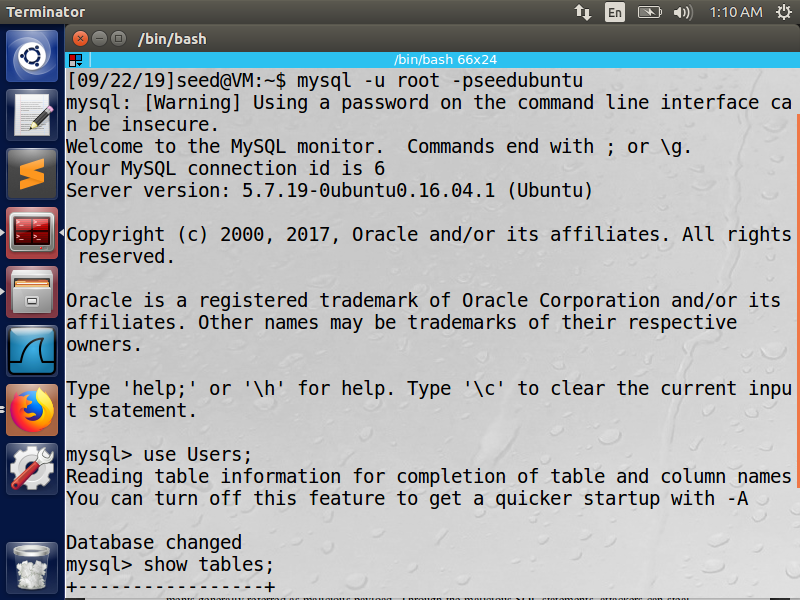
**Michael Remphry – SQL Injection Attack Lab**

# 3.1 Task 1 Task 1: Get Familiar with SQL Statements

Here I logged into mysql provided by SEED Labs using the command:

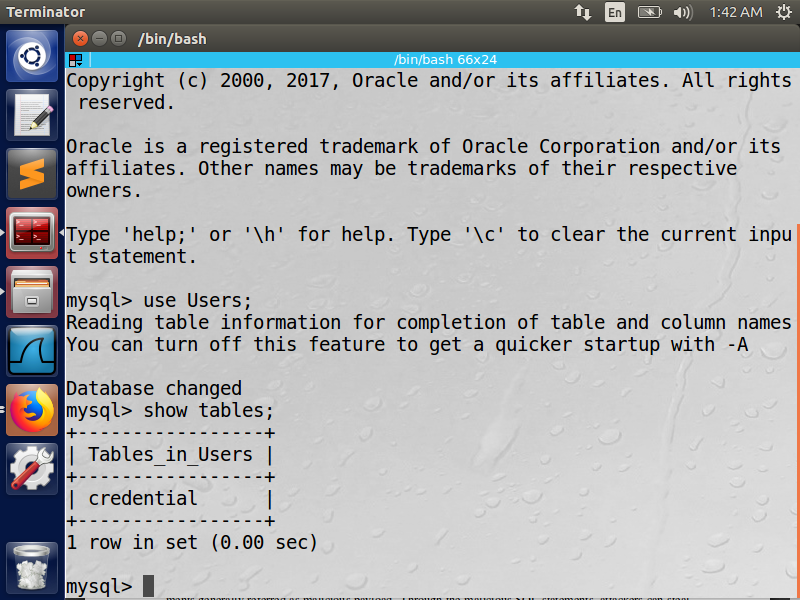
*mysql –u root –pseedubuntu*



Here I show what tables are in users, using the commands:

