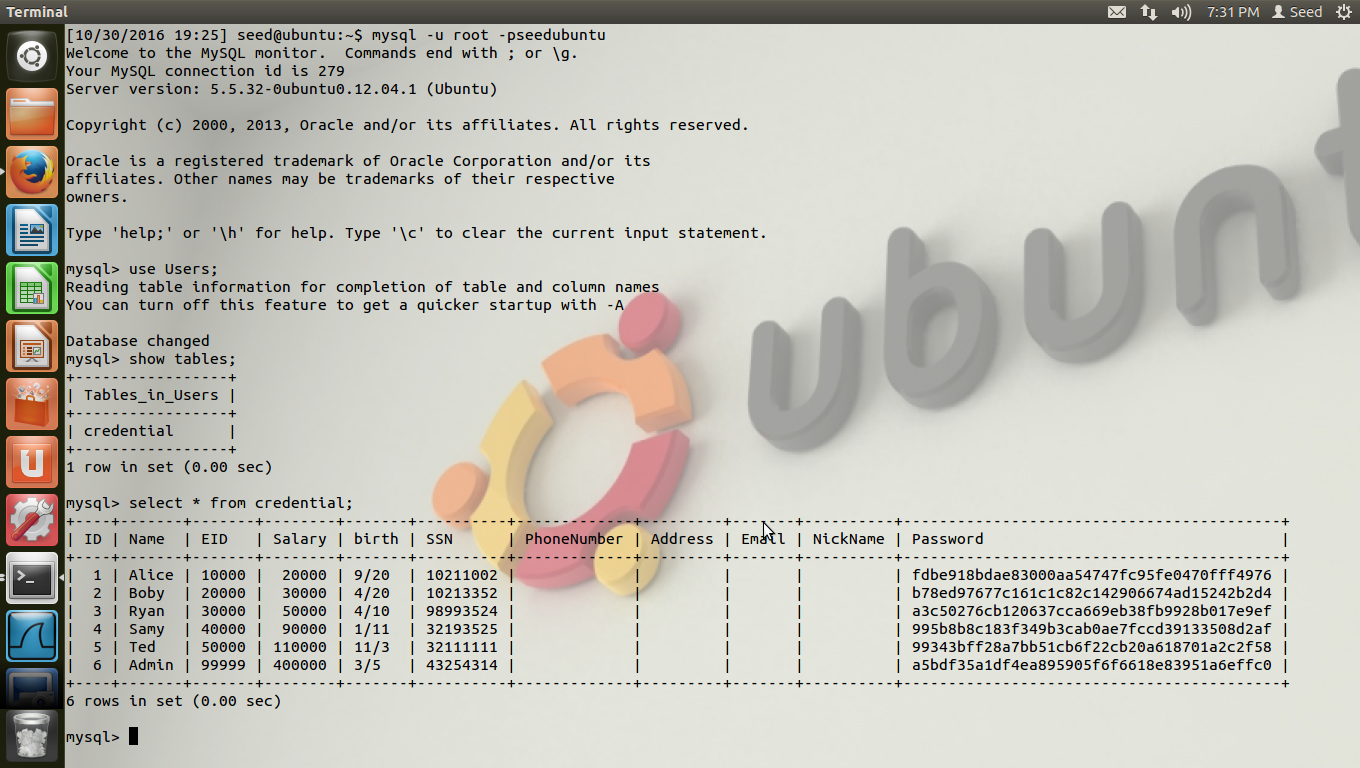
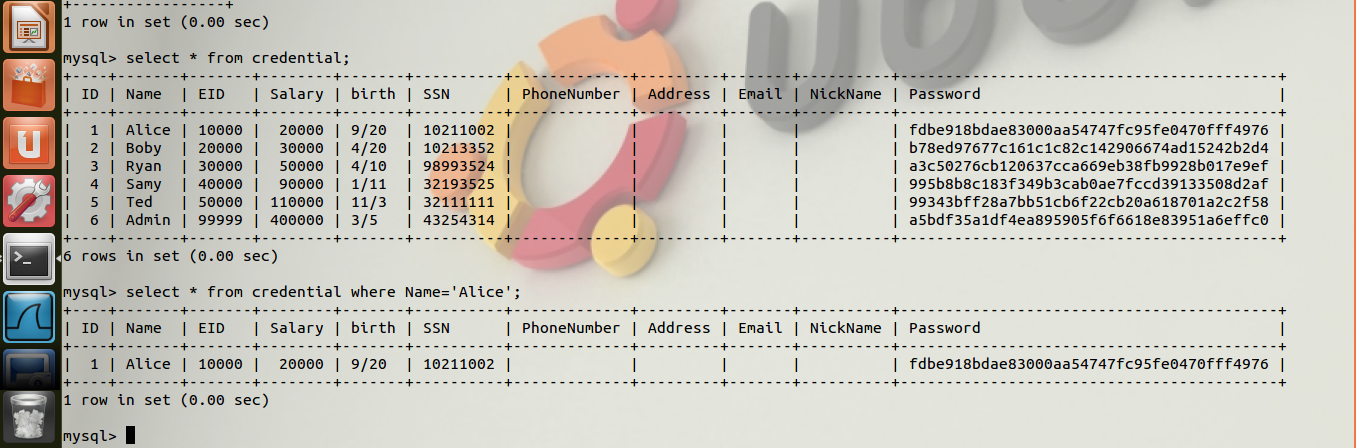
**SQL Injection Assignment**

I could login to the VM Ubuntu server and Access the website [www.SeedLabssqlInjection](http://www.SeedLabssqlInjection).

