# How to Build and Run the Application

## Important Note: Prepare a Separate Testing Folder

Before starting the conversion and build process, make a copy of your original project folder or files. Use this copied folder for testing to ensure your original files remain intact. This will help:  
- Avoid accidental modifications or data loss in the original project.  
- Keep a working backup in case issues arise during testing.

## Step 1: Prepare Your Environment

1. Install Python:

- Download and install Python 3.12 or higher from https://www.python.org/.  
- During installation, check the option to add Python to your system PATH.  
- Verify the installation:  
  
 python --version

2. Install pip (if not already installed):

- Check if pip is installed:  
  
 python -m pip --version  
  
- If pip is missing, install it:  
  
 python -m ensurepip --upgrade

3. Install Required Tools:

- Install cython and pyinstaller:  
  
 pip install cython pyinstaller  
  
- Verify the installations:  
  
 cython --version  
 pyinstaller --version

4. Install Required Libraries:

- Save the following dependencies to a requirements.txt file:  
  
 PyYAML>=6.0  
 netmiko>=4.5.0  
 rich>=13.5.2  
  
- Install them:  
  
 pip install -r requirements.txt

## Step 2: Convert main.py to Cython

1. Copy Your Files to the Testing Folder:

- Ensure the original main.py file and any dependencies are safely backed up.  
- Work in a separate copy of your project folder to avoid accidental data loss.

2. Rename Your Python File:

- Rename main.py to main.pyx to indicate it will use Cython.

3. Create a Setup File:

- Create a setup.py file in the same directory with the following content:  
  
 from setuptools import setup  
 from Cython.Build import cythonize  
  
 setup(  
 ext\_modules=cythonize("main.pyx"),  
 )

4. Compile the Cython File:

- Run the following command:  
  
 python setup.py build\_ext --inplace

- This will generate a compiled file (main.so or main.pyd, depending on your system).

5. Test the Compiled File:

- Create a new script, run.py, to use the compiled file:  
  
 import main  
 if \_\_name\_\_ == "\_\_main\_\_":  
 main.main()

- Run run.py to ensure everything works:  
  
 python run.py

## Step 3: Build the Executable

1. Open a terminal in the testing folder.  
2. Run the following command to create a standalone .exe:  
  
 pyinstaller --onefile --strip --name=CommandMate --collect-all=netmiko --collect-all=rich --collect-all=concurrent run.py

## Step 4: Locate the Executable

- After the build completes, navigate to the dist folder.  
- The executable file (CommandMate.exe) will be located there.

## Step 5: Create a Launcher (Optional)

To keep the terminal window open after execution:  
1. Create a file named CommandMate\_launcher.bat in the same directory as CommandMate.exe.  
2. Add the following content:  
  
 @echo off  
 cls  
 title CommandMate.exe Launcher  
 echo Launching CommandMate.exe...  
 echo =====================  
 CommandMate.exe  
 echo =====================  
 echo.  
 echo Program finished. Choose an option:  
 echo [1] Relaunch the program  
 echo [2] Close this window  
 echo.  
 set /p choice=Enter your choice (1-2):   
  
 if "%choice%"=="1" (  
 cls  
 echo Relaunching CommandMate.exe...  
 echo =====================  
 CommandMate.exe  
 pause  
 exit  
 ) else if "%choice%"=="2" (  
 echo Closing...  
 exit  
 ) else (  
 echo Invalid choice. Closing...  
 pause  
 )

## Step 6: Run the Application

- Double-click run\_launcher.bat to start the program with a persistent terminal window.  
- Alternatively, double-click CommandMate.exe for a temporary terminal window.

## Requirements

Ensure the following Python packages are installed before building:  
  
 PyYAML>=6.0  
 netmiko>=4.5.0  
 rich>=13.5.2

## Information

This script is parallel because it uses ThreadPoolExecutor from the concurrent.futures module to execute tasks concurrently. The line:  
  
 with ThreadPoolExecutor(max\_workers=10) as executor:

limits the number of parallel threads to a maximum of 10 workers. This means:  
- Up to 10 hosts will have commands executed simultaneously.  
- Additional hosts (if more than 10) will wait in a queue until a thread becomes available.  
  
By increasing max\_workers, you can allow more threads to run concurrently. However, keep in mind system resource constraints and potential limitations of the network devices being managed.