

**PLACEMENT EMPOWERMENT PROGRAM**

**Cloud Computing & DevOps Center**

Create a simple Backup Script: Create a script that backs up your entire Git Repo to a local folder daily.

**Name:**Mathimalar.P **Dept:** ADS

**INTRODUCTION**

In the fast-moving world of software development, safeguarding your code and version history is essential to avoid the loss of important work. Automating the process of backing up your Git repository to a local folder regularly is a simple yet effective way to ensure your entire repository, including all branches, commits, and history, is securely stored. By setting up an automated backup script, you eliminate the need for manual intervention while providing peace of mind.

Automating daily backups not only saves time but also reduces the risk of human error. In this guide, we will explore how to use basic shell scripting along with Git commands to create an automatic backup system for your Git repository. This ensures that even in the case of a system failure or data loss, you will always have a secure copy of your project, allowing for easy restoration.

**Step-by-step process**

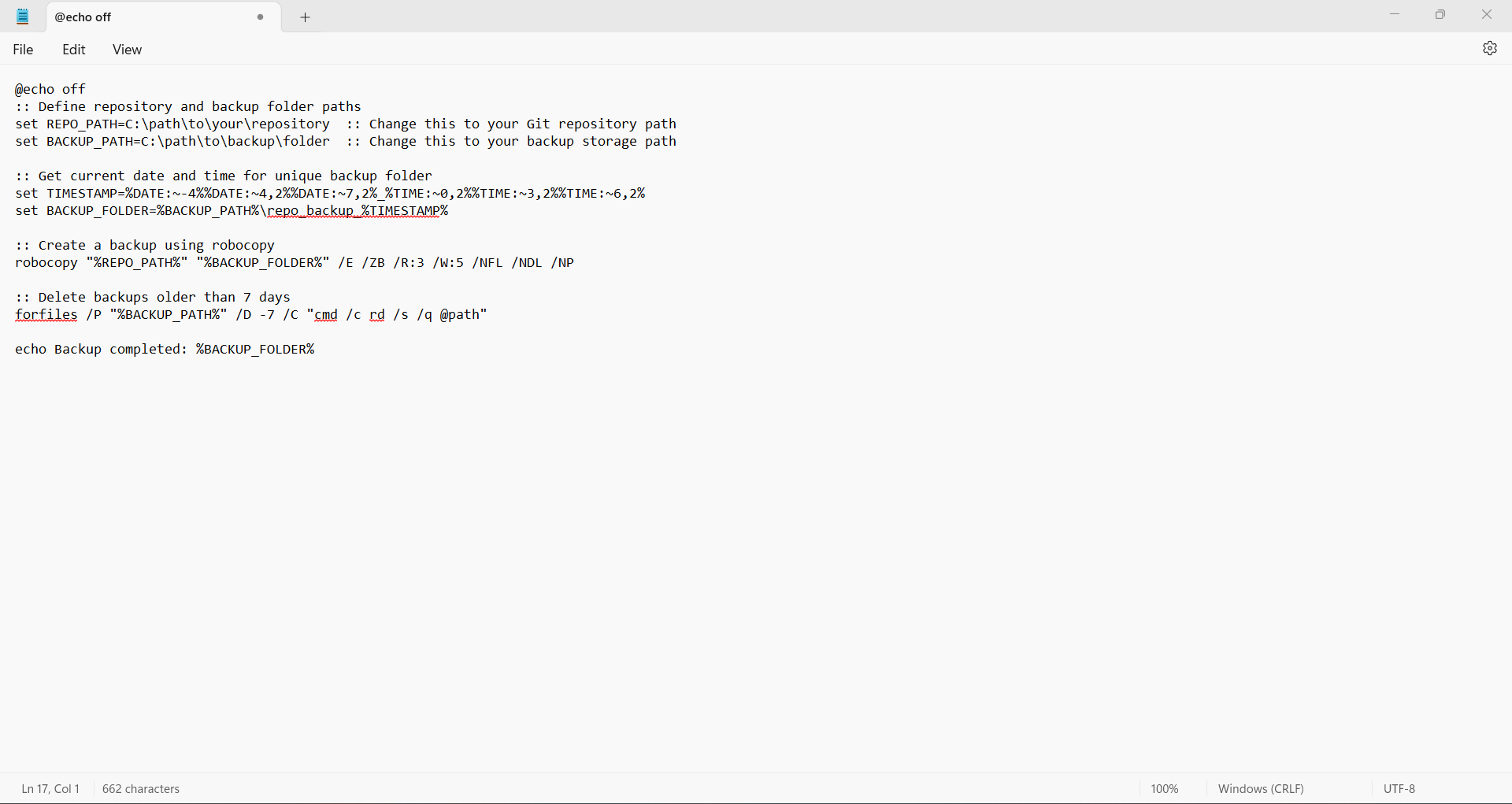
**🔹 Step 1: Create the Backup Script**

**1️ Open Notepad**

1. Press Win + R, type notepad, and press Enter.

**2️ Write the Backup Script**

Copy and paste the following script into Notepad:

**3:** **Save the File as a Batch Script**

1. Click **File** → **Save As...**.
2. In **Save as type**, select **All Files**.
3. Enter **backup.bat** as the file name.
4. Save it in a preferred location.

**Step 2: Test the Backup Script**

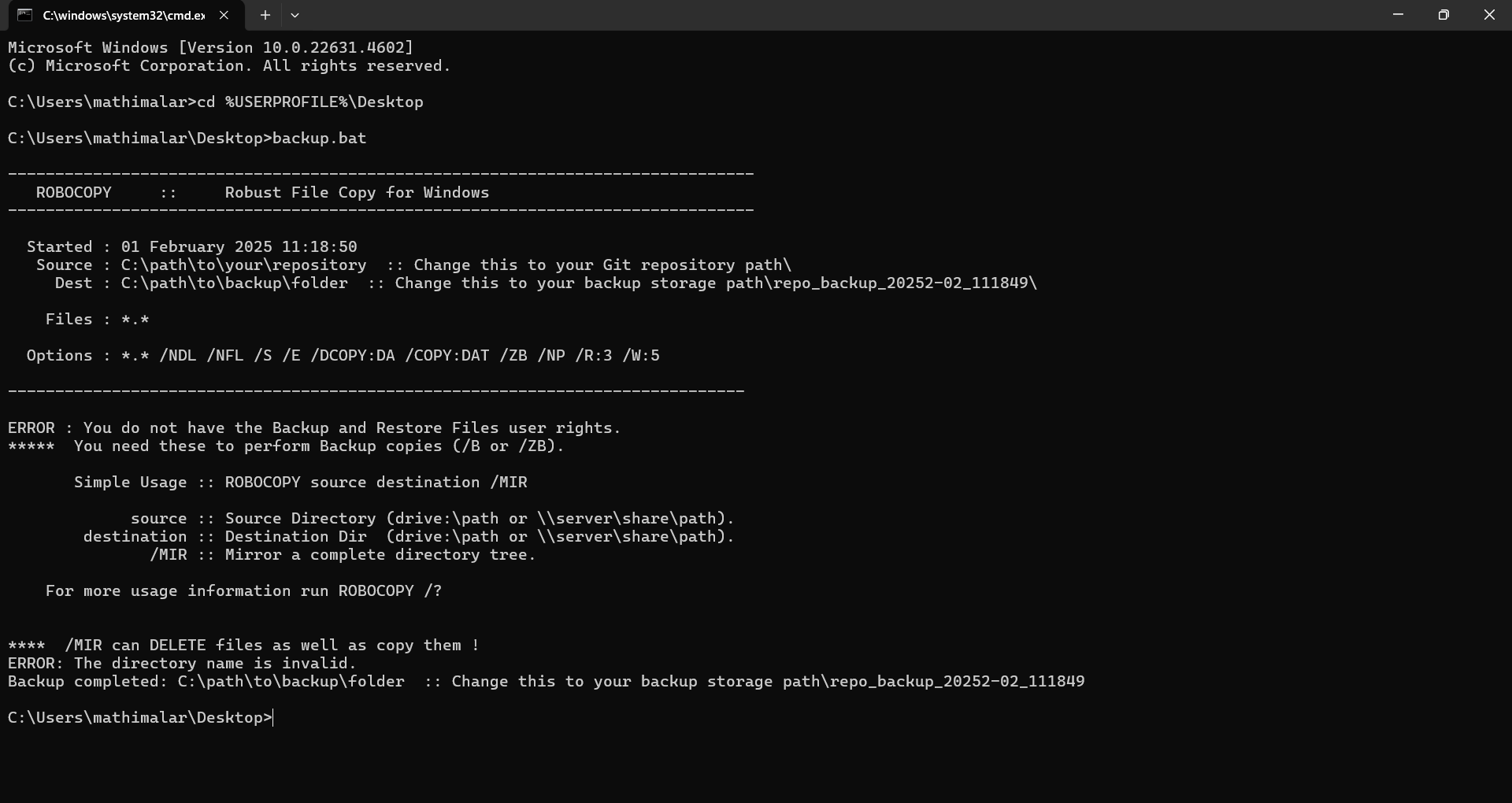
**1️ Open Command Prompt**

1. Press **Win + R**, type cmd, and press **Enter**.

**2️ Navigate to Your Desktop**

1. Type the following command and press **Enter**:

*cd %USERPROFILE%\Desktop*

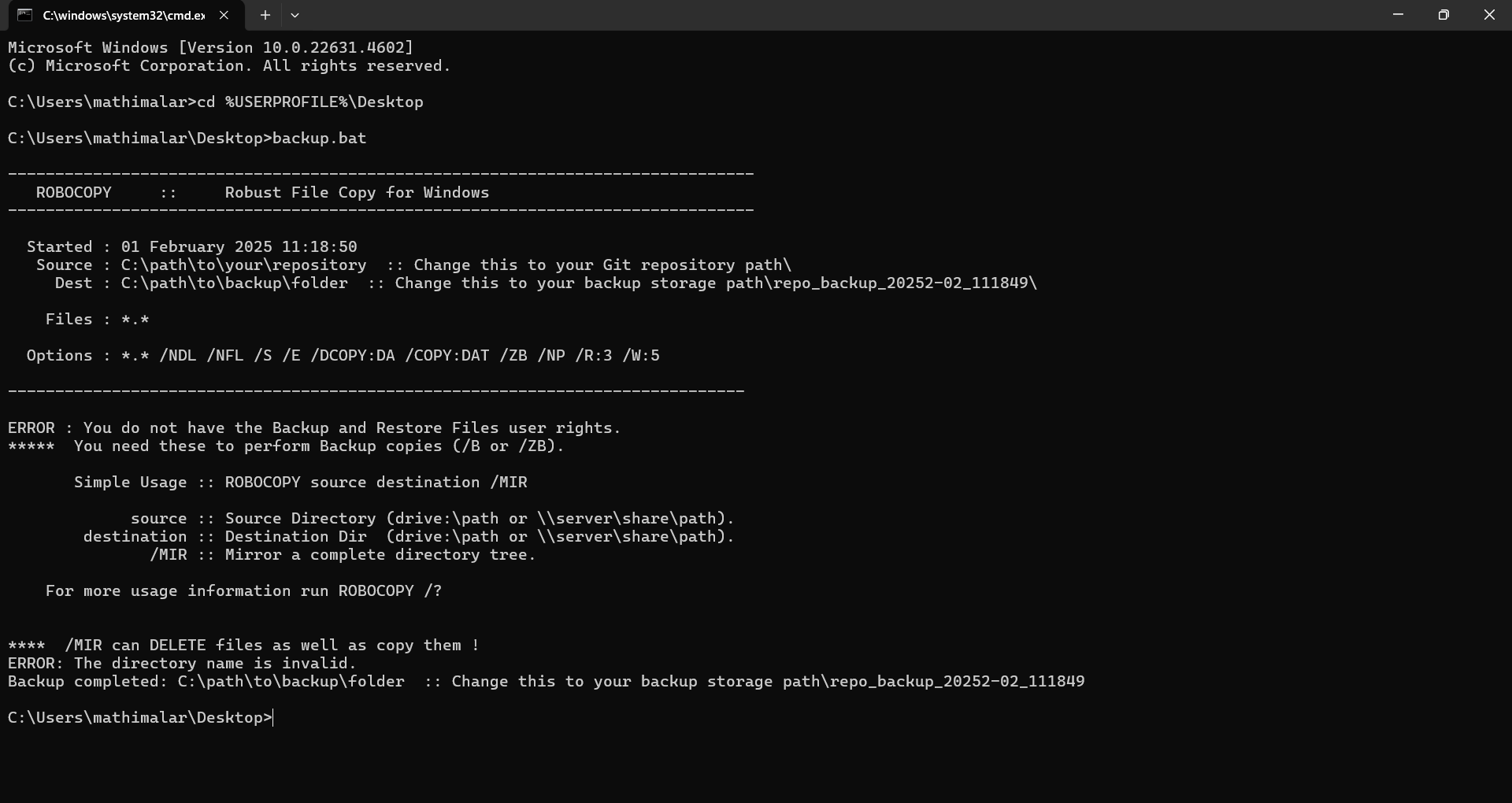


**3️ Run the Script**

1. Execute the script manually by typing:

*backup.bat*

1. If successful, you should see a new backup folder created inside the **backup location** you specified in the script.



**🔹 Step 3: Automate the Script Using Task Scheduler**

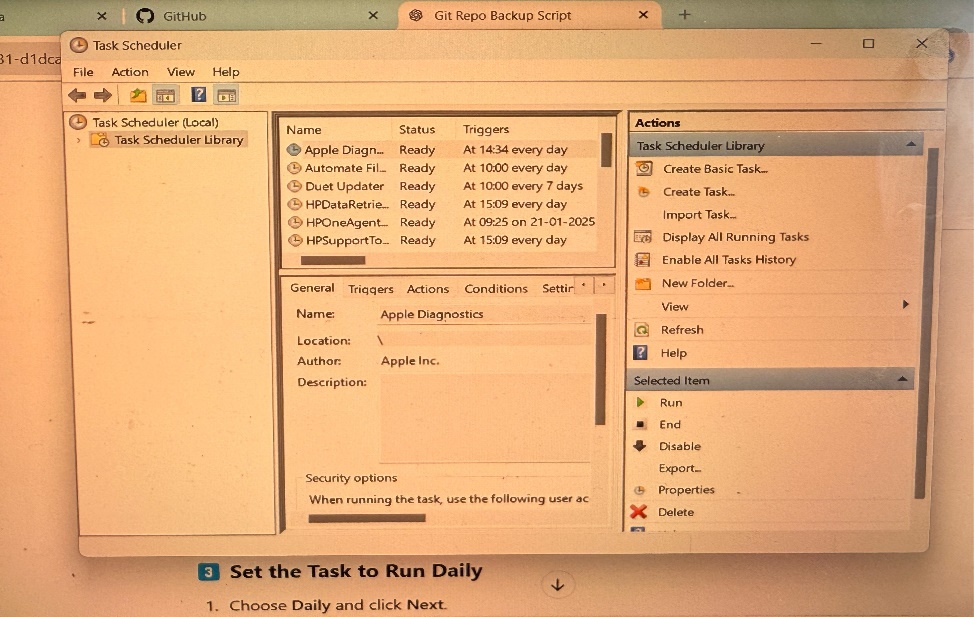
Since your script is on the **Desktop**, we need to tell Task Scheduler to run it from there.

