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 : Harshana Perianayaki B Department : IT

INTRODUCTION

Maintaining regular backups of critical data is essential for ensuring data integrity, protecting against accidental deletions, and mitigating risks posed by system failures. This is particularly important for Git repositories, which often store valuable source code, configuration files, and project history. By implementing a simple backup script, developers can automate the process of creating daily backups of their Git repositories, ensuring they always have an up-to-date and secure copy of their work.

OVERVIEW

The goal of this script is to create a simple, automated solution to back up a Git repository to a designated local folder daily. The script will compress or copy the repository and organize backups with a clear naming convention based on the current date. The solution will use platform-specific tools—`tar` for Linux and `robocopy` for Windows—and can be automated using scheduling tools like `cron` or Task Scheduler. This ensures seamless operation without manual intervention.

OBJECTIVES

- **Automate Repository Backups:**

  Develop a script that eliminates the need for manual backups, saving time and reducing the risk of human error.

- **Ensure Backup Integrity:**

The script will validate the success of the backup process, providing logs or notifications to indicate whether the operation was successful.

- **Daily Backup Organization:**

Use date-based naming conventions for backup files or folders, making it easy to track changes and locate backups from specific days.

- **Platform Compatibility:**
  Create platform-specific solutions:
  Linux: Utilize `tar` for compressing and archiving the repository.
  Windows: Leverage `robocopy`

IMPORTANCE

- **Protecting Critical Data**
  Git repositories contain source code, configuration files, and project history. Regular backups safeguard these essential assets from accidental deletions, corruption, or hardware failures.

- **Minimizing Downtime and Risks**

Automated backups reduce the impact of unexpected events, such as system crashes or ransomware attacks, ensuring quick recovery and business continuity.

- **Time and Effort Efficiency**

Automation eliminates the need for manual backups, ensuring consistency and freeing developers to focus on more important tasks.

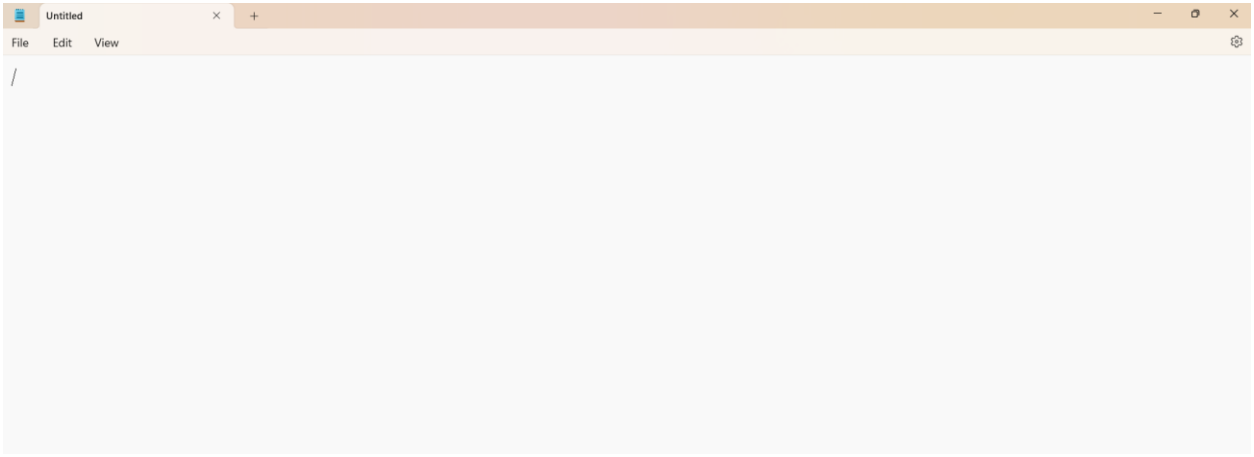
- **Simplifying Restoration**

Organized, date-stamped backups make it easy to retrieve specific versions of the repository, aiding debugging, compliance audits, and rollbacks to stable code.