Below is the task that are being given the Guidelines. Attaching screenshots for each task

**Task 1: SQL details**

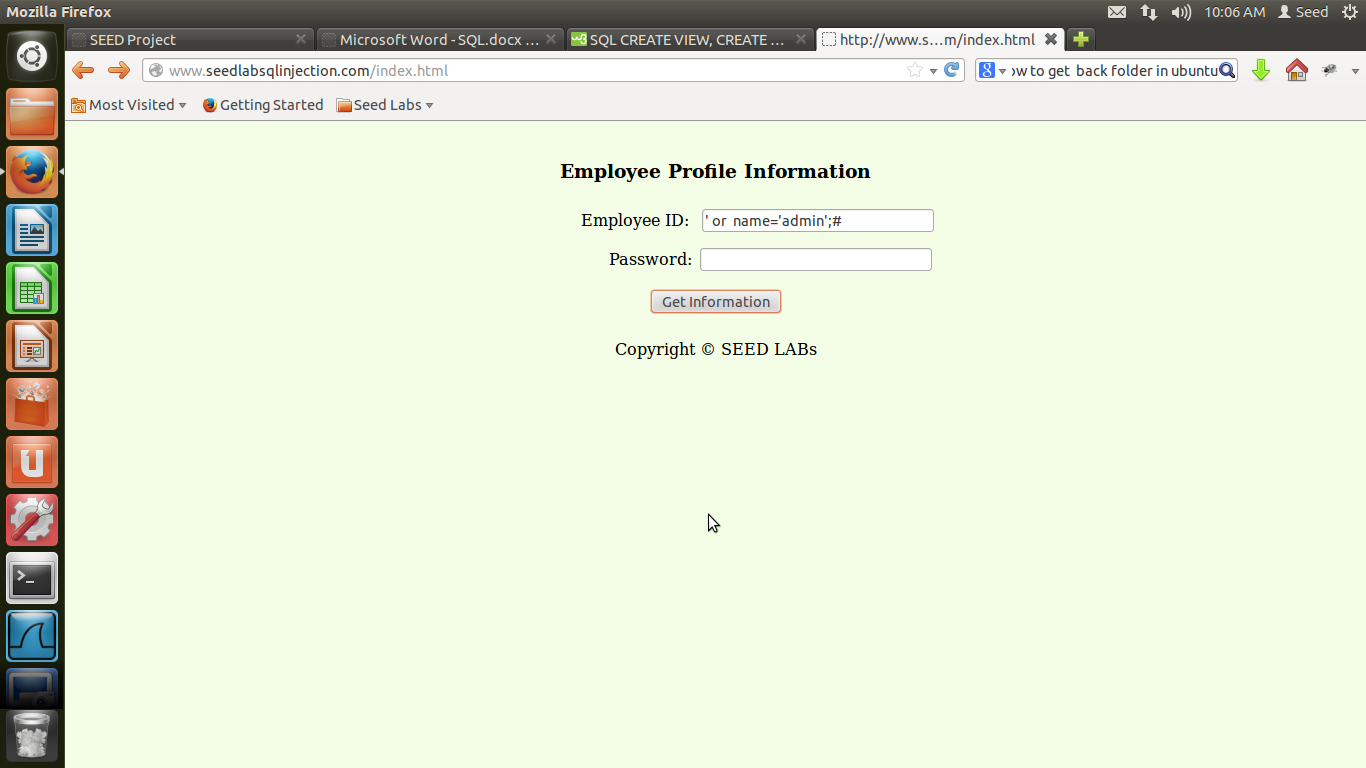


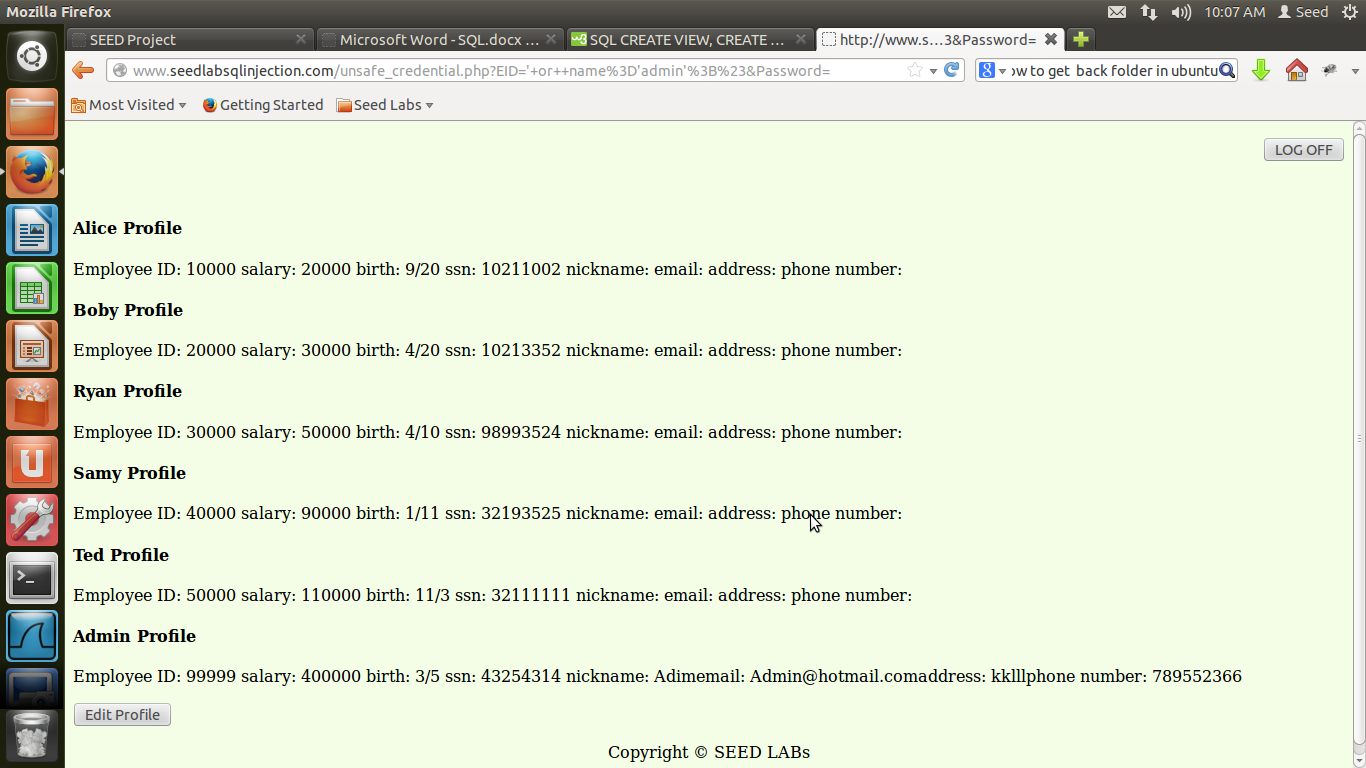


The above to screenshots are displaying the MySQL data that is being queried.

