**Network Directory Services Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Containerized MySQL, phpMyAdmin, and WordPress Lab**

**Learning activity summary**: This lab activity is intended to reinforce what you already know about containers and their ability to run services that have different purposes in a controlled way. Additionally, you will see how they can be stacked together. Lastly, while there is more we could add, but the idea here is that we’re laying the groundwork for when we go to learn more about what the structured query language (SQL) and how it is used in many different ways (ex. a website application).

**Section 1: Setting up MySQL, phpMyAdmin, and Wordpress services.** Run the following commands using *sudo* listed below and answer the following questions using as much detail as necessary to fully answer the question.

*docker run -d -e MYSQL\_ROOT\_PASSWORD=Password1 -p 3306:3306 --name mysql mysql*

*docker run -d -e MYSQL\_PASSWORD=Password1 -p 8000:80 --link mysql:db cburki/myadmin:*latest

*docker run -d -p 9000:80 --link mysql:mysql wordpress*

What makes each of these containers able to talk to each other?

Would using these environment variables like this be a good idea on a production server? What might be a better solution?

**Section 2: Verify that WordPress installed and explore its ins and outs.**

1. Navigate to <http://localhost:9000> in a web browser. Were you able to access it? Y/N
2. Run through the setup screens. Use ‘demo’ as the site title and fill in the other fields as you are asked.
3. Try Logging into WordPress at <http://localhost:9000/wp-admin> . Did it work? Y/N
4. Explore Wordpress and try changing something on the website. Note what you changed below:
5. What might be some good use cases for knowing how to set this kind of a system up?

**Section 3: Let’s check out phpMyAdmin and see what kind of data we have on our server now.**

1. Navigate to <http://localhost:8000/phpmyadmin> in a web browser. Were you able to access it? Y/N
2. Login using `root` and `Password1`. Were you able to login? Y/N
3. On the left side of the screen you should see a list of databases. Do you see one called *wordpress*? Y/N
4. Click on *wordpress* and notice how it displays a list of tables. What are the names of three of them?
5. Click on the table named *wp\_users* and write down below what you see after it says something like ‘Showing rows 0 – 0 (1 total, Query took 0.0002 seconds.)’ (Hint: The answer should start with *SELECT*)
6. Click on the Structure tab. What do you see?
7. Click on the SQL tab. Type the following statements into the white box and make note of what results you get after running them.

*select ID, user\_email from wp\_users;*

*select NOW();*

*create database ndstest;*

**Section 4: We can interact with mysql via a command prompt directly too.**

1. Head over to a terminal. We need to get the container ID for mysql. Make note of what that is.
2. Next, run the following command:

*docker exec -it <mysql\_container\_id\_goes\_here> /bin/bash*

1. You should be at a shell (inside of a running container, awesome!). What happens if you run *mysql* as a command? Why do you see this?
2. Login using: *mysql -u root –p* (When prompted, type your password). What do you see as a prompt?
3. Type the following commands at the prompt and make note of what results you see (list a few if there are multiple results):

*show databases;*

*use wordpress;*

*show tables;*

*describe wp\_users;*

*select user\_email from wp\_users;*

*use ndstest;*

*drop database ndstest;*

1. Type ‘exit’ when you are done. ‘exit’ again to exit out of the running mysql container.

**Section 5: All Done? Verify with the instructor that everything has been completed and get the magical sign-off.**

**Instructor sign-off: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**