Name: Shanne Edralyn N. Erandio **Date: 11/27/2025**
Course, Year and Section: BSIT 2A

Assessment Laboratory

Bash Scripting + Container Simulation Lab

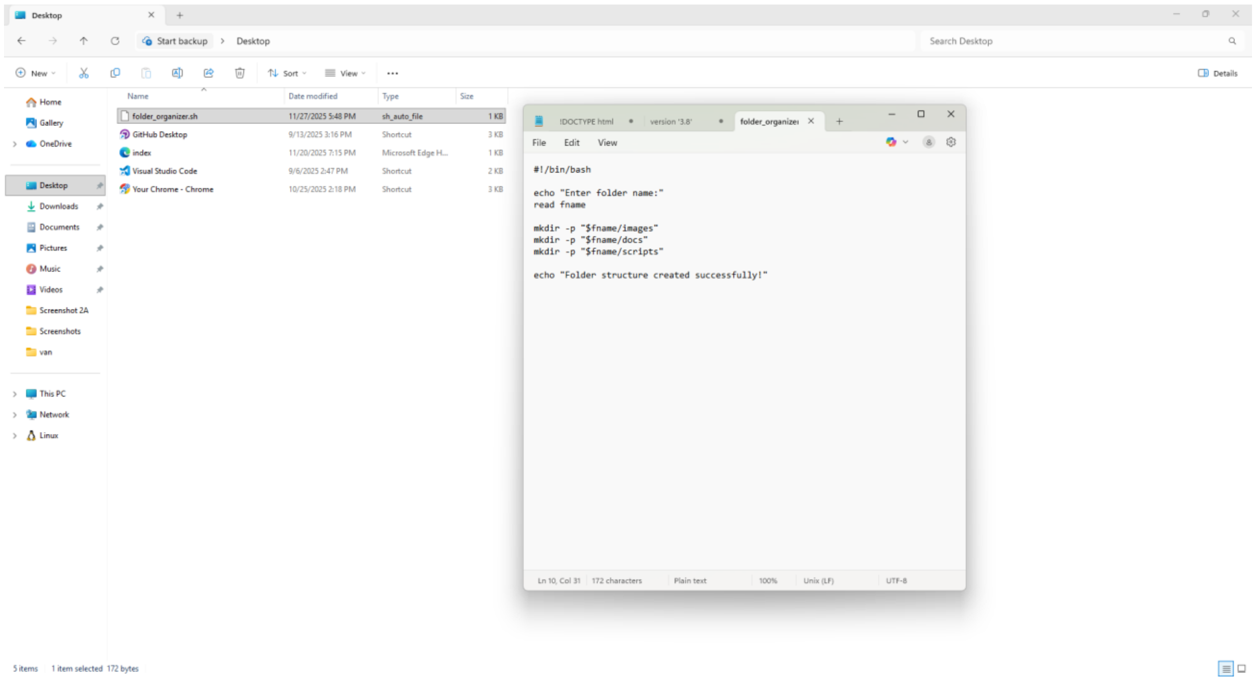
PART 1: Bash Scripting

Folder Organizer

Write a script that:

- Asks the user for a folder name
- Creates it
- Inside the folder, automatically generates:
  - images/
  - docs/
  - scripts/
- Prints "Folder structure created successfully!"

Use mkdir, echo, and read.