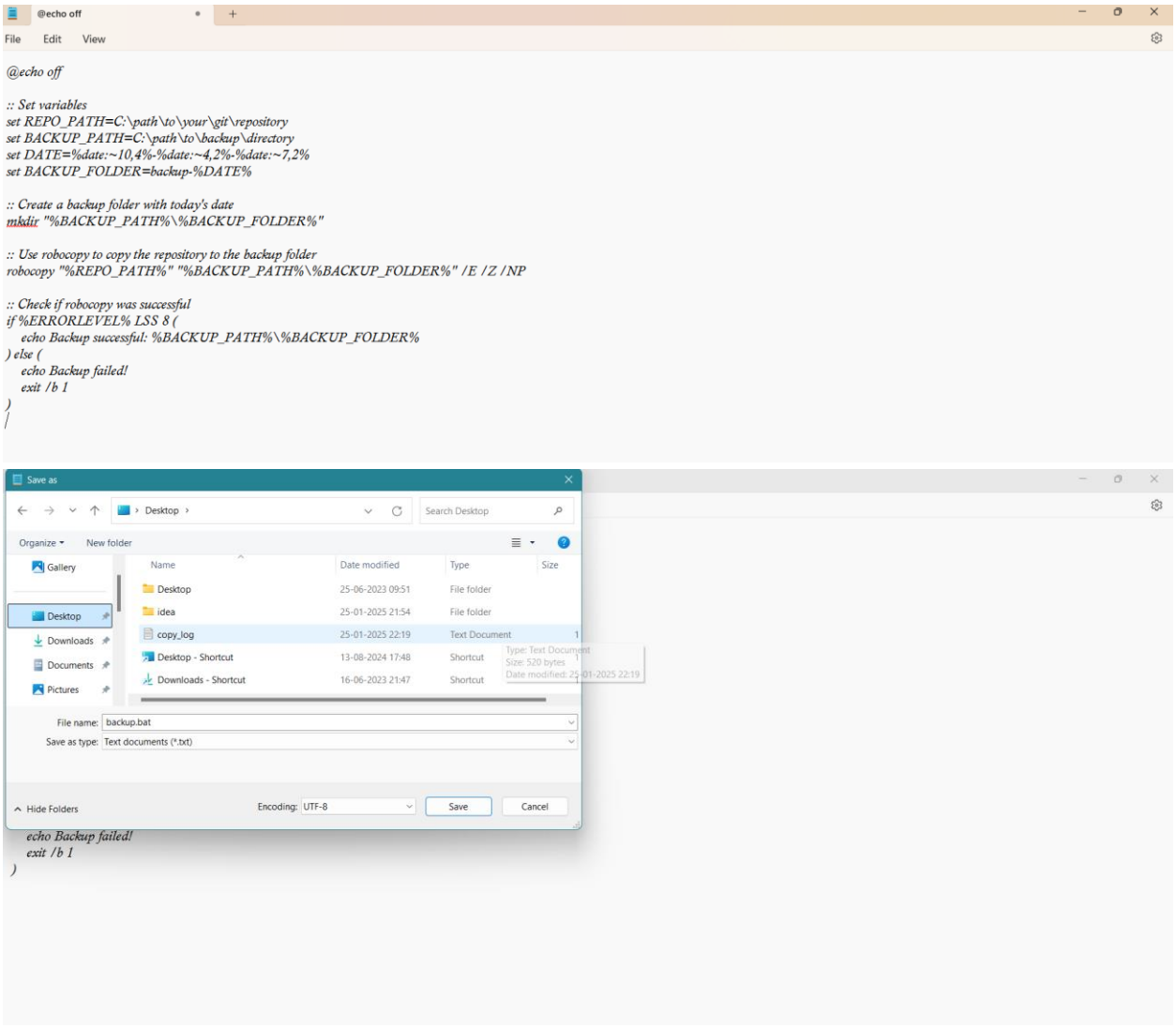
STEP BY STEP OVERVIEW

Step 1: Create a Batch Script

Open a text editor (like Notepad).



