Sure! I can **segregate your commands** into clear sections based on their purpose: setup, file management, branching, committing, pushing, and cleanup. Here's a structured version:

1️⃣ Repository Setup & Configuration

```powershell
cd ..
cd Dir               # Navigate to desired directory
git init             # Initialize a new Git repository
git status           # Check repository status
git config --global user.name "Your Name"      # Set Git username
git config --global user.email "your@email.com"  # Set Git email
git clone <path>       # Clone remote repository (if needed)
cd Folder            # Navigate into cloned folder
git status           # Verify repository status
```

2️⃣ File Creation & Tracking

```powershell
```

```
# Create text files in the repository folder
git add abcd.txt       # Stage file (if exists)
git add file2.txt
git add ab.txt
git status             # Check which files are staged
```

3️⃣ Commit Changes

```powershell
git commit -m "create ab.txt"   # Commit staged changes
git log                         # View full commit history
git log --oneline               # View condensed commit history
```

4️⃣ Push Changes to GitHub

```powershell
git push origin main            # Push local commits to main branch on GitHub
```

5️⃣ Branch Management

```powershell
git branch                    # List all branches
git branch develop            # Create new branch 'develop'
git checkout develop          # Switch to 'develop' branch
git add .                     # Stage all changes
git commit -m "added sample.txt"  # Commit changes
git checkout main             # Switch back to 'main'
git branch                    # Verify branches
git checkout develop          # Switch again if needed
```

6️⃣ Review Changes

```powershell
git status                    # Check branch status
git log --oneline             # View recent commits
```

7️⃣ Cleanup / Merge / Delete Branch

```powershell
git checkout main              # Switch to main branch
git branch -d develop          # Delete 'develop' branch
git branch                     # Verify only main exists
```

8️⃣ Handling Untracked or New Files

```powershell
git add sample.txt             # Stage untracked file
git commit -m "Add sample.txt"  # Commit the file
git push origin main           # Push changes to GitHub
```

This segregation makes it **easier to understand and follow each step**:

1. Setup

2. File management

3. Commit

4. Push

5. Branching

6. Reviewing history

7. Cleanup

8. Handling untracked files

If you want, I can also create a **flowchart or visual diagram** of this Git workflow to make it even clearer.

Do you want me to do that?