**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

Automate File Copying with a Script

Create a script to copy files from one folder to another automatically

Name: Anumitha M Department: AML



**Introduction**

In This Proof of Concept (PoC) demonstrates how to automate the process of copying files from one folder to another using a script, along with task automation and scheduling. The goal is to showcase how automation can eliminate manual intervention, ensuring consistency, saving time, and reducing human errors in repetitive tasks. To achieve this, we'll use a scripting language like PowerShell for Windows or Bash for Linux/Mac. The script will identify the source folder, copy the files to the destination, and log the operation for future reference. Error handling is included in the script to ensure that issues, such as missing source folders or insufficient space, are properly managed.

To automate the execution of this task, we’ll leverage task scheduling tools like Windows Task Scheduler (on Windows) or cron jobs (on Linux/Mac). These tools allow the script to run at specific intervals, such as every day at midnight, without manual intervention. The final outcome is that files are automatically copied from one folder to another on a regular basis, ensuring a seamless process with minimal oversight. In future enhancements, this PoC could be expanded to include file filtering, notifications, or handling more complex workflows.

**Overview**

The goal of this PoC is to automate the process of copying files from one folder to another using a script, with automation and scheduling for recurring execution.

Script: A script will be written using a scripting language like PowerShell (Windows) or Bash (Linux/Mac). The script will copy files from the source folder to the destination and include error handling for issues like missing folders or insufficient space.

Task Automation: Tools like Windows Task Scheduler (for Windows) or cron (for Linux/Mac) will be used to schedule the script to run at specific times, such as daily or at other desired intervals.

Benefits: Automating this process eliminates the need for manual intervention, ensuring consistent execution, saving time, and reducing the risk of errors in repetitive tasks.

Future Enhancements: Potential future improvements could include adding file filtering, email notifications, or handling more complex file operations to further enhance the automation process.

**Objectives**

**Automate File Copying**: To automate the process of copying files from one folder to another, reducing the need for manual intervention.

**Error Handling**: To ensure the script is robust by adding error handling for common issues such as missing folders or insufficient disk space.

**Task Scheduling**: To demonstrate how to schedule the script to run at specific times (e.g., daily or hourly) using task automation tools like Task Scheduler (Windows) or cron (Linux/Mac).

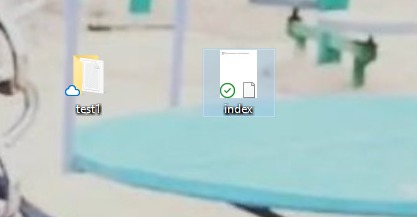
**Improve Efficiency**: To increase efficiency by automating repetitive file management tasks, saving time and reducing the potential for human errors.

**Scalability and Flexibility**: To provide a foundation for extending the automation to more complex tasks, such as filtering specific files or integrating notifications.

**Step-by-Step Overview**

**Step 1: Create Two Folders**

* Create two folders on your system where you will perform the file transfer. For example:
  + **Source Folder**: C:\Users\YourName\Documents\test
  + **DestinationFolder**: C:\Users\YourName\Documents\test1
* Place at least one text document (e.g., file1.txt) inside the **Source Folder** for testing purposes.



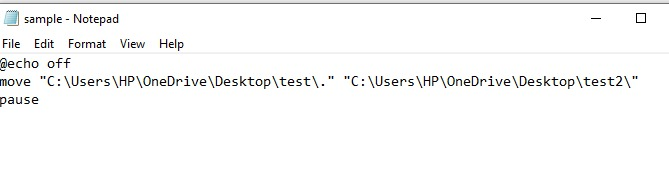
**Step 2:**

Add the Script to Transform Files from One Folder to Another

* In this step, you’ll write and save the script that will automate the process of transferring files from the Source Folder to the Destination Folder. The script will move or copy files based on the logic you define.

