CSCI 1300 CS1: Starting Computing
Ashraf, Cox, Spring 2020
Visual Studio Code

You will also use Visual Studio Code (VS Code) to write and execute your programs locally.

**Important: Before proceeding with this document, make sure that you have run Windows Update within your Windows 10 environment. You must have the latest updates installed.**
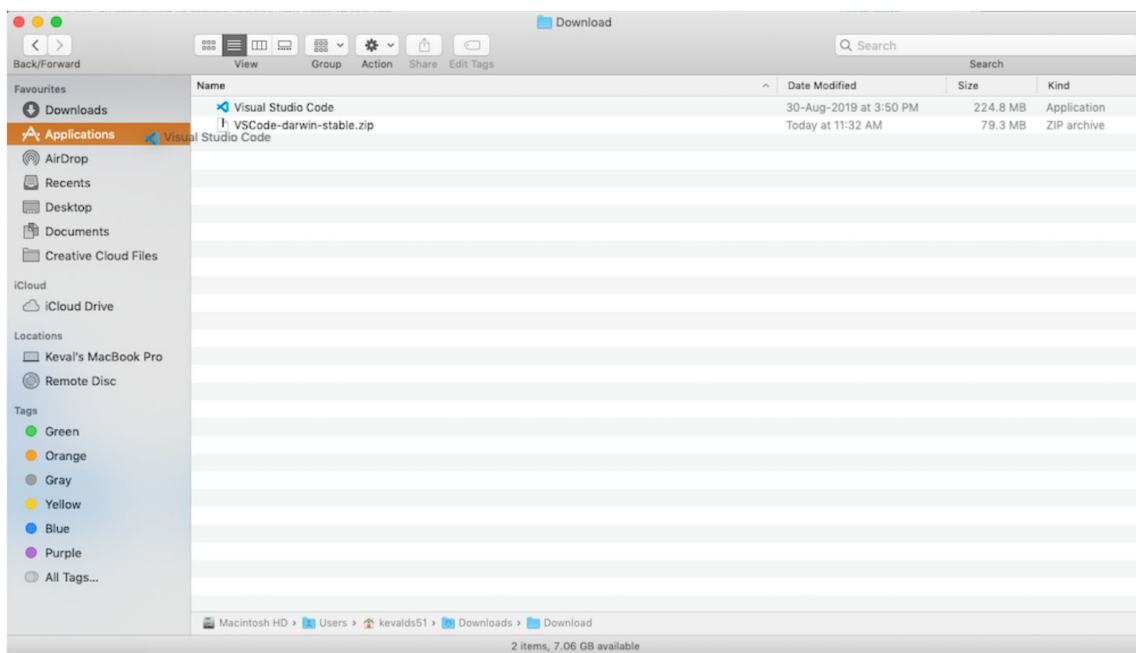
# Mac Installation Guide Part 1

**Step 1**: Go to VS code download page, and download for Mac.
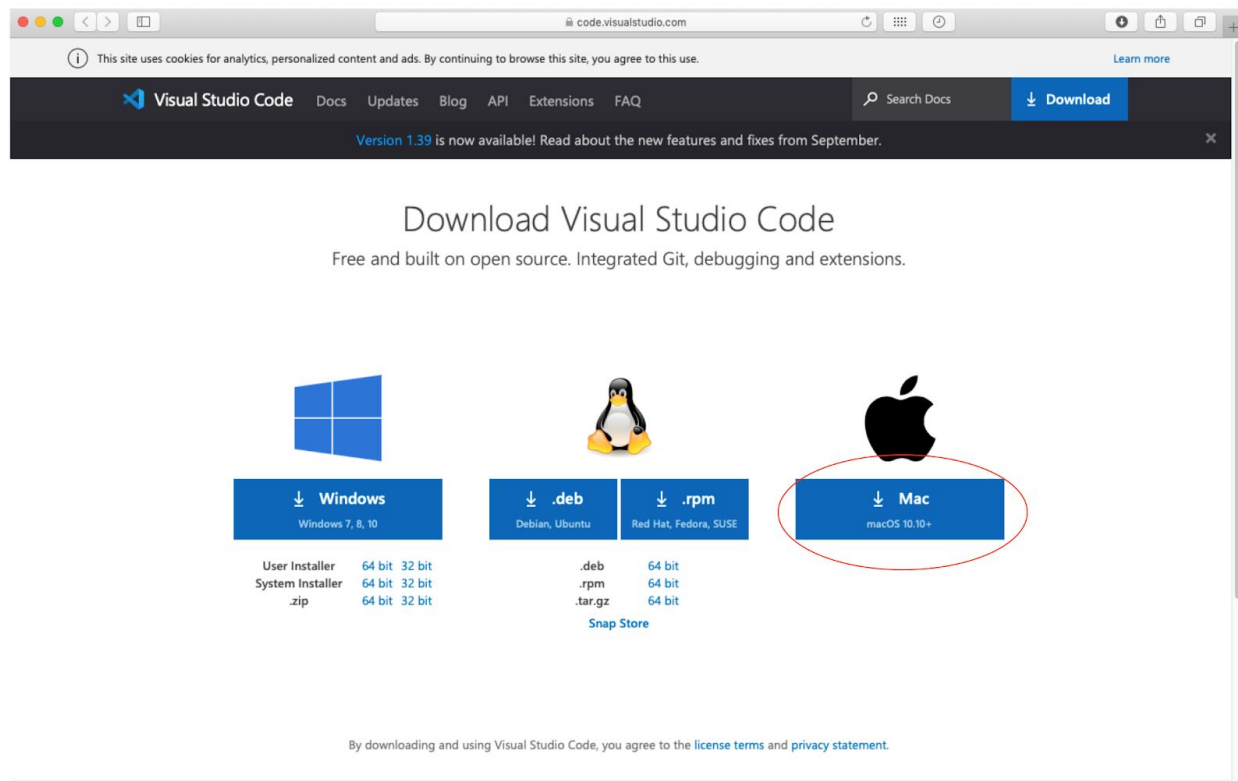
**Step 2**: After the download has finished, unzip the folder by double-clicking on it.
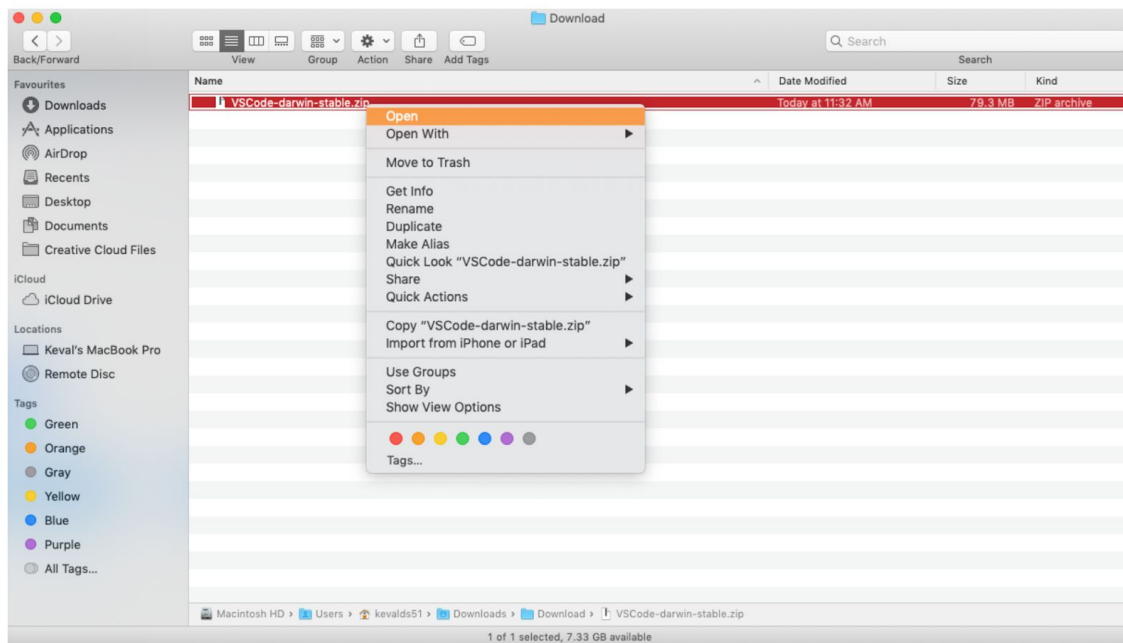


