**Assignment:** Create a test environment for Web Servers, phpMyAdmin and Content Management Systems to allow us better familiarization with CM Systems.

**Required Resources**

* Linux Mint (Created in Assignment 1)
* Naming convention (Course Resources)

**Professional Documentation**

All documentation must be done in a **professional style**. It must include:

1. Title page
2. **Updateable** Table of Contents
3. Document introduction
4. Section introductions or description, each section must be clearly identified
5. Graphics or screenshots MUST include a title with a short description
6. Any direct or copied quotes or graphics MUST be properly credited in a footnote
7. ALL sources MUST be properly cited (APA format) and placed at the end of your document in a bibliography.
8. **NO** embedded, zipped or compressed files. \*\* All scripts must be converted to text before including them in your documentation. \*\*

**\*1 Professional Word Document ONLY.**

**Research and documentation sections** -Please complete all research and question responses in your own words. Research sections not completed in your own words may result in a mark of 0 for the section.

**NOTE:** Please do NOT copy and paste responses from internet, **even with a citation**. I expect each section or response to be in your own words. Be prepared to explain your responses and demonstrate your comprehension during the marking period.

**No marks** will be given for cited or credited information included in document.

**Marking and Assignment Notes:**

* ScreenshotsMUST include user or device identifying information.
* Screenshots MUST be added to your document in the order of creation.
* Documentation must meet Professionalism requirements.
* **Automatic mark of 0 - Assignment not submitted or work not original.**

<http://www.nscc.ca/docs/about-nscc/policies-procedures/policy-studentcodeofconduct.pdf>

<https://www.nscc.ca/docs/about-nscc/policies-procedures/policy-academicintegrity.pdf>

**NOTE: This assignment may require some adaption, research and troubleshooting.**

**\*\* See APPENDIX A FOR TROUBLESHOOTING TIPS.**

**Task 1 - Install your Ubuntu Web Server**

* NO Install/Change Log required for this node.

Install Ubuntu **Server** 24.04.1 LTS (Noble Numbat)

With these settings:

* VM OS = Linux - Ubuntu 64-bit
* 100g drive
* 4g memory (8G if your host machine can support it)
* VMWare Name = Follows naming convention
* Modify the Virtual Machine Description to list:
  + Operating System:
  + Creation Date:
  + Hostname:
  + Root Password:
* Accepts defaults and disk modifications install options unless noted below
* Update to the new installer
* Server details.
  + Server host name = wb01test
  + Static IP = 192.168.208.120
  + Your Name = FirstName LastName
  + Username = webop
  + Password = Passw0rd

\***Note**: In Linux it is always recommend we use Lower Case except in regards to Your Name and password.

* Do not upgrade to Ubuntu Pro
* Install OpenSSH Server (Do not add any new users or modify your OpenSSH options)
* Do Not install any Feature Server Snaps
* Update and upgrade your Ubuntu Server.
* Set your time zone to America Halifax
* Check your time zone again to confirm it is the correct time.
* If required, install your Open VM Tools

sudoNow we will need some additional tools to support us in our Web Sever creation and configuration.

* Use the following command to install some helpful networking tools

sudo apt install net-tools

* View your IP information and then **record** your server IP information in your documentation (remember we do not require an install log or change log).
* This would be a good time to take a snapshot.

Now we will install our LAMP Stack. Remember LAMP stands for Linux, Apache, MariaDB, and PHP. Use the following guided install to install your LAMP stack with PHPMyAdmin on your Linux Web Server.

* First we will install the Linux Apache part of our LAMP stack

install apache2

* **Confirm** your service is currently installed, running and enabled
  + *Note*: You may need to “break out” of the systemctl control.

Now that we have installed apache and enabled the service let’s check that it is running and prepared to host.

* Open any browser on your Host Machine a type the following URL

<http://YourServerIP>

* Now we will install the MariaDB part of our LAMP stack by running the following command

install mariadb-server mariadb-client

* Confirm your service is currently installed, running and enabled. Modify if required.

Now we will complete the mariadb setup by enabling some basic security measures for our database by running the mysql\_secure\_installation script that is attached to the mariadb package we just installed.