Save the following as `backup.bat`:



Explanation Of Script Commands

1. set Commands:

REPO_PATH: The path to your Git repository.
BACKUP_PATH: Where backups will be stored.
DATE: Generates the current date in YYYY-MM-DD format
BACKUP_FOLDER: Names the backup folder with the current date

2. robocopy Options:

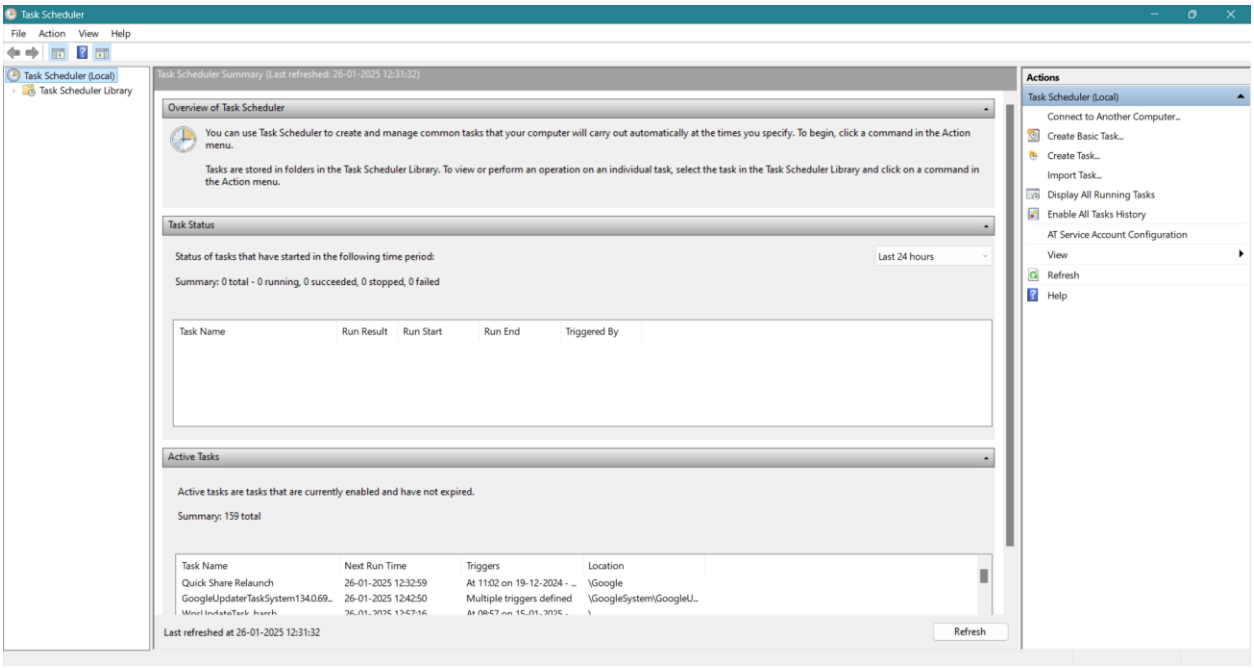
/E: Copies all subdirectories, including empty ones.

