Documentation: SQL Injection Vulnerability Allowing Login Bypass

Objective

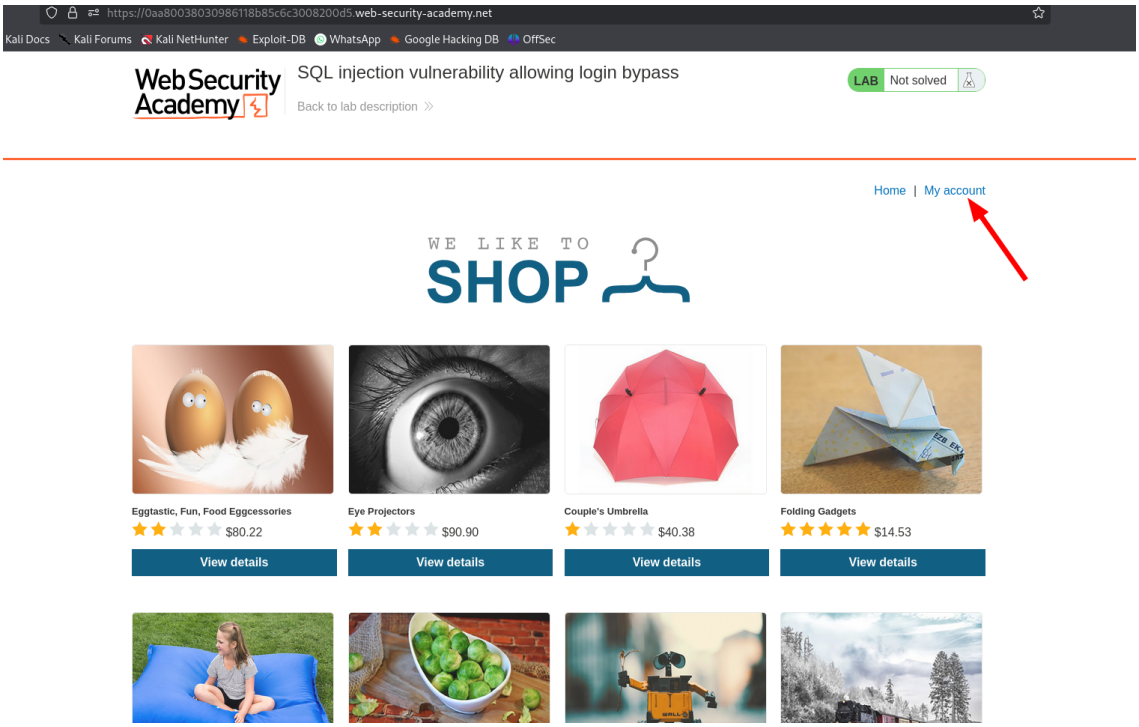
Exploit a SQL injection vulnerability in the login function to authenticate as the administrator user.

Steps to Solve

---

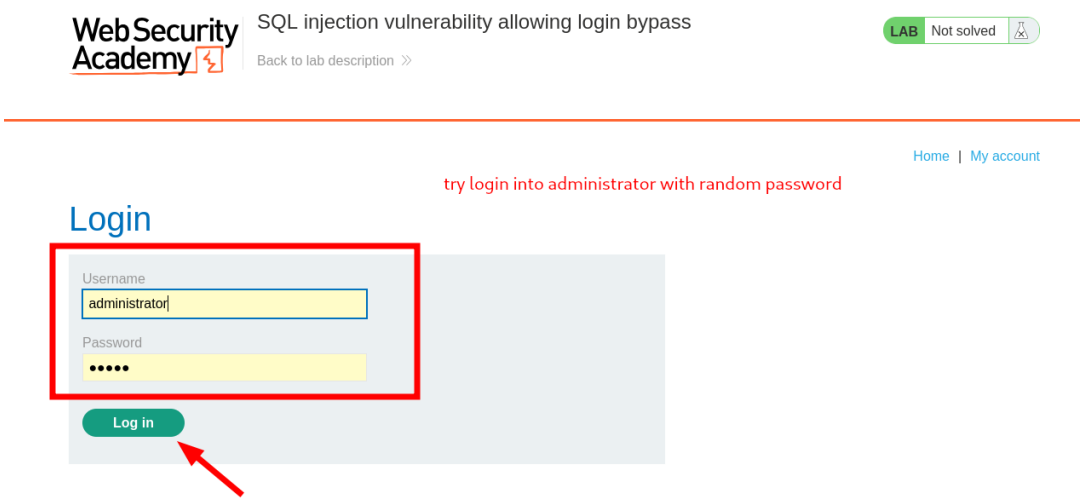
1. Preparation

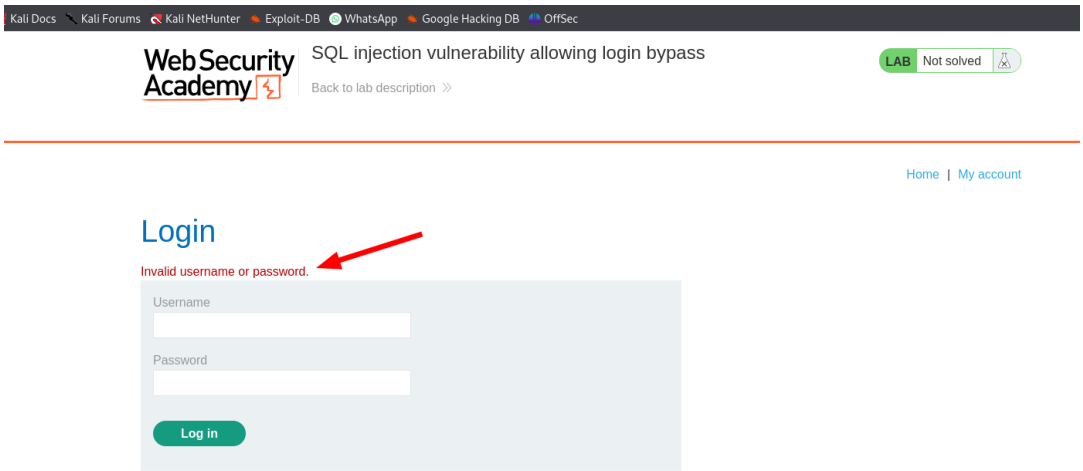
1. **Access the Lab**:
    - Open the lab URL in your web browser.
    - This will present a login page with a username and password field.



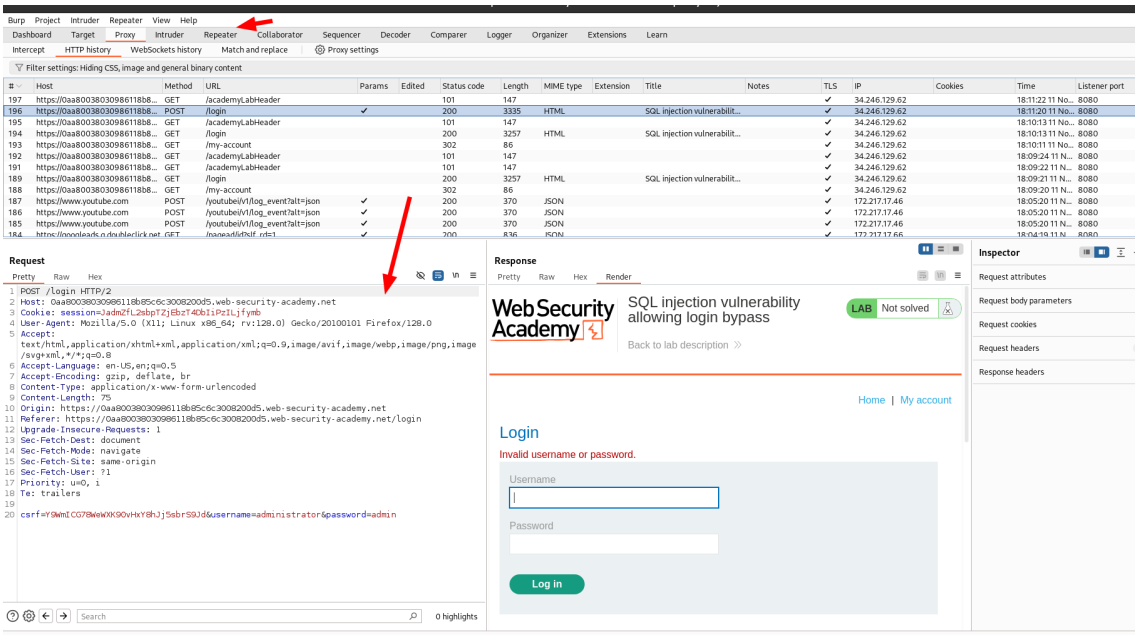
2. Analyze the Login Request

1. **Enter Dummy Credentials**:
    - In the login form, input any placeholder values like:
        - Username: administrator
        - Password: admin





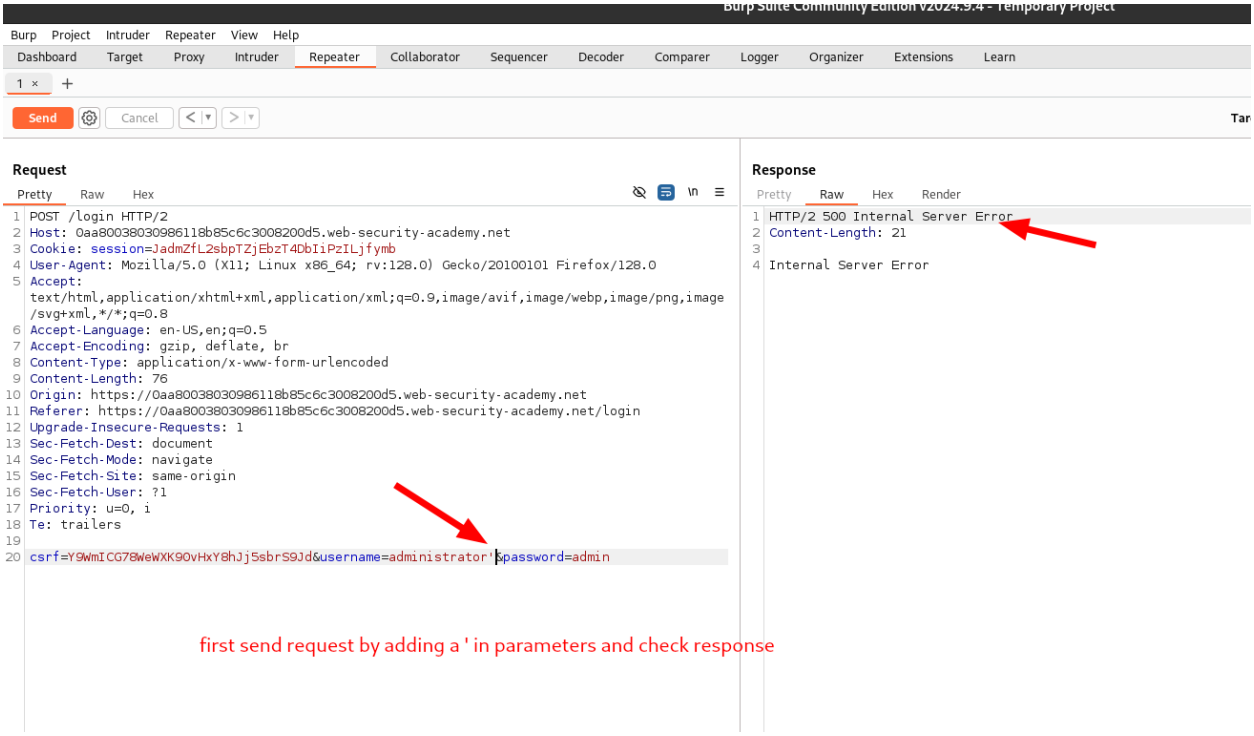
2. **Capture the Request in Burp**:
   - Submit the login form.
   - Burp Suite will intercept the HTTP request.