* First we will need to install the mysql package with the following command (notice the underscores)

sudo mysql\_secure\_installation

* + Complete the secure installation script with the responses as per directed below:
    - Enter current password for root (enter for none): Enter
    - Set a root password? [Y/n] y
    - If required, password = Passw0rd
    - Remove anonymous users? [Y/n] y
    - Disallow root login remotely? [Y/n] y
    - Remove test database and access to it? [Y/n] y
    - Reload privilege tables now? [Y/n] y
* Run the following command to access the mariadb shell. **Note**: You MUST use sudo any time you access the mariadb shell or you will receive an access error, remember we installed the secure shell with sudo.

sudo mysql -u root

* **Stop**. After accessing your mariadb shell you will be supplied some basic information including your MariaDB connection id. Take a screenshot of the result of your previous command including the information for getting ‘help’ and add it to your documentation.
* Now we will exit our mariadb with the following command

exit

* Now we will install the PHP part of our LAMP stack

install php libapache2-mod-php php-mysql

* After installing our PHP we will need to restart our apache2 service

Now you will test our apache2 is working with our PHP install.

* Using the following command to create a page called info.php in the /var/www/html/ directory

sudo nano /var/www/html/info.php

* Now add the following lines to your file

<?php

phpinfo();

?>

* Now we will test our file by opening our apache2 in any browser on your host machine and adding the /info.php to the end of your URL.
* Stop. Capture a screenshot of your php information that includes **PHP Version, System and Build Date** and add it to your documentation.

*Apache runs its process with a default user account and group. For many security reasons, it is recommended to change this and let Apache run with its own non-privileged account.*

\*\* Do NOT use quotes for settings below\*\*

* Create a group called “apachegroup”
* Use the following command to created a user called “apacheuser” and add then to your new apachegroup, since this user will be specific to our apache configuration and not a regular user they will not need a home directory so we will skip that step in our creation:

sudo useradd –d /var/www/ -g apachegroup –s /sbin/nologin apacheuser

* In order to make sure our user is active we must set the password for apacheuser, we will set our new user password to “webtest@2024” (no quotes)
* Now we will confirm our user and group was created correctly.
* Open your user file and group file confirm both are created and your user is in the correct group with the correct settings.
* Edit your /etc/apache2/envvars system file to set your “APACHE\_RUN” user and group to match your new user and group you created.
* **Stop**. Capture a screenshot of your apacheuser and apachegroup in your respective files (user file, groups file and envvars) with the correct settings.
* Restart Apache2
* Now would be a good time to take a snapshot.

**Task 2 – Install phpMyAdmin**

*“phpMyAdmin is a free software tool written in*[*PHP*](http://php.net/)*, intended to handle the administration of*[*MySQL*](http://mysql.com/)*over the Web. phpMyAdmin supports a wide range of operations on MySQL and MariaDB. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc) can be performed via the user interface, while you still have the ability to directly execute any SQL statement.” Source =* [*www.phpmyadmin.net*](http://www.phpmyadmin.net)

* We have made some changes to our system so lets get the most current updates and upgrades
* Use online install instructions such as the link below to install your phpMyAdmin. Make sure to read the instructions carefully. ***Hint****: configure your database for phpmyadmin with dbconfig-common.*

Here is one site that you might find helpful:

<https://ubuntu.com/server/docs/how-to-install-and-configure-phpmyadmin>

\*\* Remember, this is one of many sites that will help you install phpMyAdmin.

* Choose our standard super secure password “Passw0rd” for our phpmyadmin.
* Since we configured our database during the install for our local server we can skip the **Configure phpmyadmin** section.

* Complete the install with Configuration Filess section for:
  + - Symbolic link
    - Phpmyadmin.conf
    - Reload apache2

*Hint:* You may receive the output “$ln: failed to create symbolic link 'my\_link.txt': File exists” this simply notifies us that the symbolic link already existed.

*Additional Learning: Symbolic Links or symlinks is a special type of file that points to another file or directory. This is similar to a shortcut in windows. Symbolic links differ from hard links as hard links give a single file multiple names. We use the ln command to create the link and the -s to make it symbolic.*

* Let’s take a minute to confirm our symlink is present but moving the file location of the link

cd /etc/apache2/conf-available/

* Now we will use the ll command to view the contents and we should notice our symbolic link between our phpmyadmin.conf and apache.conf files.
* **Stop**. We want to keep a record of our symbolic link so take a screenshot confirming your symbolic link and add it you’re your documentation.
* Test your install by opening your host browser to <http://server_ip/phpmyadmin>
  + Log into using:
    - username= phpmyadmin
    - password= password set up during install.
  + \*\*For security reason NEVER remember your password for this site.
* Once you have logged in successfully take a look at your phpMyAdmin menu bar and mouse over each icon and take time to select each one and see what it does.