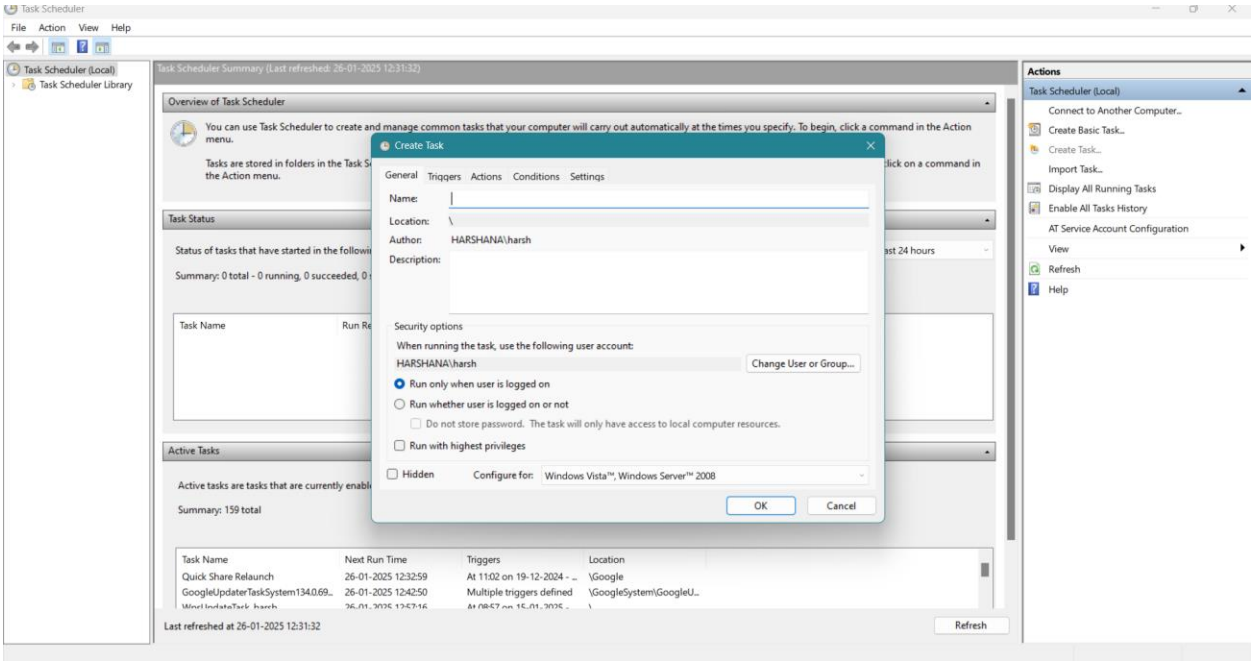
/Z: Enables restartable mode for interrupted transfers.

/NP: Suppresses the progress output.

Automating with Task Scheduler:

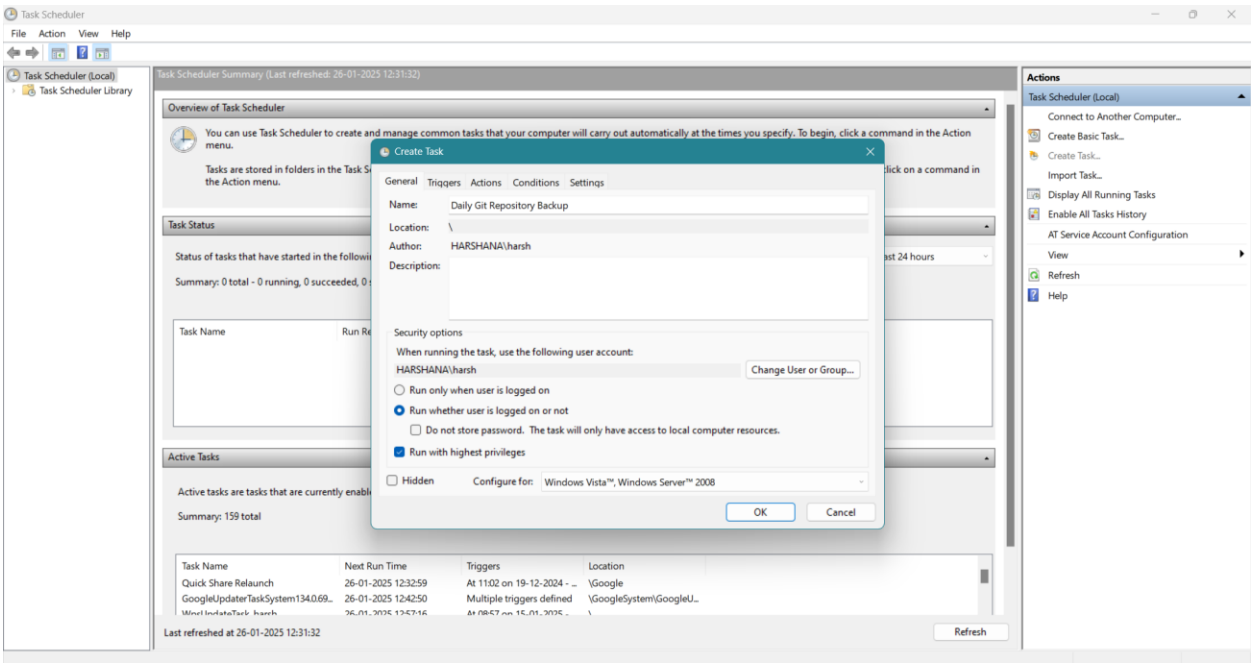
1. **Create the Task:**
    a. Open **Task Scheduler** (search "Task Scheduler" in the Start menu).
    b. Click **Create Task** (not "Basic Task").