- 

send it To Repeater and analyze the Request body

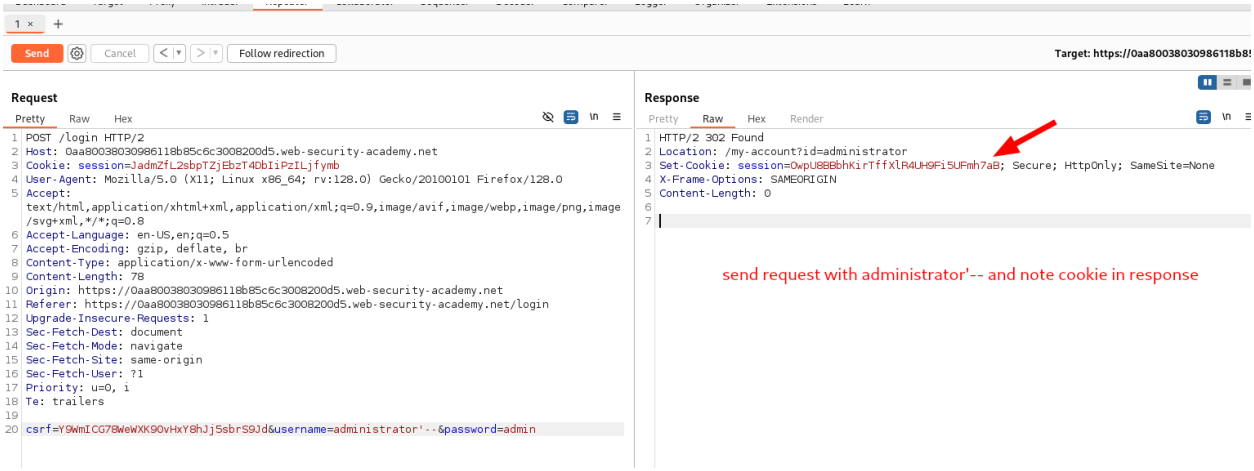
intercept the Request Parameters by adding ‘ at the end of parameter to check SQL injection option

if it gives Server error in Response then it is vulnerable for SQL injection .



After checking it now use a SQL Query attack to login as adminstrator

modify the administrator parameter in username option in Request Body :

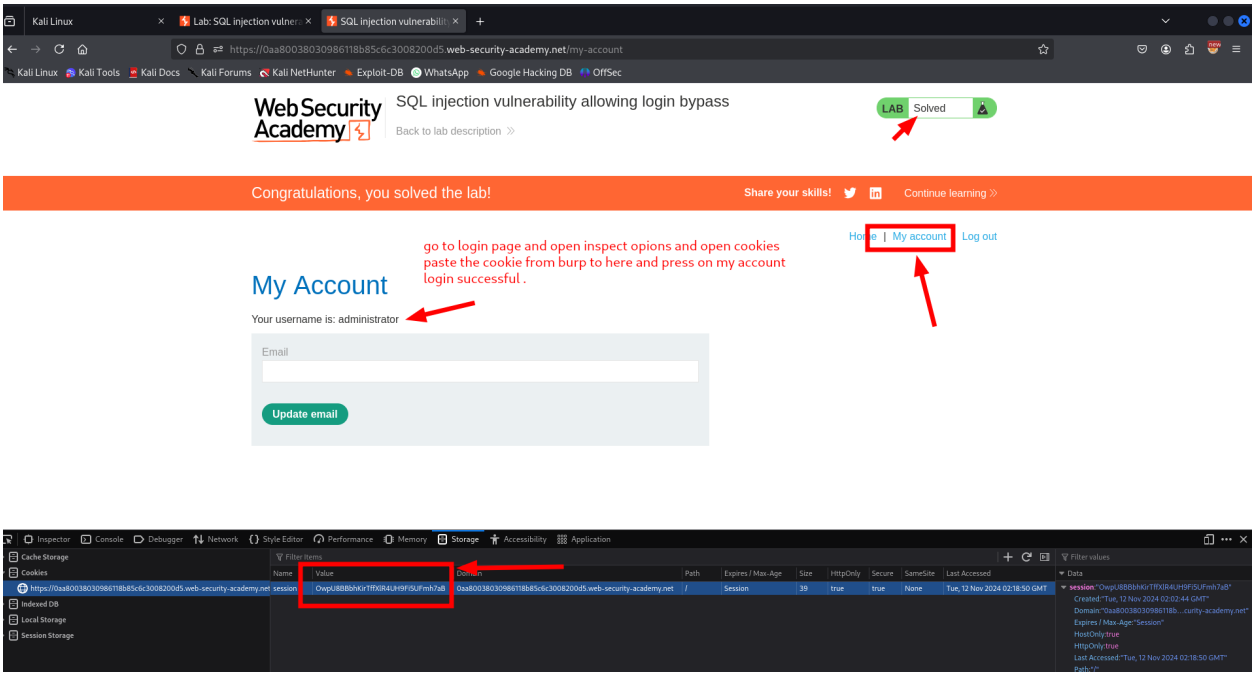


now after modifying the username parameter send Request

note Cookie session in the Response and copy it

go back to Browser and inspect the login page

go to cookie options and paste the cookie in it and Click on my Account option



you will login as administrator and Lab solved ...

Lab - SQL Injection Vulnerability in WHERE Clause Allowing Retrieval of Hidden Data

Objective

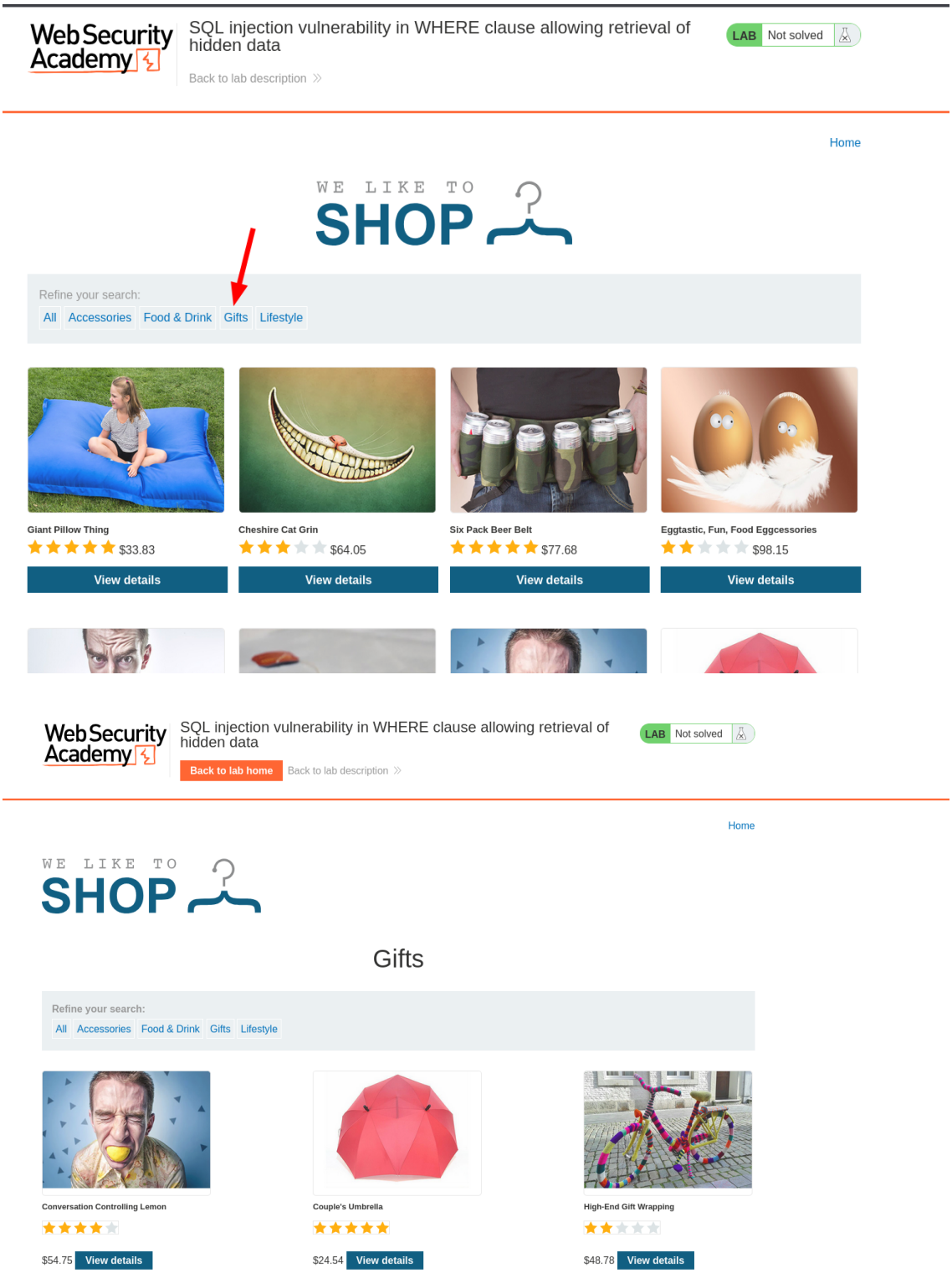
Exploit a SQL injection vulnerability in the product category filter to retrieve unreleased products.

1. Preparation

1. **Access the Lab**:
    - Open the lab URL in your browser.
    - Browse to the page containing the product category filter.
2. **Set Up Burp Suite**:
    - Launch **Burp Suite**.
    - Configure your browser to route traffic through Burp.

Analyze the Filter Request

1. **Select a Category**:
    - Choose any category from the filter (e.g., *Gifts*).
    - Observe the application reloads or updates the product list.

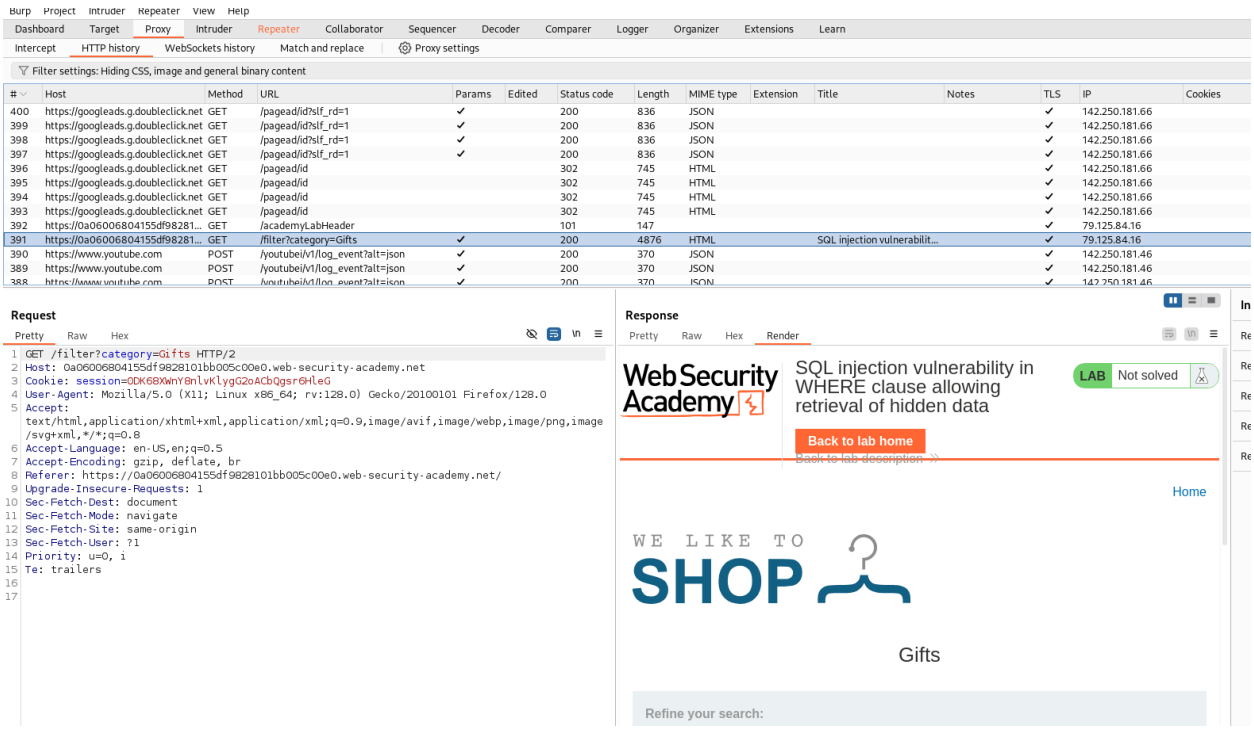


2. **Capture the Request in Burp**:

- Submit the filter request.