**Step 3**: Now you can see the "Visual Studio Code" application. Drag and drop this icon to the "Applications" folder of your computer.

**Step 4**: Double click on the "Visual Studio Code" icon to launch the application. (You might need to right click and select "open" if you cannot launch the program). Next, select the new "Terminal option" to open the terminal window.
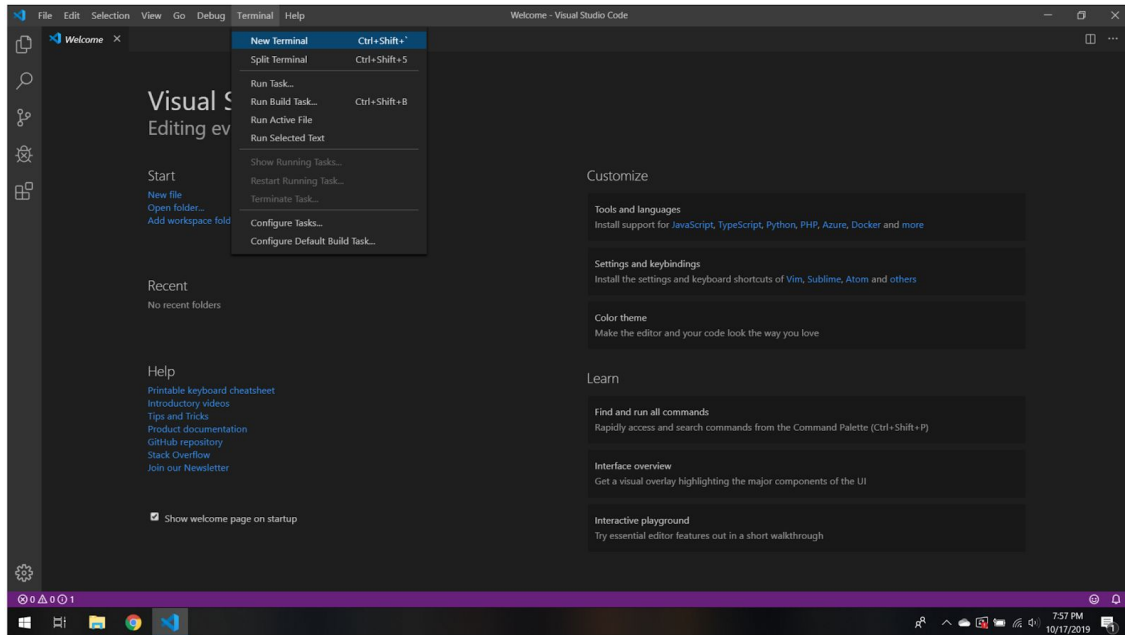


**Step 5**: Install C/C++ extension. In the toolbar on the left hand side of the screen click on the bottom icon for Extensions. Search for C/C++ and click install.
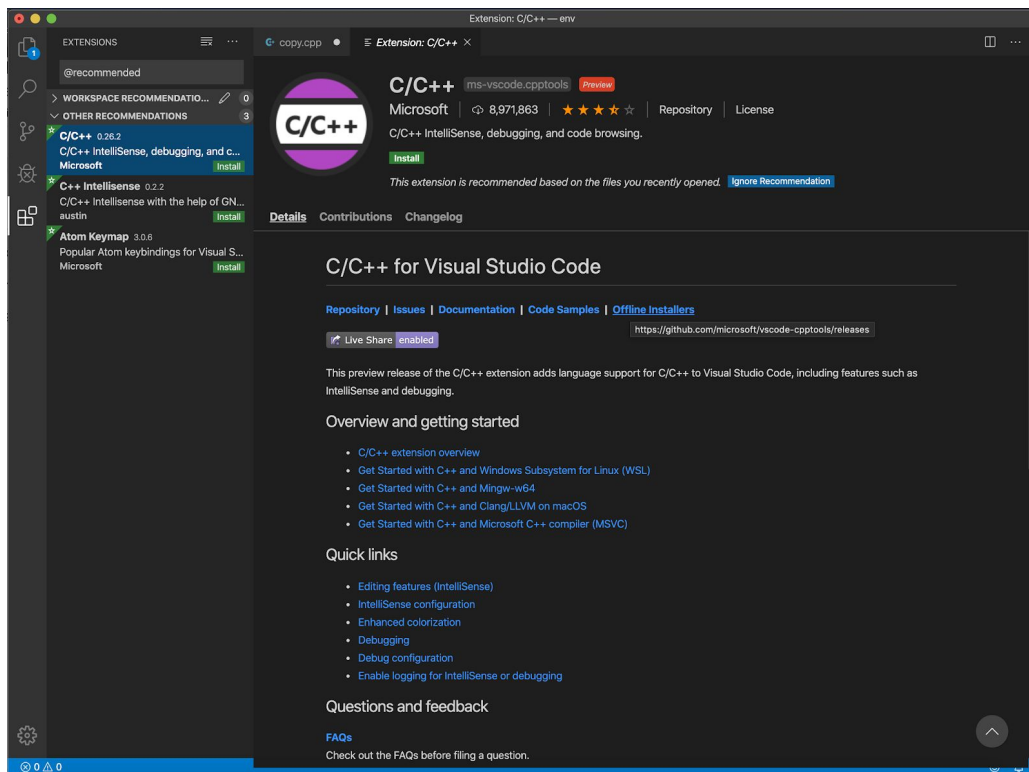
# Mac Installation Guide Part 2