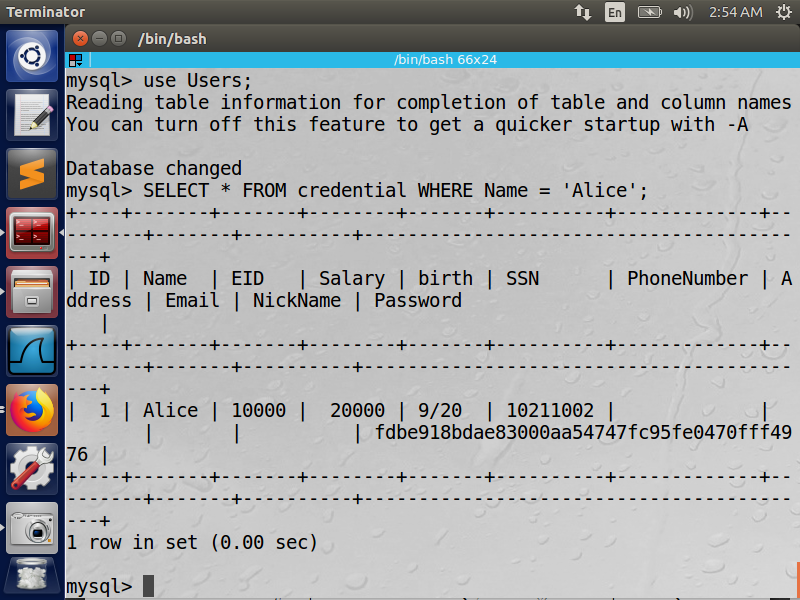
*use Users;* (This is used to load an existing database named “Users”)

*show tables;* (This is used to show the tables within the “Users” database)



Observation: Currently there is only a credentials table

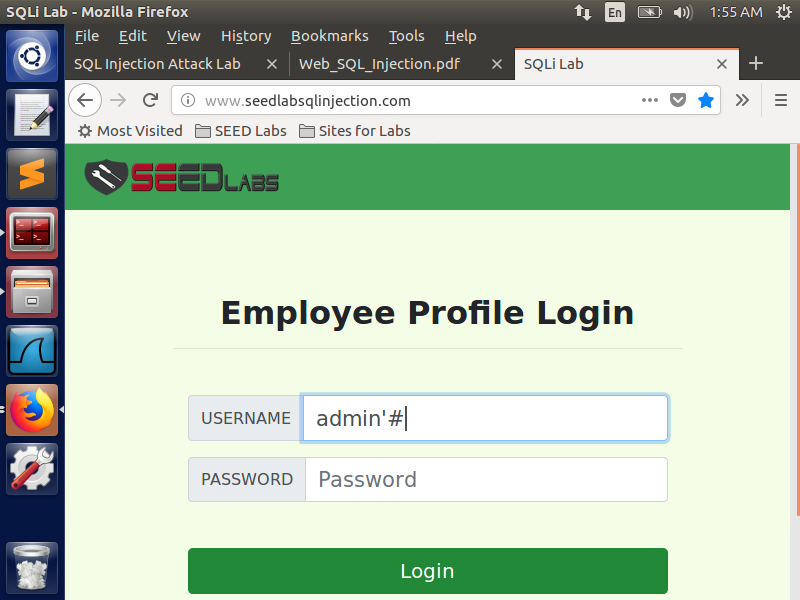
Using the SQL command, *SELECT \* FROM credentials WHERE Name = ‘Alice’;* I show all information about the user “Alice” in the credentials table



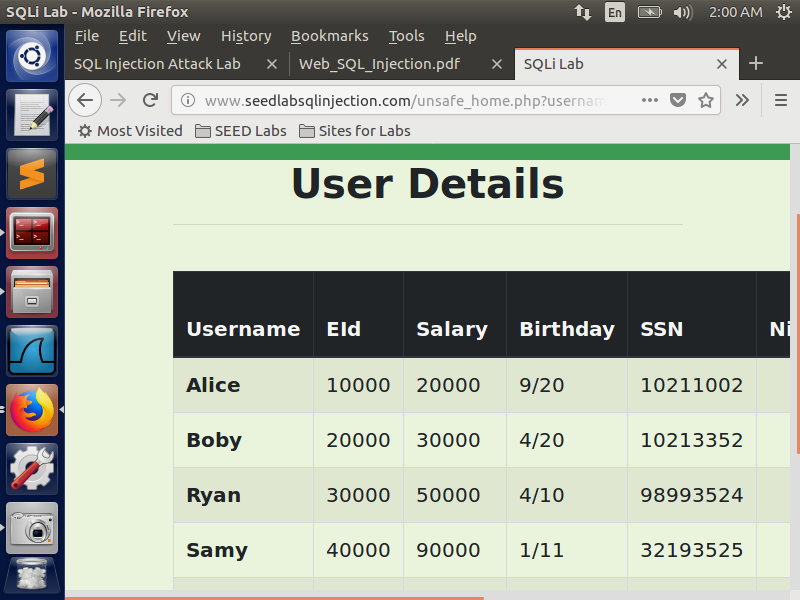
Observation: Some information seems to not be required such as phone number, address, email, and nickname. Passwords seem to be stored using some sort of hash algorithm.

