Activity: Working with Azure DevOps and Git via Command Line

Step 1: Create a New Project in Azure DevOps

1. Go to Azure DevOps and sign in.
2. Click on **New Project**.
3. Enter a project name (e.g., MyCLIProject).
4. Choose **Private** for visibility (or Public if desired).
5. Select **Git** as the version control system.
6. Click **Create**.

Step 2: Clone Repository Using Git CLI

1. Open **Command Prompt (cmd) or PowerShell**.
2. Navigate to the directory where you want to clone the repository:

```
cd path\to\your\projects\folder
```

3. Get the repository URL from Azure DevOps:
    - In ADO, go to **Repos > Files**.
    - Click on **Clone** and copy the **HTTPS URL**.
4. Run the following command to clone the repository:

    ```
    git clone https://dev.azure.com/your-org-name/MyCLIProject/_git/MyCLIProject
    ```

5. Navigate into the cloned repository:

    ```
    cd MyCLIProject
    ```

Step 3: Create a New Branch Using Git CLI

1. Create a new branch named feature-add-files:

    ```
    git checkout -b feature-add-files
    ```

2. Verify that the branch is created:

   ```
   git branch
   ```

   - The active branch will have an asterisk (* feature-add-files).

Step 4: Open the Repository in Visual Studio and Verify Branch

1. Open **Visual Studio**.
2. Click **Open a project or solution**.
3. Select the **cloned repository folder** (e.g., MyCLIProject).
4. Once the folder opens, go to **View > Git Changes** to verify that:
   - The repository is connected.
   - The current branch is feature-add-files.
5. If the branch is not feature-add-files, switch to it:
   - Click the **branch dropdown** in **Git Changes**.
   - Select **feature-add-files**.

Step 5: Add New Files in Visual Studio

1. In **Solution Explorer**, right-click the project folder.
2. Select **Add > New Item**.
3. Choose a file type (e.g., **Class**, **Text File**, **Markdown**).
4. Name the file (e.g., NewFeature.cs).
5. Add some content to the file.
6. Save the file.

Step 6: Stage and Commit Changes Using Git CLI

1. Go back to the terminal inside the repository folder.
2. Check the status of your changes:

```
git status
```

- The new file should be listed as **Untracked**.

3. Stage the new file(s):

   ```
   git add .
   ```

4. Commit the changes with a message:

   ```
   git commit -m "Added NewFeature.cs"
   ```

5. Verify the commit:

   ```
   git log --oneline -n 1
   ```

   - The latest commit message should appear.

Step 7: Push Changes to Azure DevOps

1. Push the new branch to Azure DevOps:

   ```
   git push --set-upstream origin feature-add-files
   ```

2. Verify that the branch is available in Azure DevOps by checking **Repos > Branches**.

Step 8: Create a Pull Request (PR) in Azure DevOps

1. Go to **Azure DevOps** in your browser.
2. Navigate to **Repos > Branches**.
3. Find your branch (feature-add-files).
4. Click **New Pull Request**.
5. Ensure **main** is the target branch.
6. Add a **title** and **description**.
7. Click **Create**.

Step 9: Approve and Merge the Pull Request

1. If required, assign reviewers.

2. Click **Approve** (if you have permissions).
3. Click **Complete Merge**.
4. Select **Delete branch after merge** (optional).
5. Click **Merge**.