Placement Empowerment Program

Cloud Computing and DevOps Centre

Create a Simple Backup Script: Create a script that backs up your entire Git repository to a local folder daily

Name: Venakata Sai M Department: CSE

Introduction

Backing up your Git repositories is a crucial part of version control management. It ensures that your work remains safe and accessible even in the event of unforeseen data loss, such as accidental deletions, hardware failures, or repository corruption. Automating this process saves time, reduces manual intervention, and guarantees regular updates.

Overview

This task involves creating an automated backup system for your Git repository on a Windows machine using a batch script and Task Scheduler. The script pulls the latest changes from the repository daily and stores them in a backup directory. Additionally, it compresses the repository into a timestamped archive for easy organization. The process ensures that your codebase and its version history are safely stored in a local directory.

Key Components:

1. Batch Script: A .bat file is used to execute commands such as cloning the repository, pulling updates, and compressing the backup.

2. Task Scheduler: A built-in Windows tool is used to automate the script, ensuring it runs daily without manual intervention.

Objectives

1. Automate Backups: Develop a script to back up the entire Git repository daily.

2. Minimize Data Loss: Safeguard the repository from accidental deletions or hardware failures.

3. Ease of Management: Create timestamped backups for quick identification and restoration.

4. Hands-Free Automation: Leverage Task Scheduler to eliminate the need for manual execution.

Importance

1. Disaster Recovery: In case of repository failures or accidental deletions, you can quickly restore your work from the local backup.

2. Version History Preservation: All changes and version history are secured, ensuring no progress is lost.

3. Efficient Workflow: Automating the process allows you to focus on development tasks instead of managing backups manually.

4. Organization: Timestamped backups provide a clear, structured way to keep track of changes over time.

Step-by-Step Overview

Step 1:

Create a folder named GitHub Backup Folder to store your Backup files

Step 2:

Open Notepad and type this script . Make sure that in set REPO_URL give the URL of the repository you want to backup and in set BACK_DIR give the file path of the folder which you created in first step . Then save it as **.bat format** (eg:backup.bat) in Desktop

```
@echo off

Variables

set REPO_URL=https://github.com/Venkatasaim07/website

set BACKUP_DIR=C:\Users\Hi\Desktop\GitHub Backup Folder

set CURRENT_DATE=%date:~10,4%-%date:~4,2%-%date:~7,2%

:: Ensure backup directory exists

if not exist "XBACKUP_DIR%" mkdir "XBACKUP_DIR%"

:: Navigate to the backup directory

cd /d "%BACKUP_DIR%"

:: Check if the repository is already cloned

if not exist "repo" (

echo Cloning repository for the first time...
```

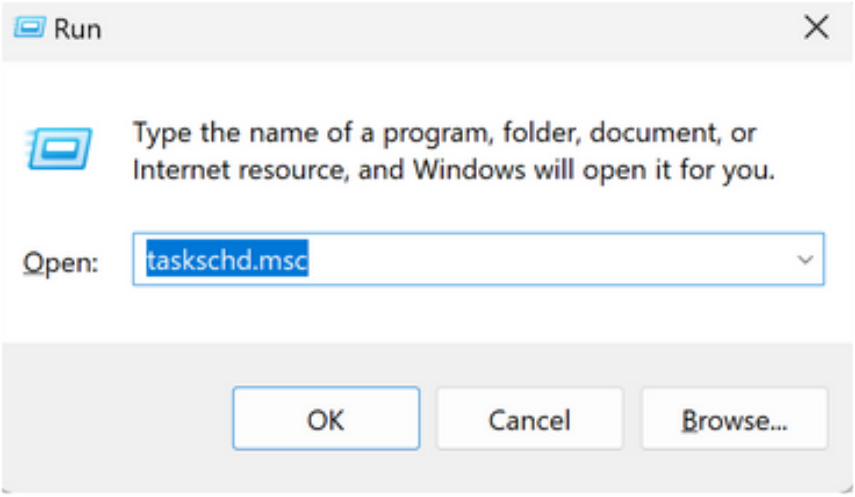
Step 3:

Press **Win + R** on your keyboard.

A small "Run" dialog box will pop up.

Type **taskschd.msc** (without quotes) in the Run box.

Press Enter or click OK.This will open the Task Scheduler window.