**Task 2.1: SQL injection Attack for webpage, Access for Admin access.**

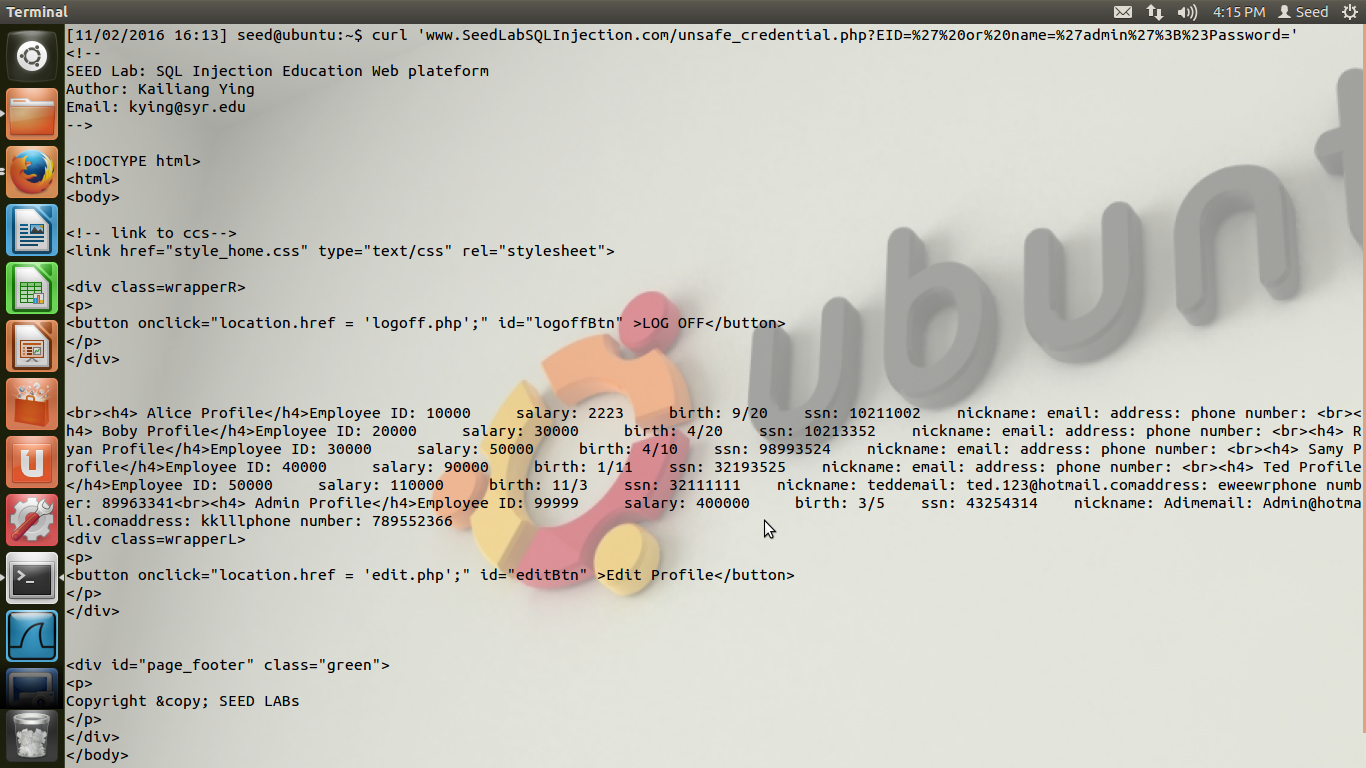
The Idea of accessing data being admin, is taken from the guidelines and the below screenshots describes it