A screenshot of a computer

Description automatically generated

* Now that you have looked a little at the options in your phpMyAdmin dashboard use the icon to **Log Out** and move back to your server.

Now that we have created our additional layer of security lets create a user in our MySQL database using phpMyAdmin. First we will need to create the user and secure the privileges in our Mariadb database.

* Create a new user named admin with the password Passw0rd and grant them full privileges using the following commands (remember Linux is case sensitive).

sudo mysql -p -u root

CREATE USER 'admin'@'localhost' IDENTIFIED BY 'Passw0rd';

GRANT ALL PRIVILEGES ON \*.\* TO 'admin'@’localhost’ IDENTIFIED BY 'adminPassw0rd' ;

FLUSH PRIVILEGES ;

SHOW GRANTS FOR 'admin'@’localhost’ ;

* Now that you have created your new user and confirmed they have been granted the full permissions exit out of your Mariadb and **restart** your apache2.
* We will need to make one more change to allow our user to the correct permissions to complete our next task.
  + Log into your phpMyAdmin as root
  + Select User Accounts
  + Edit the privileges for ‘admin’
  + Select GRANT
  + Select User Accounts and confirm admin now has GRANT
* This might be a good time to take a snapshot.

**Task 3 – Create a database and table and confirm authentication**

**Part 1 – Create a database and table**

* Open your phpMyAdmin in your host browser and login as your **new admin user.**

\*\*For security reason NEVER remember your password for this site.

Now let’s create a new database and table in our phpMyAdmin.

* Confirm you are logged into phpMyAdmin webportal as **admin** and create the following:
  + From the left hand window of your dashboard select “new”
  + Create a new Database = osys
  + Create a new table inside your osys Database
    - New Table = members
    - 3 columns
    - **Note**: You must select the “create” button on the right to create the table
  + Create your Table Structure:
    - Name = user\_id, int, length=11, Primary, A\_I (Auto\_Incremental)
    - Name = username, VARCHAR, length=50
    - Name = password, VARCHAR, length=50
    - Save your table structure
* Once you have created your new Database and Table you will need to populate it by selecting our table on the left and using the insert tab to add new entries.
  + **Insert** 2 entries into your members table
    - User\_id – value = 1
    - Username – value = Bruce
    - Password – function = MD5, value = student
    - User\_id – value = 2
    - Username – value = YourFirstName
    - Password – function = MD5, value = student
  + Select **Go** from bottom of page when to accept your new entries
  + Select your members table and confirm your two new entries
* Log out of your phpMyAdmin
* **Stop**. Be prepared to demo your databases, tables and users along with their correct settings.

**Part 2 – Use php scripts to manage authentication**

Before we can proceed with creating our PHPscript we need to transfer our script files from our Course Resources to a share on our web server.

* Create a directory to save your files to /home/firstinitiallastname/share
* View your new /share directory and view your permissions**, make sure webop not root is the owner.**
* Use SSH to copy the contents of the PHPScripts folder (from course module) to your Web Server share directory.
* Confirm your PHPscripts folder and all the subfolders and files transferred over successfully.
* Edit your **PHPscripts/lib/db\_connect.php** script with these settings:

$host = "localhost";

$db\_name = "database name here";

$tbl\_name = "table name here";

$username = "admin";

$password = "Password **you** set for admin here";

\*\*\* Note – the username and password MUST match the one you created earlier.

* Now copy the **contents** **ONLY** of your PHPScripts directory in your system directory **/var/www/html/**

\*\*Remember you will need to add the correct argument to your copy command to copy all your files, folders AND permissions (don’t forget hidden files).

* **Stop**. This is a great command to remember. Include a screenshot of the command and the results in your documentation.

Now we will need to call on our authentication page to allow for logins.

* Modify the Welcome.html or create a web page for Company Inc:
  + Add a Log in link (see example in Appendix A).
  + Link the login to your authentication page (/var/www/html/auth.php).
  + Change the webpage image from Company Inc to yours.
  + Update the date and web master

**HINT**: If you do not have auth.php in the root of /html you did not copy your contents correctly. Repeat the previous step.