- Burp Suite will intercept the HTTP request.

send it to Repeater

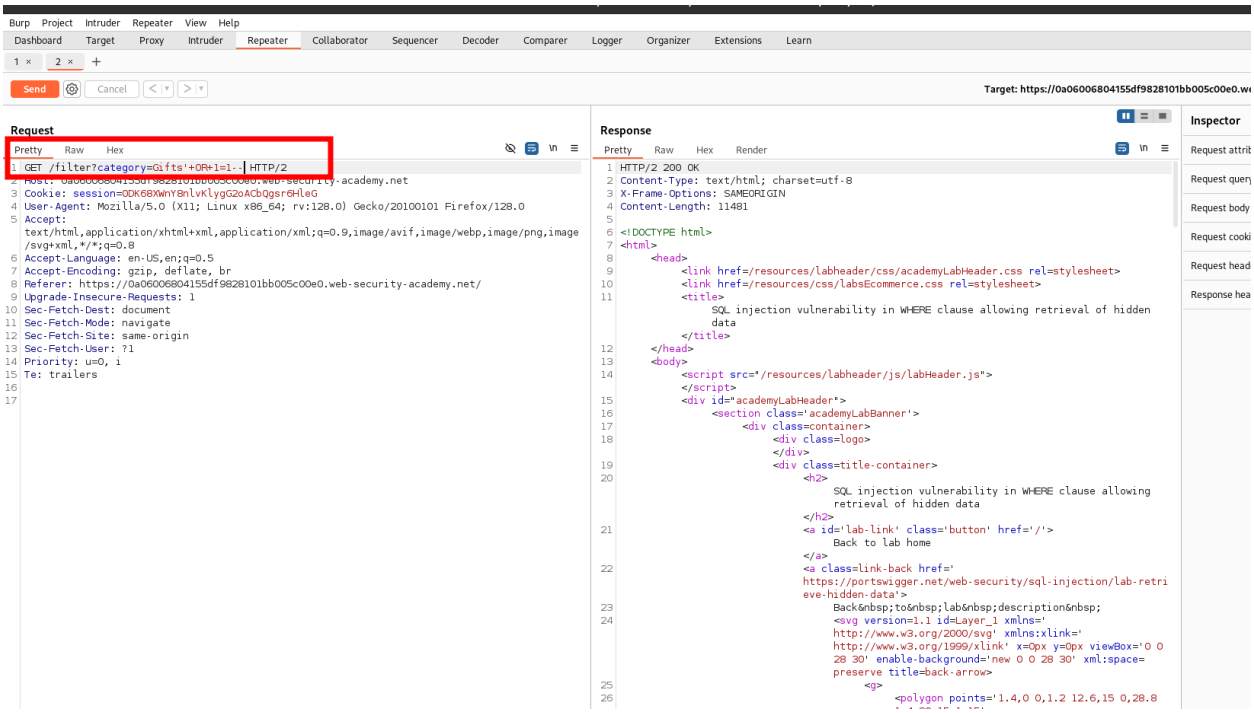


intercept the Request Parameters by adding ‘ at the end of parameter to check SQL injection option

if it gives Server error in Response then it is vulnerable for SQL injection .

After checking for the SQL injection now use SQL injection Query to show hidden data from database .

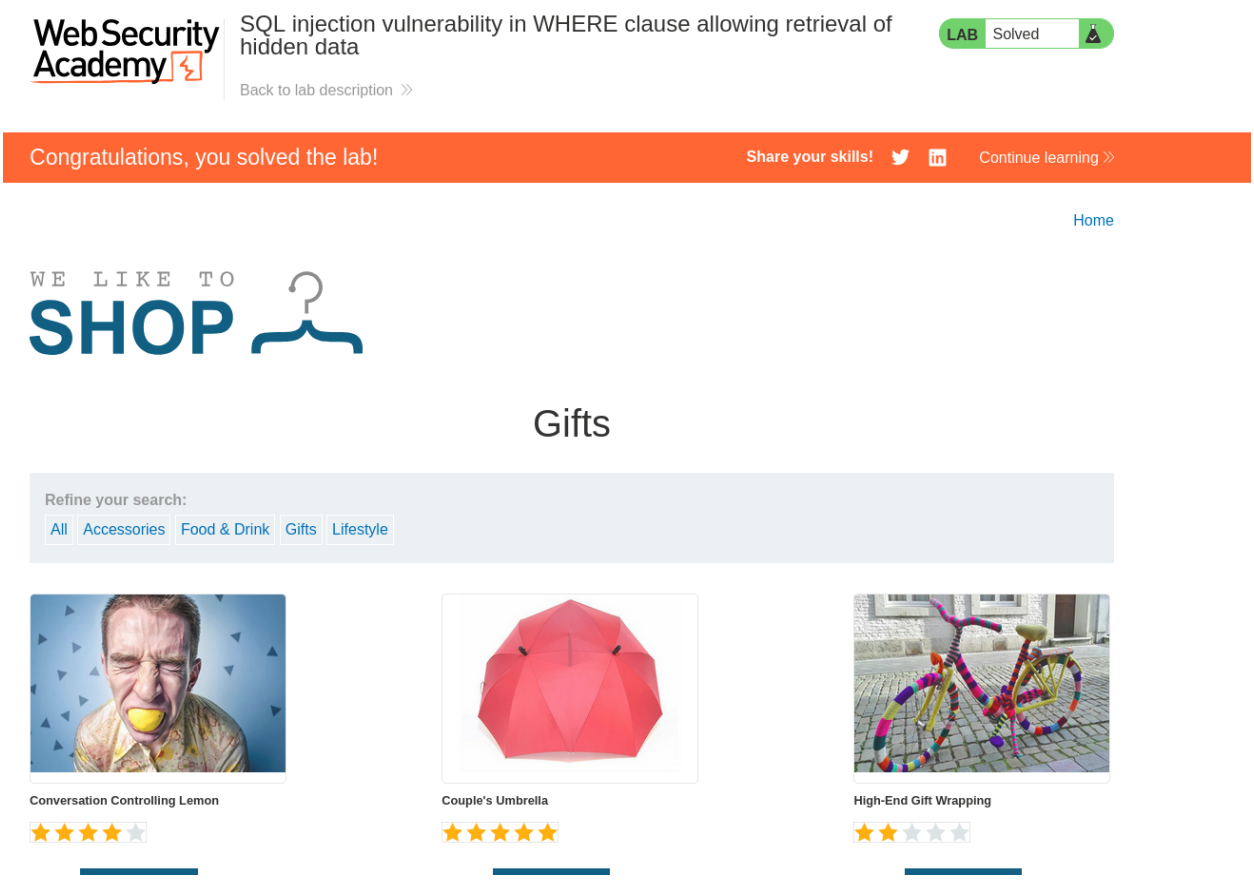
‘ OR 1=1- -



after entering this payload send Request and check Response and

open it in a browser and it will display hidden data and

Lab Solved !!



Lab - SQL Injection UNION Attack to Determine Number of Columns

Objective

Perform a SQL injection UNION attack to determine the number of columns in the query by injecting null values. This is a foundational step to retrieve data from other tables in subsequent labs.

. Preparation

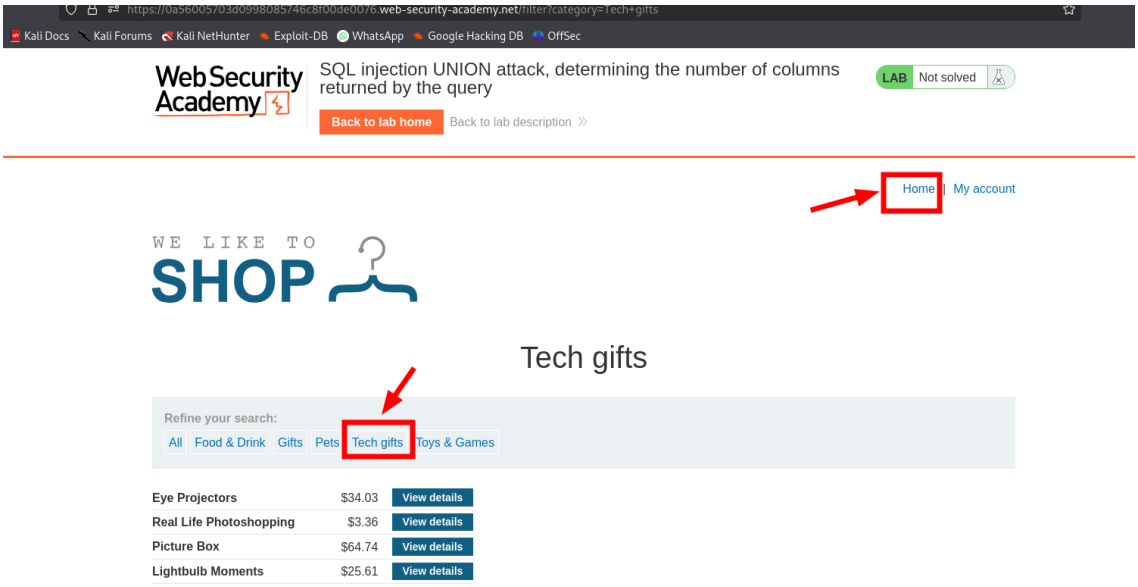
1. **Access the Lab**:
    - Open the lab URL in your browser.

- Browse to the page containing the product category filter.

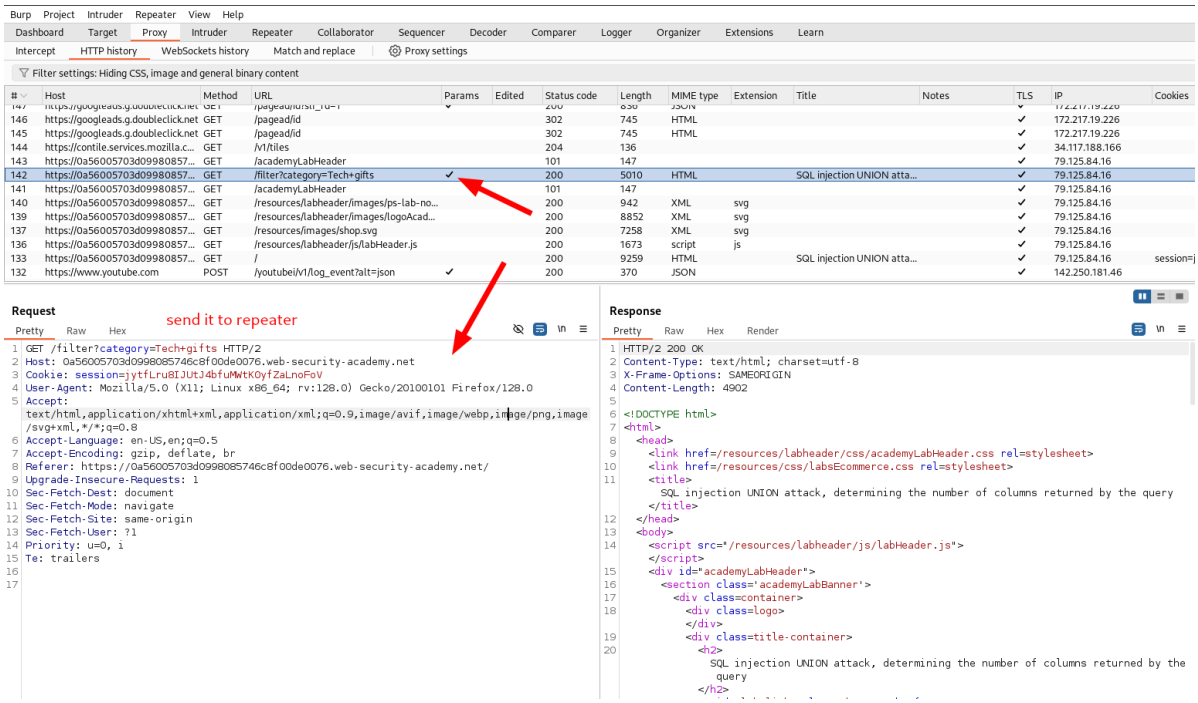
2. **Set Up Burp Suite**:
   - Launch **Burp Suite**.
   - Configure your browser to route traffic through Burp.

2. Analyze the Filter Request

1. **Select a Category**:
   - Choose any category from the filter (e.g., *Gifts*).
   - Observe the application reloads or updates the product list.



