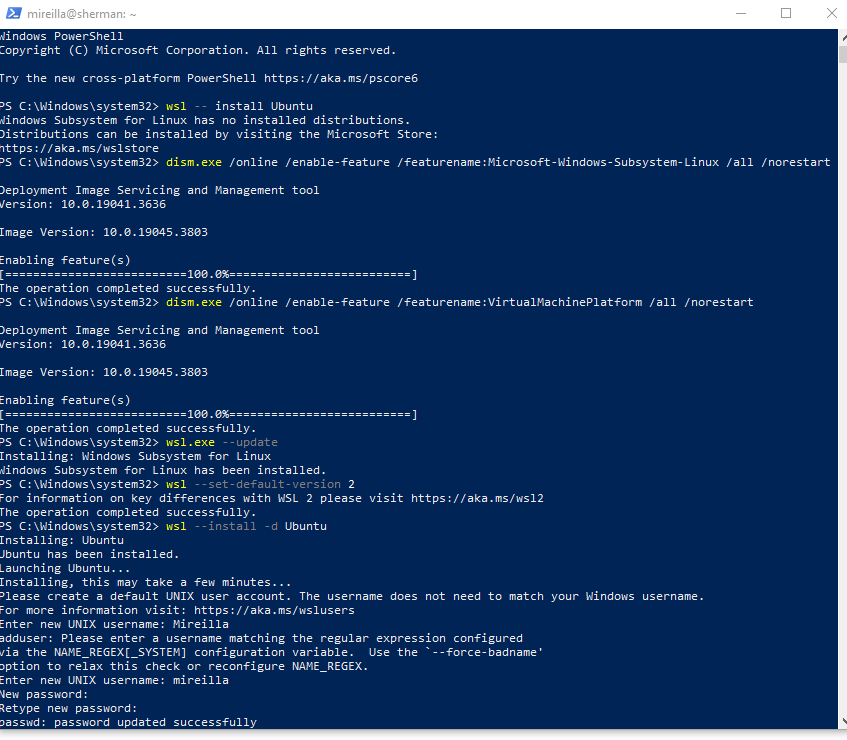
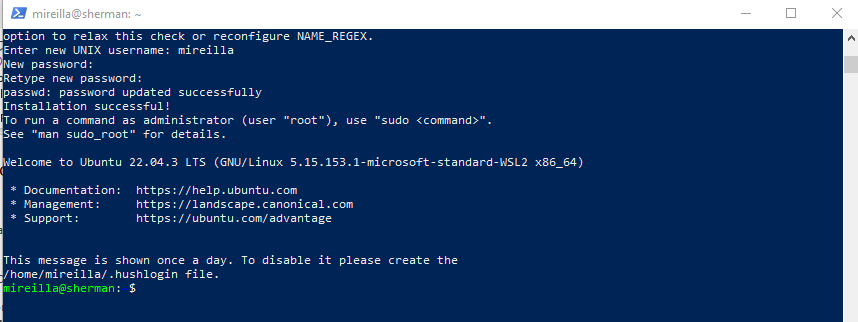
# Running Bash on your Windows device using Windows Subsystem for Linux (WSL) (those with ubuntu can skip task a)

1. Make sure you have Windows PowerShell already installed. Open Windows PowerShell (right-click on the Start button of Windows) and Install Ubuntu.

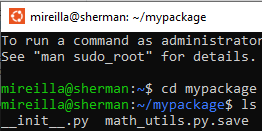




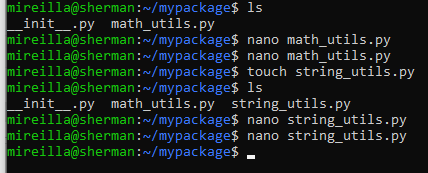
1. Create your package and modules.

To open the Ubuntu app, I searched “Ubuntu” on my Windows device.