# 3.2 Task 2: SQL Injection Attack on SELECT Statement

Here I bypass the login page by putting in the admin’s user name (admin), and comment out the password field using #. When sql executes the login process, this line it will read the # as a comment character and comment out any code that follows it. This was very easy to do, given that we know the admin username is admin.



Here is proving that I got into the admin account, which shows all information about all employees.

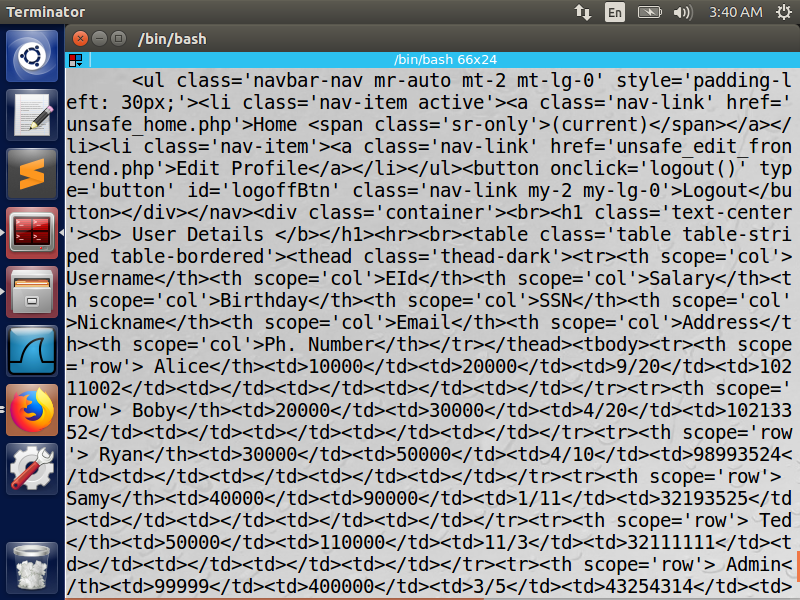


## 3.2 Task 2.2: SQL Injection Attack from command line

Here I used the command line to do the same thing as above. Using the curl command:

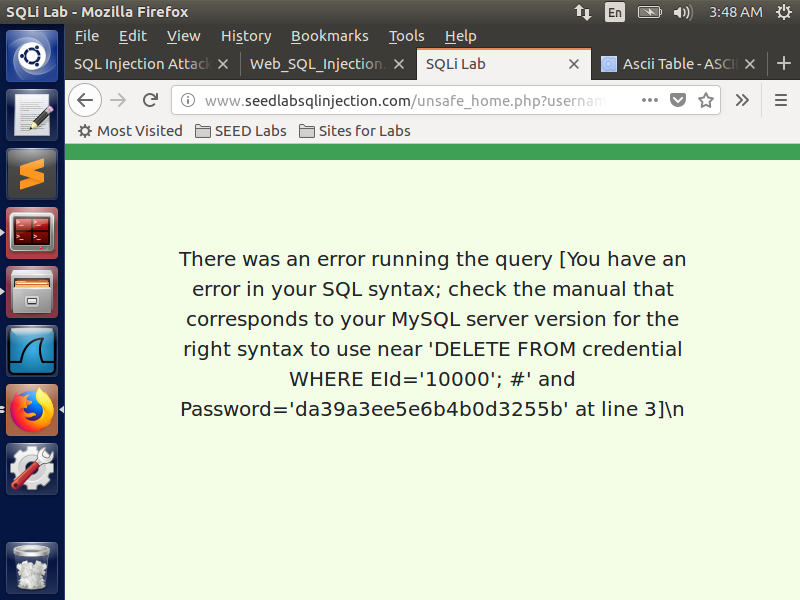
*curl ’www.SeedLabSQLInjection.com/index.php?username=admin%27%23*

Note that following admin there is %27 and %23. This is because those are the hex values for the characters ‘ and #. When typing special characters in a curl command, they need to be properly encoded, otherwise they may not work as intended.