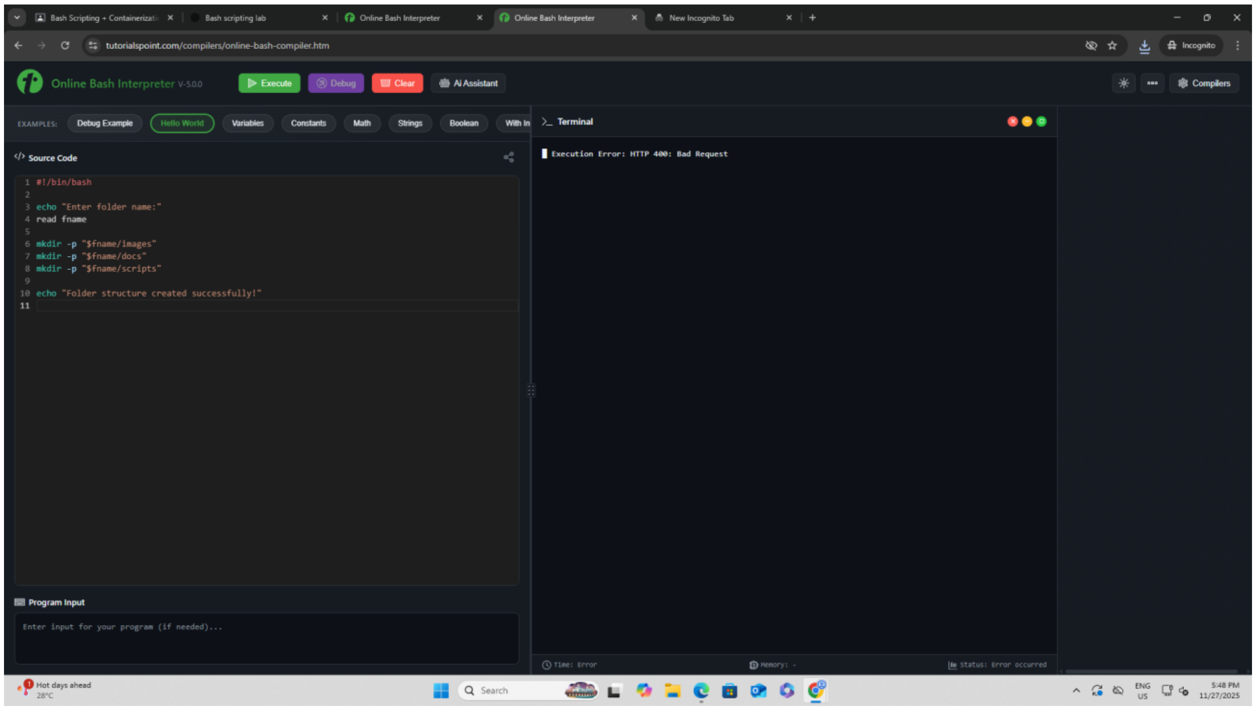
Name	Date modified	Type	Size
folder_organizer.sh	11/27/2025 5:48 PM	sh_auto_file	1 KB
GitHub Desktop	9/13/2025 3:16 PM	Shortcut	3 KB
index	11/20/2025 7:15 PM	Microsoft Edge H...	1 KB
Visual Studio Code	9/6/2025 2:47 PM	Shortcut	2 KB
Your Chrome - Chrome	10/25/2025 2:18 PM	Shortcut	3 KB

```
#!/bin/bash

echo "Enter folder name:"
read fname

mkdir -p "$fname/images"
mkdir -p "$fname/docs"
mkdir -p "$fname/scripts"

echo "Folder structure created successfully!"
```



```
#!/bin/bash

echo "Enter folder name:"
read fname

mkdir -p "$fname/images"
mkdir -p "$fname/docs"
mkdir -p "$fname/scripts"

echo "Folder structure created successfully!"
```

```
Execution Error: HTTP 400: Bad Request
```

Deliverables (Part 1)

- Bash script (.sh or text)
- Screenshot of script executed in the online IDE

PART 2: Container Simulation Task

Objective:

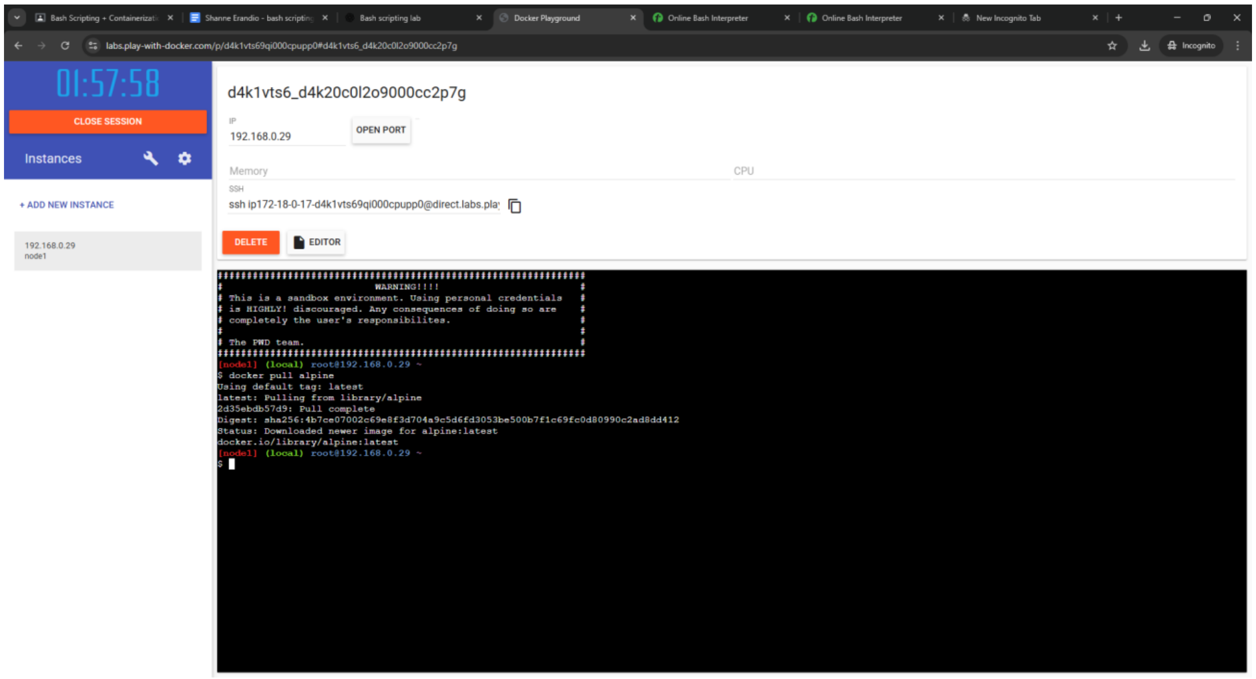
Show basic understanding of container operations using Play With Docker or Katacoda.