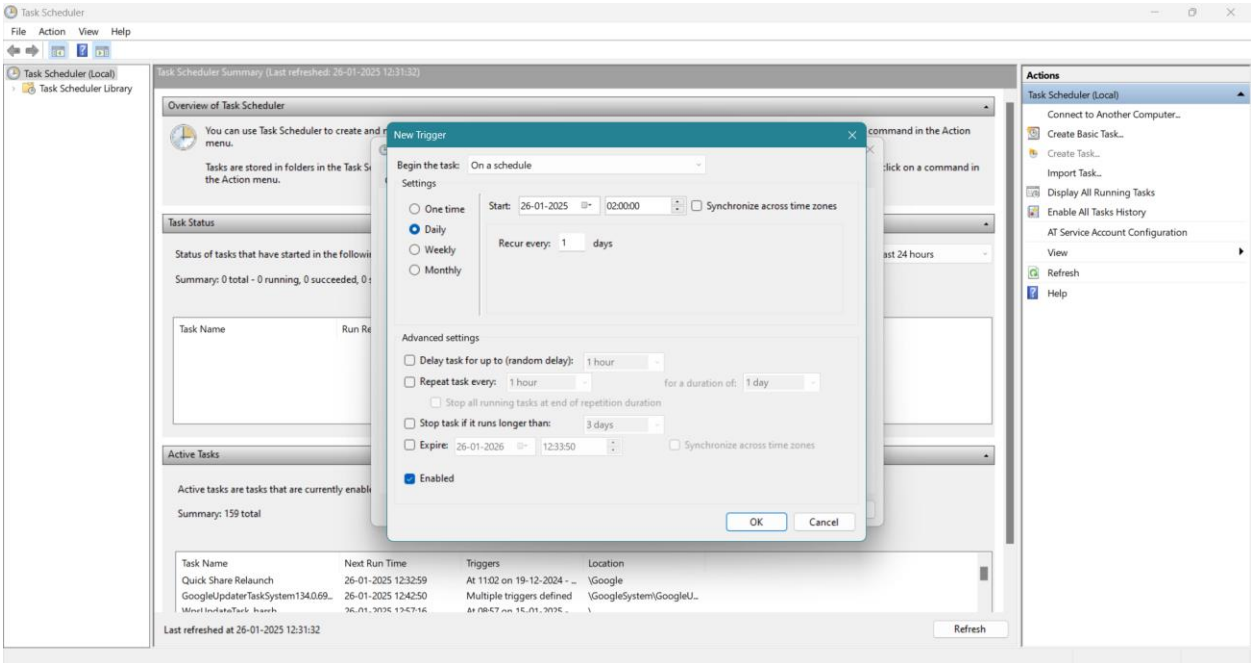
General Settings:

- Name the task (e.g., "Daily Git Repository Backup").
- Select **"Run whether user is logged on or not"**.
- Check **"Run with highest privileges"**.



Trigger:

- Go to the **Triggers** tab, click **New**, and set:
  - Start date and time (e.g., 2:00 AM).
  - Repeat every day.