**Install g++**

1. Open a Terminal window.
   Press ⌘ Command+Space, type **Terminal** in the search field, and press the Return key.
2. In the Terminal window type **g++** and press the Return key. We will see an alert box like this:
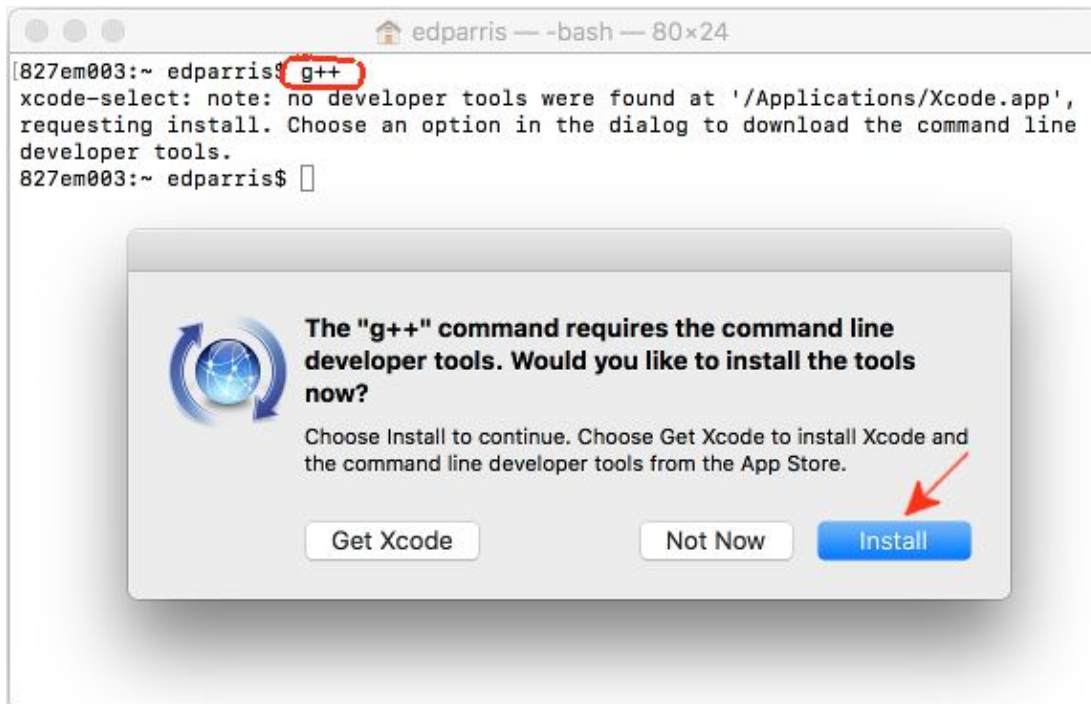


3. Choose **Install** to get only the command line tools unless you want to learn Xcode.
   Xcode can be installed later from the App Store.

