Phase 1: Getting WSL & Ubuntu Ready

1. Open an Administrator PowerShell.
2. Run `wsl --set-default-version 2` (This sets the default for any future installations).
3. Run `wsl --install Ubuntu` (This installs the fresh Ubuntu OS).
4. Once it's done, open the "Ubuntu" app from your Start Menu.
5. Complete the one-time setup by creating your UNIX username and password.

Phase 2: Building Project

🧩 Step 1 — Open your Ubuntu Terminal

As mentioned above, open the "Ubuntu" app from your Windows Start Menu. This is your command line for all the following steps.

🧰 Step 2 — Update Ubuntu and Install Core Tools

This step from the original guide is correct. It ensures your system is up-to-date.

Bash
sudo apt update && sudo apt upgrade -y
sudo apt install -y python3.12 python3.12-venv python3-pip git build-essential wget bash

🧱 Step 3 — (Skipped) Do NOT Upgrade Base Python Tools

We are intentionally skipping the original Step 3. As explained above, we will not modify the system's Python.

📁 Step 4 — Go to Your Linux Home Directory (The Right Place for Projects)

Instead of navigating to `/mnt/c/`, we'll work from your WSL home directory. It's cleaner, faster, and avoids permission errors. The `~` symbol is a shortcut for your home directory (e.g., `/home/vadan/`).

Bash
cd ~

You are now in the perfect place to store your projects.

📦 Step 5 — Clone the Repository

Now, run the `git clone` command here using your personal repository link. This will work without any permission errors and move you into the new project directory.

Bash
git clone https://github.com/VadanKhan/fusion-neutronics-modelling.git && cd fusion-neutronics-modelling

Your project now lives at `~/fusion-neutronics-modelling` inside your Linux environment.

🧪 Step 6 — Create and Activate a Virtual Environment

This step is correct. It creates an isolated Python "bubble" for your project.

Bash
python3.12 -m venv .venv
source .venv/bin/activate

You'll know it's active because your terminal prompt will change to show `(.venv)`.

⚙️ Step 7 — Install Python Dependencies (The Correct Way)

Now that your virtual environment is active, this is the correct place to upgrade your build tools and then install the project's requirements.

Bash
First, upgrade pip, setuptools, and wheel INSIDE the venv
python -m pip install --upgrade pip setuptools wheel

Now, install the project packages from requirements.txt
python -m pip install -r requirements.txt

☢️ Step 8 — Download Nuclear Data

This step remains the same. It runs a script to download necessary data.

Bash
bash postBuild

This will download data to a new folder in your Linux home directory called `~/nuclear_data`.

🧬 Step 9 — Launch Jupyter Lab

This step remains the same for running the project directly from the terminal. For the recommended VS Code method, see the next step.

Bash
jupyter lab

When Jupyter starts, it will give you a URL like `http://localhost:8888/lab?token=...`. Copy this full URL and paste it into a web browser (like Chrome or Edge) on Windows. It will connect to the Jupyter server running inside WSL.

💻 Step 10 — Open in VS Code (The Recommended Workflow)

For a much better development experience, you can open this entire WSL project in VS Code on Windows.

Prerequisites:

1. **VS Code Installed:** You must have Visual Studio Code installed on Windows.
2. **WSL Extension:** You need the official **WSL extension** from Microsoft. If you don't have it, VS Code will likely prompt you to install it during this process.

Instructions:

1. **Navigate to Project Directory:** Make sure your Ubuntu terminal is still in the project directory (`~/fusion-neutronics-modelling`) and the virtual environment is active (`(.venv)` should be visible).
2. **Launch VS Code:** In the **Ubuntu terminal**, type the following command and press Enter:

   Bash
   code .
3. **What Happens Next:**
   - This command tells WSL to launch your Windows VS Code application and automatically connect it to the current Linux directory (`.`).
     i. **This may have to install vs code on wsl first.**
   - VS Code will open on your Windows desktop. Look at the bottom-left corner of the VS Code window. You should see a green button that says **"WSL: Ubuntu"**. This confirms you are connected to your Linux environment.
     i. **You may have to reinstall some extensions for wsl integration, this is normal behaviour.**
4. **Work with Your Files:**
   - You can now use the VS Code file explorer on the left to open and edit files (like `.ipynb` notebooks or `.py` scripts). All changes are saved directly within the WSL file system.
   - When you open a Jupyter Notebook (`.ipynb` file), VS Code's built-in notebook editor will be used. It may ask you to select a Python kernel. Be sure to select the one that includes `.venv` in its path. This ensures you are using the packages you installed in your virtual environment.
5. **Connect to Your Git Repository:**
   - VS Code automatically detects that your folder is a Git repository.
   - Click on the **Source Control** icon in the Activity Bar on the left (it looks like three dots with branches).
   - Here, you can see all your file changes, stage them, write commit messages, and sync (push/pull) with your GitHub repository

(`VadanKhan/fusion-neutronics-modelling`) directly from the VS Code interface.

i. Before you can push you may have to setup git profile Git stuff