**Task 2.2 Curl Injection**

**Observation:** Curl was not installing, initially need to update completely the drivers and file present in Ubuntu (sudo apt-get updates), which will update all the files/directories and the use apt-get install curl to install the curl.



**Task 2.3 To bypass the data in login page and modify the statement with the second SQL**

I tried out many combinations to bypass and join the 2 SQL statements, but was unable to completely crack it. Below is an SQL query that I was finally conclude too and still was not able to derive the output:

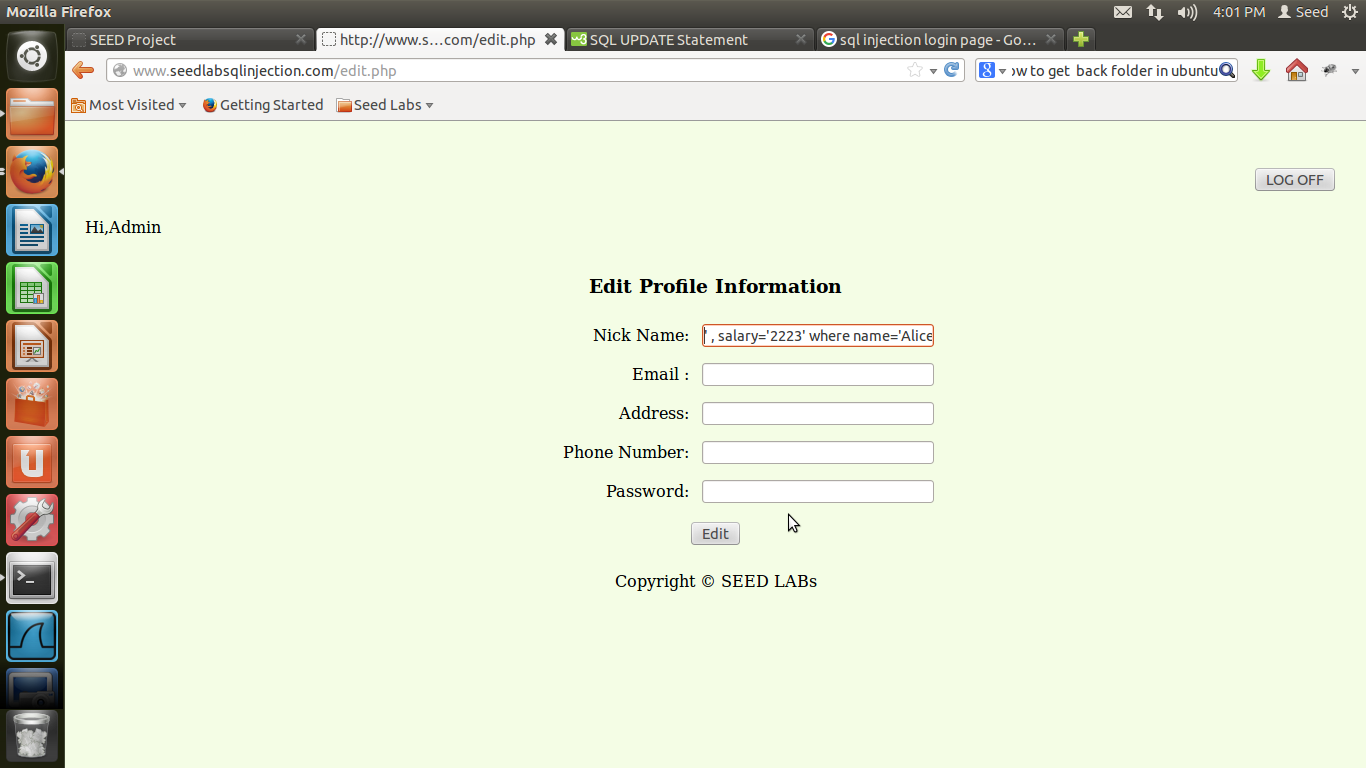
' or name='admin'; ' or salary='5555' where name='Alice';#