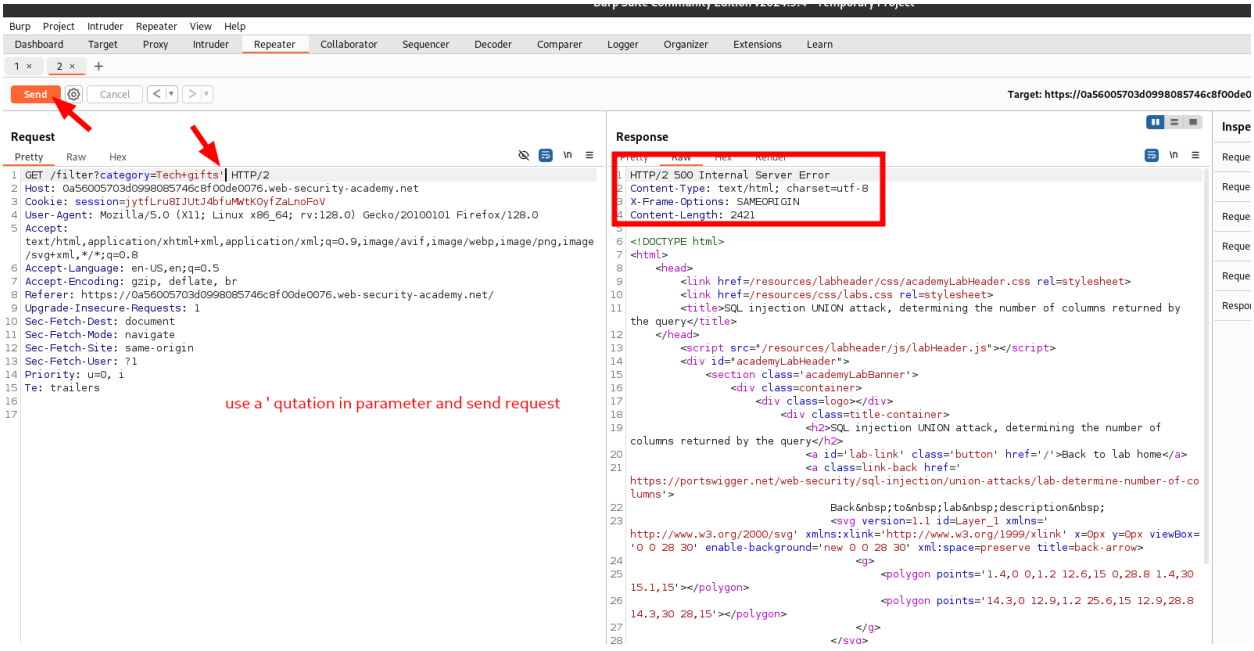
- **Capture the Request in Burp**:
  - Submit the filter request.
  - Burp Suite will intercept the HTTP request.

- 

send it to Repeater

in Repeater note Request body closely and add ‘ (single qutation mark) at the end of

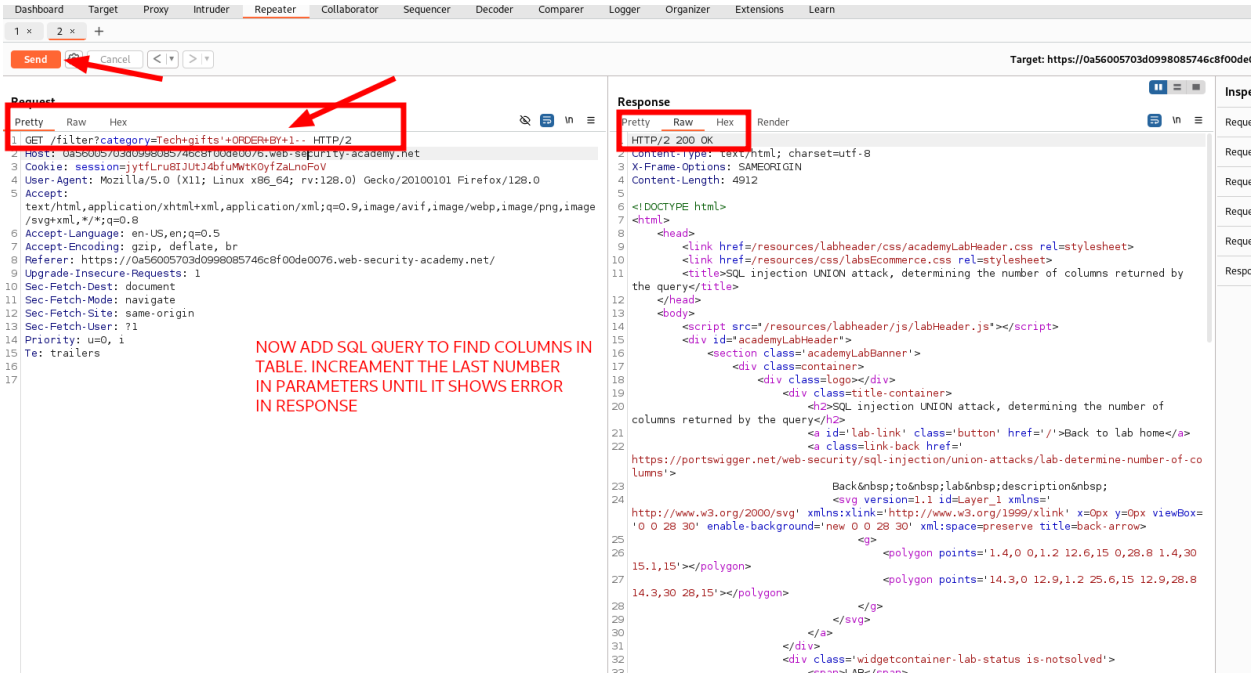
Request Search Parameter and send Request and note the Response to Check it for SQL injection .



Perform SQL Injection to Determine Column Count:

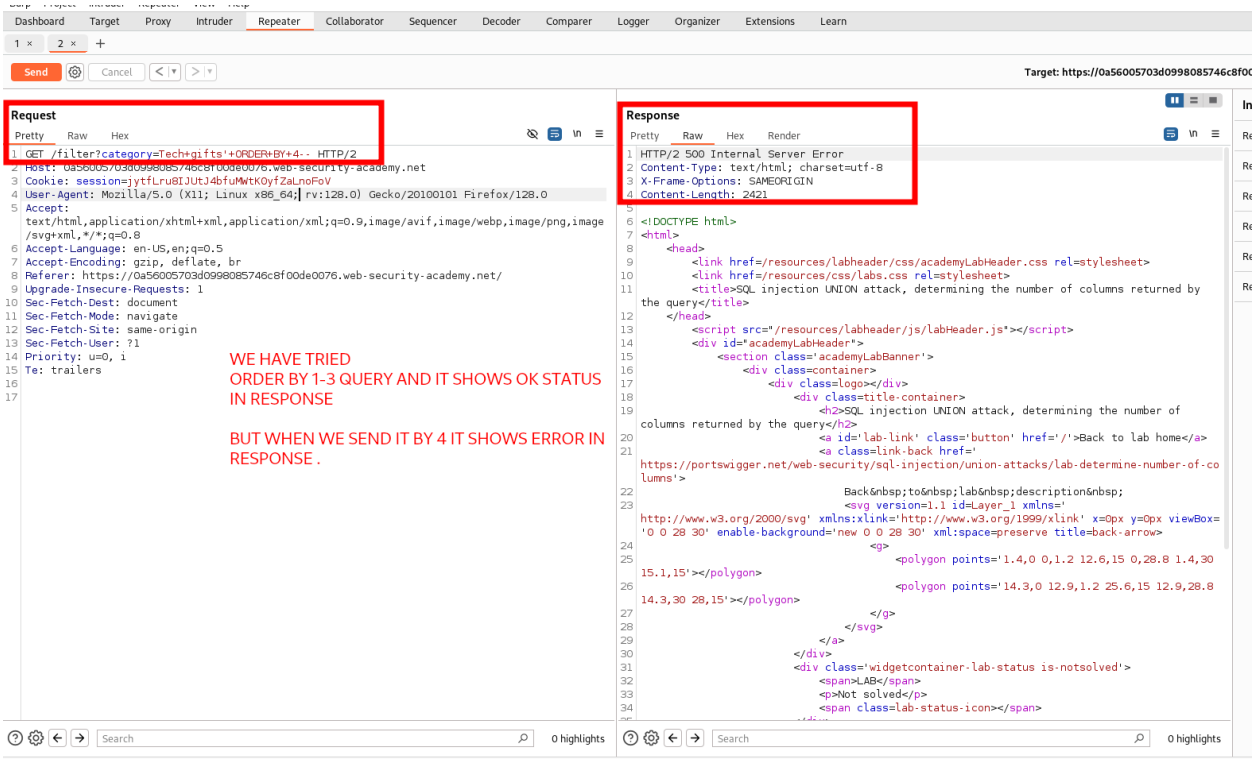
add SQL Query : ‘ ORDER BY 1- - to find out Columns in Table

, note : increase 1 until it shows Error in Response



we have Tried ORDER BY 1-3 Query and it Shows OK status in Response

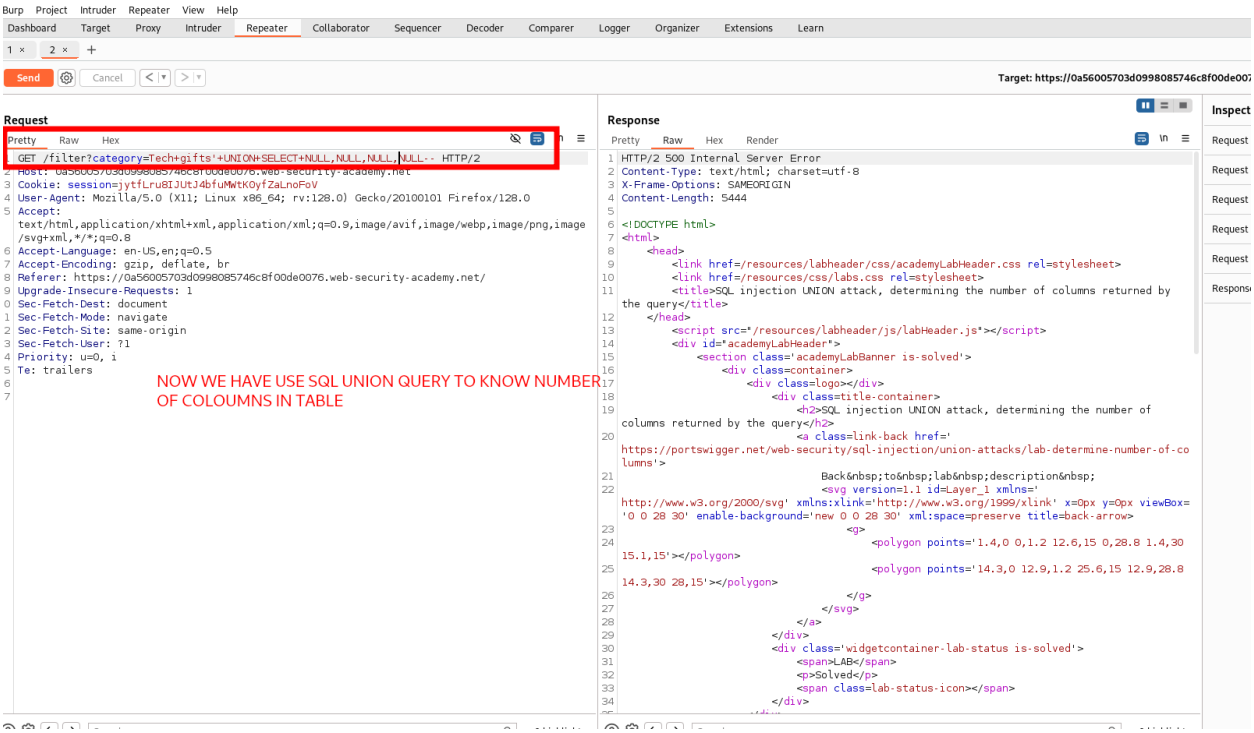
but When we send it by 4 it shows error in Response .