I connected to the website and successfully logged in as the admin. I obtain the same results, just in a very difficult format to read.

3.2 Task 2.3: Append a new SQL statement

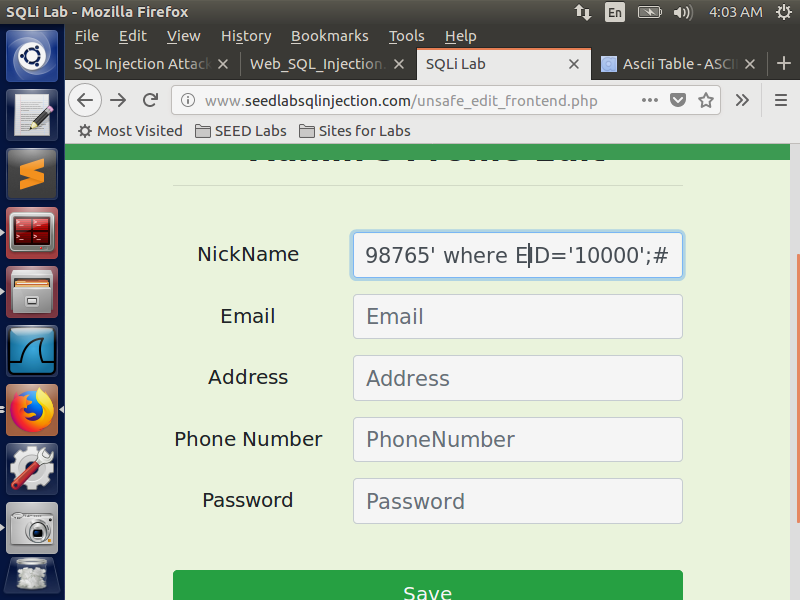
Here I attempt to append multiple sql statements in the login page. It appears that there are some limitations or security measures set up to reject multiple sql commands. After many attempts of different commands and testing different fields, I was unsuccessful to gain access.



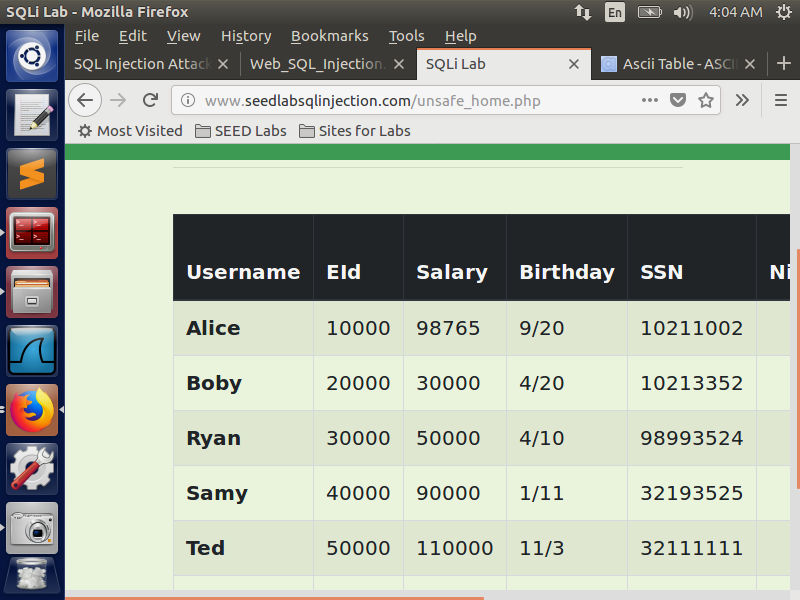
3.3 Task 3.1: Modify your own salary

By using Alice’s edit info page, I was able to run a sql command to change Alice’s salary to $98,765. Given that I was Alice in this scenario and that I know the EID, I used EID as the field to insert the salary into the correct field. I also found out that I could also use the username of the individual.