   After installation, type **g++** in the Terminal, press the Return key, and verify the terminal prints the message, "no input files".
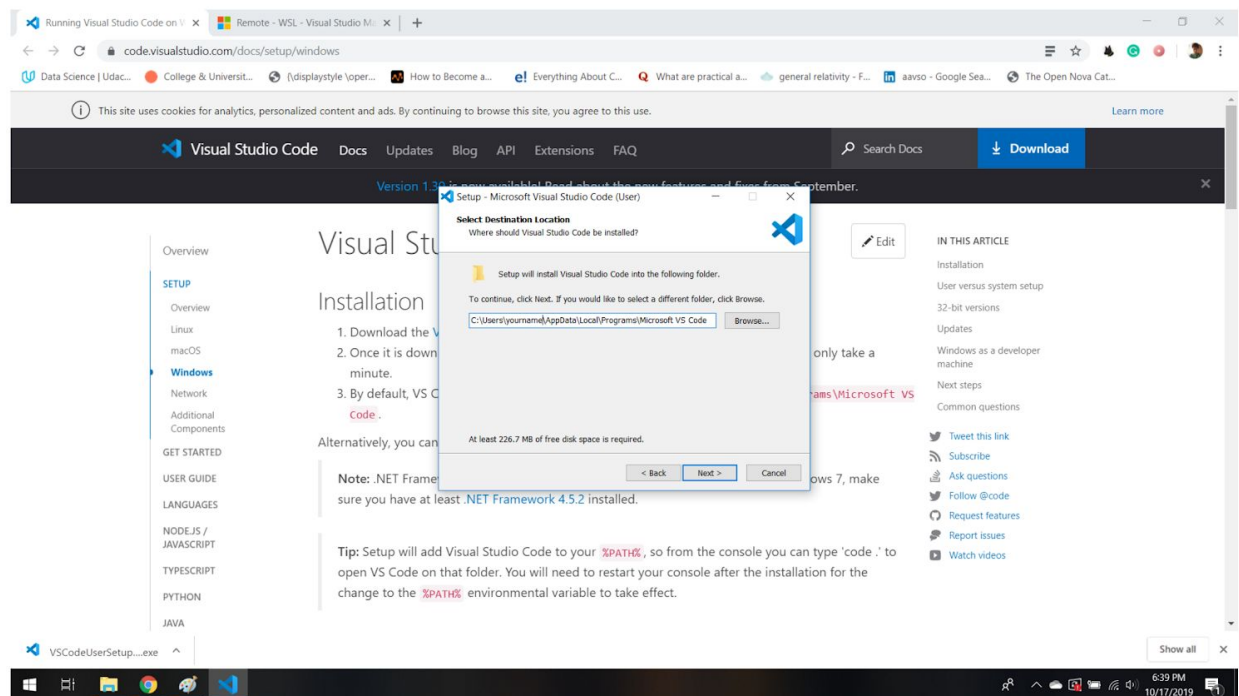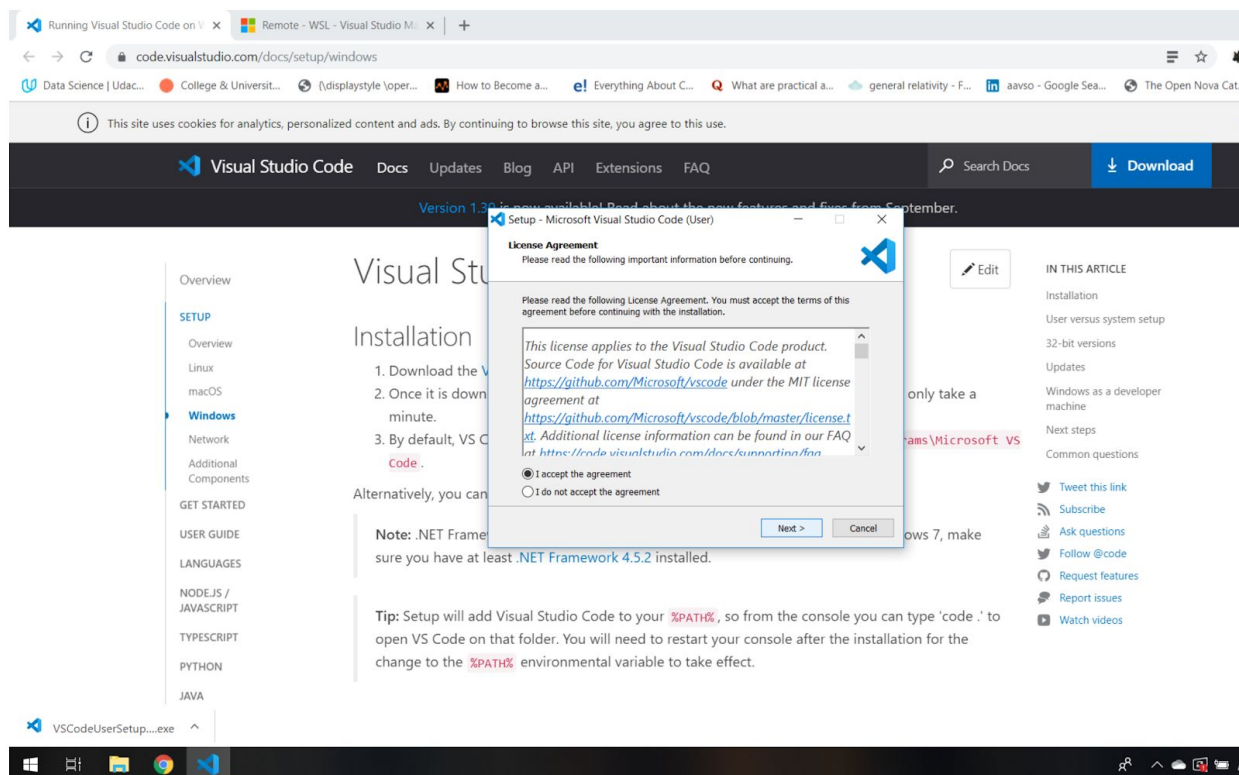   > **$ g++**
   > **clang: error: no input files**

Well done! Go to the section Compiling Code In Terminal (For Both Mac & Windows) to compile code on your computer.

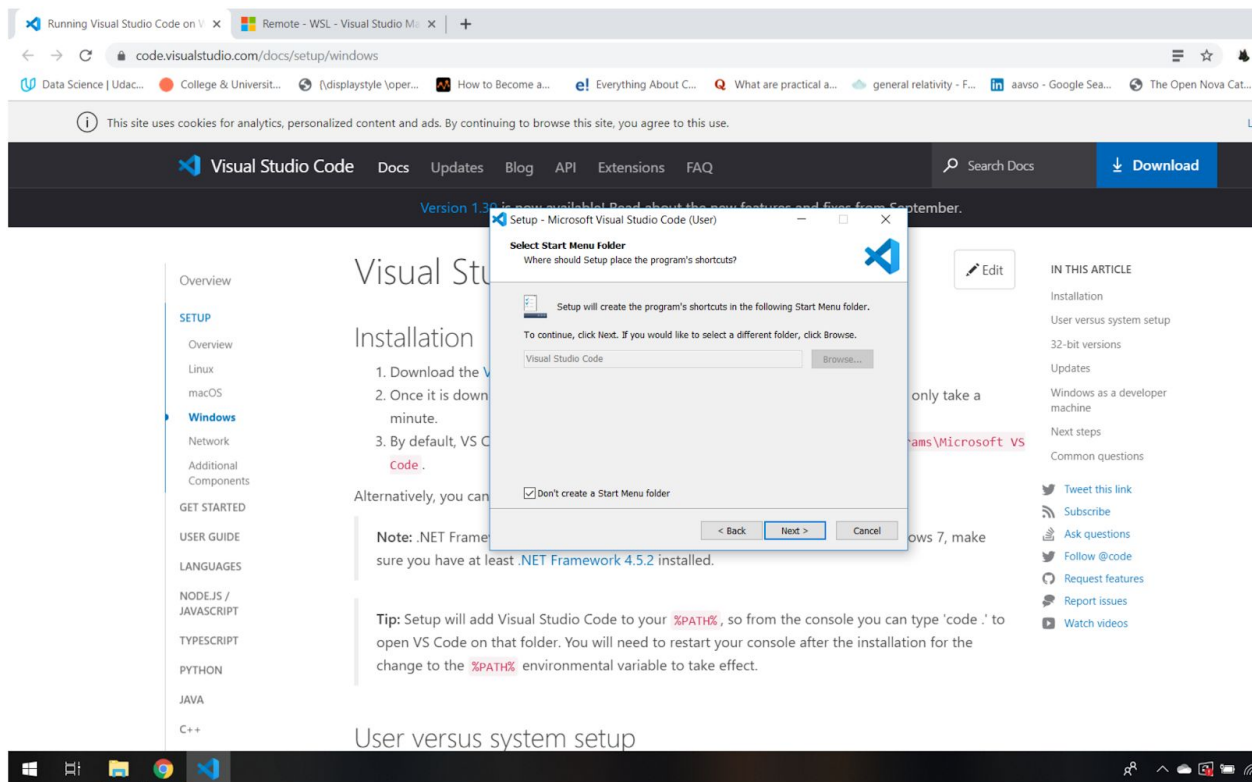# Windows Installation Guide (part 1)