Instructions:

Perform the following in the container simulation environment:

1. Pull a Docker image
   Type this:

   docker pull alpine


Bash Scripting + Containerizati
Shanne Erandio - bash scripting
Bash scripting lab
Docker Playground
Online Bash Interpreter
Online Bash Interpreter
New Incognito Tab
labs.play-with-docker.com/p/d4k1vts69qi000cpupp0#d4k1vts6_d4k20c0l2o9000cc2p7g
Incognito
01:57:58
CLOSE SESSION
Instances
+ ADD NEW INSTANCE
192.168.0.29
node1
d4k1vts6_d4k20c0l2o9000cc2p7g
IP
192.168.0.29
OPEN PORT
Memory
CPU
SSH
ssh ip172-18-0-17-d4k1vts69qi000cpupp0@direct.labs.pla
DELETE
EDITOR

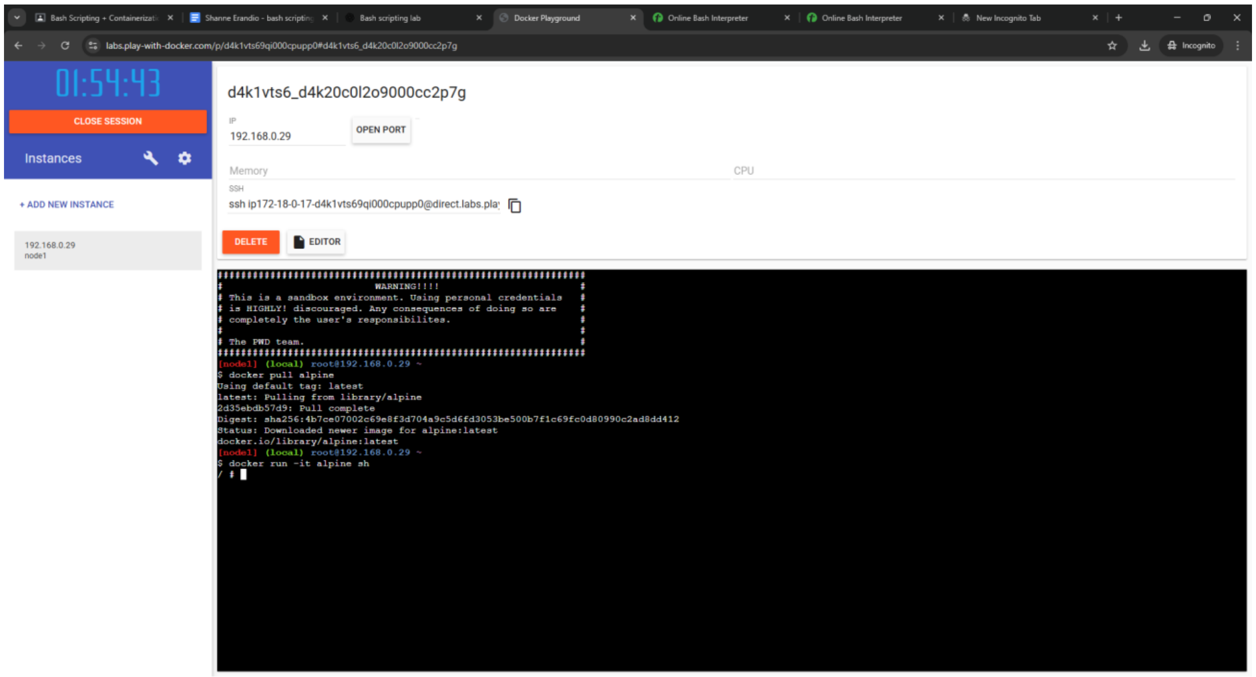
```
###################################################################
#                          WARNING!!!!                            #
# This is a sandbox environment. Using personal credentials      #
# is HIGHLY! discouraged. Any consequences of doing so are       #
# completely the user's responsibilites.                          #
#                                                                 #
# The PWD team.                                                   #
###################################################################
[node1] (local) root@192.168.0.29 ~
$ docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
2d35ebdb57d9: Pull complete
Digest: sha256:4b7ce07002c69e8f3d704a9c5d6fd3053be500b7f1c69fc0d80990c2ad8dd412
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
[node1] (local) root@192.168.0.29 ~
$ 
```