Now we use SQL UNION Injection to check for columns in Table with NULL parameter

SQL UNION INJECTION :

‘ UNION SELECT NULL,NULL,NULL- -


Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send Cancel
Request
Pretty Raw Hex
GET /filter?category=Tech+gifts'+UNION+SELECT+NULL,NULL,NULL,NULL-- HTTP/2
Host: 0a56005703d0998085746c8f00de0076.web-security-academy.net
Cookie: session=jytfLru8IJUtJ4bfuMWtKOyfZaLnoFoV
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://0a56005703d0998085746c8f00de0076.web-security-academy.net/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers
NOW WE HAVE USE SQL UNION QUERY TO KNOW NUMBER OF COLOUMNS IN TABLE
Response
Pretty Raw Hex Render
HTTP/2 500 Internal Server Error
Content-Type: text/html; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 5444
<!DOCTYPE html>
<html>
<head>
<link href=/resources/labheader/css/academyLabHeader.css rel=stylesheet>
<link href=/resources/css/labs.css rel=stylesheet>
<title>SQL injection UNION attack, determining the number of columns returned by the query</title>
</head>
<script src="/resources/labheader/js/labHeader.js"></script>

<section class='academyLabBanner is-solved'>

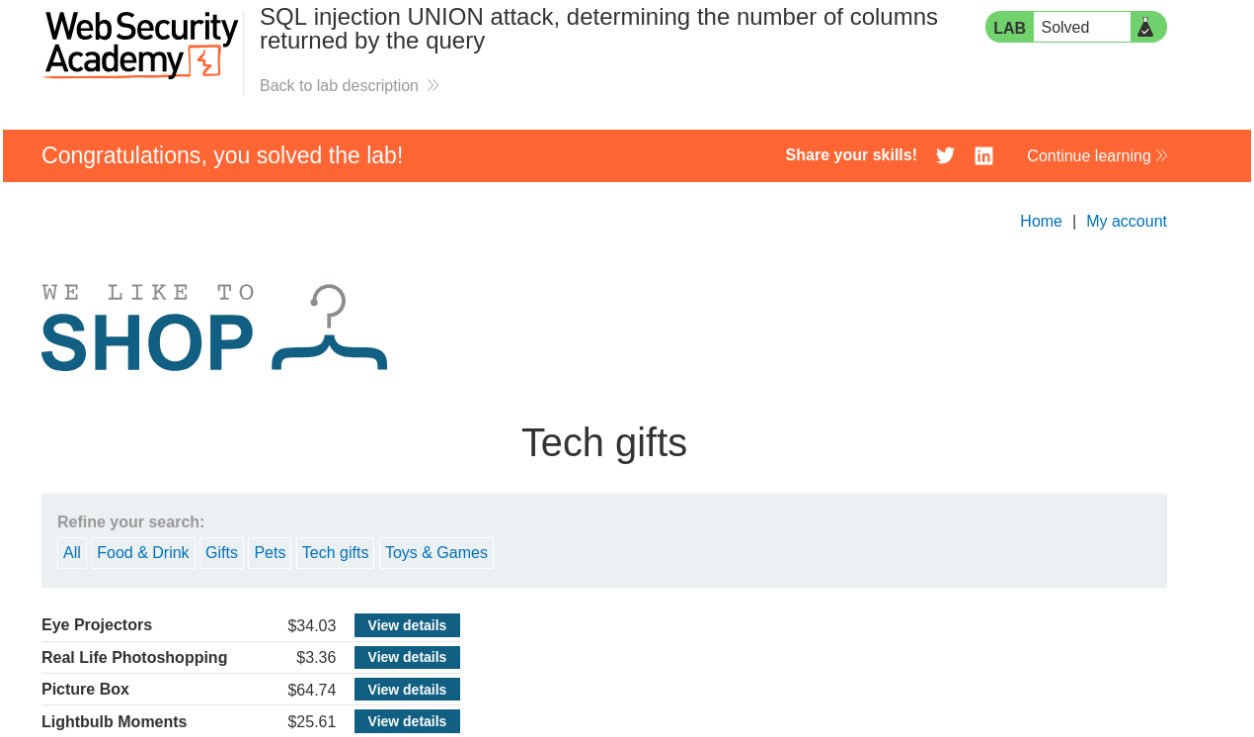
<h2>SQL injection UNION attack, determining the number of columns returned by the query</h2>

Back to lab description
<svg version=1.1 id=Layer_1 xmlns='http://www.w3.org/2000/svg' xmlns:xlink='http://www.w3.org/1999/xlink' x=0px y=0px viewBox='0 0 28 30' enable-background='new 0 0 28 30' xml:space=preserve title=back-arrow>
<g>
<polygon points='1.4,0 0,1.2 12.6,15 0,28.8 1.4,30 15.1,15'></polygon>
<polygon points='14.3,0 12.9,1.2 25.6,15 12.9,28.8 14.3,30 28,15'></polygon>
</g>
</svg>

LAB
Solved

Inspect
Request
Response

Lab Solved !!


Web Security Academy
SQL injection UNION attack, determining the number of columns returned by the query
LAB Solved
Back to lab description
Congratulations, you solved the lab!
Share your skills!
Continue learning
Home | My account
WE LIKE TO SHOP
Tech gifts
Refine your search:
All Food & Drink Gifts Pets Tech gifts Toys & Games
Eye Projectors $34.03 View details
Real Life Photoshopping $3.36 View details
Picture Box $64.74 View details
Lightbulb Moments $25.61 View details

Lab - SQL Injection UNION Attack to Find a Column Containing Text

Objective

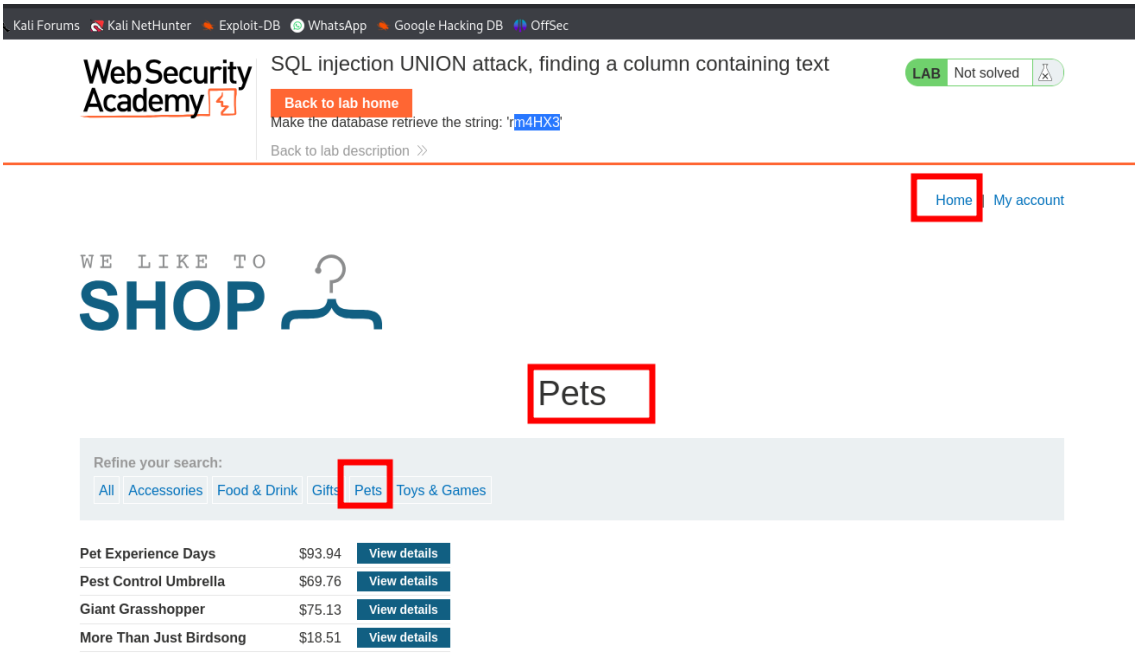
Perform a SQL injection UNION attack to identify a column compatible with string data by injecting a random value provided by the lab. This allows further exploitation, such as retrieving sensitive data.

1. Preparation

1. **Access the Lab**:
    - Open the lab URL in your browser.
    - Take note of the random string value provided by the lab (e.g., `abcdef`).
2. **Set Up Burp Suite**:
    - Launch **Burp Suite**.
    - Configure your browser to proxy traffic through Burp.

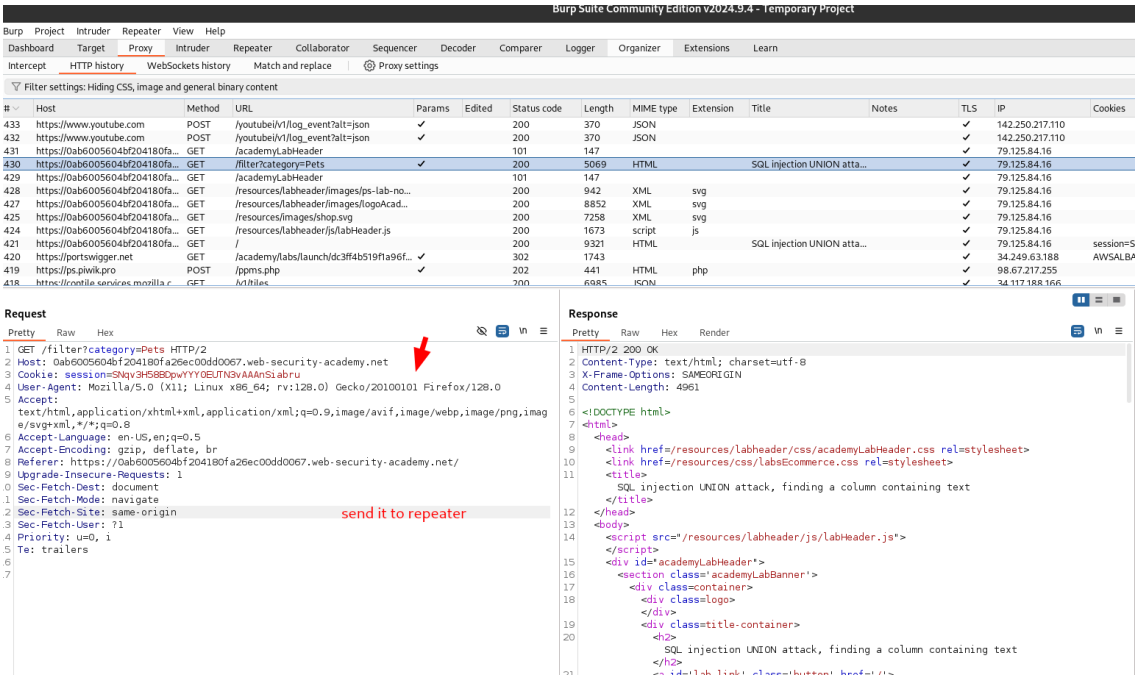
Analyze the Filter Request

1. **Select a Category**:
    - Choose any category from the product filter (e.g., Pets).
    - Observe the application reloading or updating the product list.



-

2. **Capture the Request in Burp**:
   - Submit the request for the selected category.
   - Burp Suite will intercept the HTTP request.



send it to REPEATER

in Repeater check the Request body and note the Request Search/GET parameters

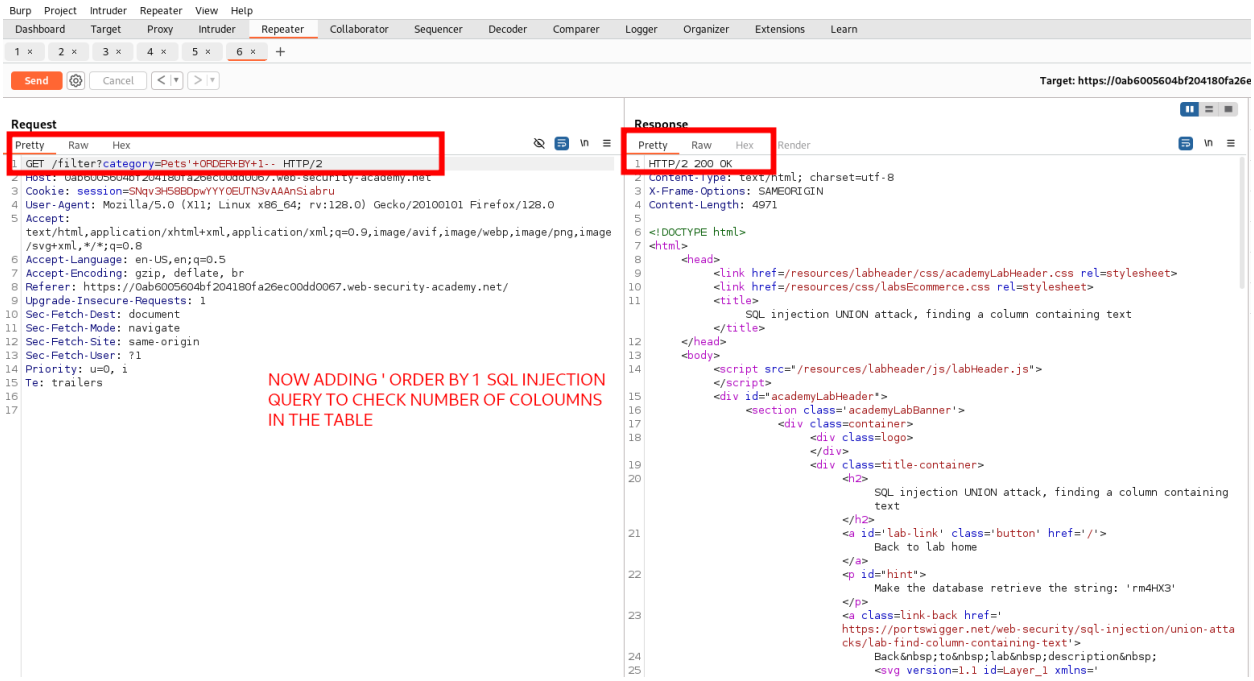
Add ‘ (single qutation mark) at the end of