Also, in order to get to the edit page, the lab says to go to “unsafe\_edit\_backend.php”, however, I would get a 404 page not found error when using this in the url. Instead you have to use “unsafe\_edit\_frontend.php”.

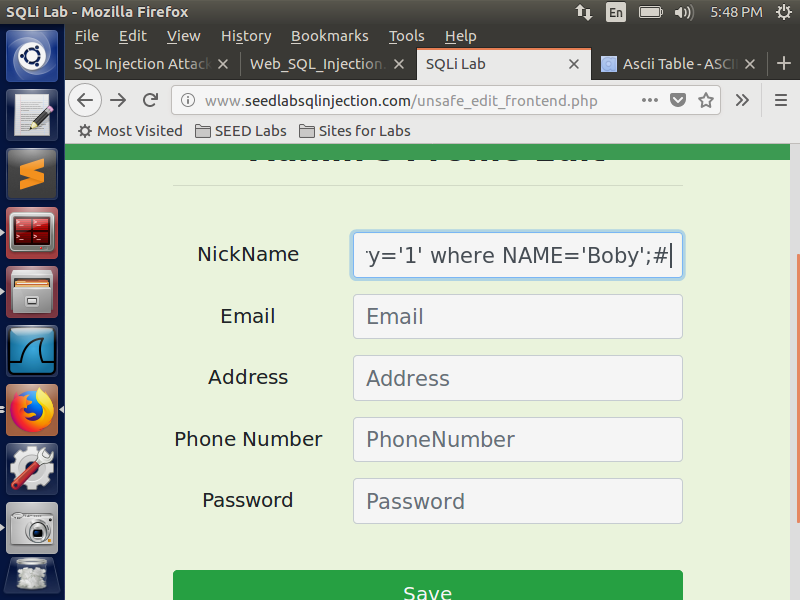


Here I prove that I successfully changed Alice’s salary

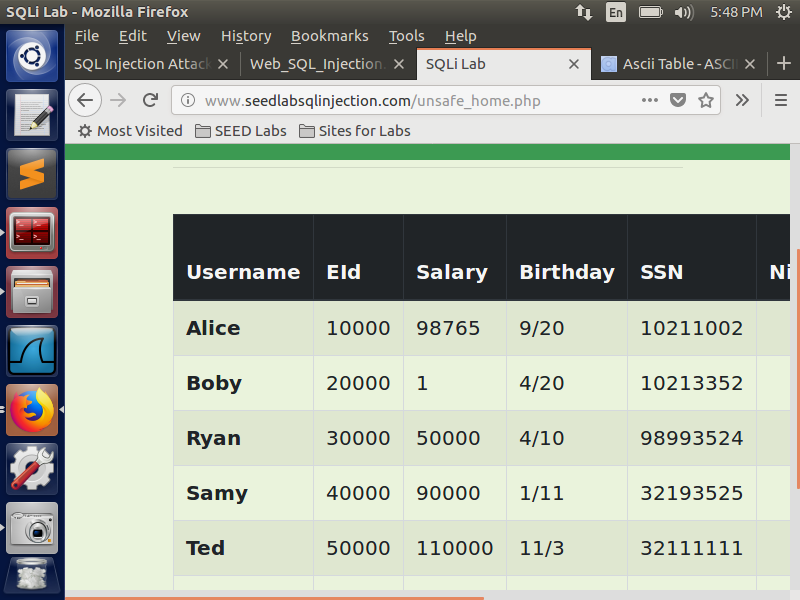


3.3 Task 3.2: Modify other people’s salary

Here I changed Boby's salary to 1. Since I know Boby's name and that "Name" is a field in the database, it is easy to change Boby's salary using his name for the WHERE condition using the name field.