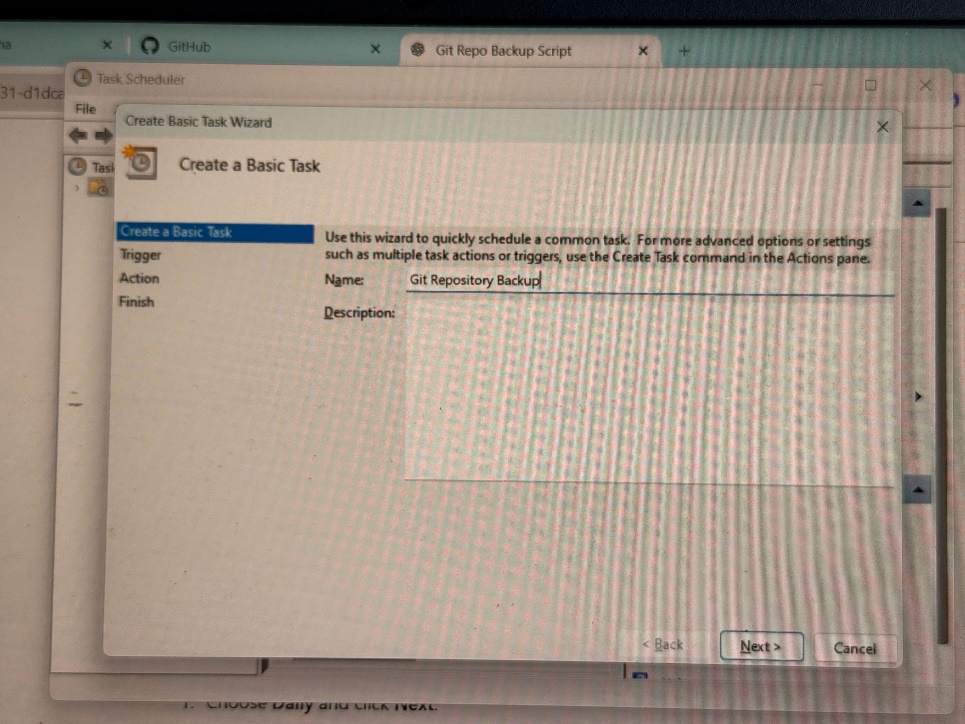
**1️Open Task Scheduler**

1. Press **Win + R**, type taskschd.msc, and press **Enter**.

**2️Create a New Task**

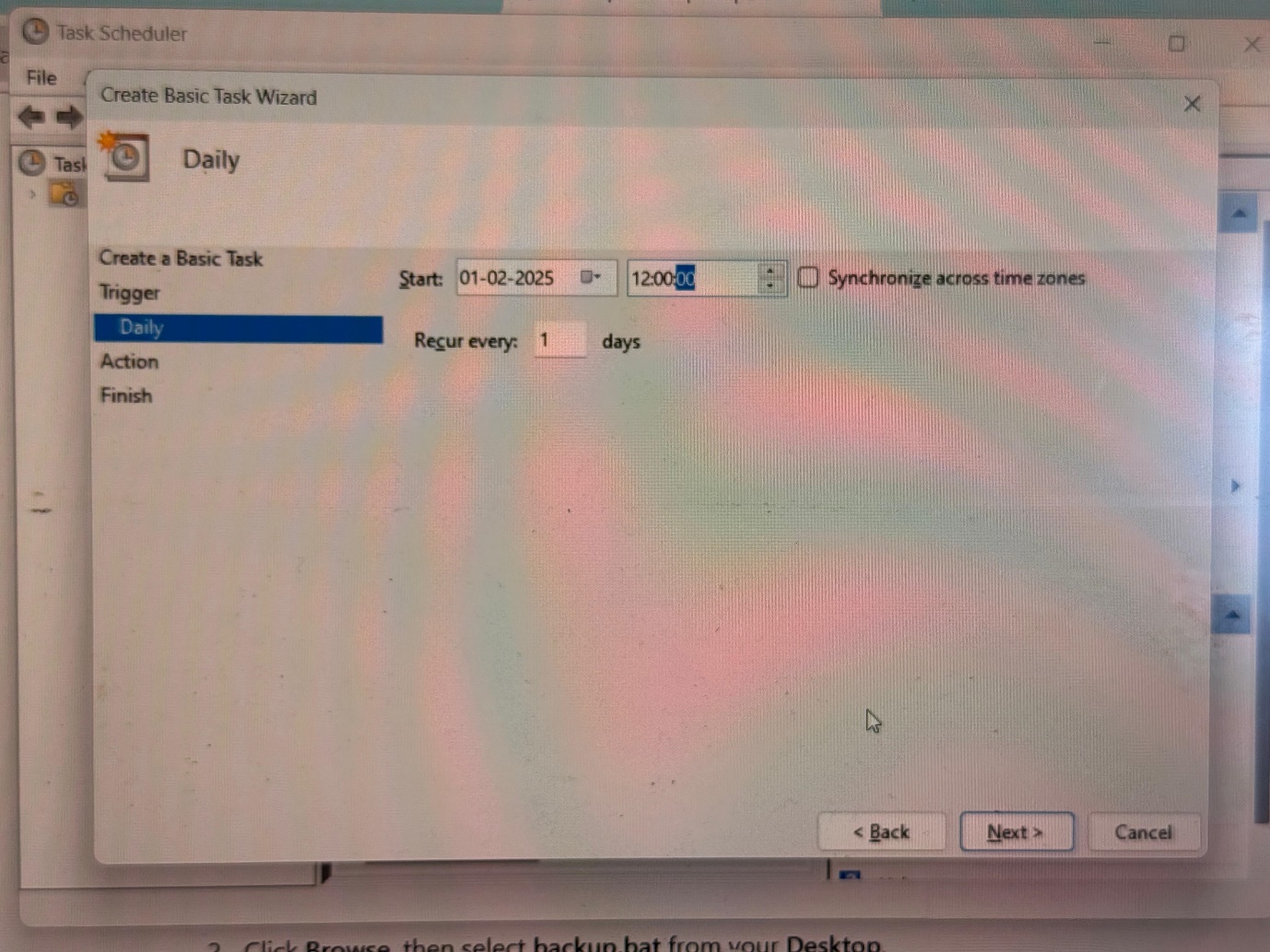
1. Click **Create Basic Task** on the right panel.



2.Enter a name for the task, such as **Git Repository Backup**, then click **Next**. 

**3️Set the Task to Run Daily**

1. Choose **Daily** and click **Next**.
2. Set the **Start time** (e.g., 12:00 AM) and repeat every **1 day**.
3. Click **Next**.