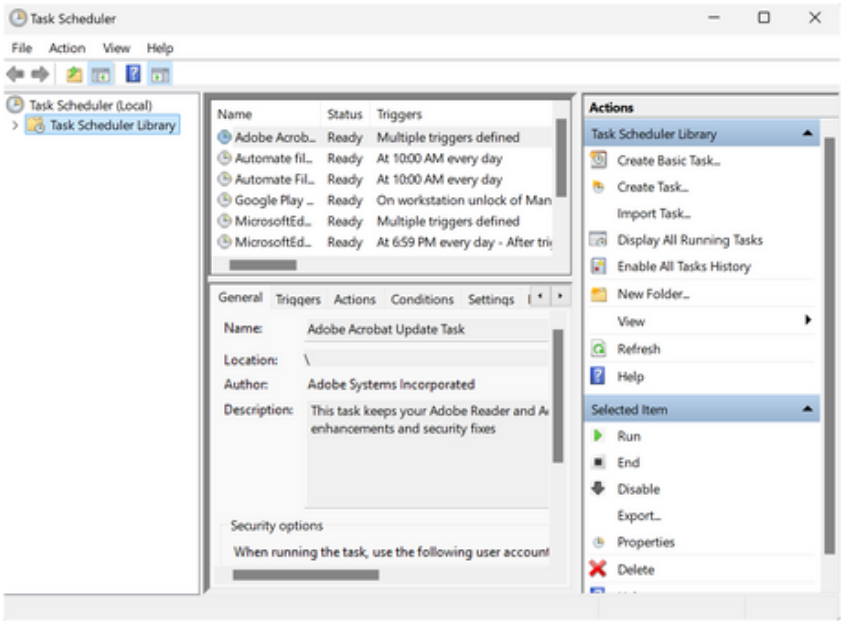


Step 4:

In the Task Scheduler window, look to the right-hand side for a button called "Create Basic Task".

Click it.

A wizard will open to guide you through the setup.



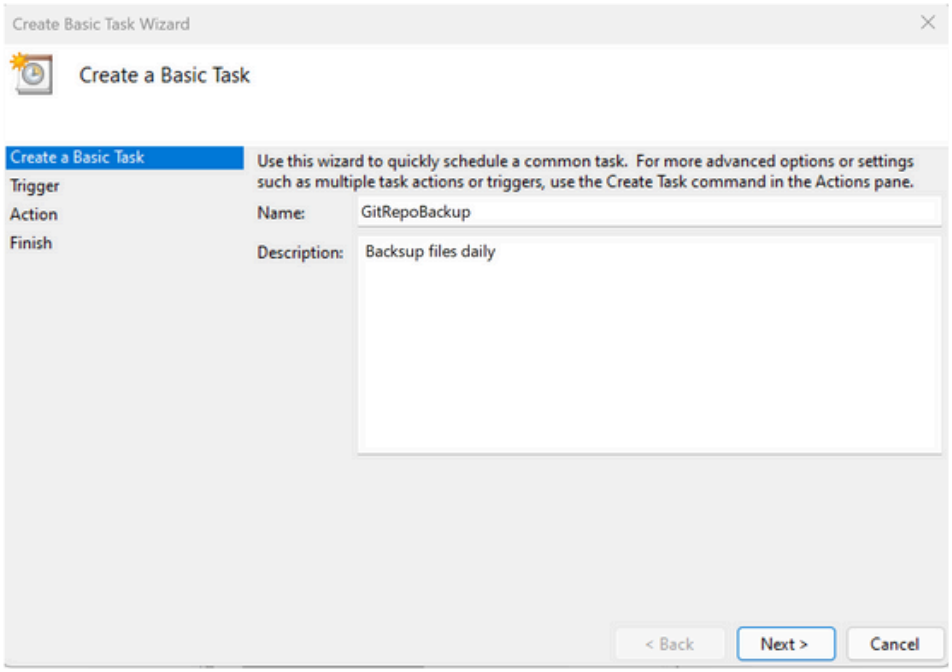
Step 5:

1. Enter a Name for the Task:

For example: "GitRepoBackup".(This can be anything that helps you remember what the task does.)

Optionally, you can add a description like "Backsup files daily".

