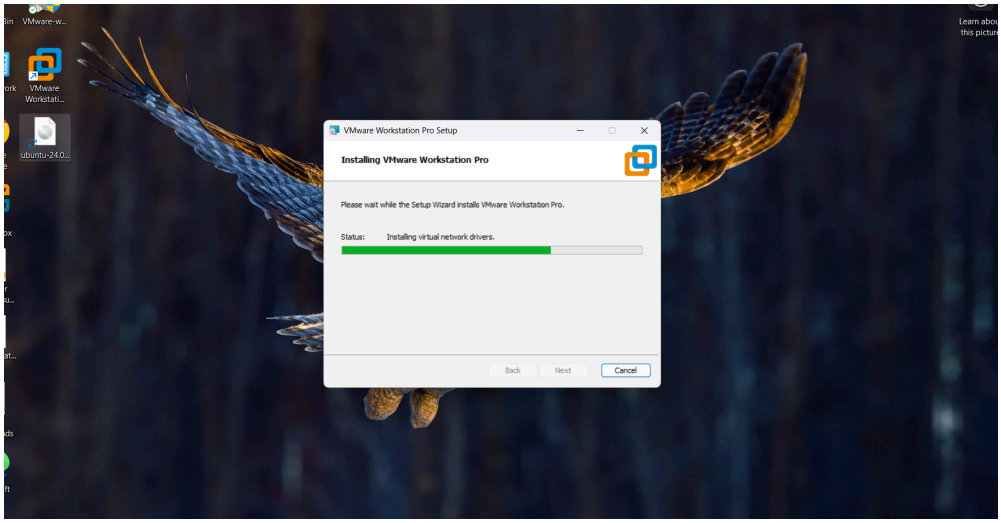
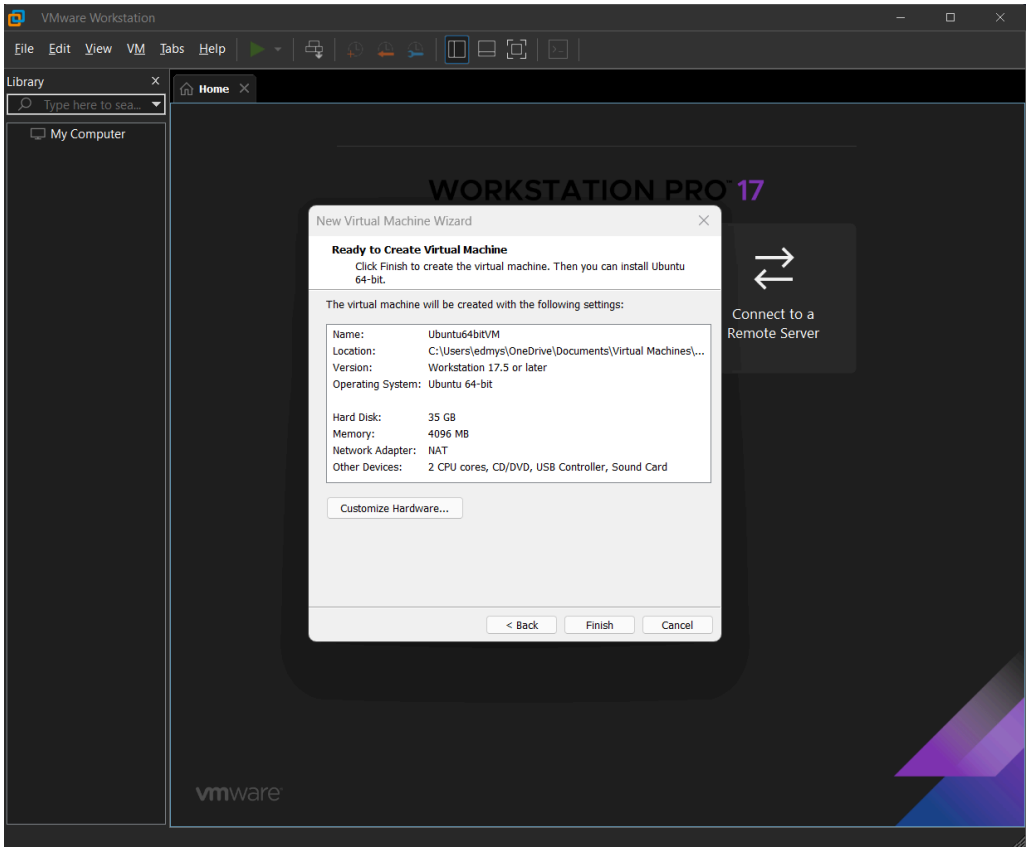
DELIVERABLES FOR TASK #1