**4️Select the Action**

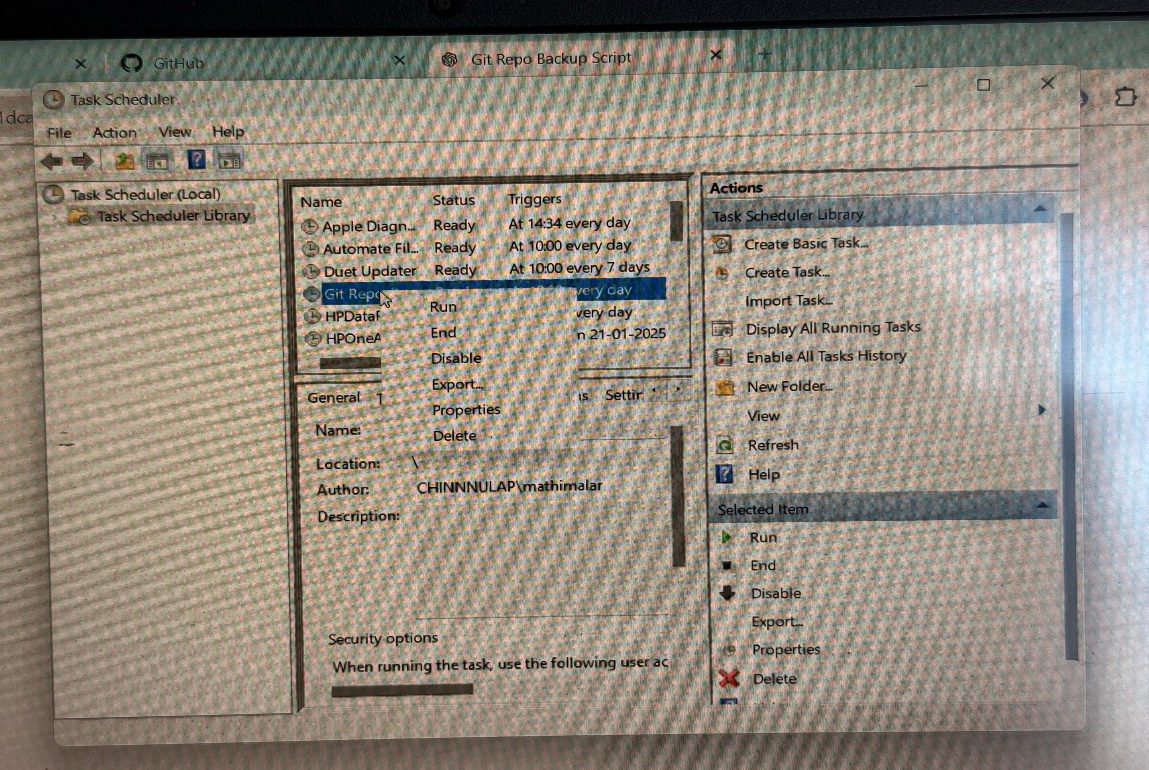
1. Choose **Start a Program**, then click **Next**.
2. Click **Browse**, then select **backup.bat** from your **Desktop**.
3. Click **Next**.

**5️ Enable High Privileges**

1. Check **Open Properties Dialog before finishing** and click **Finish**.
2. In the **Properties** window:
   * **General Tab** → Check **Run with highest privileges**.
   * **Conditions Tab** → Uncheck **Start the task only if the computer is on AC power**.
   * Click **OK**.

**🔹 Step 4: Test the Scheduled Task**

1. In **Task Scheduler**, find the task under **Task Scheduler Library**.
2. Right-click the task and select **Run**.
3. Check the backup folder to confirm that the script ran successfully.



✅ **Now your Git repository will be backed up daily!** 🚀

The outcome of automating the Git repository backup process is a reliable, time-saving solution that ensures the safety and integrity of your code. By setting up an automated backup script, you achieve:

1. **Consistent Backups**: Your repository, including all branches, commits, and history, is backed up automatically on a daily basis, reducing the risk of losing important work.
2. **Time-Saving**: The process is automated, eliminating the need for manual intervention, so you can focus on development without worrying about backup tasks.
3. **Peace of Mind**: In the event of system failures or data loss, you can easily restore your project using the secure, up-to-date copy of your repository stored in the backup location.
4. **Reduced Human Error**: Automation minimizes the potential for mistakes, ensuring that backups are always performed correctly and consistently.

This approach provides a robust safeguard for your project, ensuring its availability and continuity even in challenging situations.