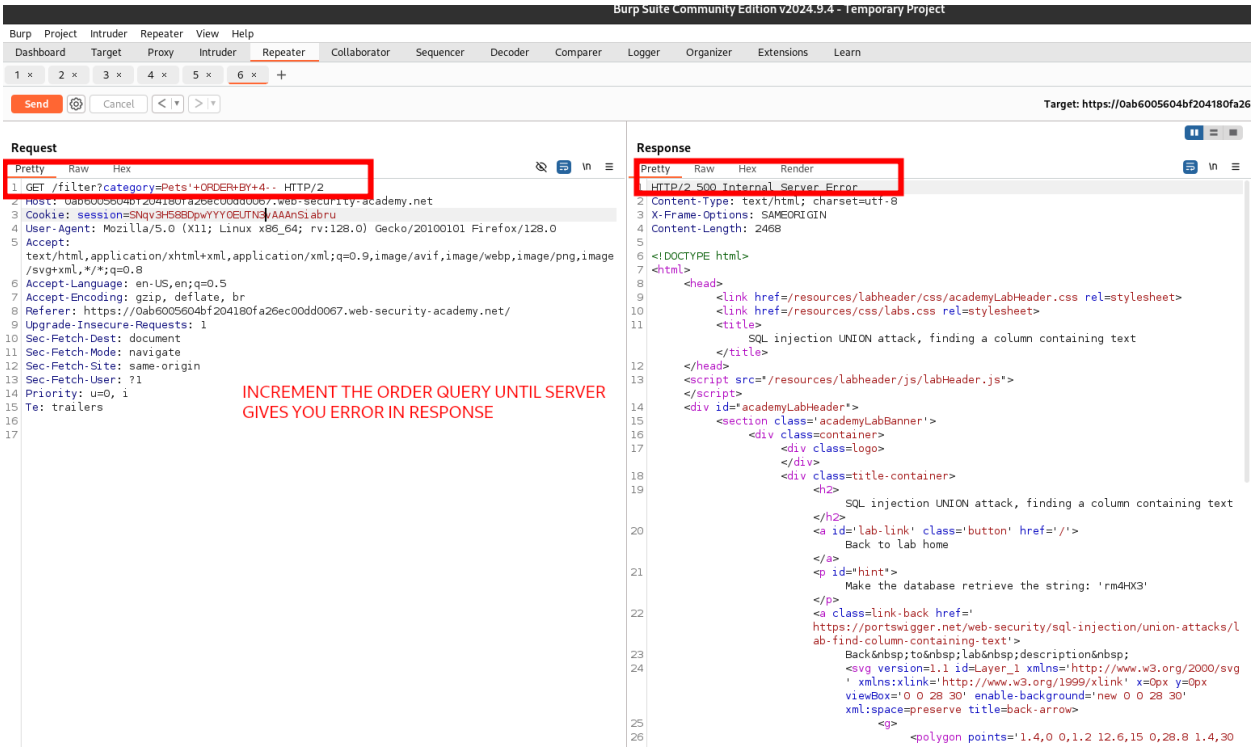
Request Search Parameter and send Request and note the Response to Check it for SQL injection

Perform SQL Injection to Determine Column Count:

add SQL Query : ‘ ORDER BY 1- - to find out Columns in Table

, note : increase 1 until it shows Error in Response

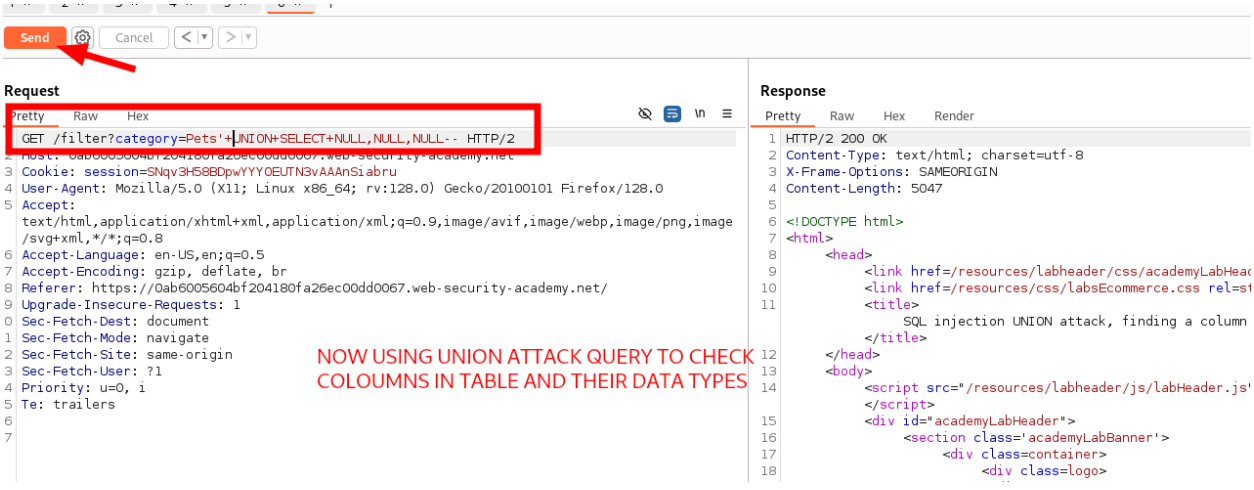




Now we use SQL UNION Injection to check for columns in Table with NULL parameter

SQL UNION INJECTION :

‘ UNION SELECT NULL,NULL,NULL- -



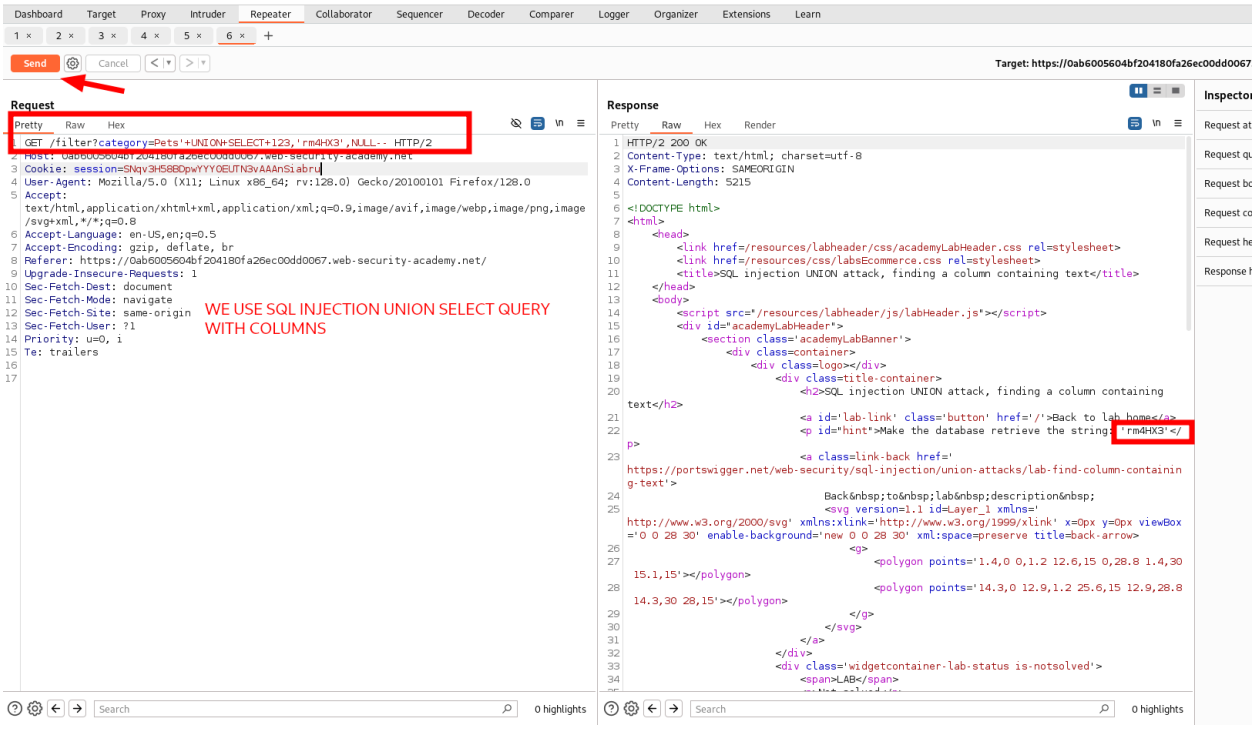
Identify a Column Compatible with String Data

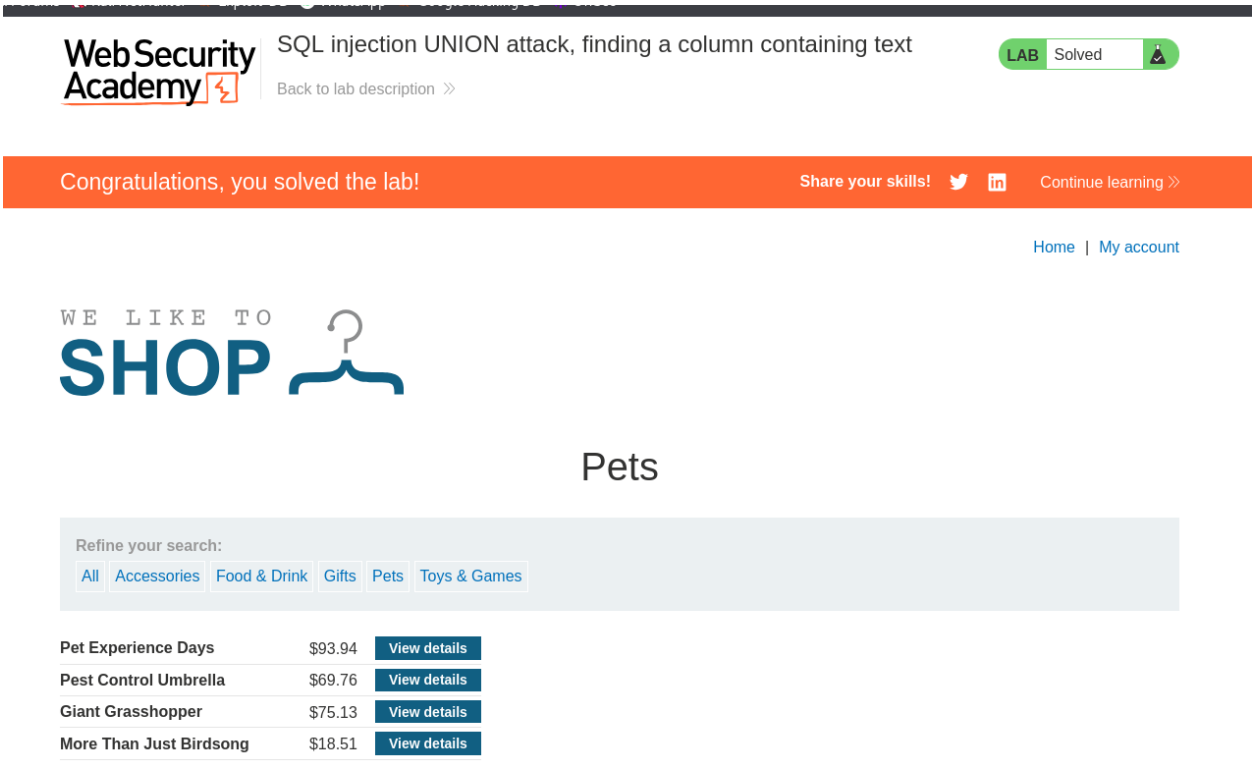
1. **Inject the Random String**:

- Replace one `NULL` value in the query with the provided string (e.g., `abcdef`):

Test Each Column:

- If the query returns an error or the string doesn't appear in the response, move the string to the next column and retry:
  - `'+UNION+SELECT+NULL,'abcdef',NULL--`
  - `'+UNION+SELECT+NULL,NULL,'abcdef'--`





Verify the Exploit:

- Ensure the success message appears, indicating the lab is complete.

Lab - SQL Injection UNION Attack, Retrieving Data from Other Tables

Objective

Exploit a SQL injection vulnerability in the product category filter to retrieve data from other tables. This lab teaches how to extract sensitive information (e.g., usernames and passwords) using a SQL injection UNION attack.