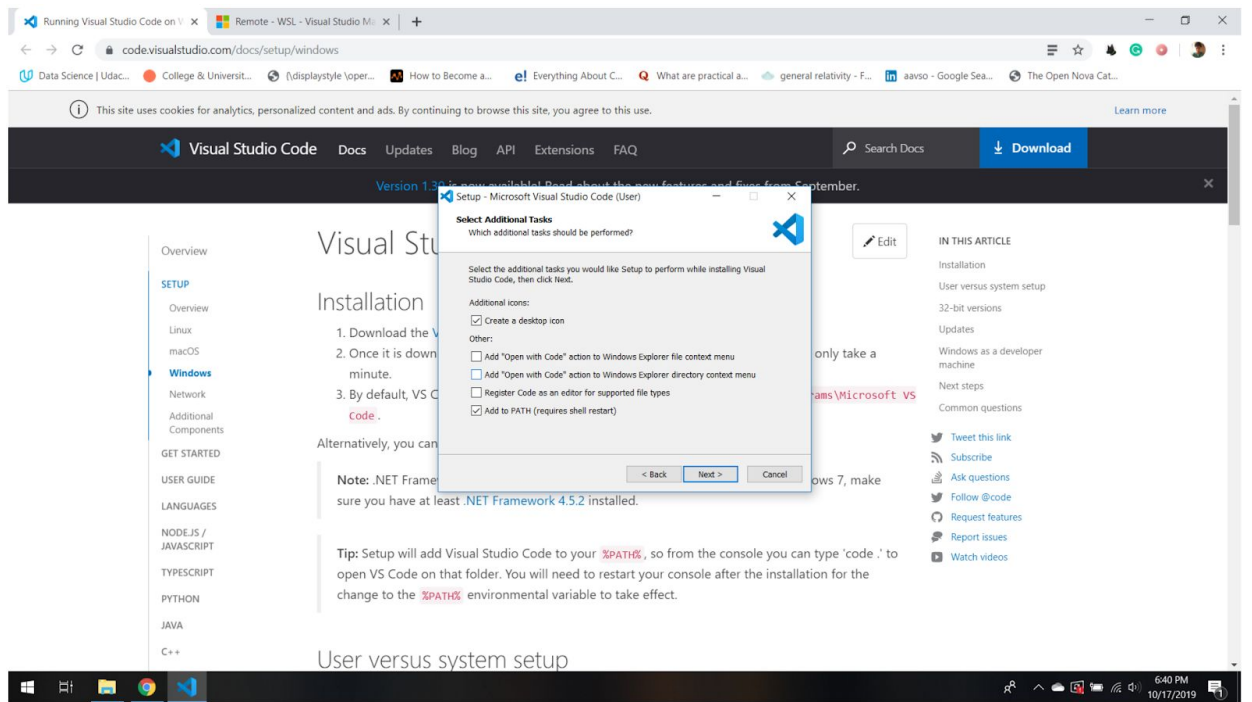**Step 1**: Go to VS code [download page](#), and download for Windows.
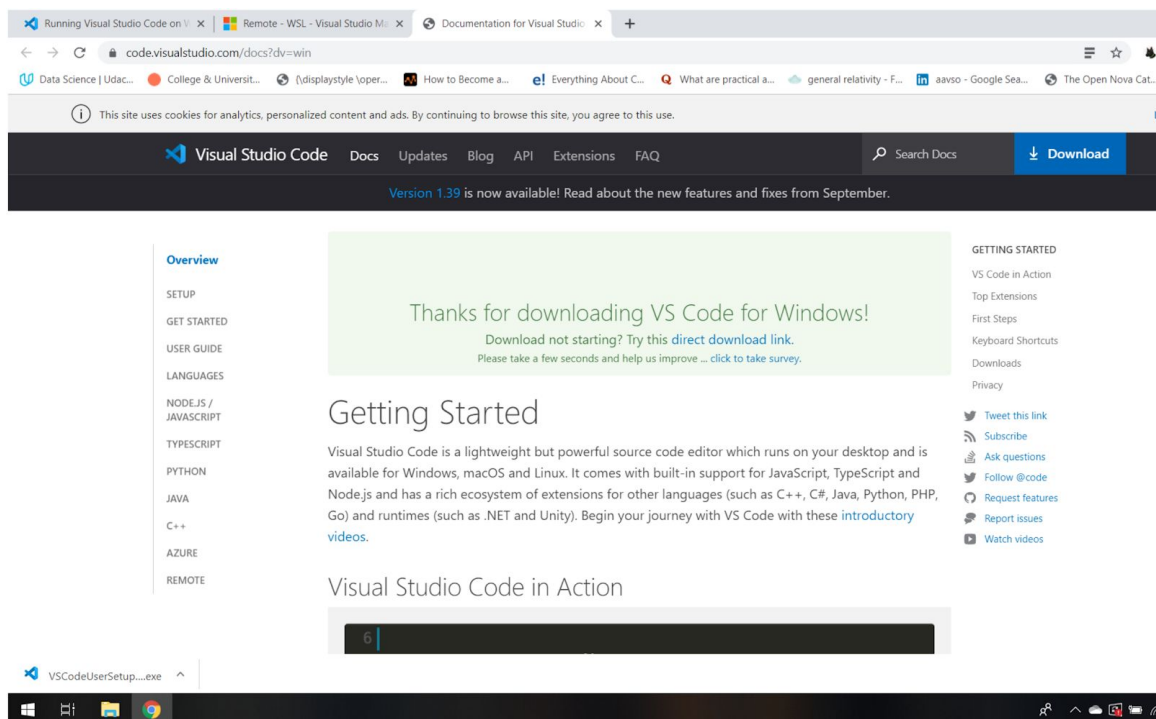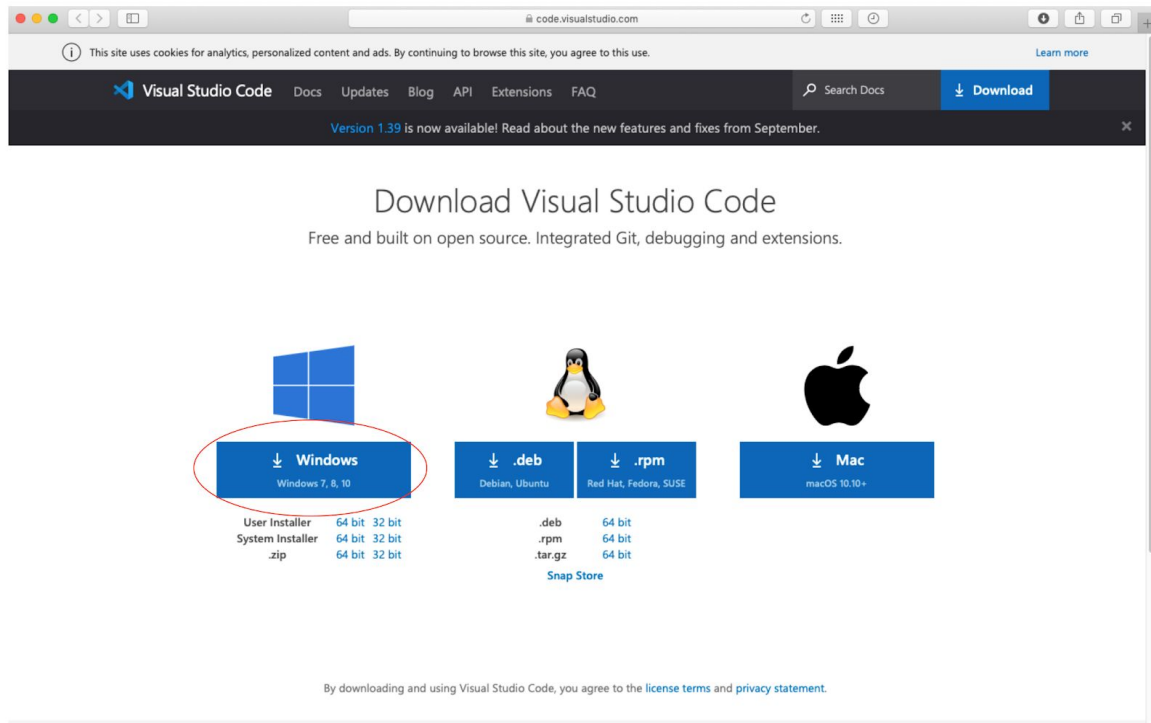
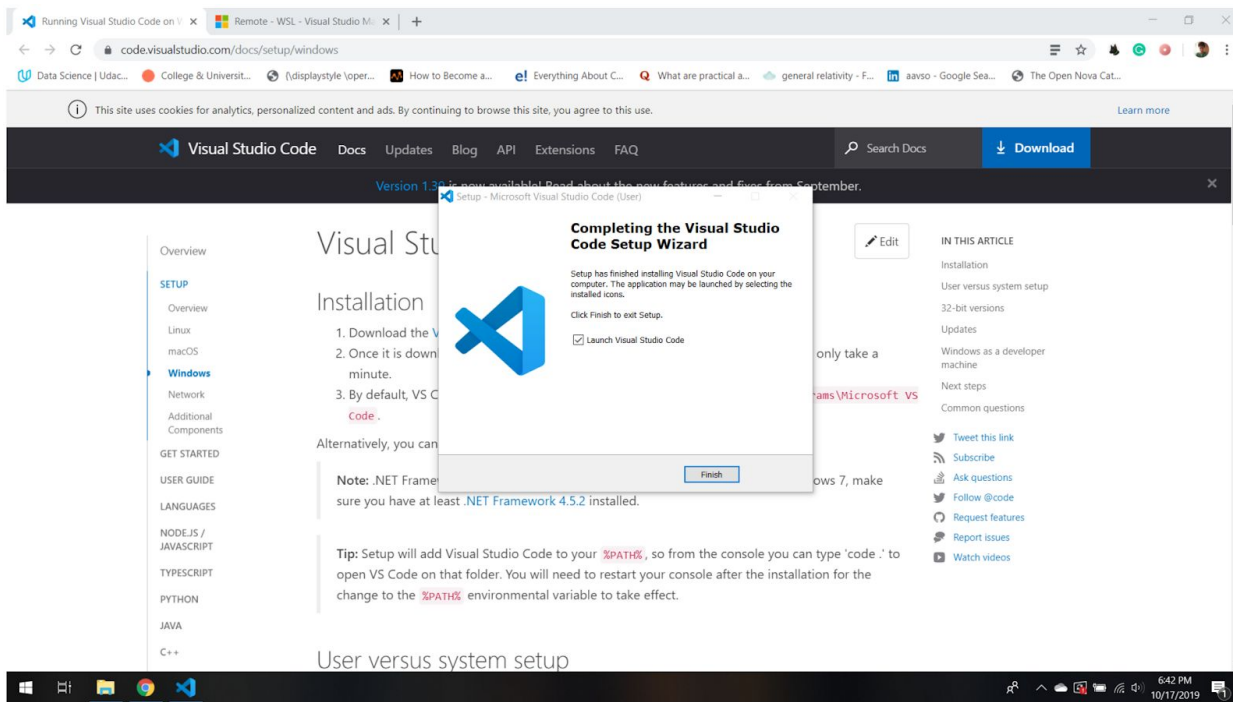**Step 2**: Accept the user agreement and proceed to follow the steps for installation