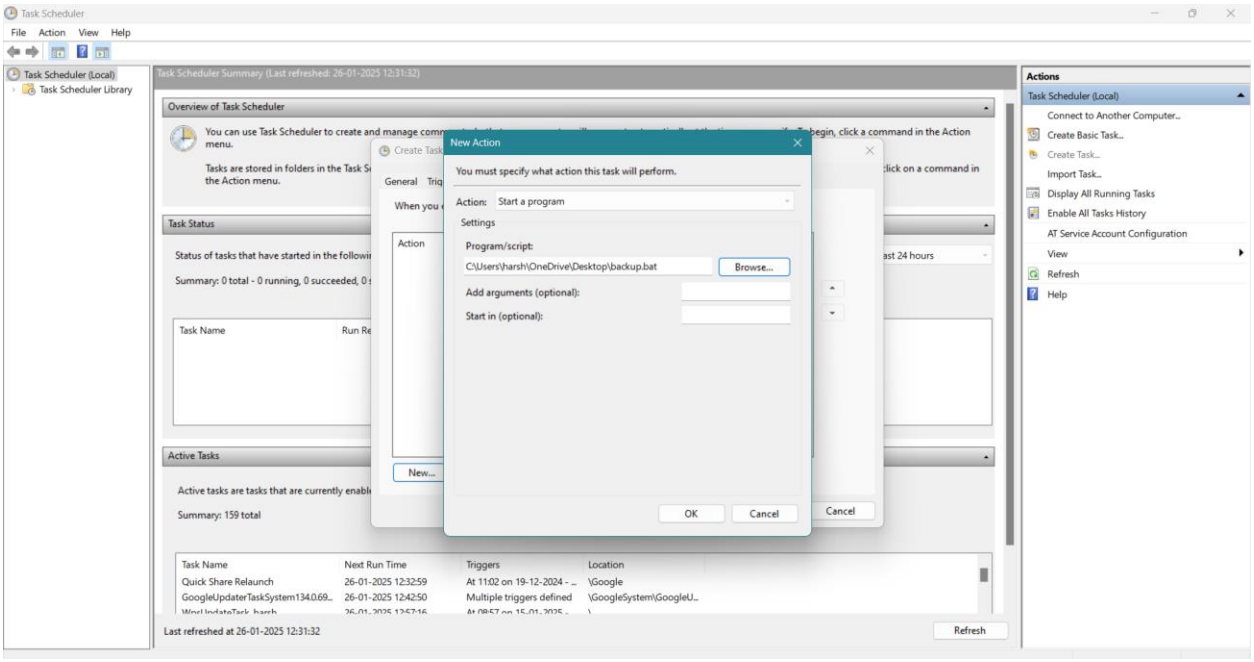


Action:

Go to the **Actions** tab, click **New**, and set:

Action: "Start a Program".

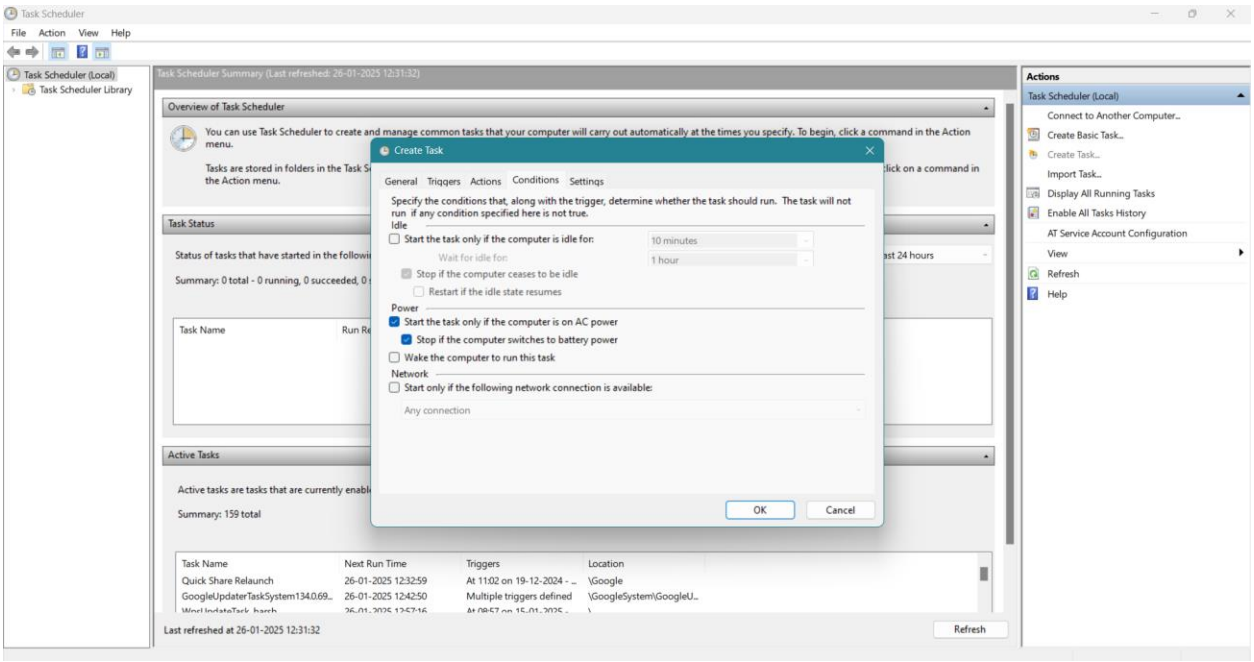
Program/script: Browse to your backup.bat file.

