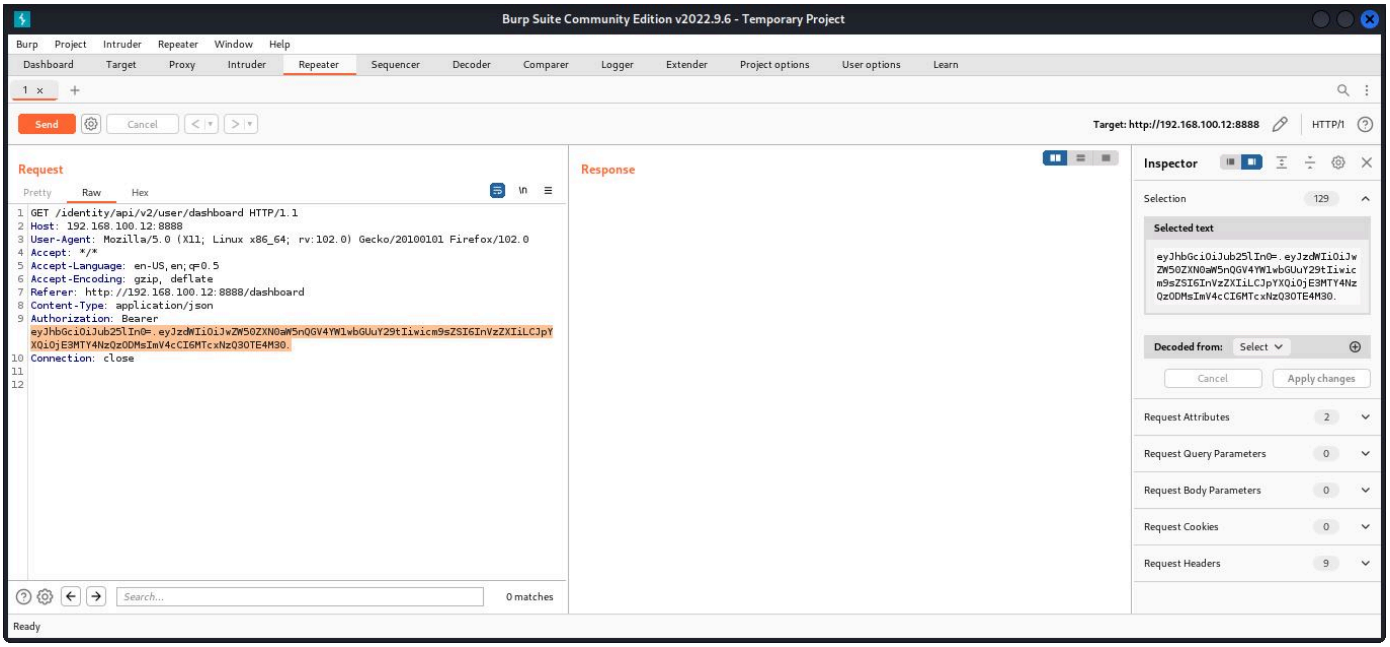


- Change `alg` from `RS256` Be a `none` then block and select Encode as Base64



- Remove the signature section on JWT Token then block the entire new JWT Token. Press `Ctrl+C` to copy

- Back to the Repeater tab, change the JWT Token in the auto-ization section with the modified JWT Token



- If you have pressed **Send**the Send button. After the request is sent we managed to be stopped even though we remove the signature on the JWT Token