Screenshot of installation in progress on Task 1.1

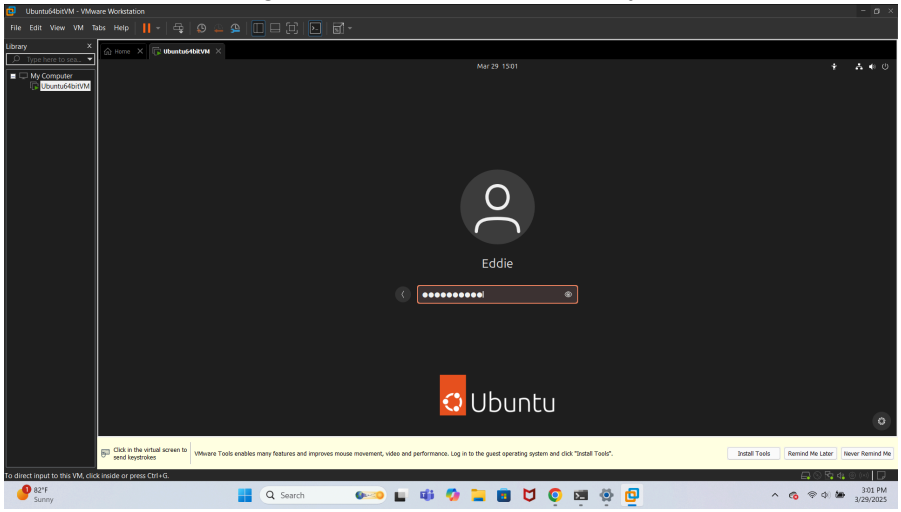


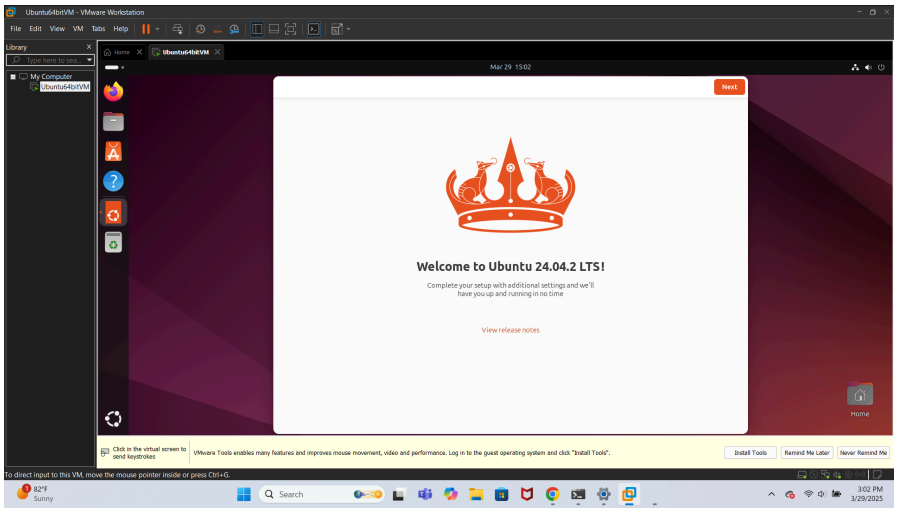
Screenshot of window that shows you the summary of your VM Configuration in Task 1.2