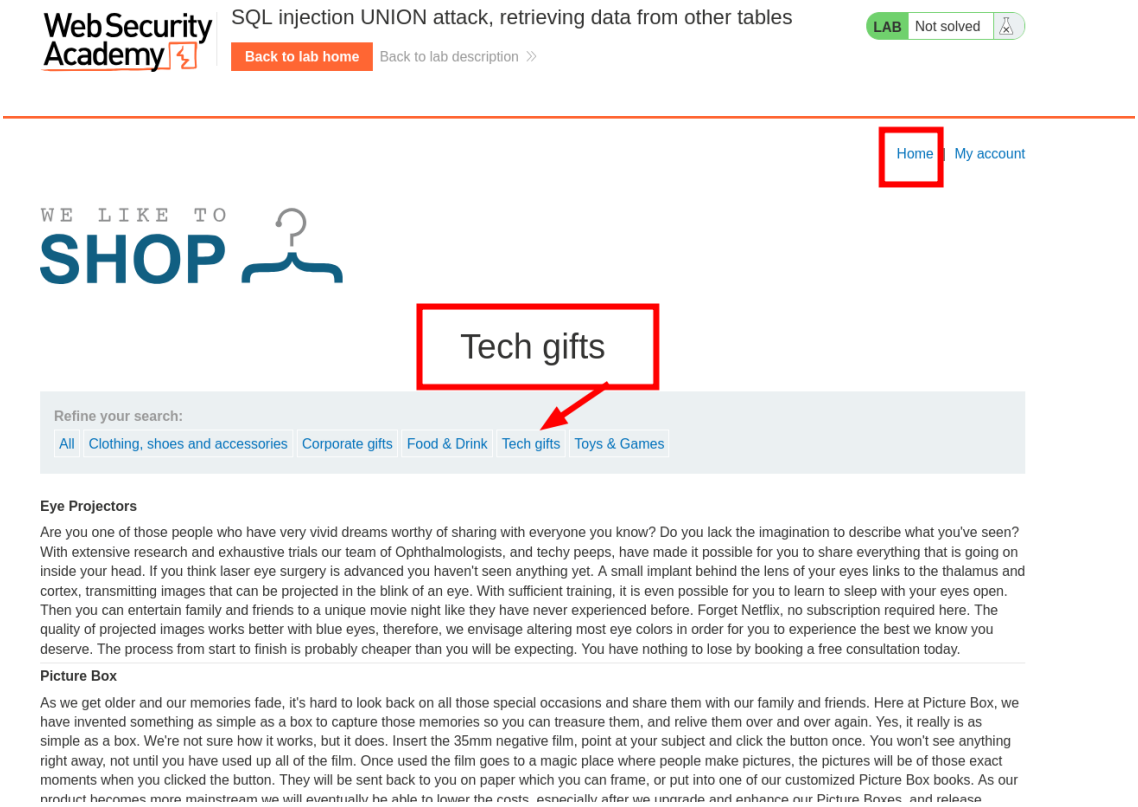
Preparation

1. **Access the Lab**:

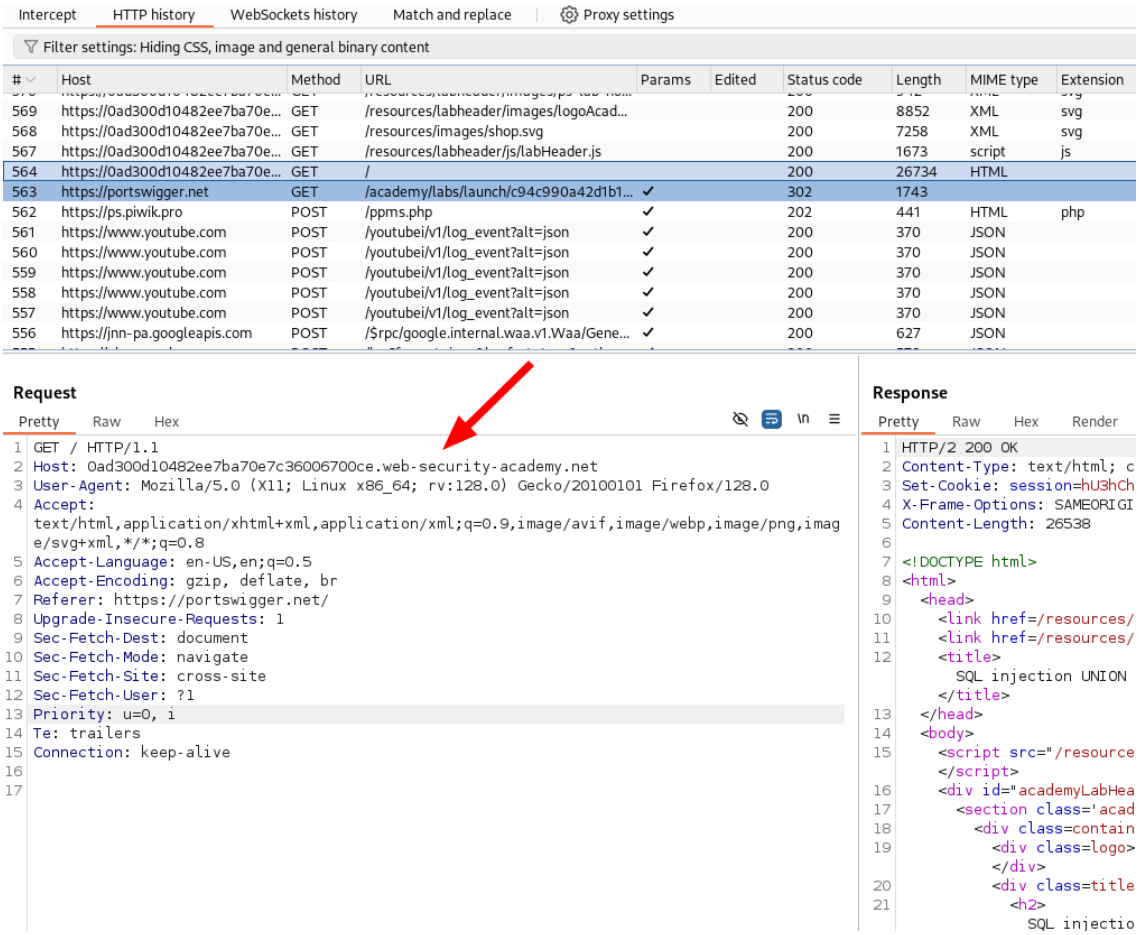
- Open the lab link in your browser.

2. **Set Up Burp Suite**:
   - Launch **Burp Suite**.
   - Configure your browser to proxy traffic through Burp.

Analyze the Request

1. **Select a Category**:
   - Choose any product category from the filter (e.g., TECH Gift).
   - Observe the application loading or updating the product list.
   - 

2. **Intercept the Request**:
   - Capture the HTTP request for the selected category using Burp Suite.

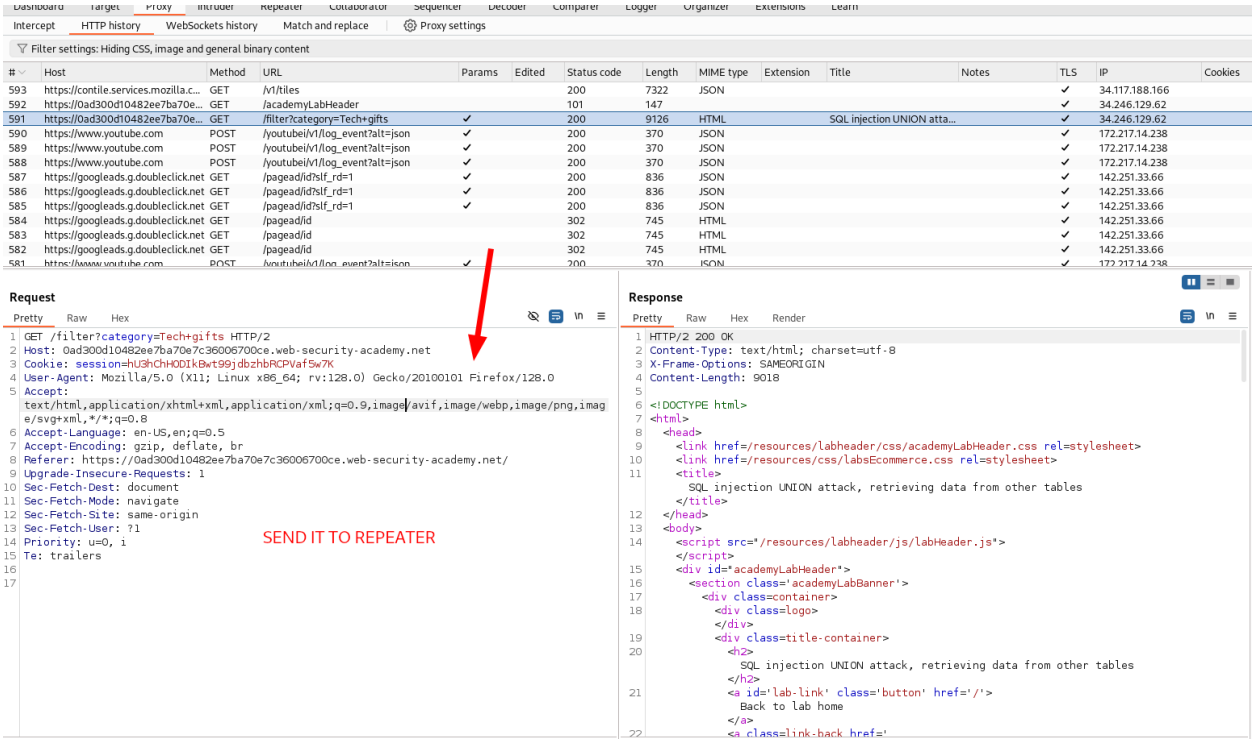


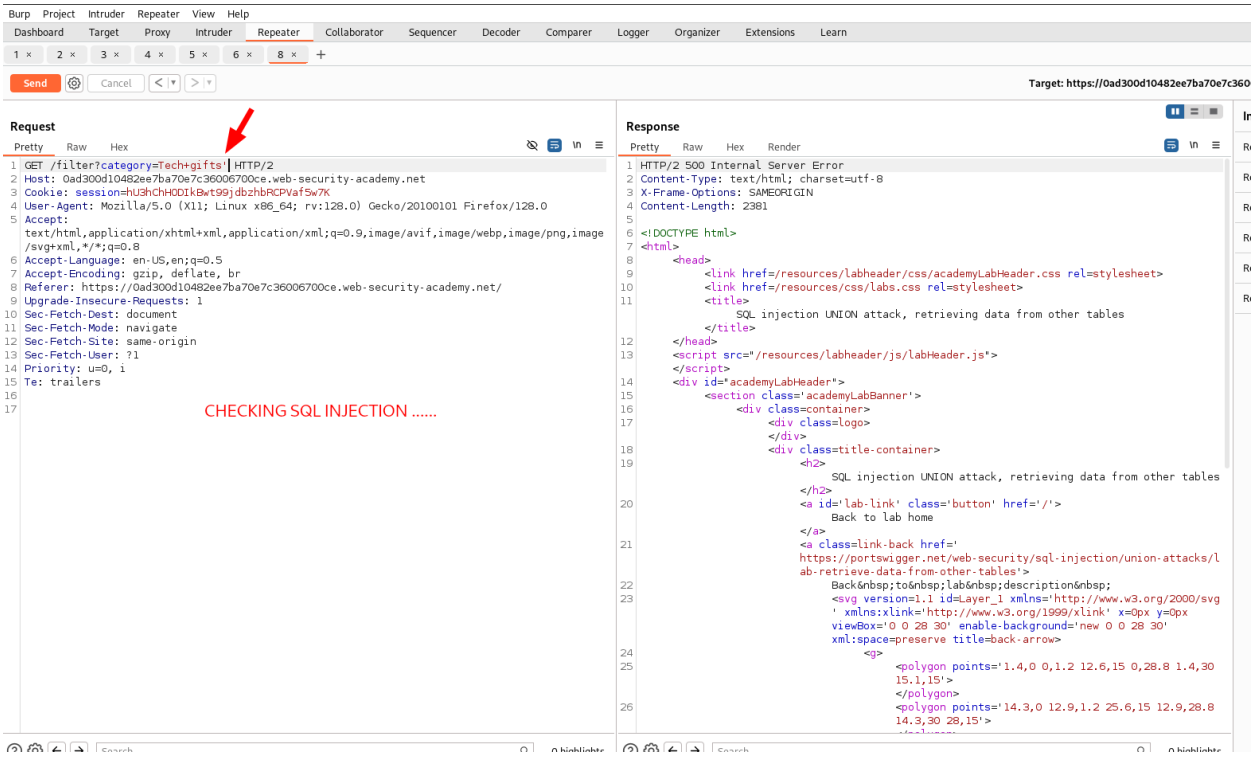
- Send it to Repeater and note Request body