2. Run a container interactively

   Type this:

   docker run -it alpine sh

   ✓ You should see this as a prompt: / #
   ✓ Seeing that means you are inside the container.



```
################################################################
#                          WARNING!!!!                          #
# This is a sandbox environment. Using personal credentials    #
# is HIGHLY! discouraged. Any consequences of doing so are     #
# completely the user's responsibilites.                        #
#                                                               #
# The PWD team.                                                 #
################################################################
[node1] (local) root@192.168.0.29 ~
$ docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
2d35ebdb57d9: Pull complete
Digest: sha256:4b7ce07002c69e8f3d704a9c5d6fd3053be500b7f1c69fc0d80990c2ad8dd412
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
[node1] (local) root@192.168.0.29 ~
$ docker run -it alpine sh
/ #
```

3. Inside the container, run:

   Type this:

   - pwd
     - ✓ You should see this as output: /

   And then

   - Create a file named hello.txt

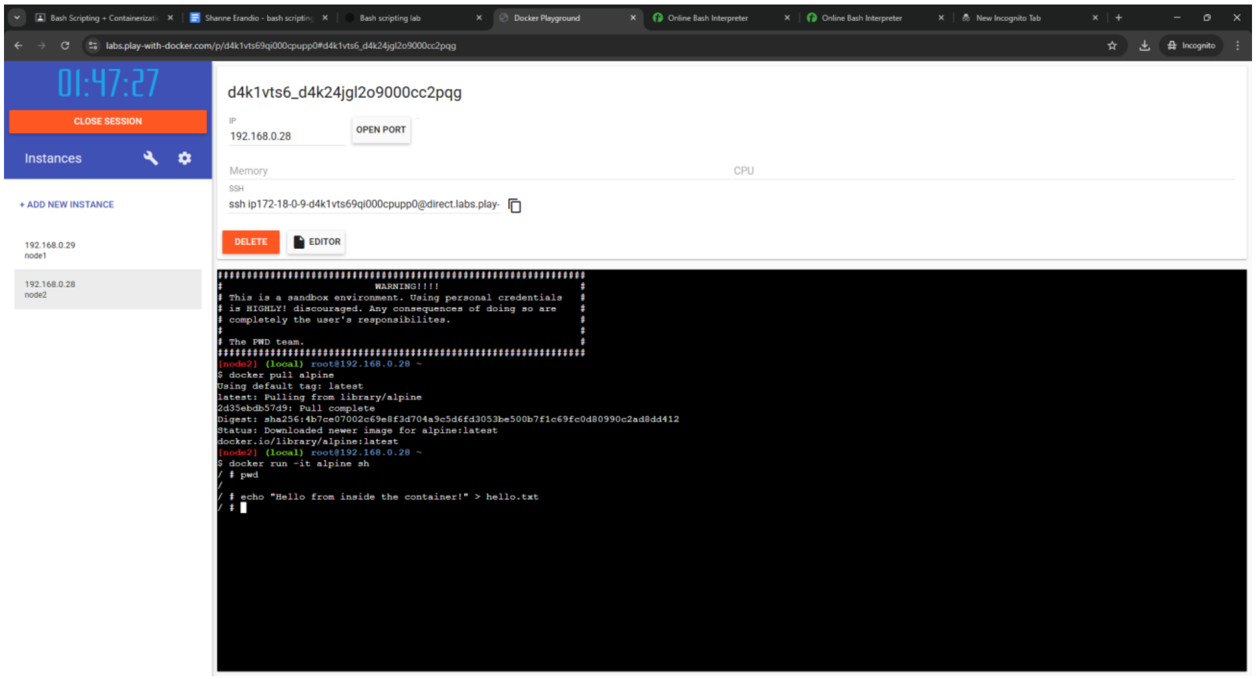
   Type this:

- echo “Hello from inside the container!” > hello.text
  - ✓ Verify if it exists by typing: ls
  - ✓ You should see this: hello.txt