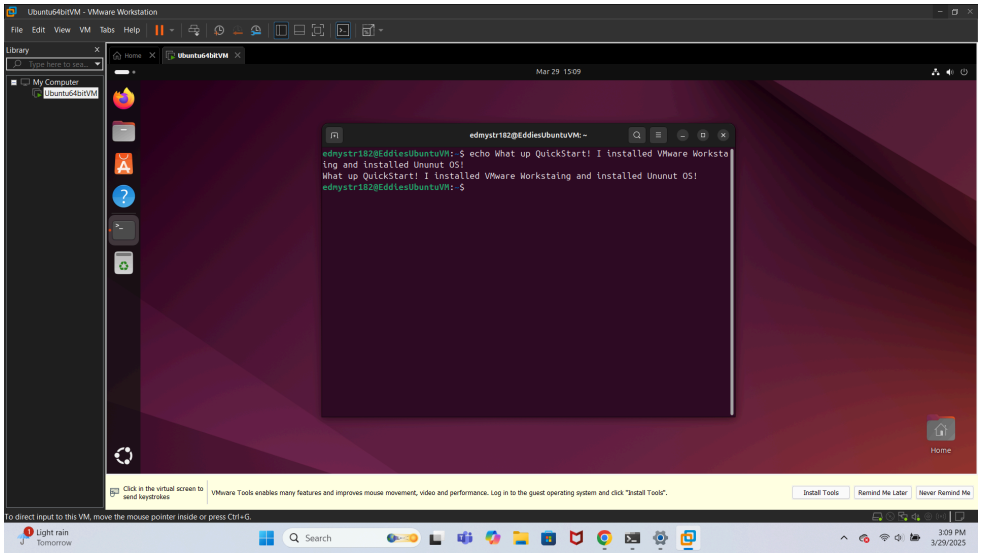


DELIVERABLES FOR TASK #2

Screenshot of the login screen of Ubuntu where you need to enter the username and password.







DELIVERABLES FOR TASK #3

1. Exercise 1 Tasks:

a.

```
edmystr182@EddiesUbuntuVM:~$ pwd
/home/edmystr182
edmystr182@EddiesUbuntuVM:~$
```

edmystr182@ = **Username**
EddiesUbuntuVM = **VM name (Host Name)**
~ = **Home directory (/home/edmystr182)**

---

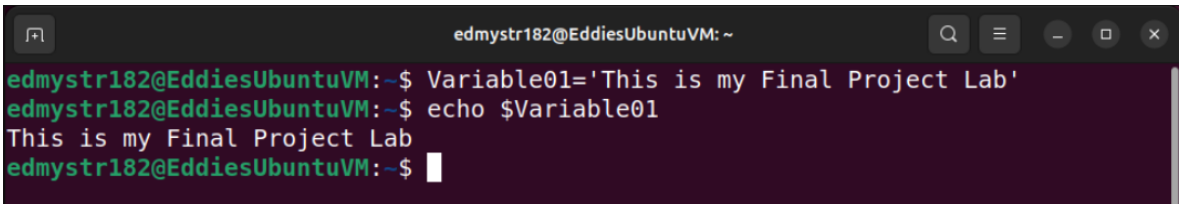
b. ls stands for “list” and it lists the files and directories in the current working directory

- ls -a: List hidden files
- ls -l: Displays long listing format: permissions, owner, size, date, and name.
- ls -la: Combines commands -l and -a
- ls -r: Reverses the order of the output (works with other options too)
- ls -R: Lists contents of subdirectories as well, going deep into folder structures.
- ls *.txt: Only lists files that match the pattern (e.g., all .txt files).

---

c. A variable is like a label that stores data for use later.

- In the below example the label “Variable01” is given to the data “This is my Final Project Lab”. Then it is called on by the command “echo” and then the data is printed on the screen. It is useful data that is called on often.

