**LINUX EXERCISE (LAB 06)**

Exercise 1. Manage services using systemd:

1. Check whether systemd is available in your Linux system (systemctl command)

A picture containing text, screenshot, font

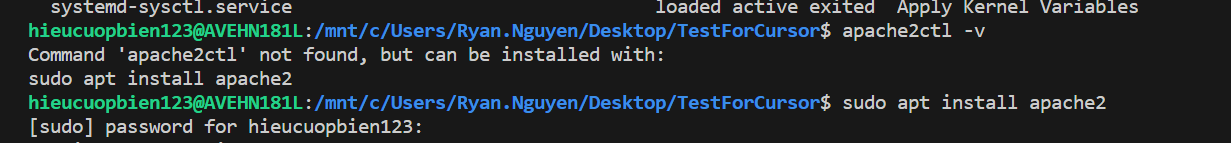
Description automatically generated

1. List all services in your Linux OS

A screen shot of a computer

Description automatically generated with low confidence

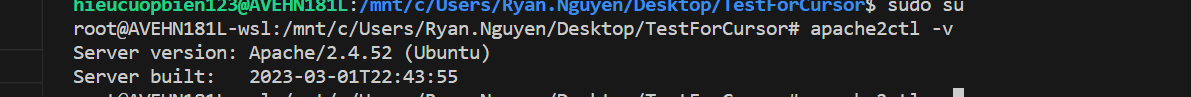
1. Check whether Apache server is installed or not (httpd). If not, please install Apache server

When I hadn’t install apache:  


Install it:  
A screen shot of a computer

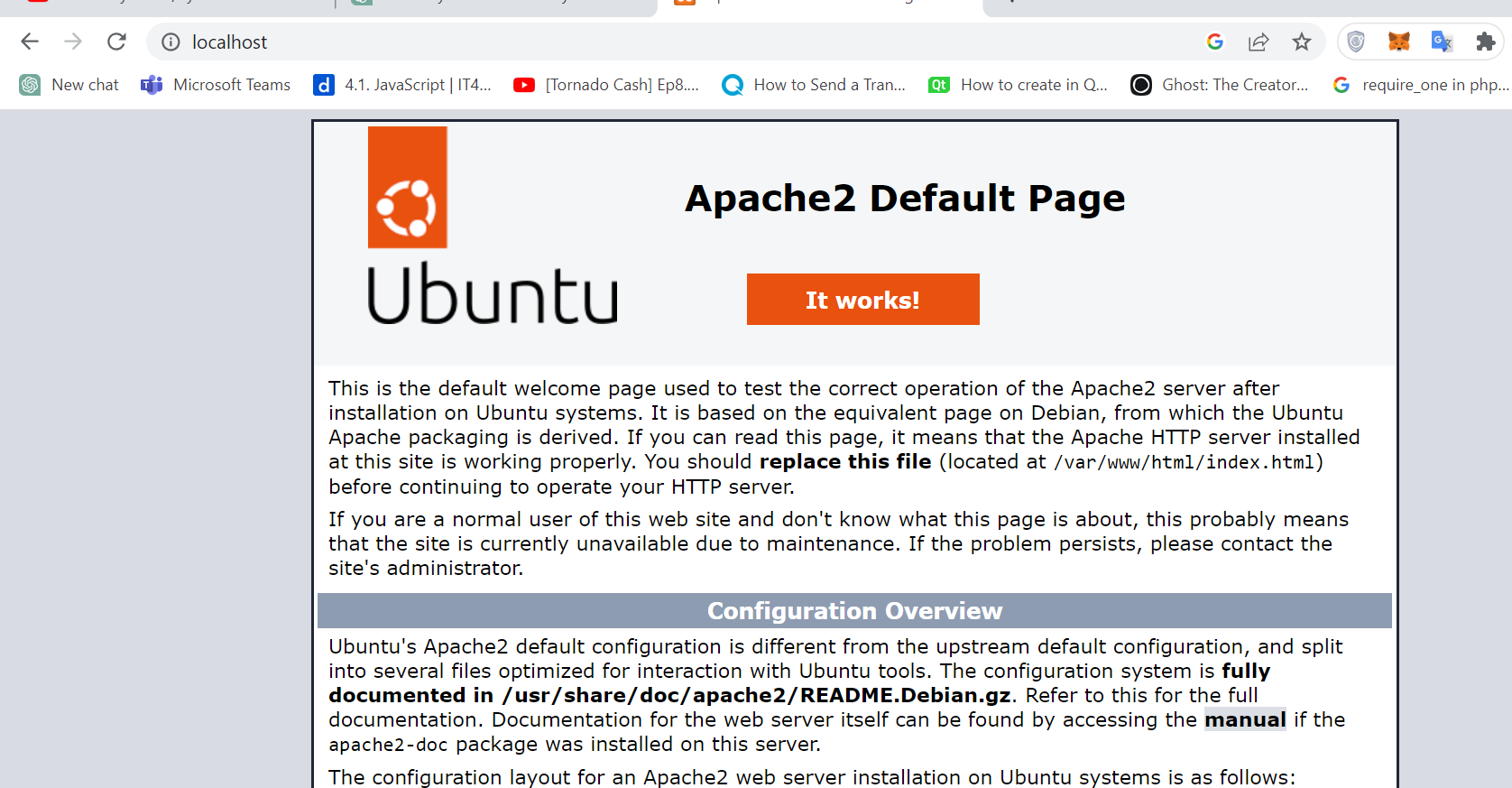
Description automatically generated with low confidence

After it is installed:



1. Start Apache server

sudo service apache2 start



1. Show all using files of Apache server

A screen shot of a computer

Description automatically generated with low confidence

1. Try to access to your web server from another computer using IP address. If we want access the web server using the domain name (test.com), which steps should we do?