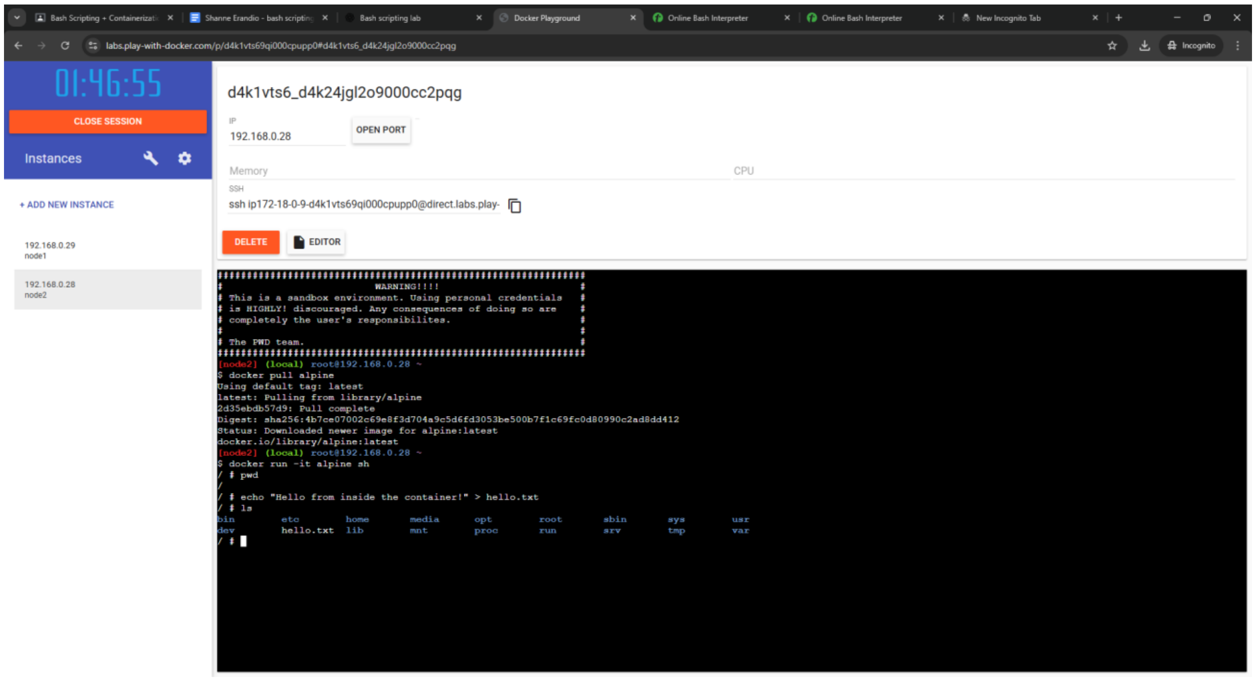
   - Display its contents using cat

   Type this:

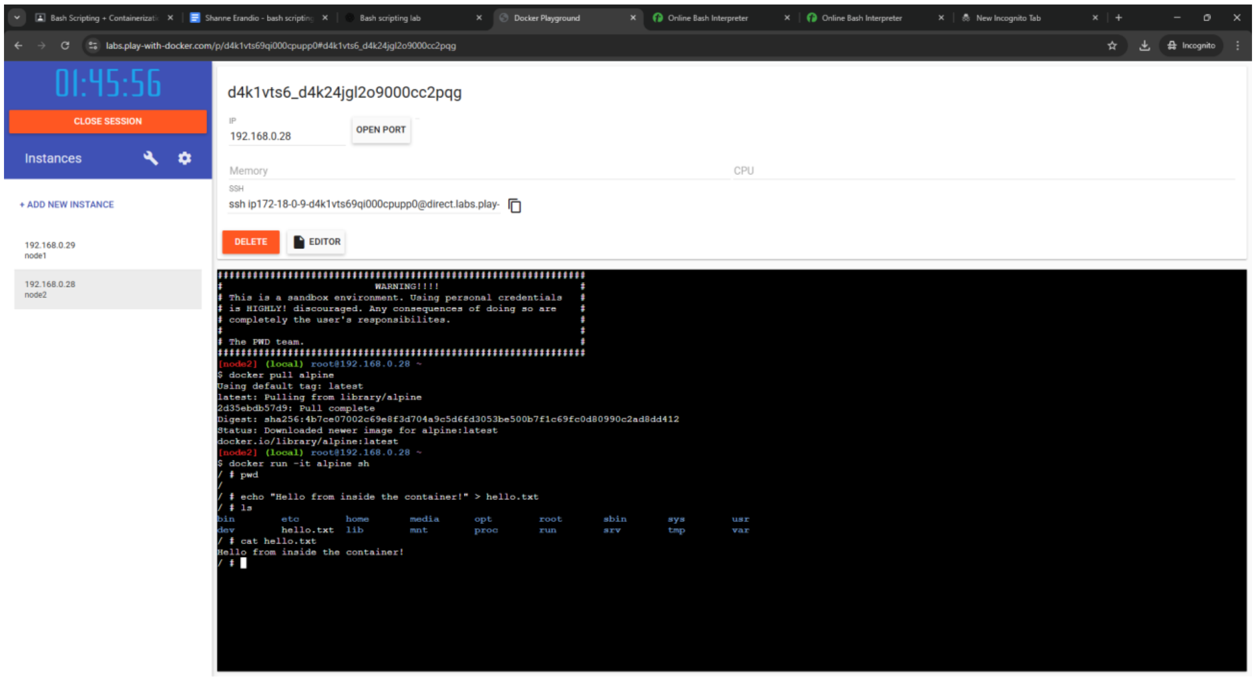
     - cat hello.txt
     - You should see this: Hello from inside the container!



```
################################################################
#                          WARNING!!!!                         #
# This is a sandbox environment. Using personal credentials   #
# is HIGHLY! discouraged. Any consequences of doing so are     #
# completely the user's responsibilites.                       #
#                                                              #
# The PWD team.                                                #
################################################################
[node2] (local) root@192.168.0.28 ~
$ docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
2d35ebdb57d9: Pull complete
Digest: sha256:4b7ce07002c69e8f3d704a9c5d6fd3053be500b7f1c69fc0d80990c2ad8dd412
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
[node2] (local) root@192.168.0.28 ~
$ docker run -it alpine sh
/ # pwd
/
/ # echo "Hello from inside the container!" > hello.txt
/ # 
```



```
################################################################
#                          WARNING!!!!                         #
# This is a sandbox environment. Using personal credentials   #
# is HIGHLY! discouraged. Any consequences of doing so are     #
# completely the user's responsibilites.                       #
#                                                              #
# The PWD team.                                                #
################################################################
[node2] (local) root@192.168.0.28 ~
$ docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
2d35ebdb57d9: Pull complete
Digest: sha256:4b7ce07002c69e8f3d704a9c5d6fd3053be500b7f1c69fc0d80990c2ad8dd412
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
[node2] (local) root@192.168.0.28 ~
$ docker run -it alpine sh
/ # pwd
/
/ # echo "Hello from inside the container!" > hello.txt
/ # ls
bin        etc        home       media      opt        root       sbin       sys        usr
dev        hello.txt  lib        mnt        proc       run        srv        tmp        var
/ # 
```