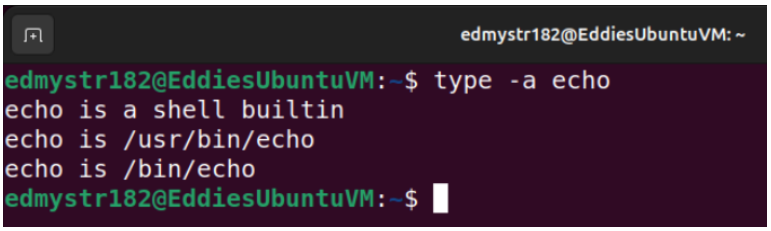


2. Exercise 2 Tasks:

a. Define Internal and External commands with examples.

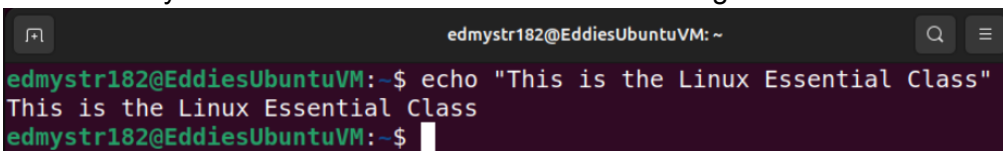
Internal Tasks	External Tasks
Definition: Built into the shell (like bash), executed directly by the shell without needing an external program. Faster to execute since they don’t require launching a new process. **Examples:** cd – Changes the current directory. echo – Displays a message or variable value.	**Definition:** Stored as separate executable files on the system (usually in /bin, /usr/bin, etc.). The shell calls the program from the filesystem. **Examples:** ls – Lists directory contents. cat – Displays the contents of a file.

b. Type -a echo



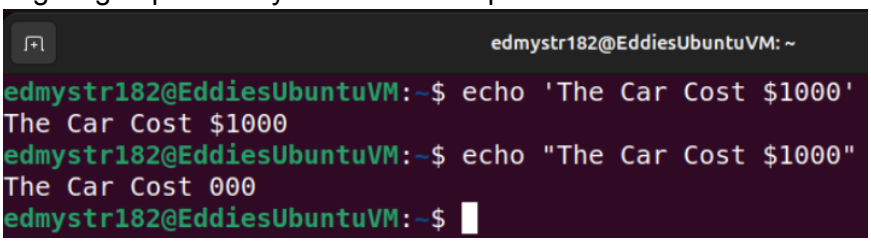
These results mean that the echo command is built into the shell, but there is also an external binary version at /usr/bin/echo and /bin/echo

c. Double quoting: Allows variable expansion, command substitution, and escape sequences like \n. It is useful when you want to include variable values in strings.



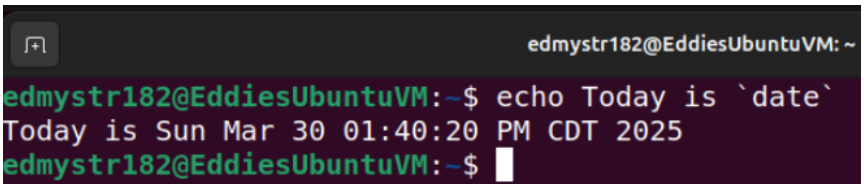
d. Single Quoting: Everything is taken literally—no variable expansion, no command substitution. It is useful when you want to preserve exact content.

e. By using single quotes. If you use double quotes the command will read the '$' as a variable.



f. The **backslash (\)** is used in Bash as an escape character, meaning it tells the shell to treat the next character literally.
Single Quotes: The backslash loses its meaning and will be printed with the data.
Double Quotes: Backslash can escape certain characters: $, ", `, \, and newline. Useful to include those special characters in a string.

g.

