

Setting Up a Development Environment with WSL, VS Code, and Python

1 Introduction

In today's software development landscape, having a robust and efficient development environment is essential for productivity and collaboration. This guide aims to walk you through the process of setting up a powerful development environment using Windows Subsystem for Linux (WSL), Visual Studio Code (VS Code), and Python. By leveraging these tools, developers can enjoy the flexibility of Linux-based development alongside the familiar interface of Windows, coupled with the extensive features and extensions provided by VS Code.

2 Installing Ubuntu 22.04 on WSL

2.1 Open PowerShell

PowerShell serves as the command-line interface for executing administrative tasks and commands in Windows. It provides powerful scripting capabilities and facilitates system management tasks.

```
1 wsl --install -d Ubuntu-22.04
```

2.2 Search and Launch Ubuntu

Once the installation is complete, Ubuntu can be accessed by searching for it in the Windows search bar and launching it like any other application.

2.3 Provide Username and Password

Upon launching Ubuntu for the first time, you'll be prompted to create a new username and password for your Ubuntu environment. For example, username: `user123`, password: `*****`.

```
1 user123@LAPTOP-VTGNBQQR:~$
```

3 Setting up VS Code on WSL

3.1 Open Ubuntu Terminal

With Ubuntu running on WSL, open the Ubuntu terminal to execute commands and perform system tasks.

```
1 bash
```

3.2 Navigate to Desired Directory

```
1 user123@LAPTOP-VTGNBQQR:~$ cd /
2 user123@LAPTOP-VTGNBQQR:/$ cd home/user123
3 user123@LAPTOP-VTGNBQQR:~$ mkdir AiMLprojects
4 user123@LAPTOP-VTGNBQQR:~$ cd AiMLprojects
5 user123@LAPTOP-VTGNBQQR:~/AiMLprojects$ mkdir project1
6 user123@LAPTOP-VTGNBQQR:~/AiMLprojects$ cd project1

1 user123@LAPTOP-VTGNBQQR:~/AiMLprojects$ code .
```

3.3 Extension Installation

Upon opening VS Code, it may prompt you to install recommended extensions for WSL development. These extensions enhance the functionality of VS Code for Linux-based development.

4 Installing Python in WSL

4.1 Confirm Python3 Installation

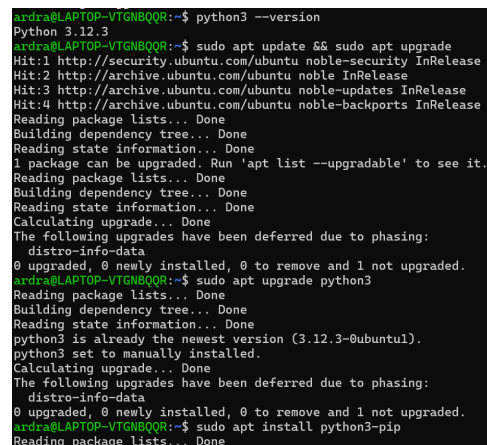
```
1 user123@LAPTOP-VTGNBQQR:~$ python3 --version
```

4.2 Update Ubuntu and Python

```
1 user123@LAPTOP-VTGNBQQR:~$ sudo apt update && sudo apt upgrade
```

4.3 Install pip (Python Package Manager)

```
1 user123@LAPTOP-VTGNBQQR:~$ sudo apt install python3-pip
```



```
ardra@LAPTOP-VTGNBQQR:~$ python3 --version
Python 3.12.3
ardra@LAPTOP-VTGNBQQR:~$ sudo apt update && sudo apt upgrade
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following upgrades have been deferred due to phasing:
  distro-info-data
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
ardra@LAPTOP-VTGNBQQR:~$ sudo apt upgrade python3
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.12.3-0ubuntu1).
python3 set to manually installed.
Calculating upgrade... Done
The following upgrades have been deferred due to phasing:
  distro-info-data
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
ardra@LAPTOP-VTGNBQQR:~$ sudo apt install python3-pip
Reading package lists... Done
```

Figure 1: Example Image

4.4 Install venv (Python Virtual Environment)

```
1 user123@LAPTOP-VTGNBQQR:~$ sudo apt install python3-venv
```

```
ardra@LAPTOP-VTGNBQQR:~$ sudo apt install python3-venv
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  python3-pip-whl python3-setuptools-whl python3.12-venv
The following NEW packages will be installed:
  python3-pip-whl python3-setuptools-whl python3-venv python3.12-venv
0 upgraded, 4 newly installed, 0 to remove and 1 not upgraded.
Need to get 2424 kB of archives.
```

Figure 2: Example Image

4.5 What is a Virtual Environment?

A virtual environment in Python is a self-contained directory that contains a Python installation for a particular version of Python, plus a number of additional packages. It allows developers to work on multiple projects with different dependencies simultaneously, without those dependencies interfering with each other or with the system-wide Python installation. In other words virtual environment is like a separate workspace for your Python projects. Imagine having multiple projects, each requiring different versions of Python or specific packages. Instead of installing everything globally on your computer, which could lead to conflicts, a virtual environment keeps everything isolated. It's like having a mini Python universe just for that project. When you activate a virtual environment, it sets up an environment where you can install specific versions of Python and libraries without affecting other projects. This way, you can work on different projects with different requirements without worrying about compatibility issues. It's like having different rooms in a house for different activities, keeping everything organized and preventing chaos.