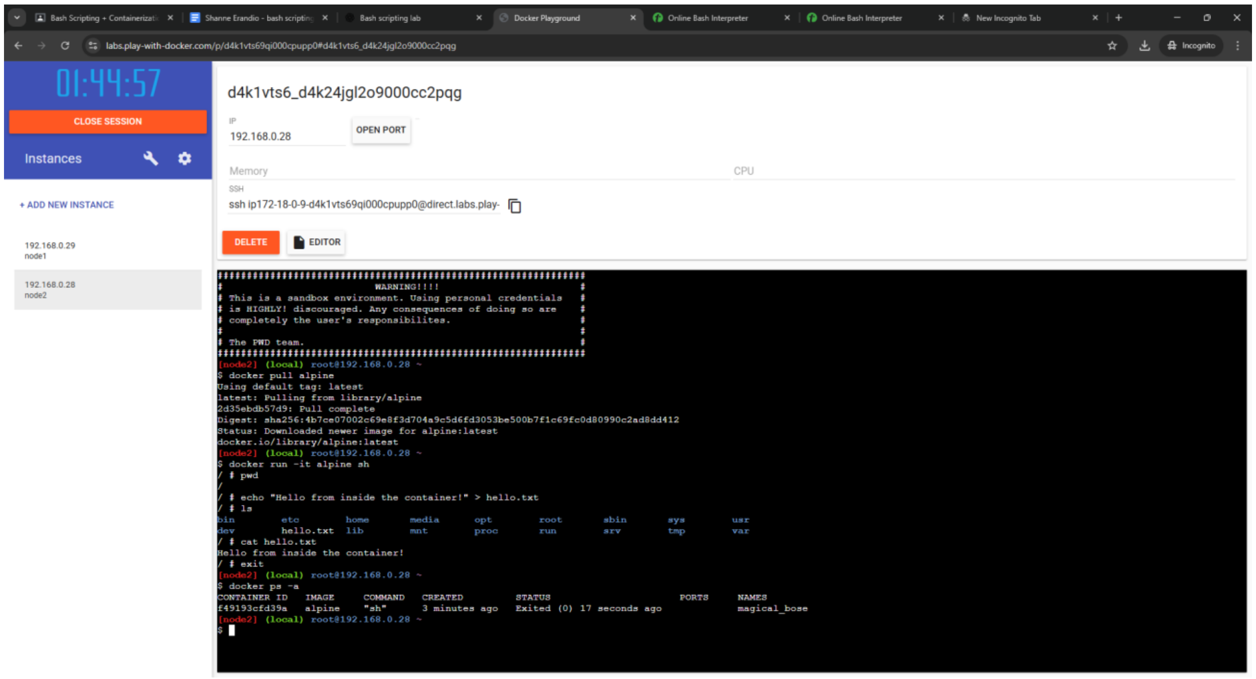


```
###############################################################
#                          WARNING!!!!                        #
# This is a sandbox environment. Using personal credentials   #
# is HIGHLY! discouraged. Any consequences of doing so are    #
# completely the user's responsibilites.                      #
#                                                             #
# The PWD team.                                               #
###############################################################
[node2] (local) root@192.168.0.28 ~
$ docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
2d35ebdb57d9: Pull complete
Digest: sha256:4b7ce07002c69e8f3d704a9c5d6fd3053be500b7f1c69fc0d80990c2ad8dd412
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
[node2] (local) root@192.168.0.28 ~
$ docker run -it alpine sh
/ # pwd
/
/ # echo "Hello from inside the container!" > hello.txt
/ # ls
bin         etc         home        media       opt         root        sbin        sys         usr
dev         hello.txt   lib         mnt         proc        run         srv         tmp         var
/ # cat hello.txt
Hello from inside the container!
/ #
```