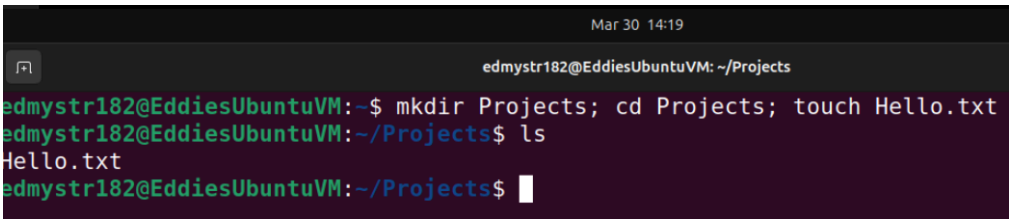


3. Exercise 3 Tasks:

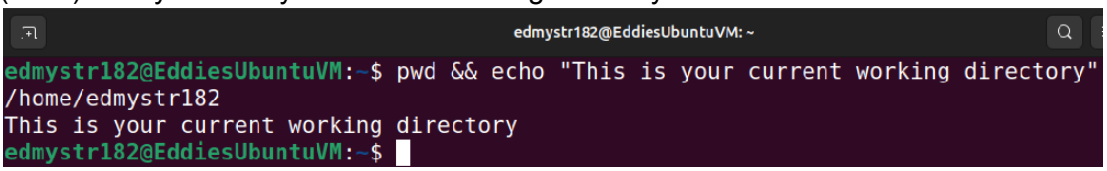
a. **Definition:** Control statements are special syntax in Bash that control the flow of command execution. They determine when and how commands run, especially in sequences, conditions, and loops.

b. **Basic Control Statements:**
(;) Semicolon: Run multiple commands one after another, regardless of success.
(&&) Double Ampersand: Run the next command only if the previous one succeeds.
(||) Double Pipe: Runs the next command only if the previous one fails
(if) If Statement: Executes commands based on a true/false condition.
(elif) Else If Statement: Adds more conditions to an if block.
(else) Else Statement: Executes if none of the previous if or elif conditions were true.
(for) For Loop: Repeats commands for a list of items.
(while) While Loop: Repeats commands while a condition is true.

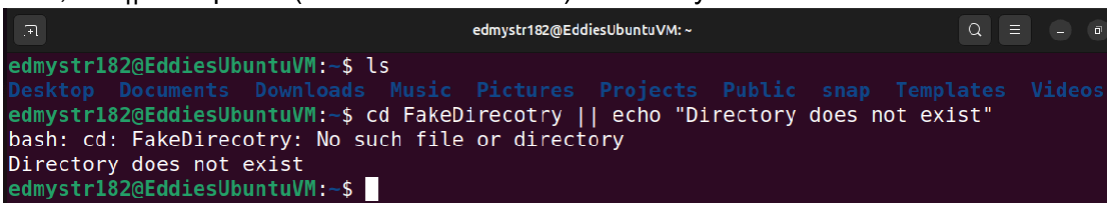
c. **Example of (;):** The below command created a directory, moved into the directory, and created a .txt file in that new directory by using (;) between commands.



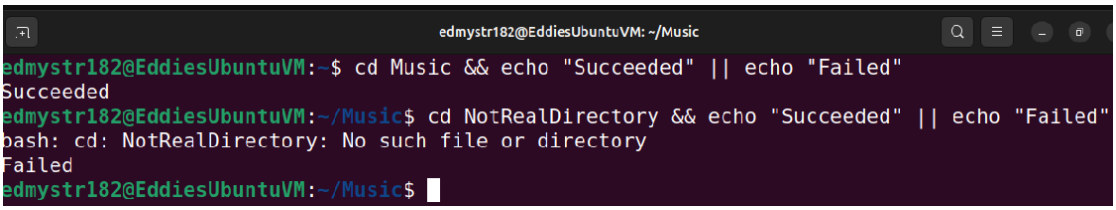
Example of (&&): The below command printed the current directory, then if (pwd) succeeds used (echo) to say “This is your current working directory”



Example of (||): The below command attempts to change directories into a directory that does not exist, the || then prints (if the command fails) “Directory does not exist”



Bonus (&&) with (||) together: The below command uses both and prints “Succeeded” if the command works and “Failed” if it does not.