**Step 3**: Choose Options based on your preferred settings. If you're not sure, you can stick with the ones used in this tutorial.

**Step 4**: Click on Install and wait for Visual Studio Code to finish installing.
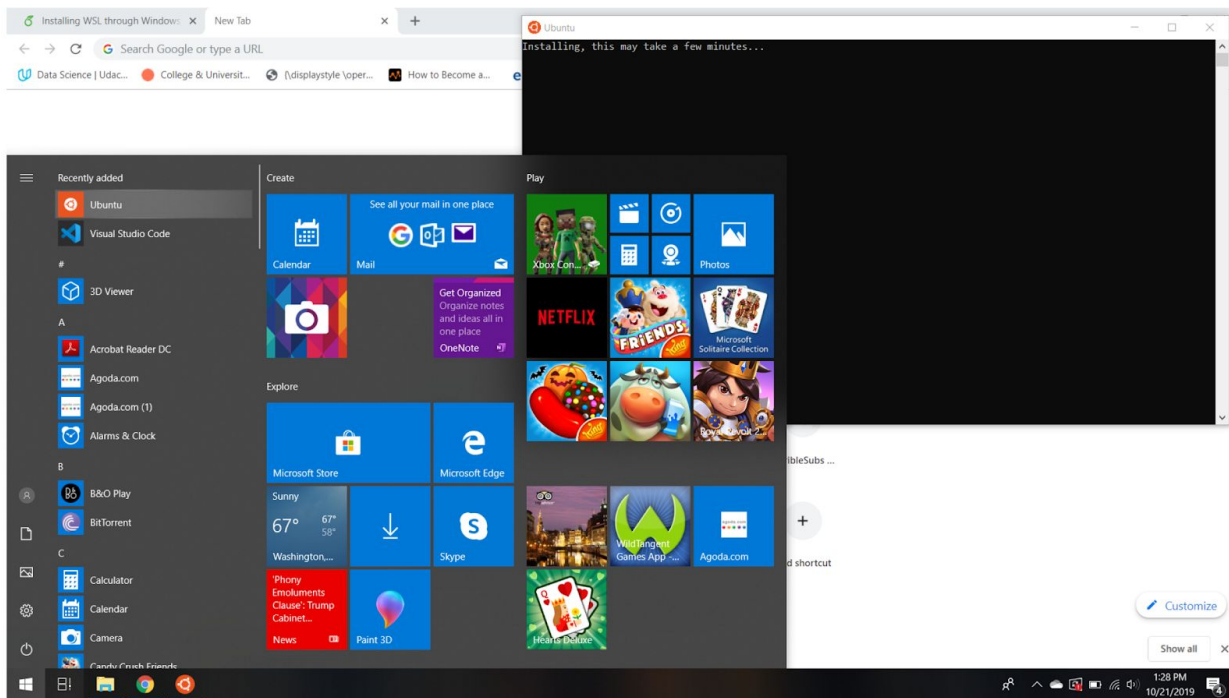
**Step 5**: You may check out Visual Studio Code by launching it.