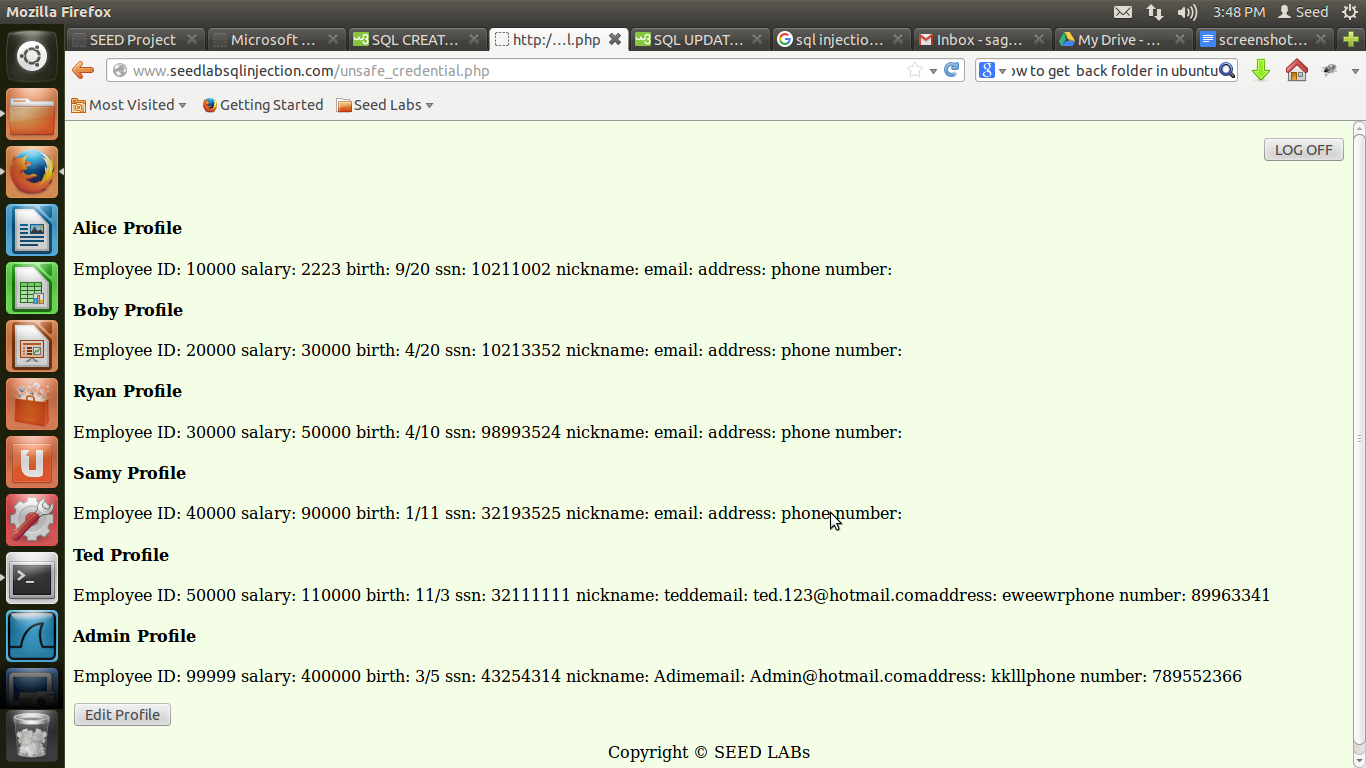
**Task 3 .1 SQL Injection Update statement- Salary**

Query for updating Salary:

The Admin has the right to change the salary structure of an employee and the SQL injection query was passed in edit part and is shown below.