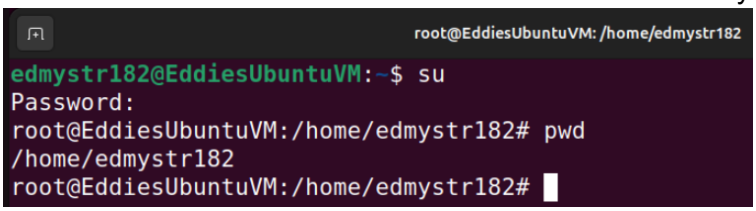


DELIVERABLES FOR TASK #4

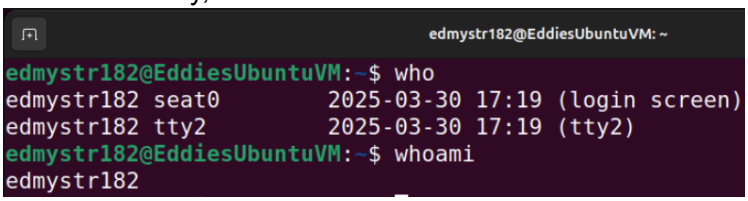
1. What is the Administrator account?
    a. The administrator account refers to the root user. The root user is the superuser with full control over the system. The root user can install software, modify system files, create/delete users, and override permissions.
2. What is meant by switching accounts?
    a. You are usually not logged in directly as root. Instead, use the (sudo) command. This runs a command with root privileges, if your user is in the sudo group.
3. What are system accounts?
    a. System accounts are user accounts created by the system, not by human users. They are used to run background services, daemons, and handle specific system tasks.
    b. Key Characteristics:
        - ➢ Usually have user IDs (UIDs) below 1000 (on most Linux distros).
        - ➢ Do not have a home directory or login shell (or use /usr/sbin/nologin or /bin/false).
        - ➢ Not meant for interactive login.
        - ➢ Help isolate processes and improve security by assigning services their own identities.
    c. Common System Accounts:
        - ➢ **root:** Superuser / Admin
        - ➢ **nobody:** Used for processes with the least privilege
        - ➢ **www-data:** Runs the web server (e.g., Apache, Nginx)
        - ➢ **daemon:** Generic account for daemons
        - ➢ **sshd:** Runs the SSH service

➢ **systemd-network:** Manages network services

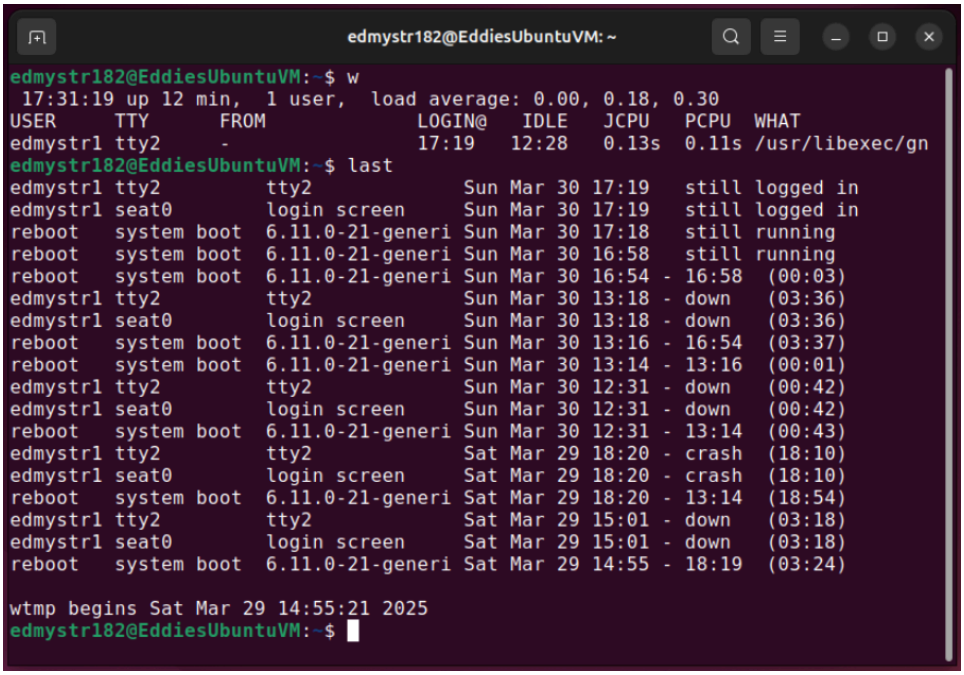
4. What does the (su) command do?
   a. The su command stands for "substitute user". It is usually used to switch to root.