1. Click Next to continue.



Step 6:

Choose a Schedule:

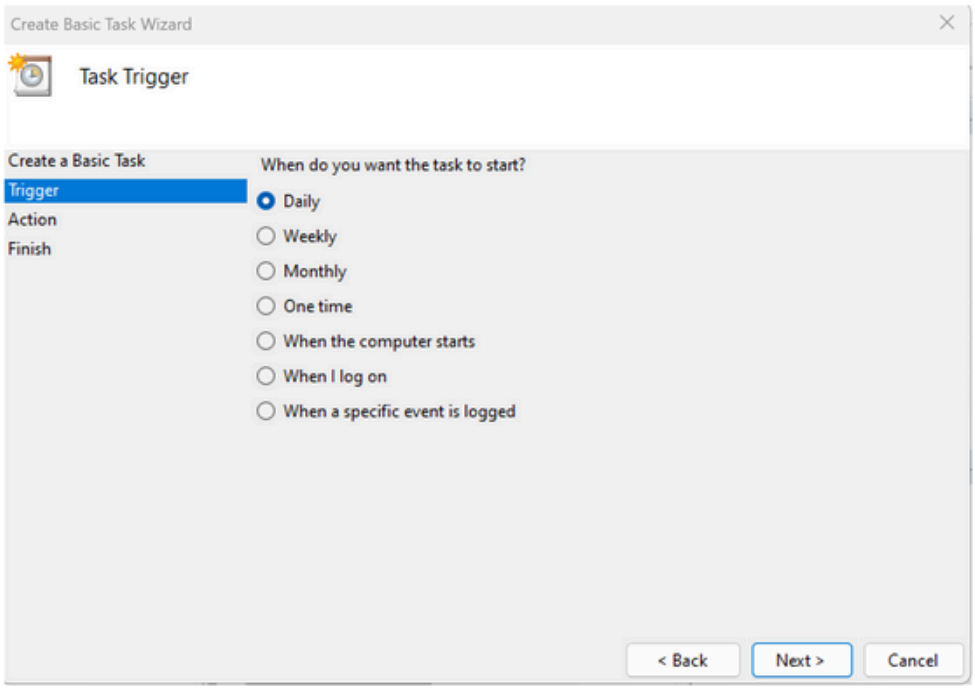
You will see options like:

Daily (runs every day).

Weekly (runs once a week).

One time (runs only once at a specific time).

Choose what works for you (e.g., Daily) and click Next.



Step 7:

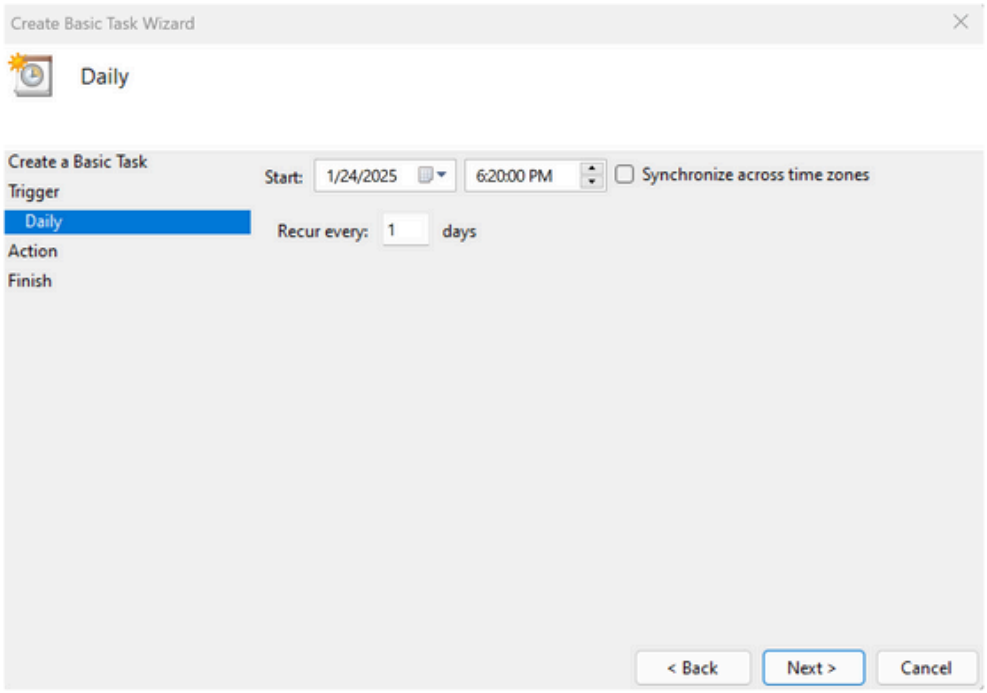
Set the Time and Frequency:

If you chose Daily, specify:

The start date (it defaults to today).