- Look for the `category` parameter in Repeater Request body and

- Add ‘ (single qutation mark) at the end of

- Request Search Parameter and send Request and note the Response to Check it for SQL injection

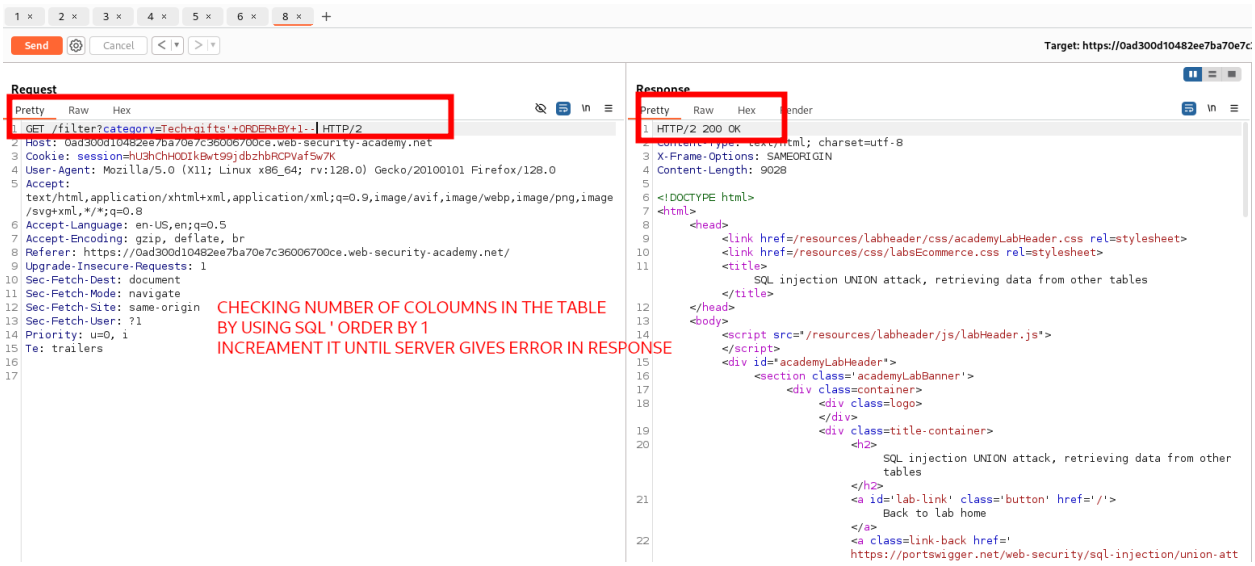


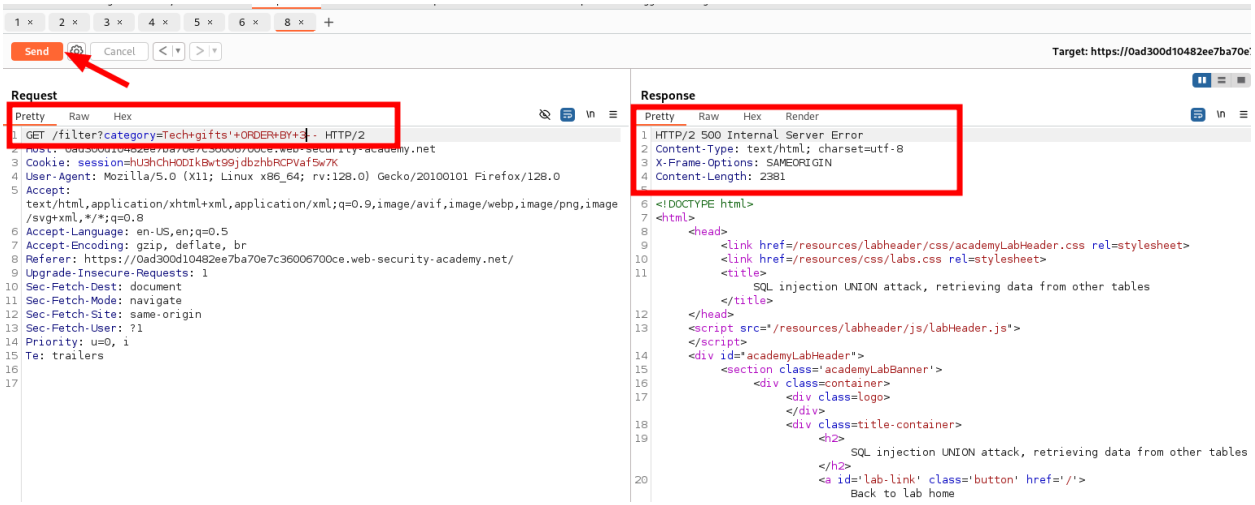


Perform SQL Injection to Determine Column Count:

add SQL Query : ‘ ORDER BY 1- - to find out Columns in Table

, note : increase 1 until it shows Error in Response



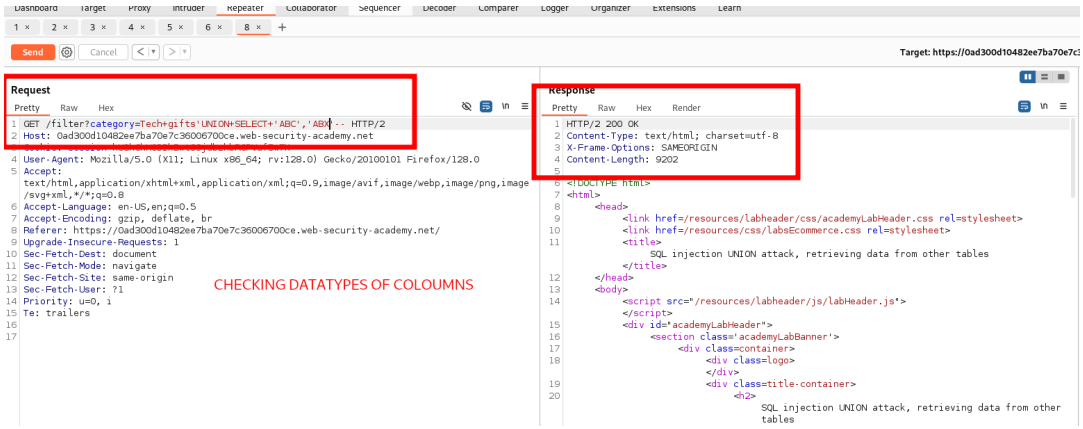


Identify a Columns Data Type

1. **Inject the Random String**:
   - Replace one `NULL` value in the query with the provided string (e.g., `abcdef`):

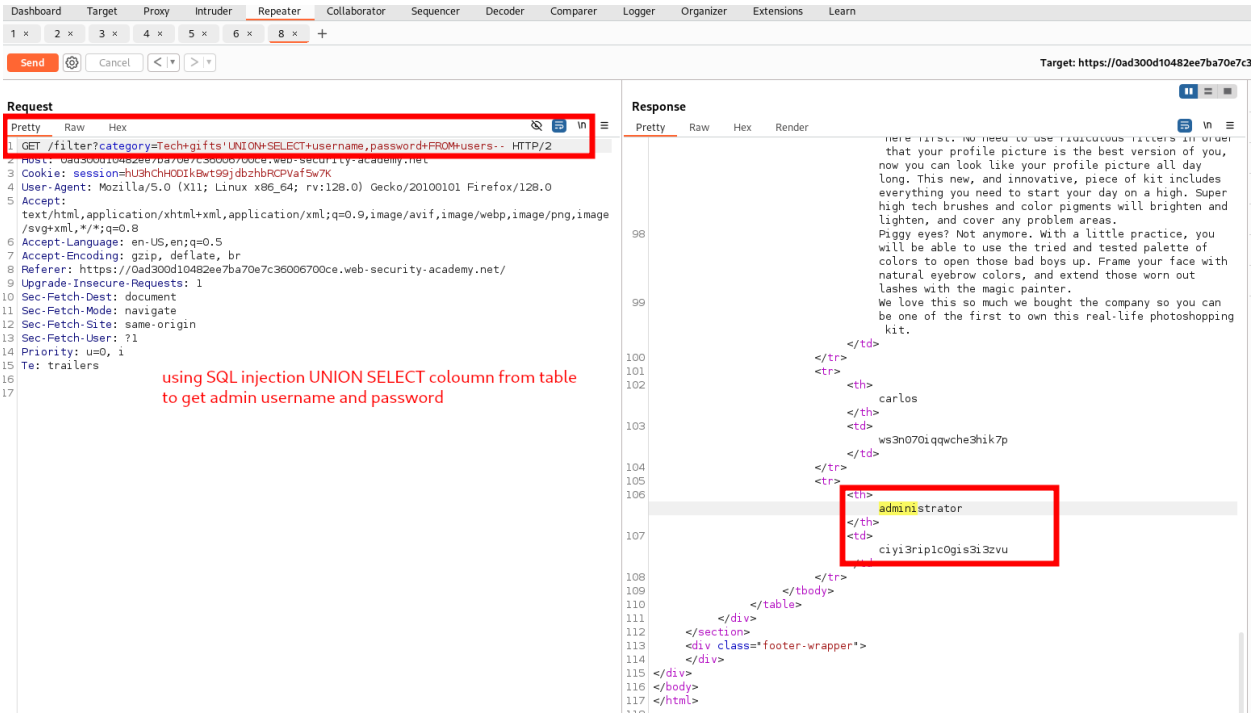
   Test Each Column:
   - If the query returns an error or the string doesn’t appear in the response, move the string to the next column and retry:

- `'+UNION+SELECT+NULL,'abcdef',NULL--`
- `'+UNION+SELECT+NULL,NULL,'abcdef'--`



- **dentify Relevant Tables**:
  - Look for table names related to users, e.g., `users`.
- **Extract User Data**:
  - Query the `users` table to extract usernames and passwords. For example:

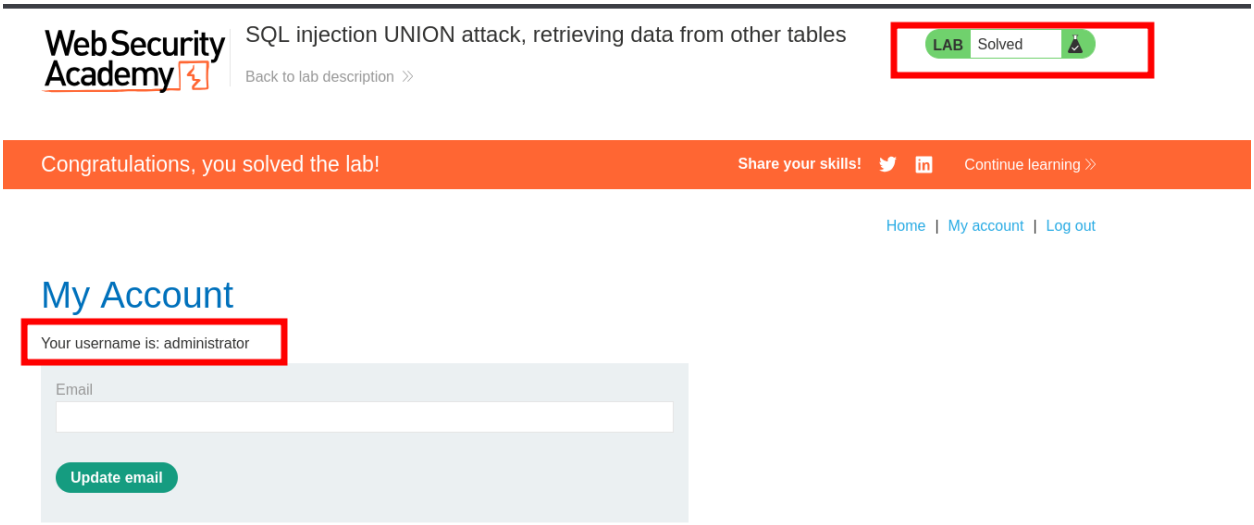
'+UNION+SELECT+username,password,NULL+FROM+users--



check Response and find out Administrator and its Password

copy the user name Administrator and Password from Response

go Back to Lab in Browser and try Login in Administrator account with username and Password



Lab Solved !!

Lab - SQL Injection UNION Attack, Retrieving Multiple Values in a Single Column

Objective

Exploit a SQL injection vulnerability in the product category filter to retrieve multiple pieces of data in a single column. This attack targets an application where the results of a query are returned in a single column of the application's response.