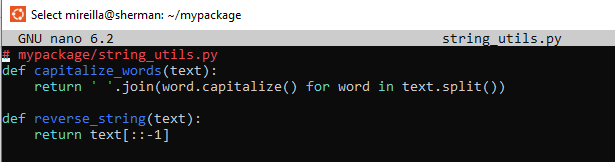
I created a directory **mypackage** using **mkdir mypackage.** Then, I moved inside it using the command **cd mypackage.** Inside the package, I added an empty **\_\_init\_\_.py** file in the directory **mypackage** to indicate that it is a package. I created **math\_utils.py** and **string\_utils.py** modules using the command **touch**. Use **ls** to verify what is inside **mypackage**.



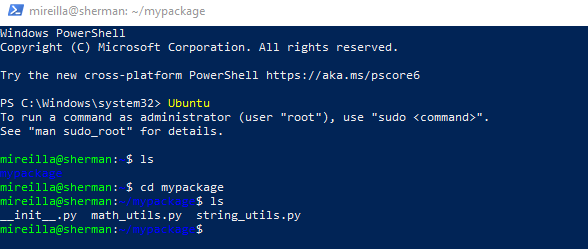
I used **nano** to open each of my modules and edit it.



I used **nano** command to open the file string\_utils.py and entered the text I wanted:



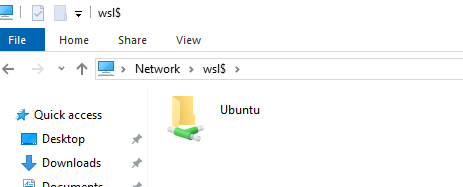
You can open Ubuntu from your PowerShell and see the same files.



**Ensure Jupyter Notebook is Running in the Correct Environment**: Make sure that Jupyter Notebook is running in the same environment where your package is installed or accessible. If your package is not in the default Python path, you need to add the path to your package in your Jupyter Notebook.

Windows provides a way to access WSL files directly from Windows applications, including Jupyter Notebook. Here’s a step-by-step guide on how to set this up:

1. Open Windows File Explorer.
2. Type the path **\\wsl$** into the address bar to access the root of your WSL file system.

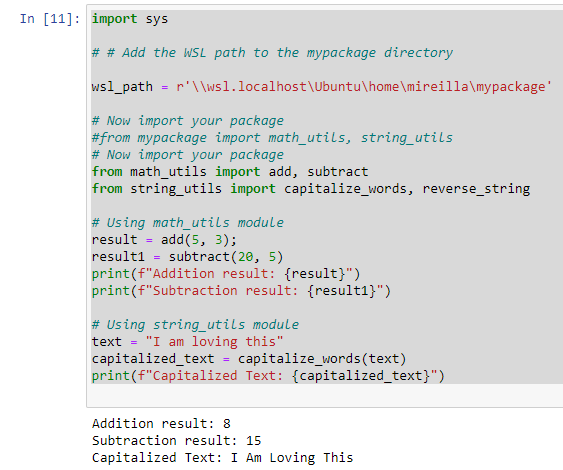


1. Navigate through the directories to find your package, e.g., \\wsl$\Ubuntu\home\username\mypackage.

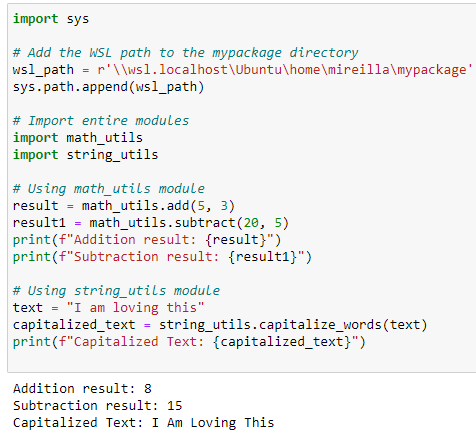
A screenshot of a computer

Description automatically generated

1. Copy and add your package path to your code. In this example, I have successfully added the WSL path to mypackage and **imported modules directly**.



You could also import entire modules



## Using a relative path

You can also make sure that your path is relative. A relative path is useful when you want the code to be portable and not tied to a specific directory structure on a particular machine.