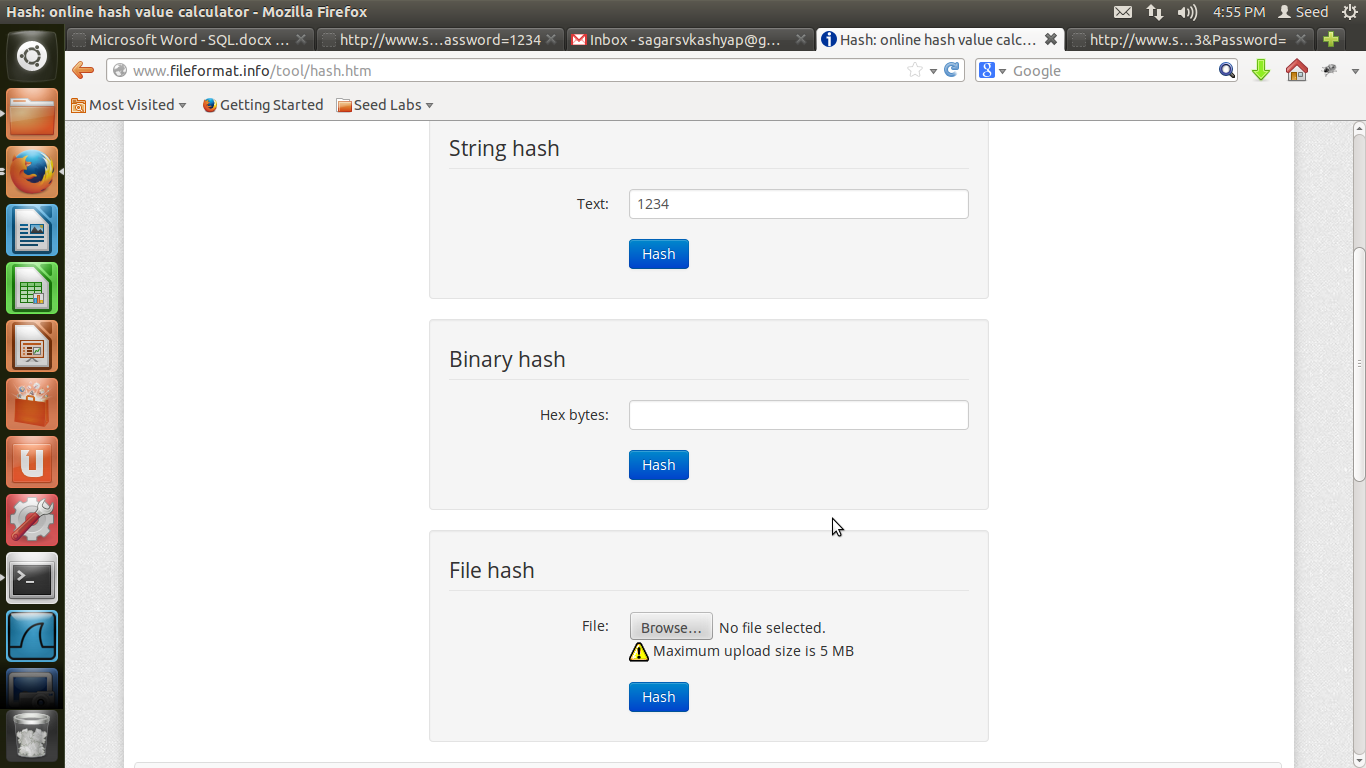
' , salary='2223' where name='Alice';#

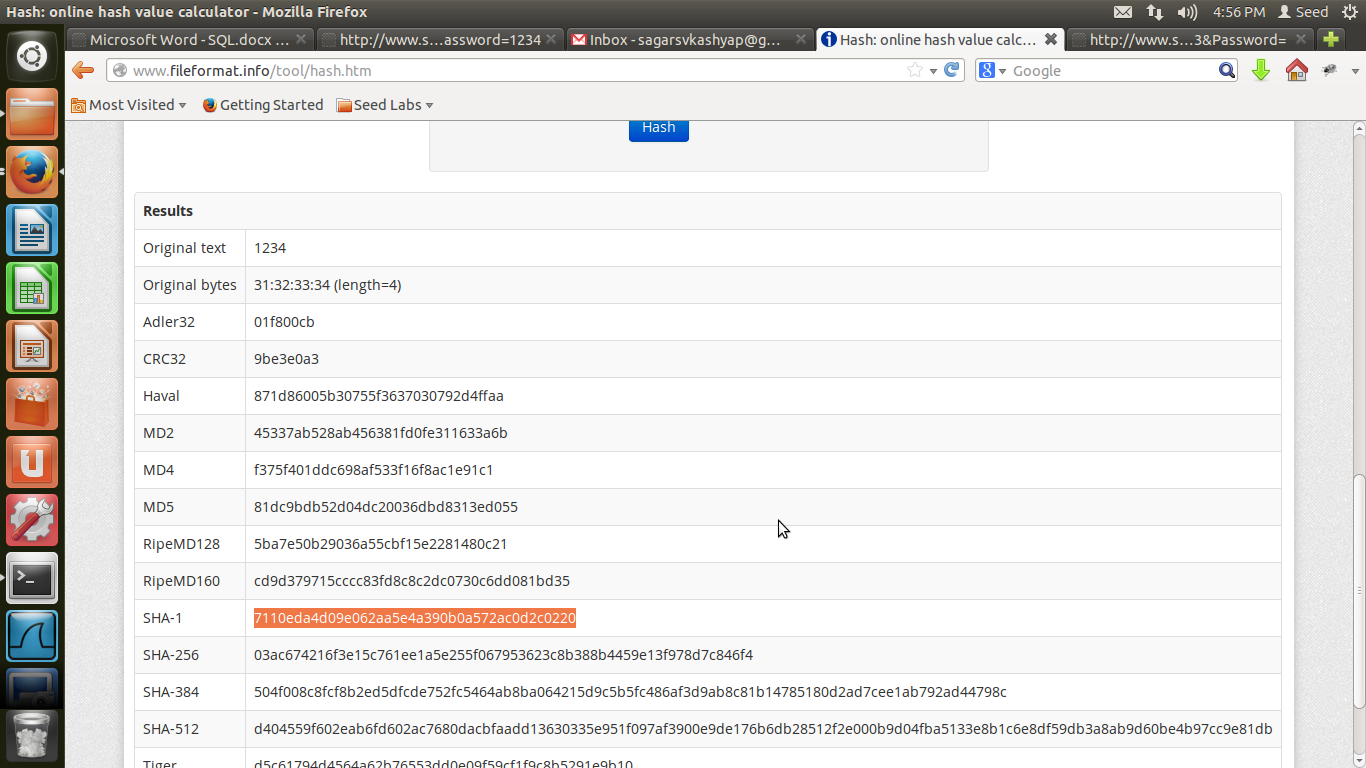




**Task 3.2: SQL Injection update of password**

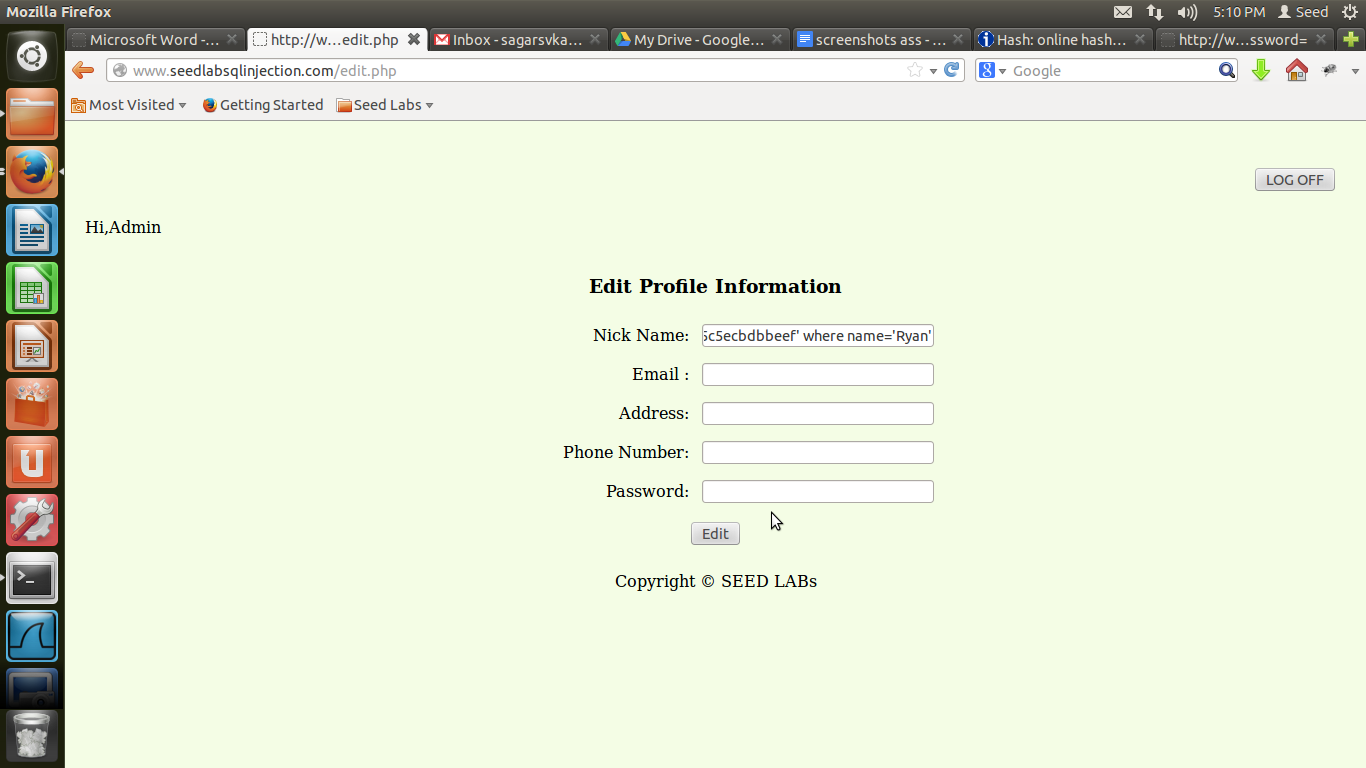
This one was tricking one as we had to find the hash value of the password that was given the MYSQL table and the find update the password / hash value of SHA1 into the query. The Password got updated and I could login.