4. Exit the container by typing exit

5. List all containers

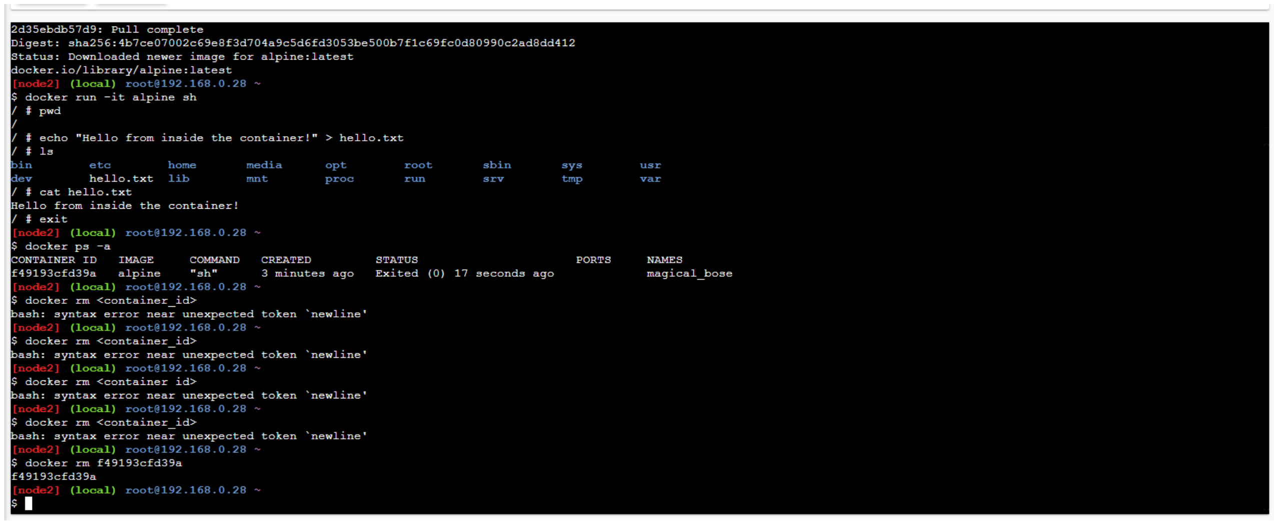
   Type this: docker ps -a



```
###############################################################
#                          WARNING!!!!                        #
# This is a sandbox environment. Using personal credentials   #
# is HIGHLY! discouraged. Any consequences of doing so are    #
# completely the user's responsibilites.                      #
#                                                             #
# The PWD team.                                               #
###############################################################
[node2] (local) root@192.168.0.28 ~
$ docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
2d35ebdb57d9: Pull complete
Digest: sha256:4b7ce07002c69e8f3d704a9c5d6fd3053be500b7f1c69fc0d80990c2ad8dd412
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
[node2] (local) root@192.168.0.28 ~
$ docker run -it alpine sh
/ # pwd
/
/ # echo "Hello from inside the container!" > hello.txt
/ # ls
bin         etc         home        media       opt         root        sbin        sys         usr
dev         hello.txt   lib         mnt         proc        run         srv         tmp         var
/ # cat hello.txt
Hello from inside the container!
/ # exit
[node2] (local) root@192.168.0.28 ~
$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED         STATUS                      PORTS     NAMES
f49193cfd39a   alpine    "sh"      3 minutes ago   Exited (0) 17 seconds ago             magical_bose
[node2] (local) root@192.168.0.28 ~
$
```