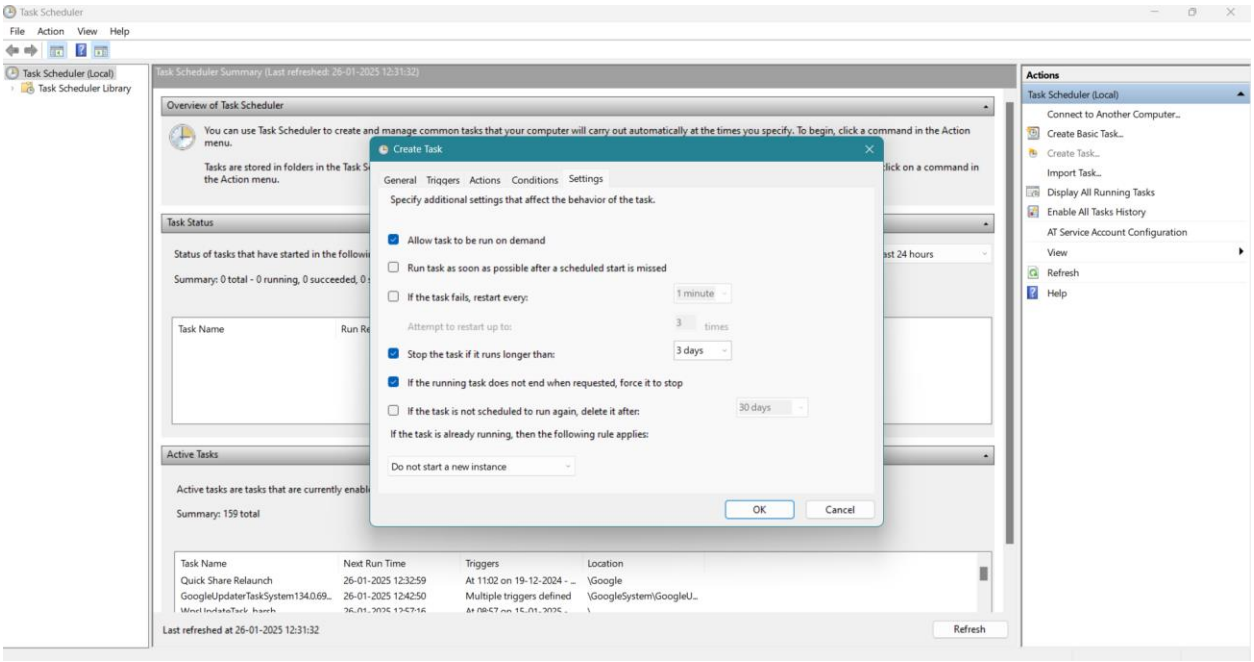


Conditions & Settings:

In **Conditions**, uncheck "Start the task only if the computer is on AC power" (if you're on a laptop).

