**Name Rimsha Amzat**

**Course DevOps**

**Submitted to Sir Sajjad**

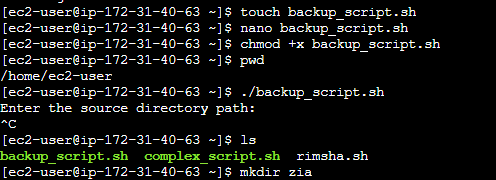
Assignment

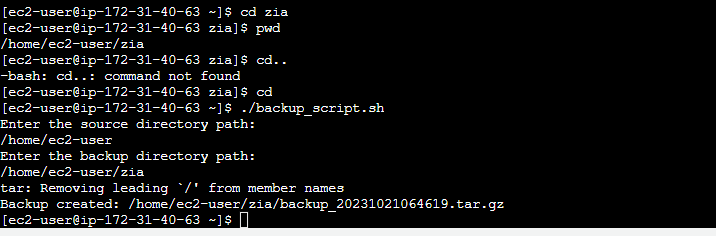
# Bash Script: Perform a Backup of a Directory by Creating a Compressed Archive

Save the below code in a file with a .sh extension, such as backup\_script.sh. Make the file executable by running the command chmod +x backup\_script.sh in the terminal. Then you can execute the script and provide the source directory and backup directory paths.

The script prompts the user for the source directory and backup directory paths. It then checks if the source directory exists. If the backup directory doesn't exist, it creates it using the mkdir command. The perform\_backup function is called with the source directory and backup directory paths as arguments. Inside the function, a timestamp is generated to create a unique backup filename. The tar command is used to create a compressed archive of the source directory in the specified backup directory.

After the backup is created, the script displays the backup filename and path to the user.





You can further customize this script by adding error handling, excluding specific files or directories from the backup, or implementing additional backup strategies according to your requirements.

#!/bin/bash

# Function to perform directory backup

perform\_backup() {

source\_directory="$1"

backup\_directory="$2"

# Check if the source directory exists

if [ ! -d "$source\_directory" ]; then

echo "Source directory '$source\_directory' does not exist."

exit 1

fi

# Create a timestamp for the backup filename

timestamp=$(date +"%Y%m%d%H%M%S")

# Generate the backup filename

backup\_filename="backup\_$timestamp.tar.gz"

# Create the backup by excluding certain files or directories (if needed)

tar -czf "$backup\_directory/$backup\_filename" --exclude="\*.log" "$source\_directory"

if [ $? -eq 0 ]; then

echo "Backup created: $backup\_directory/$backup\_filename"

else

echo "Backup failed."

exit 1

fi

}

# Prompt the user for the source directory and backup directory

echo "Enter the source directory path:"

read source\_directory

echo "Enter the backup directory path:"

read backup\_directory

# Check if the backup directory exists, if not, create it

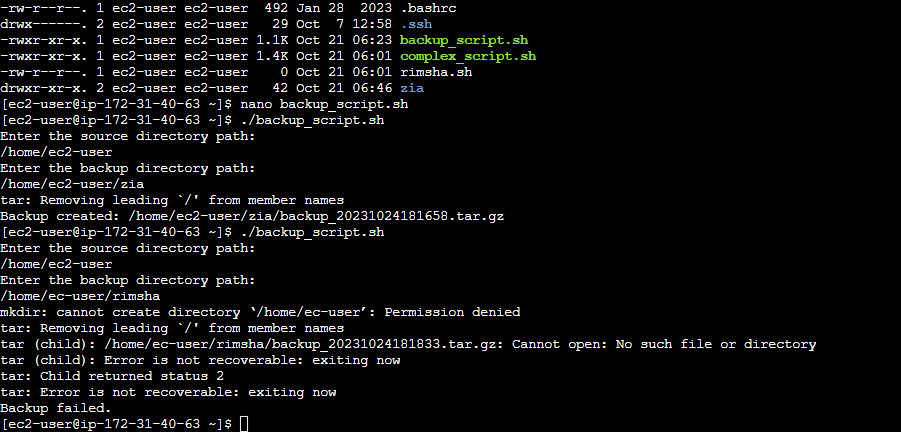
if [ ! -d "$backup\_directory" ]; then

mkdir -p "$backup\_directory"

fi

# Call the function to perform the backup

perform\_backup "$source\_directory" "$backup\_directory"



Here are the changes made:

1. Error Handling: The script now checks if the source directory exists before proceeding with the backup. If the source directory doesn't exist, it will display an error message and exit.
2. Excluding Files or Directories: You can modify the **tar** command to exclude specific files or directories by using the **--exclude** option. In this example, it excludes files with the **.log** extension. You can adjust this as needed.
3. Error Checking: After the backup operation, the script checks the exit code of the **tar** command to ensure the backup was successful. If the backup fails, it displays an error message and exits.

These improvements make the script more robust and capable of handling errors and customizing what gets included or excluded from the backup.