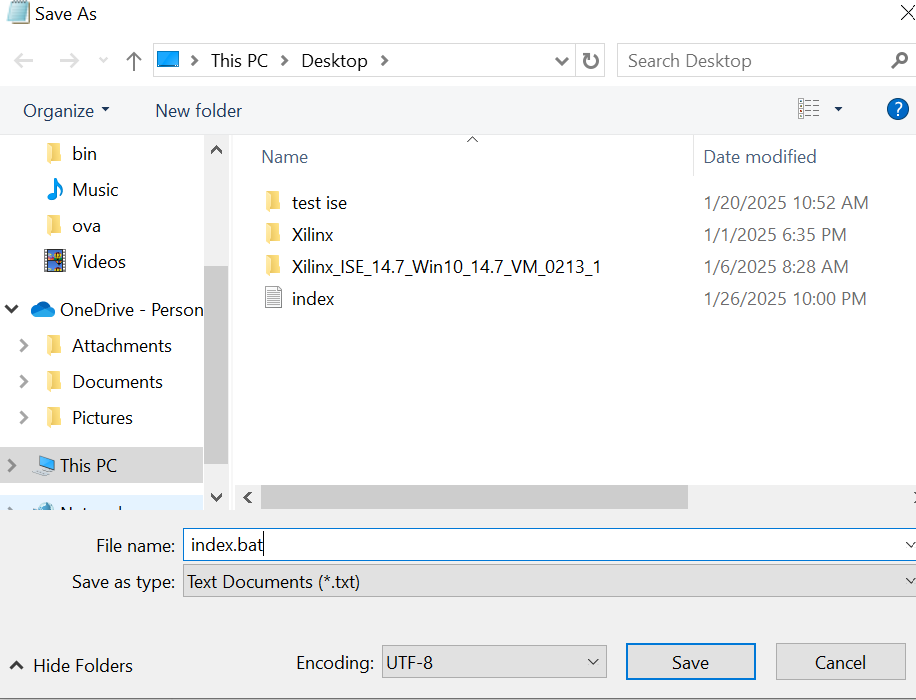
For PowerShell (Windows):

* Open Notepad or your preferred text editor and write the following PowerShell script:

****

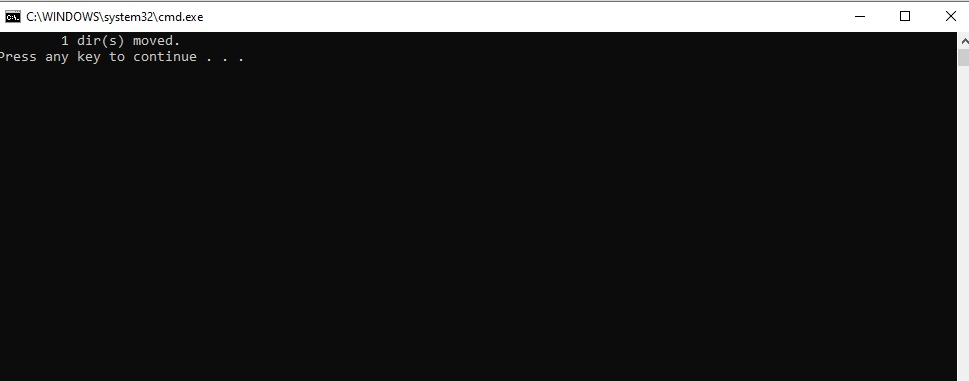
Step 4:

Save the file in a directory with bat extension:

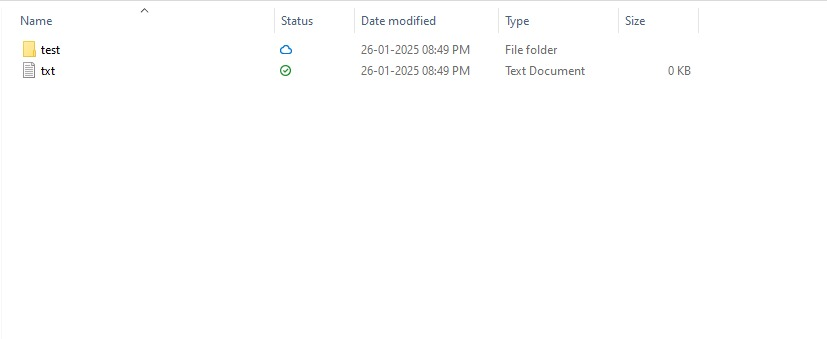


Step 5:

Open the text file in a command prompt to attempt the script to copy the files from one folder to another