* Make a backup copy of your /var/www/html/index.html called index.html.old then replace your original index.html with your Welcome page **NOTE:** If have not already done so , you will need to rename your Welcome page index.html
* Your /var/www/html directory should now contain:
  + auth.php
  + authv2.php
  + change\_password.php
  + checklogin.php
  + create\_account.php
  + delete\_account.php
  + index.html.old
  + index.html (your welcome page)
  + lib (directory)
  + login\_fail.php
  + login\_success.php
  + logout.php
  + styles.css
  + “your graphic”
* Test your authentication setup by opening your host browser to http://server\_ip/index.html
* This would be a great time to take a snapshot.
* **Stop**. You will need to be able to complete the following tasks for comprehensive marking:

1. Index loads welcome page that includes:
   * Welcome <centered>
   * Your Graphic <centered>
   * Your Name as Web Master
   * Creation/Modification Date
   * Link to Login page (auth.php)
2. Successfully log in as Bruce
3. Successfully Log out
4. Create new account
5. Successfully log in with new account
6. Change Password for new user
7. Logout and login with the new password
8. Successfully delete new user
9. Cannot log in with deleted account

**Task 4 – Install wordpress and create a Blog page.**

*Additional Learning: Web Content Managers are software designed to help organize and manage web contents including sites, media, etc.*

*WordPress is a Content Manager originally created to create Blogs but has grown to support publishing all forms or web pages and forums. It supports a wide variety of web based applications and API.* [*https://wordpress.com/*](https://wordpress.com/)

*Additional Learning: In order to get our new content manager application we will need to download a tar.gz file and then unzip it, but what is a .tar.gz file?*

***tar*** *stands for tape archive which is file that contains multiple files, sometimes referred to as a “tarball”.*

***.gz*** *is the extension used to reference GZIP which compresses and uncompresses the file.*

*When unzipping or expanding our tarball we have several options available:*

*c – create a archive file.*

*x – extract a archive file.*

*v – show the progress of archive file.*

*f – filename of archive file.*

*t – viewing content of archive file.*

*j – filter archive through bzip2.*

*z – filter archive through gzip.*

*r – append or update files or directories to existing archive file.*

*W – Verify a archive file.*

* Use the **wget** command to get an updated copy of WordPress from site below and save it to your share folder your created in the last assignment

wget -P /pathtoshare/share <https://wordpress.org/latest.tar.gz>

**ATTENTION**: Make sure to replace your /pathtoshare/ with the correct path to the share your created earlier.

* Unzip your new application. **Note:** it will automatically unzip into a folder called WordPress in the same location as the downloaded file. Ex. share\wordpress

tar -xzvf latest.tar.gz

Before we can install WordPress we will need to create a database to support it.

* Using phpMyAdmin
* Log in as admin
* Create a new Database:
  + Name = wordpress
  + We do not need a table for wordpress

Now we will need to create a WordPress database manager.

* Using phpMyAdmin
* Login in as your Mariadb root user
* Create a new user
  + User name = wpmgr
  + Host = Local localhost
  + Password = same as root
  + Select “Go” to create the user
  + **Global privileges** = Check all
  + Select “Go” to apply privileges
  + Now give wpmgr privileges to your wordpress database
  + Edit **Privileges** for wpmgr
  + **Database** privileges
  + Add privileges on the following database: wordpress
  + Database-specific privileges = Check All
* **Stop**. Confirm you have created your new user with the correct privileges by selecting your wpmgr user account and choose “Export”. If all the privileges are correct then take a screenshot and add it to your documentation for record keeping.
* **Stop**. Be prepared to demo the wordpress database and user with correct permissions.
* Log out of your phpMyAdmin

Now we will pre-configure our install to link to our new database and user.

* If required log on to your web server
* Edit your /wordpress/wp-config-sample.php under the section

//\*\*Database settings – You can get this information from your web host\*\*//

* + Modify the following settings:

‘database\_name\_here’

‘username\_here’

‘password\_here’

* + Using the database, user and password that we just created in phpMyAdmin

\*\*Note: Do not make any other changes to your file except those noted above.

* Save your edited file as **wp-config.php**

We are almost ready to install our WordPress.

* Using the command from earlier copy the contents of your wordpress directory to /var/www/html/

\*\*Note: Do **NOT** copy the wordpress folder, just the contents.