5. The who command in Linux is used to show who is currently logged into the system.
   What It Displays:
   a. Username of logged-in users
   b. Terminal (TTY) they're using
   c. Date and time they logged in
   d. Possibly, their IP address or hostname



6. (**w) command**: Shows a detailed overview of who is logged in and what they are doing on the system.
   (**last) command**: Shows a history of logins on the system — including user logins, reboots, and shutdowns.



Bonus Work: Personalization and Scripting

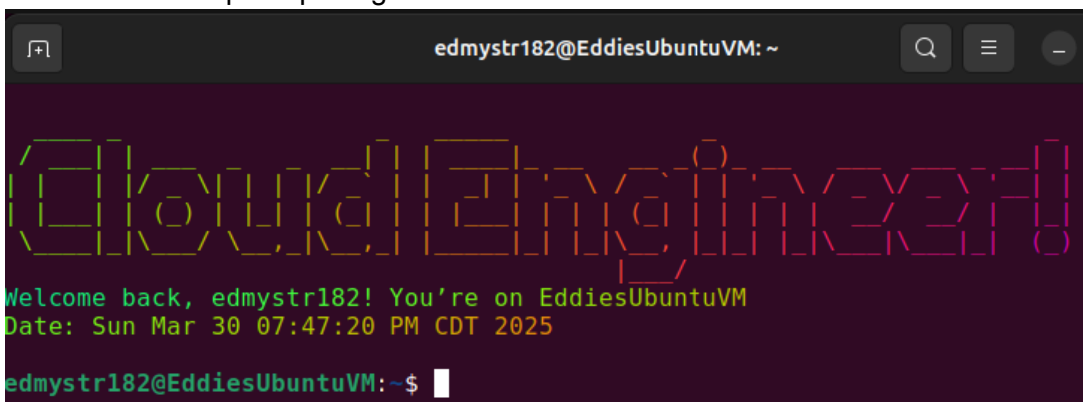
Objective #1: Create a personalized welcome message

Steps:

1. Download figlet, toilet, and lolcat
   a. sudo apt install figlet (Bubble Letters)
   b. sudo apt install toilet (Colors and Effects)
   c. sudo apt install lolcat (Rainbow Colors)
2. Create Welcome Script
   a. Run command: “nano ~/.bashrc” and add to the bottom
   b.

```
# === Custom Welcome Banner ===
echo
figlet "Cloud Engineer!" | lolcat
echo "Welcome back, $USER! You're on $(hostname)" | lolcat
echo "Date: $(date)" | lolcat
echo
```

3. Show welcome banner upon opening terminal
   a.



Objective #2: Check disc space and add a warning if disc space is over a 80% Threshold.

1. Add the below script to ~/.bashrc

```
# === Disk Space Check ===
THRESHOLD=80
USAGE=$(df / | grep / | awk '{ print $5 }' | sed 's/%//g')

if [ "$USAGE" -gt "$THRESHOLD" ]; then
  echo "⚠️ Warning: Disk usage is above ${THRESHOLD}% (${USAGE}%)" | lolcat
else
  echo "✅ Disk usage is healthy: ${USAGE}%" | lolcat
fi
```