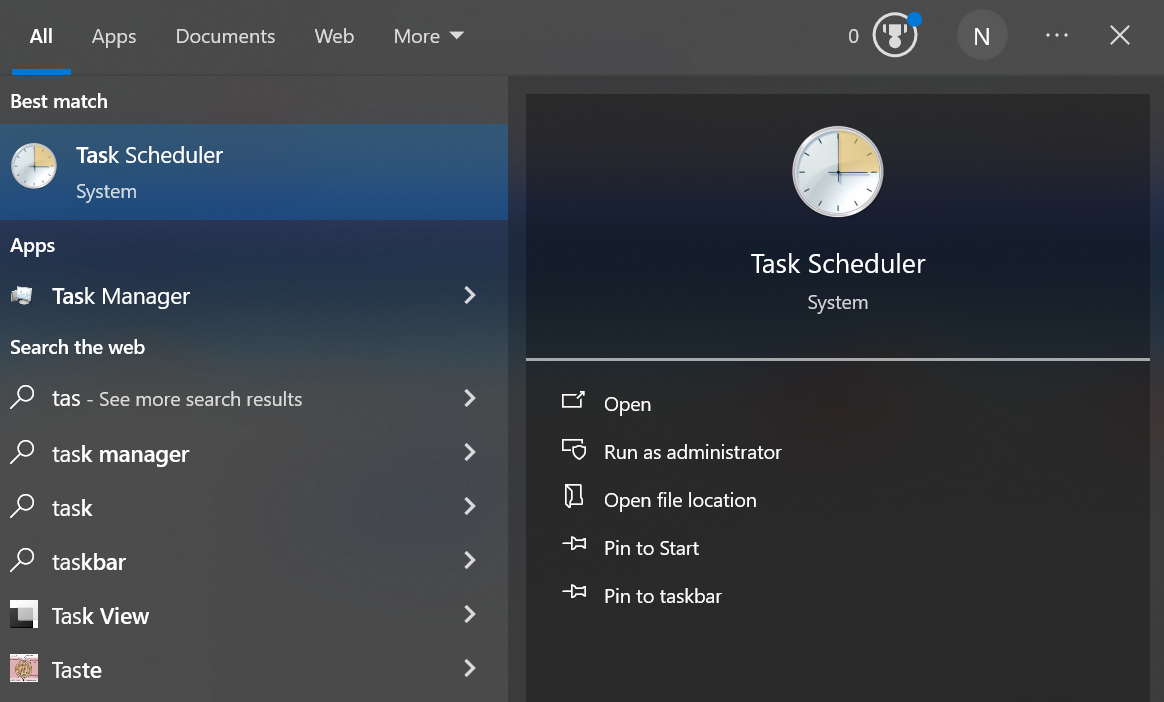
File is copied succesfully from one directory to another



Step 6: Open Task Scheduler Using the Open Button

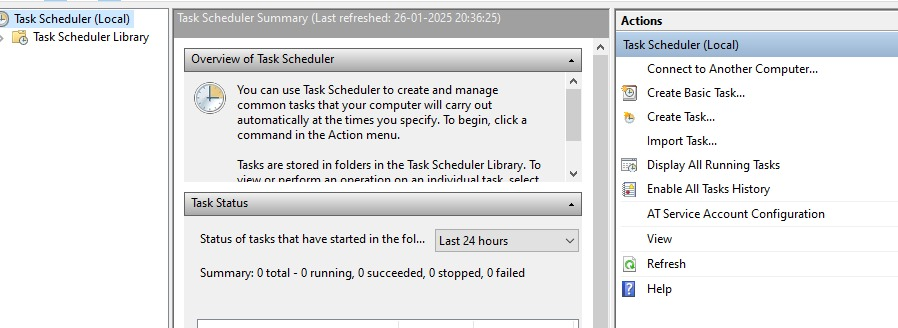
To schedule your script to run automatically at specified intervals, you need to open Task Scheduler and set up a new task. Here’s how you can do it:

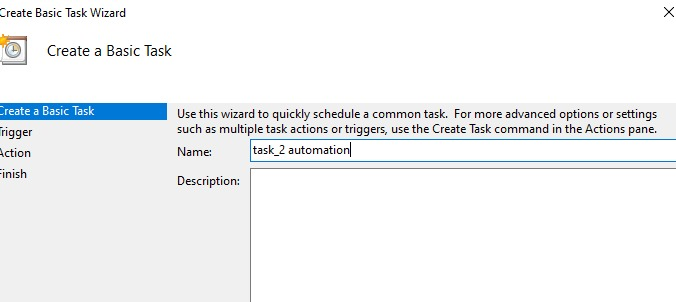
1. Open Task Scheduler:
   * Option 1: Press the Windows key on your keyboard, type “Task Scheduler” in the search bar, and then click on the Task Scheduler app when it appears in the search results.
   * Option 2: Alternatively, press Win + R to open the Run dialog box. Type taskschd.msc and press Enter. This will open Task Scheduler directly.

****

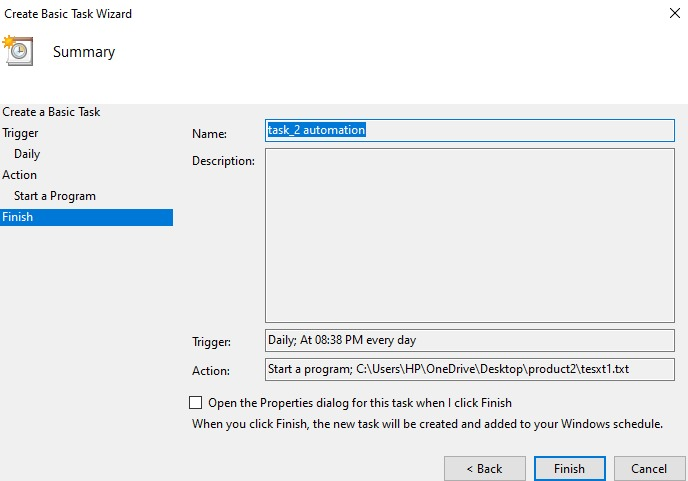
Create a New Task:

* In the Task Scheduler window, look at the right-hand panel under Actions and click on "Create Basic Task...". This will open a wizard that guides you through the task creation process.

****

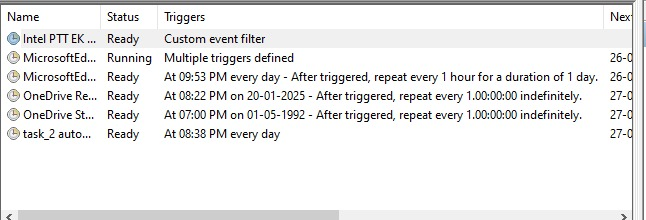
****

**Finish it by clicking the finish button after entering all the credentials.**

****

Step 7:

After creating the task scheduler,view the backup auotomated task in history



**Expected Outcome**

By completing this PoC of setting up a local Git repository, you will:

1. **Successful Automation of File Transfer**:  
   After setting up the task in Task Scheduler, the **script** (whether written in PowerShell or Bash) will automatically run according to the schedule you've defined (e.g., daily at midnight). The task will transfer files from the **Source Folder** to the **Destination Folder** without manual intervention.
2. **File Transfer Execution**:
   * The script will check the **Source Folder** for files, and then **copy** (or move) the specified files to the **Destination Folder**.
   * The task will only run when the conditions you've set (time, frequency) are met, ensuring the automation occurs consistently.
3. **Error Handling**:
   * If the source folder or file is missing, the script will log an error message or display a message indicating that the file transfer was unsuccessful.
   * If the file already exists in the destination, the script will either skip the transfer or overwrite the file (depending on your script's configuration).
4. **Scheduled Execution**:
   * The task will execute automatically at the scheduled time, with no manual intervention required.
   * You will see the task appear in **Task Scheduler Library**, and you can check the task’s **History** tab for logs or execution status (successful or failed).
5. **Consistency**:
   * The process will be consistent every time the task runs, transferring files reliably without the need for user interaction.
   * In case of any failures (like missing folders, permissions issues, etc.), the error handling will provide feedback, and corrective actions can be taken.
6. **Efficiency Gains**:
   * The automation eliminates the need for manual file transfers, saving time and effort.
   * It reduces the risk of human error, ensuring files are copied on time and correctly each time.