**Hint**: remember you require a command switch to copy directories and their contents including hidden files and permissions to a system directory.

Now we will need to created a location to save our images in Wordpress.

* Create the **/var/www/html/wp-content/uploads** directory with the permissions 777.

We are finally ready to install our WordPress.

* Open your browser and go to:
  + http://[serverIP/wp-admi](http://localhost/wp-admin)n/install.php
  + Site Title = yourfirstname\_osys3030 (all lowercase)
  + Username = wpmgr
  + Password = change to password set previously for wpmgr
  + Check Confirm use of weak password if required
  + Email = “[webserverdefaultuser”@osys3030.ca](mailto:WebServerDefaultUser@OSYS2040.ca) (no quotes, all lowercase)
    - ex. webop@osys3030.ca
  + **Check** Discourage search engines from indexing this site
  + Install WordPress
  + Log in with your wordpress credentials identified above
  + Do NOT remember your password
* Use the following site to help complete the demo requirements for Wordpress. **Make sure your pages, posts, etc. have content**.

<https://wordpress.org/documentation/support-guides/>

* **Stop**. You will need to be able to complete the following tasks for comprehensive marking:

1. Log into Wordpress
2. Demo an applied theme.
3. Demo page with your created with content.
4. Demo at least one post you created with content.
5. Demo at least one comment you created with content.
6. Demo at least one media file you hosted (must be able to access and include content).
   1. Media file MUST be created by you.
   2. Media file MUST be no longer than 2 minutes.

It is important to keep an up to date record of all changes and modifications made to your server and have a reliable copy available as backup.

* Take a final snapshot of your server in the **OFF** state.
* Capture a screenshot of your Gold copy properties and add it to your documentation.
* Upload your documentation to Brightspace.

Marking Rubric

|  |  |
| --- | --- |
| **Value** | **Task** |
|  | **Comprehensive Marking** |
| 6 | Demo your phpMyAdmin database, table and users along with their correct settings:   * Database * Table * Users x2 |
| 3 | Demo Webpage:  Index loads welcome page that includes:   * + Welcome <centered>   + Your Graphic <centered>   + Your Name as Web Master   + Creation/Modification Date   + Link to Login page (auth.php) |
| 4 | Successfully log/ log out in as Bruce |
| 1 | Create new account |
| 1 | Successfully log in with new account |
| 1 | Change Password for new user |
| 1 | Logout and login with the new password |
| 1 | Successfully delete new user |
| 1 | Cannot log in with deleted account |
| 2 | Demo the wordpress database and user with correct permissions |
| 17 | 1. Log into Wordpress (1 mark) 2. Demo an applied theme. (2 marks) 3. Demo page with your created with content. (4 marks) 4. Demo at least one post you created with content. (3 marks) 5. Demo at least one comment you created with content. (3 marks) 6. Demo at least one media file you hosted (must be able to access and include content). (4 marks)  * Media file MUST be created by you. * Media file MUST be no longer than 2 minutes. |
| 2 | Snapshots taken (2 or more) |
| **40** | **Total Marks for comprehensive** |
|  | **Documentation as submitted to Brightspace** |
| 2 | Record your server IP information in your documentation (remember we do not require an install log or change log). |
| 2 | Screenshot of the result of your previous command including the information for getting ‘help’ and add it to your documentation. |
| 4 | Screenshot of your apacheuser and apachegroup in your respective files (user file, groups file and envvars) with the correct settings. |
| 2 | Screenshot confirming your symbolic link and add it you’re your documentation. |
| 4 | Screenshot of the command and the results that copies your files and folder and permissions and does not forget about hidden files in your documentation. |
| 2 | Screenshot of wpmgr user account with correct privileges and add it to your documentation |
| 2 | Screenshot of Gold copy with correct properties. |
| 2 | Documentation professionalism meets requires from page 1 of assignment, |
| **20** | **Total Marks for Brightspace Submissions** |
|  |  |
| **60** | **Total Assignment Marks.** |

# Appendix A

1. Where is the login link required on web page.

A screenshot of a company logo

Description automatically generated

Image 1 – Example of Link to log into OSYS database

1. I received an error when logging into phpMyAdmin with admin user.

A screenshot of a computer

Description automatically generated

Image 2 – Ensure you admin has full privileges to mysql database.