The time (e.g., 06:20 PM).

Click Next to move on.



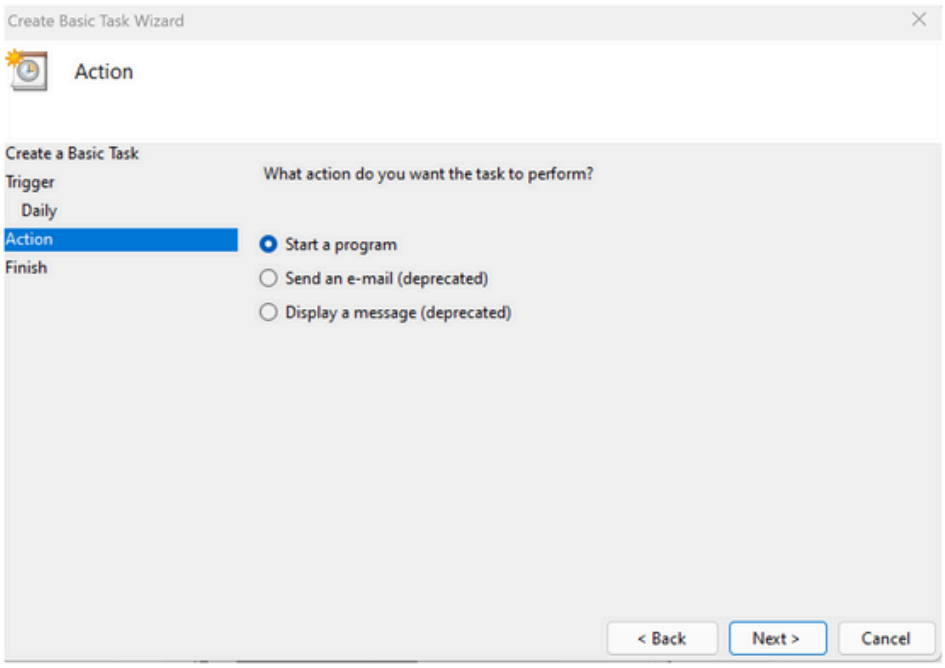
Step 8:

Set the Action

Now, we tell Task Scheduler what to do when it runs.

Select "Start a Program":

On the "Action" screen, select the option **"Start a Program"** and click Next.



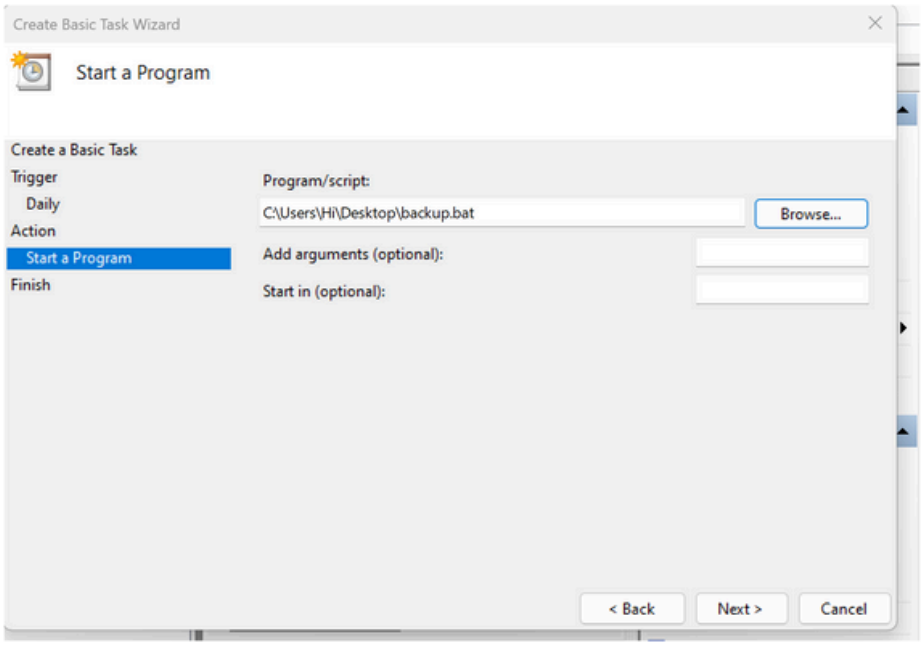
Step 9:

Point to the Program or Script:

In the Program/script field, click **Browse** and navigate to the location of your .bat file.