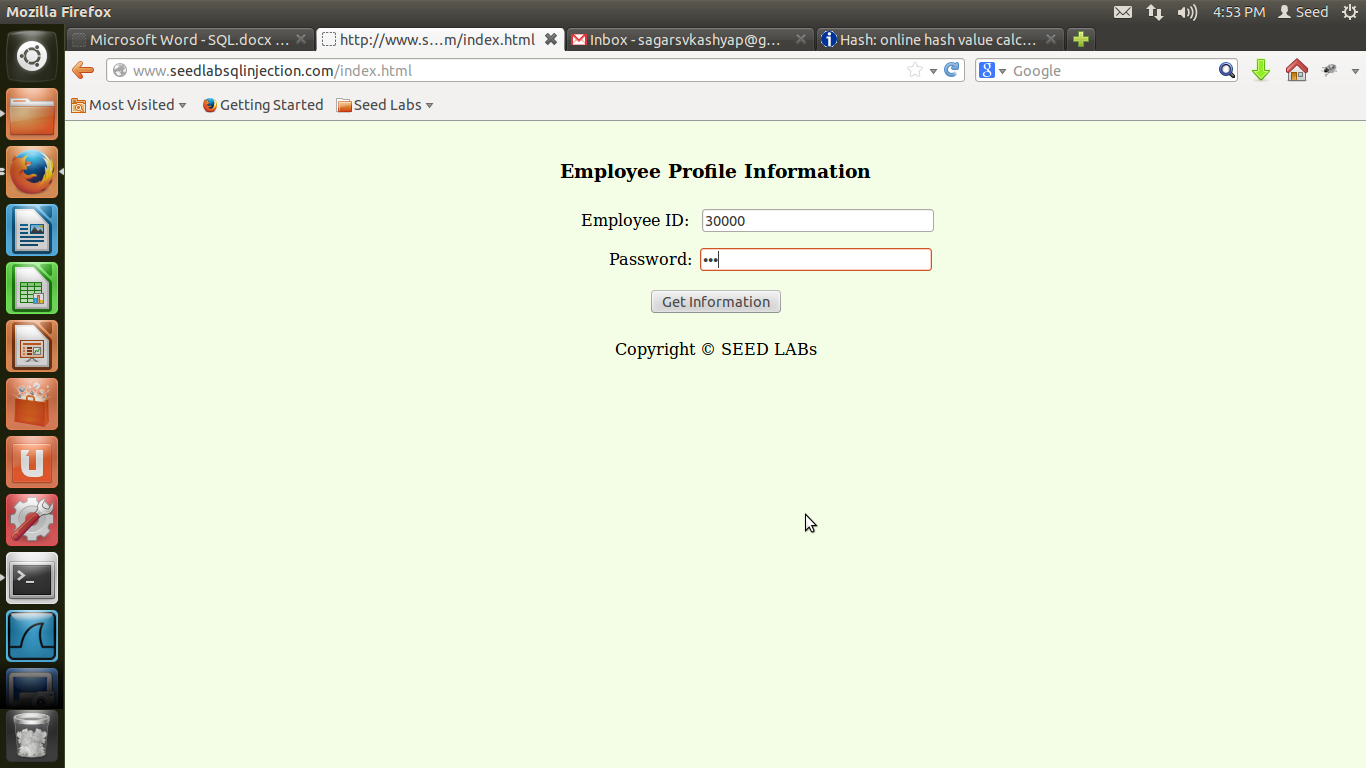




Query used in the block for updating the password:

‘, Password='40bd001563085fc35165329ea1ff5c5ecbdbbeef' where name='Ryan'

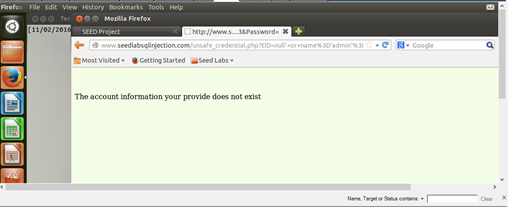






**Task 4 Counter-measures Magic packets and prepared statements**

**Magic Packets**



When magic packets is turned on, You cannot access/bypass the application

**Prepared statements,**

<?php

$input\_eid = $\_GET['EID'];

$input\_pwd = $\_GET['Password'];

$input\_pwd = sha1($input\_pwd);

// check if it has exist login session

session\_start();

if($input\_eid=="" and $input\_pwd==sha1("") and $\_SESSION['name']!="" and $\_SESSION['pwd']!=""){

$input\_eid = $\_SESSION['eid'];

$input\_pwd = $\_SESSION['pwd'];

}

$conn = getDB();

/\* start make change for prepared statement \*/

$stmt= $conn->prepare("SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password

FROM credential

WHERE eid= ? and Password=?");

$stmt->bind\_param("is",$input\_eid,$input\_pwd);

$stmt->execute();

$stmt->bind\_result($id,$name,$eid,$salary,$birth,$ssn,$phoneNumber,$address,$email,$nickname,$Password);

$stmt->fetch();

if (!$result = $conn->query($stmt)) {

die('There was an error running the query [' . $conn->error . ']\n');

}

The above code was used to bind the parameter and was not able to completely login.