6. Remove the container

   Type this: docker rm <container_id>



```
2d35ebdb57d9: Pull complete
Digest: sha256:4b7ce07002c69e8f3d704a9c5d6fd3053be500b7f1c69fc0d80990c2ad8dd412
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
[node2] (local) root@192.168.0.28 ~
$ docker run -it alpine sh
/ # pwd
/
/ # echo "Hello from inside the container!" > hello.txt
/ # ls
bin        etc        home       media      opt        root       sbin       sys        usr
dev        hello.txt  lib        mnt        proc       run        srv        tmp        var
/ # cat hello.txt
Hello from inside the container!
/ # exit
[node2] (local) root@192.168.0.28 ~
$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED         STATUS                       PORTS     NAMES
f49193cfd39a   alpine    "sh"      3 minutes ago   Exited (0) 17 seconds ago              magical_bose
[node2] (local) root@192.168.0.28 ~
$ docker rm <container_id>
bash: syntax error near unexpected token `newline'
[node2] (local) root@192.168.0.28 ~
$ docker rm <container_id>
bash: syntax error near unexpected token `newline'
[node2] (local) root@192.168.0.28 ~
$ docker rm <container id>
bash: syntax error near unexpected token `newline'
[node2] (local) root@192.168.0.28 ~
$ docker rm <container_id>
bash: syntax error near unexpected token `newline'
[node2] (local) root@192.168.0.28 ~
$ docker rm f49193cfd39a
f49193cfd39a
[node2] (local) root@192.168.0.28 ~
$ 
```

Deliverables (Part 2)

- Screenshot of pulling the image
- Screenshot of running the container
- Screenshot inside the container
- Screenshot of listing and removing the container

REFLECTION

While creating the script, I learned how Bash handles user input and directory creation. Using `mkdir -p` was helpful because it prevents errors even if folders already exist. Testing the script in the online Bash environment also helped me understand execution permissions and how to run scripts properly using `chmod +x` and `./filename.sh`.

This part taught me how containers work in an isolated environment. I understood how images serve as templates and how containers act as lightweight virtual environments. I also learned the importance of correct syntax, especially when removing containers (using the real container ID instead of placeholder brackets).

The activity helped me become more familiar with container lifecycle commands such as **pull**, **run**, **inspect**, **exit**, **list**, and **remove**.

This laboratory activity strengthened my foundational skills in Bash scripting and container-based workflows. The hands-on practice provided a clearer understanding of automation, file operations, and Docker container management. Overall, this lab improved my confidence in using command-line tools and understanding containerization concepts, which are essential in modern computing environments.

LEARNING OUTCOMES

Students will be able to:

- Write and execute Bash scripts in an online environment
- Use basic Bash constructs: variables, loops, conditionals, input/output
- Demonstrate familiarity with container operations: pull, run, exec, list, remove
- Understand the concept of isolated environments
- Apply automation and command-line skills

Criteria (20 pts each)	**Excellent (20 pts)**	**Very Good (19-17 pts)**	**Good (16-14 pts)**	**Needs Improvement (13-10 pts)**	**Poor (9-0 pts)**
1. Code Correctness & Functionality	Program runs flawlessly, meets all case study requirements, produces correct output in all test cases.	Runs with minor issues but overall correct; 1 small error that does not break functionality.	Produces partial correct output; some logic errors but main idea works.	Major errors, incorrect logic, or missing required features; program barely runs.	Does not run, incomplete, or does not match chosen case study.
2. Use of Assembly Concepts (loops, conditionals, segments)	Demonstrates excellent understanding; uses loops/conditionals correctly; segments structured properly.	Proper use of concepts with minor mistakes; logic is mostly correct.	Uses only required structures but with noticeable mistakes or inefficiencies.	Incorrect or missing loops/conditionals; poor segment structure.	No use of required Assembly concepts.
3. Code Organization & Comments	Code is clean, well-organized, clearly commented; comments explain purpose of major instructions.	Well-commented but missing small clarifications; organization still clear.	Some comments present but incomplete or unclear; formatting inconsistent.	Very few comments; messy code structure; hard to read.	No comments; disorganized and difficult to understand.
4. Output Quality (Screenshot & Execution)	Clear screenshot showing correct input and output; format is professional and complete.	Screenshot shows output clearly but missing minor details (e.g., prompt).	Screenshot present but slightly unclear or incomplete.	Screenshot is low-quality, barely readable, or missing key parts.	No screenshot or completely unreadable.
5. Reflection & Understanding	Reflection is insightful, specific, and explains concepts learned and	Good reflection with clear understanding; missing a bit of detail.	Basic reflection; mentions learning but not specific.	Reflection lacks clarity or detail; shows minimal understanding.	No reflection or irrelevant submission.

	challenges. Shows deep understanding.				