# Installing Windows Subsystem for Linux (WSL) (part 2)

Let us now install WSL so that we can emulate linux commands and functionality on our Windows Machine

**Step 1**: Open Settings from your Start menu and click on Update and Security.
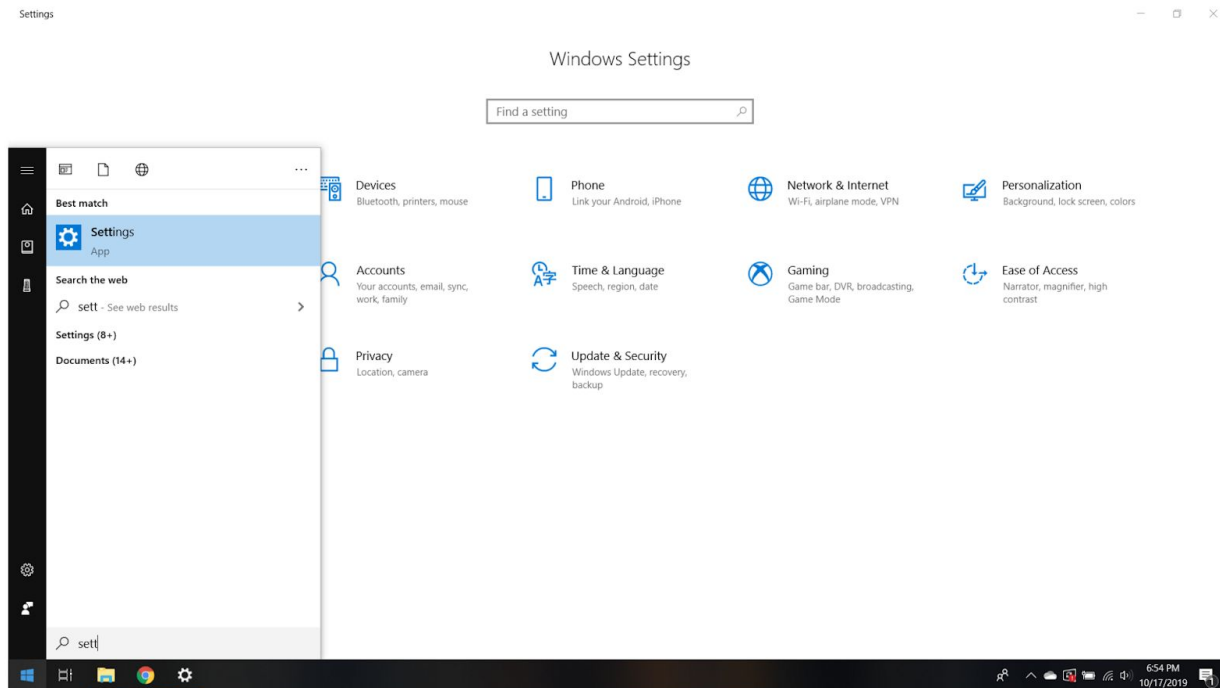
**Step 2**: Click on the For Developers Tab on the left side and toggle the Developer Mode option. Click "Yes" to turn on developer mode.
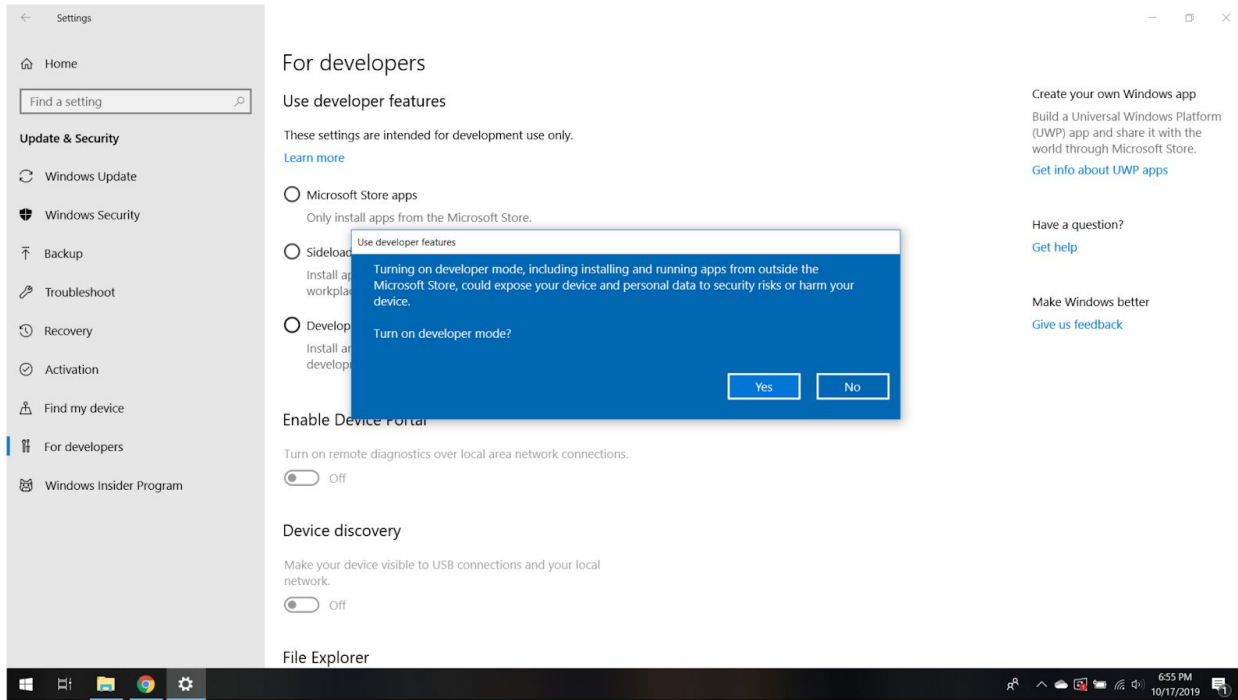


