pytunnel: A Command Line Tool for Directory Synchronization

Introduction

pytunnel is a command-line tool that connects two directories (A and B), allowing a seamless flow of files between them. Directory A acts as the stable, Git-tracked project folder, while Directory B serves as a development sandbox for making changes without affecting the main project. This eliminates the hassle of dealing with Git branches or modifying gitignore files.

Primary Features

1. **Directory Synchronization**:
   - When pytunnel is initiated, it copies all files from A to B.
   - B acts as a development folder where tweaks and builds can be made without impacting A.
2. **Traverse Feature**:
   - When specific files from B are ready to be merged back into A, the user can use the `--traverse` command to backtrack the changes, copying selected files from B to A.
3. **Active Tunnel Environment**:
   - pytunnel operates as an active environment. Once activated between A and B, it remains in effect until manually deactivated with the `--break-link` command.
   - Any file changes in either directory while the tunnel is active will be monitored and can be synced.
4. **Real-Time Sync Monitoring**:
   - File changes in A and B are monitored in real-time (using a tool like `watchdog`), allowing smooth syncing and tracking during development.

Technology Stack

Core Development Language

- **Python**: Python will be the core language used for pytunnel. The ease of file system handling, cross-platform capabilities, and extensive library support make it a good choice.

File and Directory Management

- **shutil**, **os**, **pathlib**: These libraries will handle the copying and movement of files between directories.
- **watchdog**: This will be used to monitor changes in real-time between directories A and B.

Command-Line Interface (CLI)

- **Click** or **argparse**: For building the command-line interface to manage commands such as `--traverse` and `--break-link`.

Real-Time Sync

- Using **watchdog** to listen for any file creation, modification, or deletion in both directories while the environment is active.

Minimal Viable Product (MVP) V0.0.1

Features Included:

1. **Initial Directory Sync**:
   - Copies all files from directory A to directory B.
2. **File Monitoring**:
   - Real-time file change monitoring using the `watchdog` library, tracking both directories for changes.
3. **Selective File Traversal**:
   - Users can manually select files to "traverse" from B to A using a simple command.
4. **Tunnel Activation and Deactivation**:
   - Activate the tunnel between directories A and B.
   - Deactivate the tunnel with a command like `--break-link`.

MVP V0.0.1 Flow:

1. **Activate Tunnel**:
   - Run the command `python pytunnel.py activate_tunnel` to sync directory A to B and start monitoring changes in both directories.
2. **Modify Files in Directory B**:

- Files can be modified freely in directory B for development.

3. **Traverse Files**:
   - Use `python pytunnel.py traverse <file1> <file2>` to selectively traverse files from directory B back to directory A.
4. **Deactivate Tunnel**:
   - Use `python pytunnel.py break_link` to stop the tunnel and deactivate real-time monitoring.