4.6 Why Do We Need It?

We need virtual environments primarily to manage dependencies and avoid conflicts between different Python projects. Without virtual environments, installing Python packages globally could lead to version mismatches or dependency clashes. For example, Project A might require a specific version of a package, while Project B needs a different project version. If both projects share the same global environment, installing one version could break the other project. Virtual environments provide a solution by creating isolated environments for each project. This isolation ensures that each project has its own set of dependencies and Python interpreter, preventing conflicts. Additionally, virtual environments promote reproducibility by allowing you to recreate the exact environment in which a project was developed, making it easier to share and collaborate on code. In essence, virtual environments serve as sandboxed spaces where you can experiment, develop, and deploy Python projects without worrying about interference from other projects or system-level dependencies.

4.7 How Is It Beneficial for MLOps?

Virtual environments are essential in MLOps for creating controlled, reproducible environments tailored to machine learning projects. They enable isolation of dependencies, ensuring consistency across various project stages like data preprocessing, training, and deployment. By encapsulating dependencies, virtual environments promote collaboration and reproducibility, facilitating easy sharing of work among team members. Moreover, they aid in version control and dependency management, mitigating compatibility issues and ensuring smooth deployment in production environments. Overall, virtual environments streamline the machine learning workflow, enhancing collaboration, efficiency, and model reliability in MLOps.

4.8 Create a Virtual Environment

```
1 user123@LAPTOP-VTGNBQQR:~/AiMLprojects$ python3 -m venv .venv
```

4.9 Activate the Virtual Environment

```
1 user123@LAPTOP-VTGNBQQR:~/AiMLprojects$ source .venv/bin/activate
```

4.10 Create a Test Python File

```
1 (.venv) user123@LAPTOP-VTGNBQQR:~/AiMLprojects$ touch test.py
```

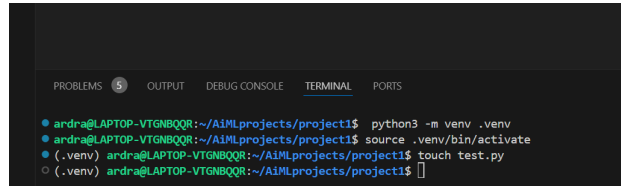


Figure 3: Example Image

4.11 Install Required Packages

```
1 (.venv) user123@LAPTOP-VTGNBQQR:~/AiMLprojects$ pip install pandas numpy
```

Explanation: The command `pip install` is used to install Python packages. `pandas` is a powerful data analysis and manipulation library, while `numpy` is a fundamental package for scientific computing with Python.

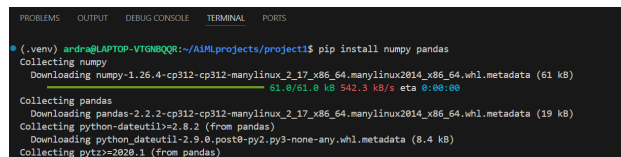


Figure 4: Example Image

4.12 Write and Run Example Program

Write a sample Python program using the installed packages (`pandas` and `numpy`) to verify their functionality within the project environment. This step ensures that the packages are installed correctly and can be imported and used in Python scripts.

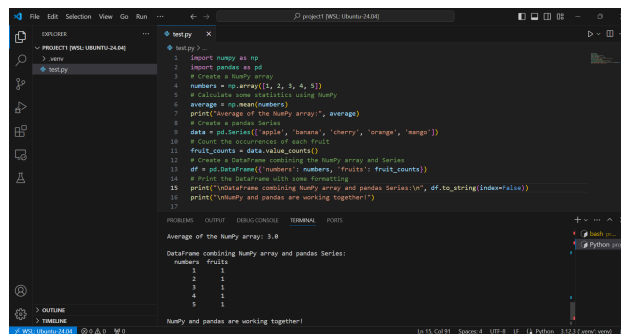
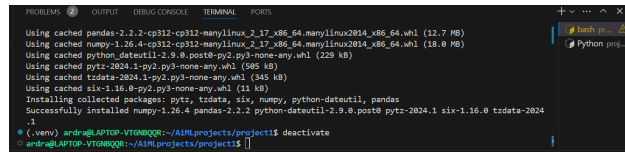


Figure 5: Example Image

4.13 Deactivate the Virtual Environment

```
1 (.venv) user123@LAPTOP-VTGNBQQR:~/AiMLprojects$ deactivate
```

Note: If you encounter a "permission denied" error when deactivating the virtual environment, ensure that the virtual environment was created within your project folder. Use the command `source deactivate` to deactivate in such cases.



```
Using cached pandas-2.2.2-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.7 MB)
Using cached numpy-1.26.4-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.0 MB)
Using cached python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
Using cached pytz-2024.1-py2.py3-none-any.whl (505 kB)
Using cached tzdata-2024.1-py2.py3-none-any.whl (345 kB)
Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: pytz, tzdata, six, numpy, python-dateutil, pandas
Successfully installed numpy-1.26.4 pandas-2.2.2 python-dateutil-2.9.0.post0 pytz-2024.1 six-1.16.0 tzdata-2024.1
(.venv) andrag@LAPTOP-VTQNRQQR:~/AIHL/projects/project1$ deactivate
andrag@LAPTOP-VTQNRQQR:~/AIHL/projects/project1$
```

Figure 6: Example Image

5 Conclusion

In conclusion, setting up a development environment with WSL, VS Code, and Python offers a seamless and efficient workflow for software development on Windows machines. By leveraging the power of Linux-based development alongside the versatility of Python and the feature-rich environment of Visual Studio Code, developers can enjoy a productive and integrated development experience. With the step-by-step instructions provided in this guide, you can easily configure your development environment and embark on your coding journey with confidence.