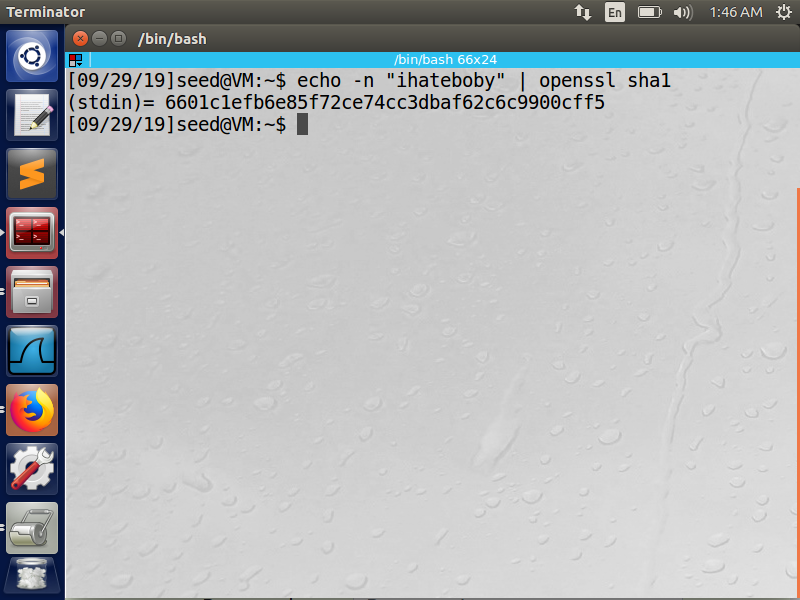
Here is proof that I the code worked



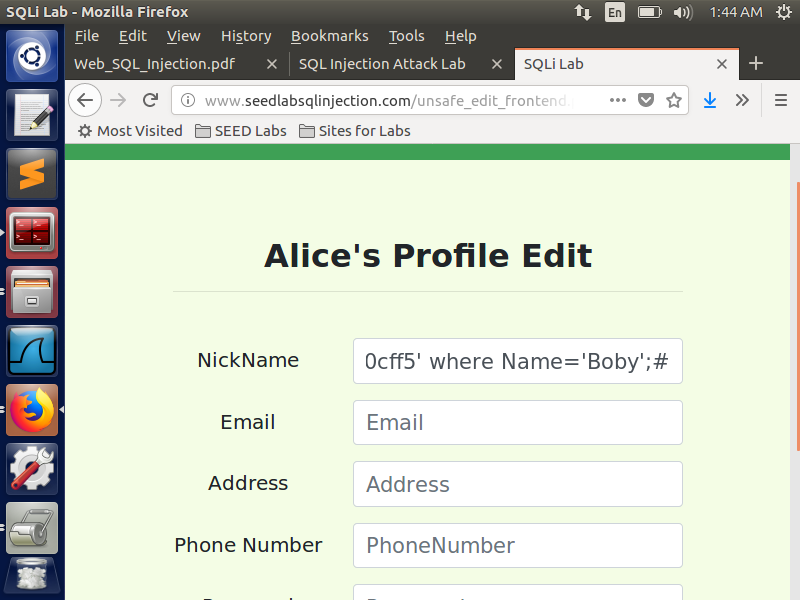
3.3 Task 3.3: Modify other people’s password

Knowing that the passwords are stored using a SHA1 hashing algorithm (you can find this by looking at the *unsafe\_edit\_backend.php*) I use the following command to give me the hash value to change boby’s password to “ihateboby”:

***echo –n “ihateboby” | openssl sha1***

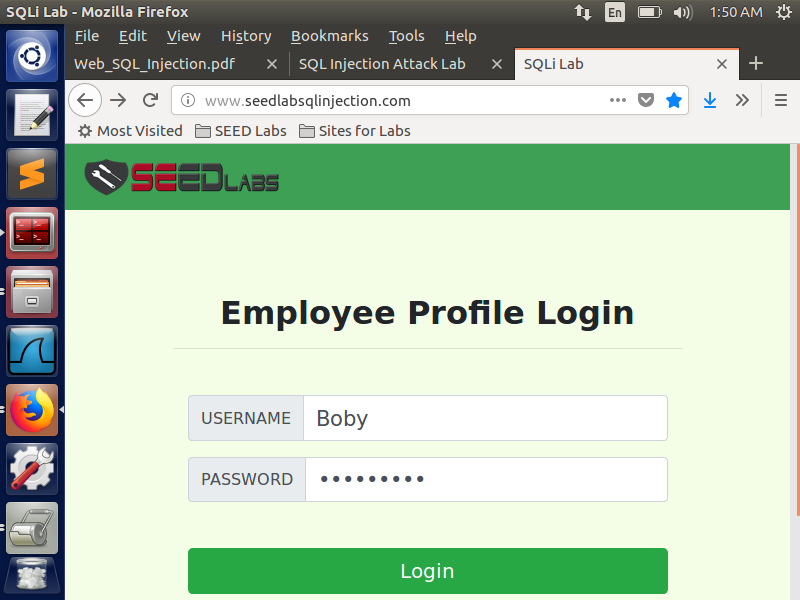


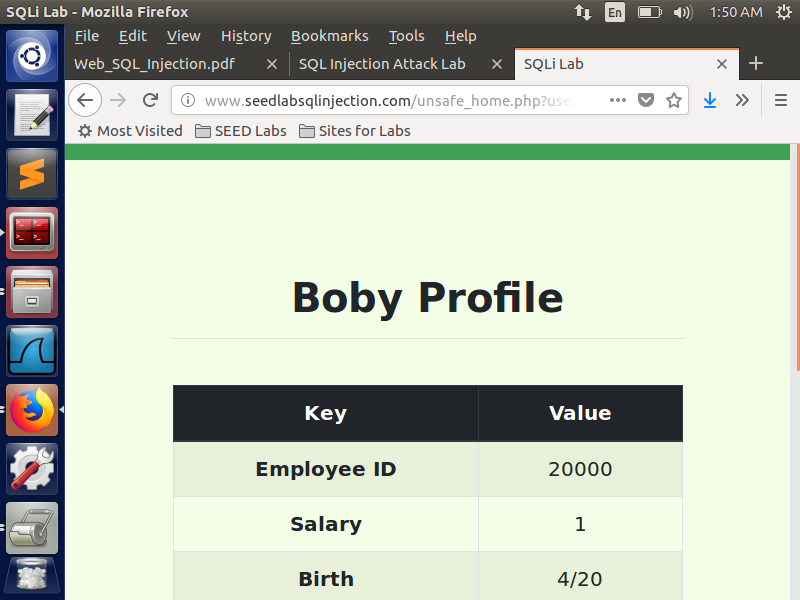
I then send an update command to change Boby’s password



3.3 Task 3.3 Cont..

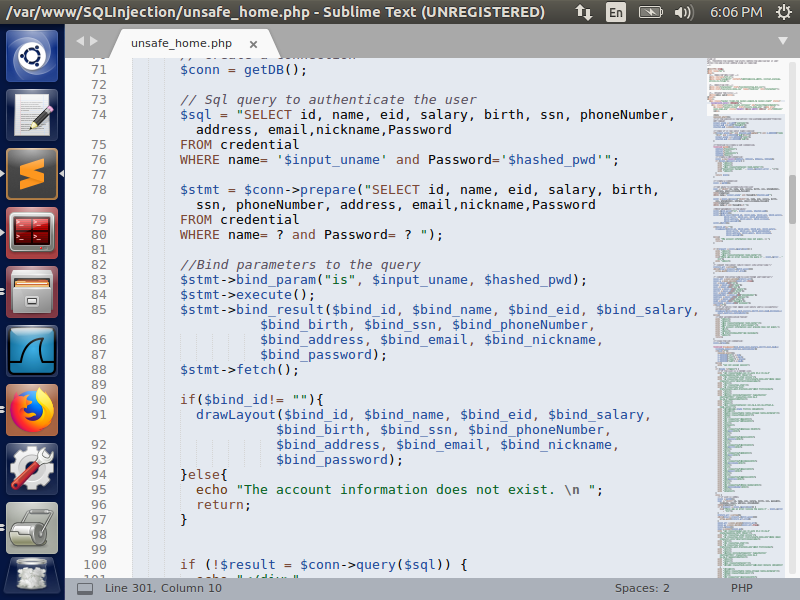
Here I show myself using the new password to log in as Boby, which the following image shows Boby’s profile after successfully logging in.





3.4 Task 4: Countermeasure – Prepared Statement

Here I implement a prepared statement to separate the code that is input into user fields from actual data to be used in the database.



3.4 Task 4 Cont..

After implementing the prepared statement, I attempt to log back into the admin account, but am unsuccessful (indicated by the blank page in the second image.