In the same network, just enable firewall and allow access from outside computer

In the different network, we have to config router

If we want to show domain name, we have to register a new domain. We can use local DNS by adjusting file hosts in computer to show the domain name on the computer we

1. Turn off the Apache server’s service, try to access to the Apache server from another computer.

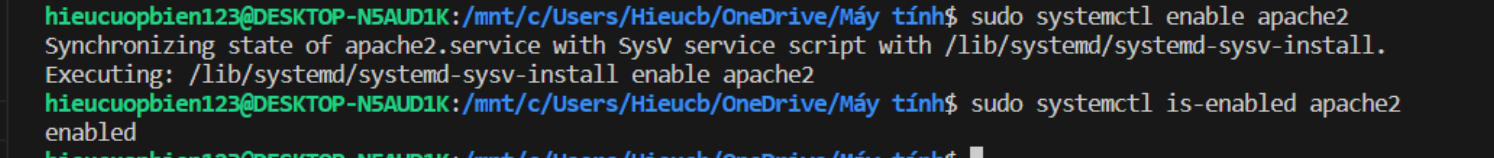
Show error page

A screenshot of a computer error

Description automatically generated with medium confidence

1. Configure your system so the Apache server boots up along with your computer

sudo systemctl enable apache2



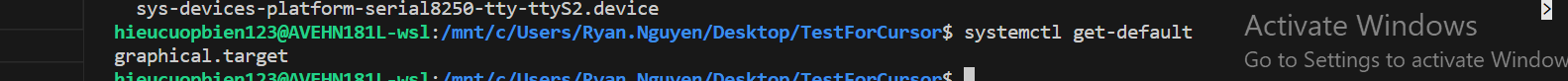
1. Restart your computer using the command line. Login to your system. How could we know whether the Apache server is working or not?

sudo reboot

Open the link: <http://localhost> if ok, the server started successfully

Exercise 2. Change the run level of your system using systemd

1. Check the default run level of your system



1. Display all run levels of your system

A screen shot of a computer

Description automatically generated with medium confidence

1. Change the default run level of your system from GUI to console (or reverse).

Here I change from gui to console and then console to gui

A screen shot of a computer screen

Description automatically generated with low confidence

1. What is the notification when you try to change the default run level? Is there any other way to change the default run level of your system without using systemctl?

Simple notification: The concept of runlevels is obsolete. Please use 'systemctl' to set the default target. For example, use 'systemctl set-default graphical.target' to set the default target to the graphical environment

In older Linux distribution that still relies on traditional System V (SysV) init scripts, we can modify the /etc/inittab file to change the default runlevel.

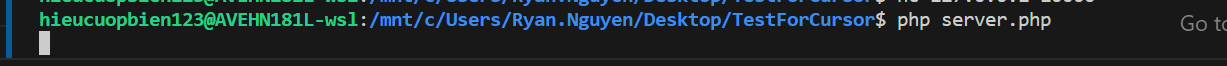
Exercise 3. Create and manage a new service

1. Create a new file server.php with the following content. Try to guess the purpose of this file? (Hint: you can do it after executing the step 3)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | <?php | |  |  | |  | $sock = socket\_create(AF\_INET, SOCK\_DGRAM, SOL\_UDP); | |  | socket\_bind($sock, '0.0.0.0', 10000); | |  |  | |  | for (;;) { | |  | socket\_recvfrom($sock, $message, 1024, 0, $ip, $port); | |  | $reply = str\_rot13($message); | |  | socket\_sendto($sock, $reply, strlen($reply), 0, $ip, $port); | |  | } | |

This PHP code creates a socket using the UDP protocol and binds it to port 10000 on the local machine. It then enters an infinite loop where it waits for incoming messages using socket\_recvfrom(). Once a message is received, it applies the str\_rot13() function to the message, which performs a simple character substitution by rotating each letter in the message by 13 places in the alphabet. The resulting string is then sent back to the original sender using socket\_sendto() => a simple UDP echo server

1. Run “php server.php” to start the php server. (You might need to install php server – php\_cli)



1. Check the status of php server by login to another account (different terminal). Run the command “nc 127.0.0.1 10000” and then type “Hello world” after connecting with local php server. Is the printed message “Uryyb, jbeyq”?

Yes

1. To convert the PHP server to a service, create a file /etc/systemd/system/rot13.service with the following content

|  |
| --- |
| [Unit] Description=ROT13 demo service After=network.target StartLimitIntervalSec=0[Service] Type=simple Restart=always RestartSec=1 User=centos ExecStart=/usr/bin/env php /path/to/server.php  [Install] WantedBy=multi-user.target |

Create this file

1. Try to start the service rot13 using systemctl

systemctl start rot13

1. Use systemctl to configure rot13 so it can boot up along with the Linux OS

systemctl enable rot13

Exercise 4. (Optional) Please check which bootloader your Linux system is using. Change it to a different bootloader (from GRUB to LILO and vice versa, or another bootloader like Slim Bootloader). We recommend students to use a virtual operating system and not to perform direct operations on physical machines.