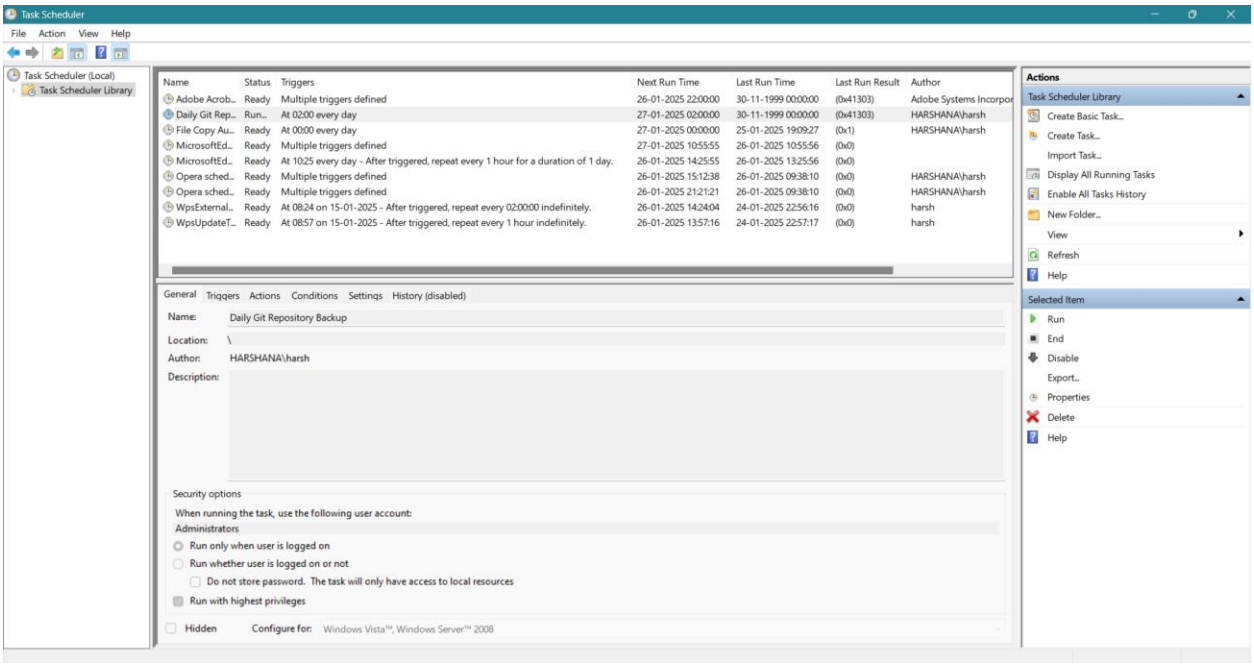
In **Settings**, ensure "Allow task to be run on demand" is checked

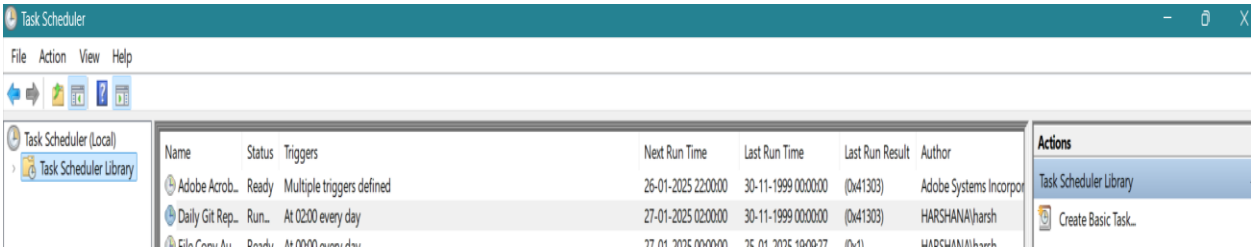




Save and Enter Password:

- Click **OK** and enter your user account password to confirm.





Once this is set up, the script will run daily at the scheduled time, creating a dated backup of your Git repository in the specified location.

OUTCOMES

Outcomes of Implementing a Git Repository Backup Script

1. **Secure and Reliable Backup System**

   A robust and automated backup system ensures that the Git repository is consistently backed up, safeguarding valuable project data from loss or corruption.

2. **Improved Recovery Time**
   With organized, date-stamped backups, recovering specific versions of the repository becomes quick and straightforward, reducing downtime in case of data loss.

3. **Peace of Mind for Developers**
   Automation eliminates the need to worry about manual backups, allowing developers to focus on coding and innovation without fear of losing progress.

4. **Enhanced Data Management and Organization**
   Backups stored with clear naming conventions ensure easy tracking of changes over time, helping teams manage project versions and history efficiently.

5. **Risk Mitigation and Business Continuity**
   The backup system minimizes the impact of unexpected events, such as hardware failures, accidental deletions, or cyberattacks, ensuring the project's continuity and security.