**Lab 4: Bash Loops with Docker Image Management**

Objective:*Implement loop constructs in Bash for Docker image management*

Tasks:

1. Write a script that lists all Docker images.

2. Use a loop to remove each image interactively.

Documentation:

- Bash loop structures.

- Docker image management through Bash

Prerequisites:

1- An AWS account with administrative access.

2- Docker Deep Dive Course

3- Bash Script Deep Dive Course

Implementation:

1- Create a file with ‘.sh’ extension e.g. **‘docker\_image\_management.sh’.**

Script used in Lab 4 that demonstrates how to use Bash loops for Docker image management:

| #!/bin/bash  # Task 1: List all Docker images echo "Listing all Docker images:" docker images  # Task 2: Remove each image interactively using a loop echo "Do you want to remove the Docker images? (yes/no)" read confirmation  if [ "$confirmation" == "yes" ]; then  # Loop through the images and remove them one by one  echo "Removing Docker images..."  for image\_id in $(docker images -q); do  echo "Removing image $image\_id"  docker rmi $image\_id  done  echo "All Docker images have been removed." else  echo "No Docker images have been removed." fi |
| --- |

2- Save the file with ‘.sh’ extension and grant it execute permission **(chmod +x docker\_image\_management.sh).**

3- Run it using **( ./docker\_image\_management.sh )** in a terminal.

Explanation:

We start by listing all Docker images using the docker images command.

We then ask the user for confirmation to remove the Docker images interactively. The read command is used to get the user's input.

If the user confirms by typing "yes", a loop is initiated using the for loop construct. The loop iterates through the list of image IDs obtained from docker images -q.

Inside the loop, each image is removed using the **docker rmi** command with the respective image ID.

After all images have been removed, a message is displayed indicating that all Docker images have been removed.

If the user did not confirm image removal, a message is displayed stating that no images were removed.

Instructions:

Make sure to save this script in a .sh file **(e.g., docker\_image\_management.sh)**, grant it execute permissions **(chmod +x docker\_image\_management.sh)**, and then run it using **./docker\_image\_management.sh** in a terminal.

Note:

Please note that using docker rmi will permanently remove images from your system. Be cautious when using this script on production systems or valuable images. It's a good practice to have backups or be certain of what you're removing.