Example: If your script is named backup.bat and saved on the desktop, navigate to that file and select it.

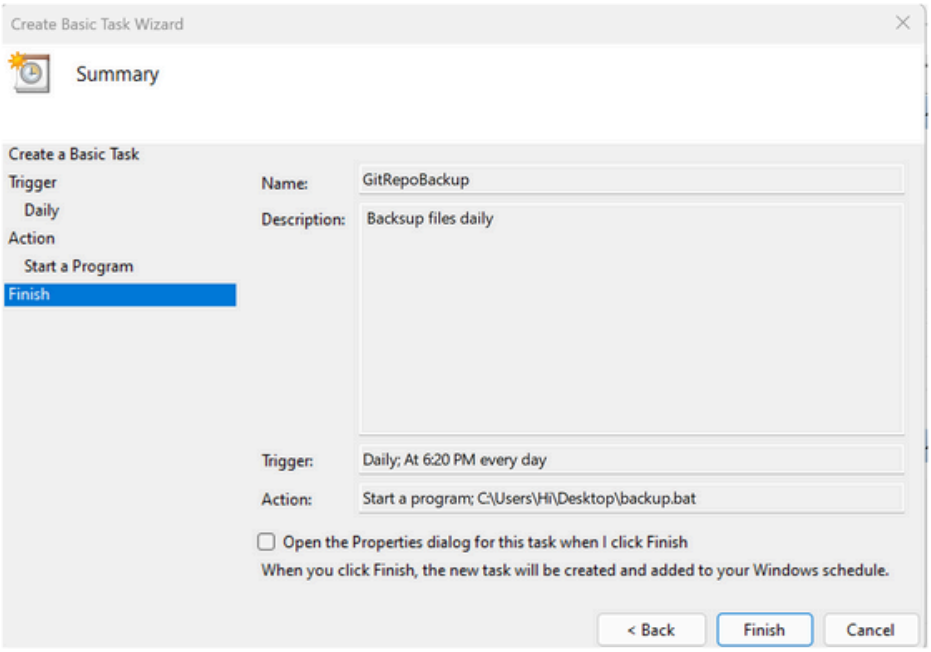
Click Next.



Step 10:

Review and Finish

Click **Finish** to save and schedule the task.



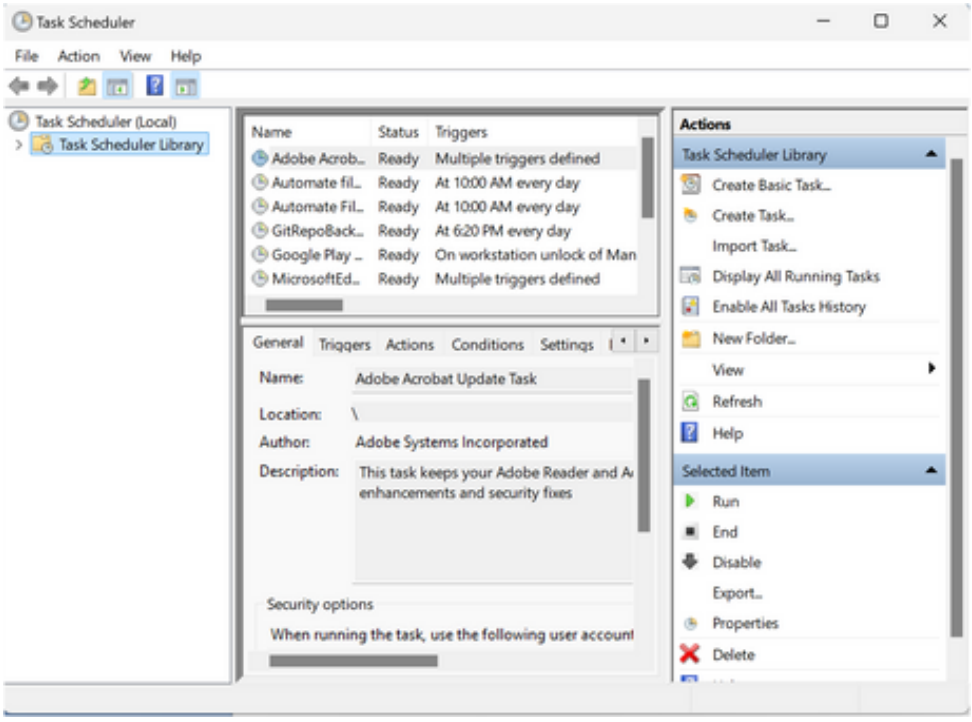
Step 11:

In Task Scheduler, go to the **Task Scheduler Library** (on the left-hand side).

Find your task (it should have the name you gave it, e.g., "GitRepoBackup").

Right-click the task and select **Run**.

This will manually trigger the task immediately

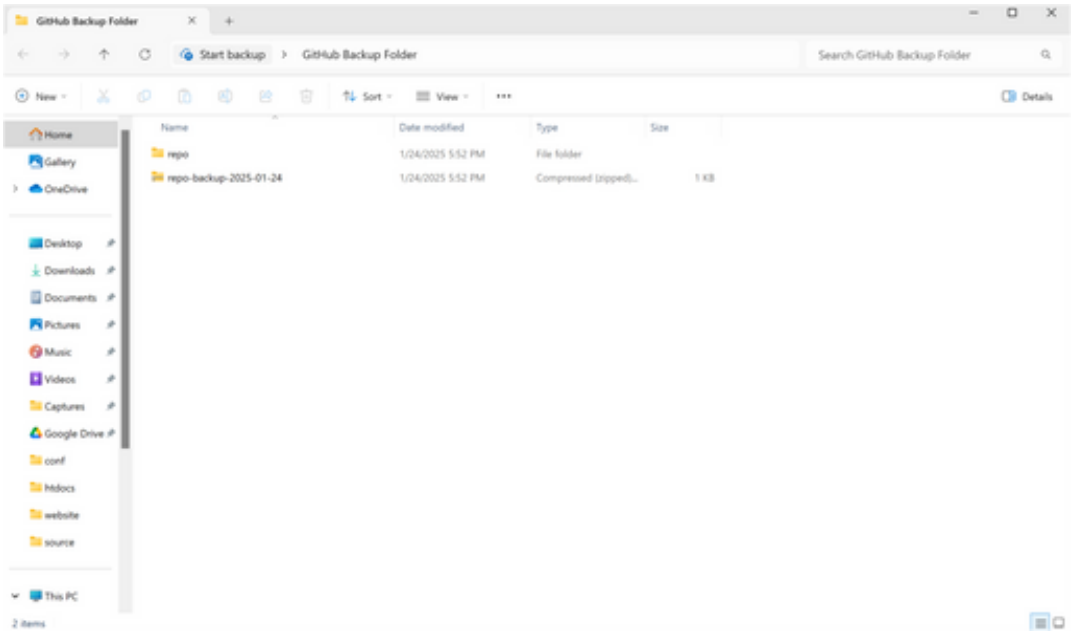



```
Cloning repository for the first time...
Cloning into 'repo'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
Creating a compressed backup: repo-backup-2025-01-24.zip
Backup complete: repo-backup-2025-01-24.zip

[process exited with code 0 (0x00000000)]
You can now close this terminal with Ctrl+D, or press Enter to restart.
```

Step 12:

Now u can see the folder which you created (GitHub Backup Folder) in the first step will now contains the files which is in your repository.



Outcomes

By completing this Proof of Concept (PoC) of automating Git repository backups, you will:

Successfully implement a backup system for Git repositories:Automate the process of creating daily backups for your Git repositories, ensuring that all updates and changes are securely stored in a local folder.

Master the use of batch scripting for task automation:Learn to create and execute a .bat script that clones, pulls updates, and compresses a Git repository into timestamped backup archives.

Understand Task Scheduler's automation capabilities:Gain practical experience with Task Scheduler, learning how to set triggers, define actions, and configure conditions to automate repetitive tasks seamlessly on a Windows system.