**Step 3**: This will start a brief installation of developer packages.
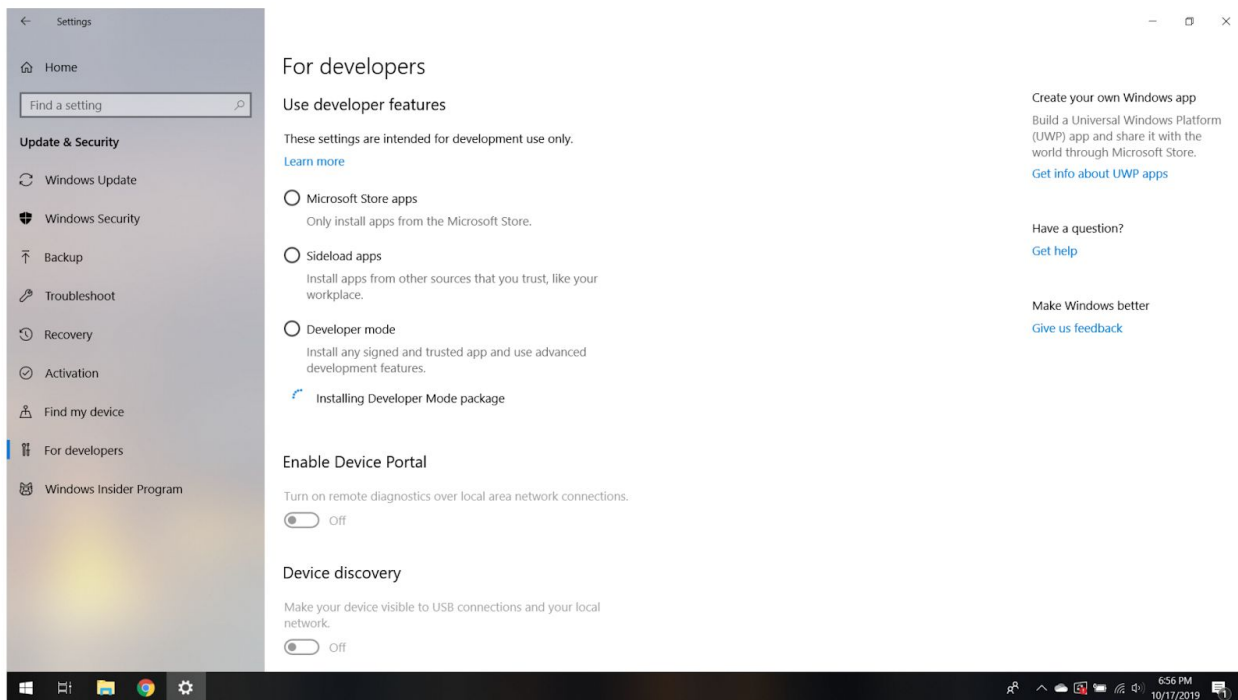
**Step 4**: Click on the Start menu again and search for "Turn Windows features on or off"
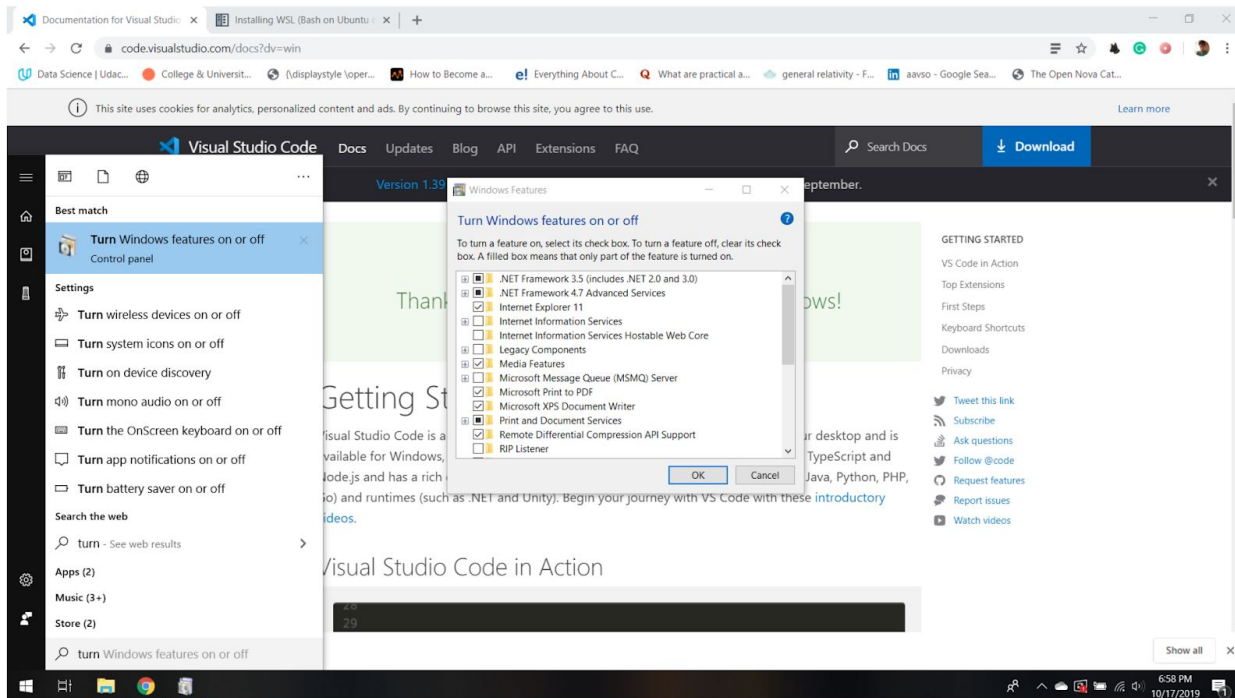


**Step 5**: Scroll down the list in that window and enable Windows Subsystems for Linux. This will trigger a brief loading screen and asks you to restart your computer. Click on "restart now".
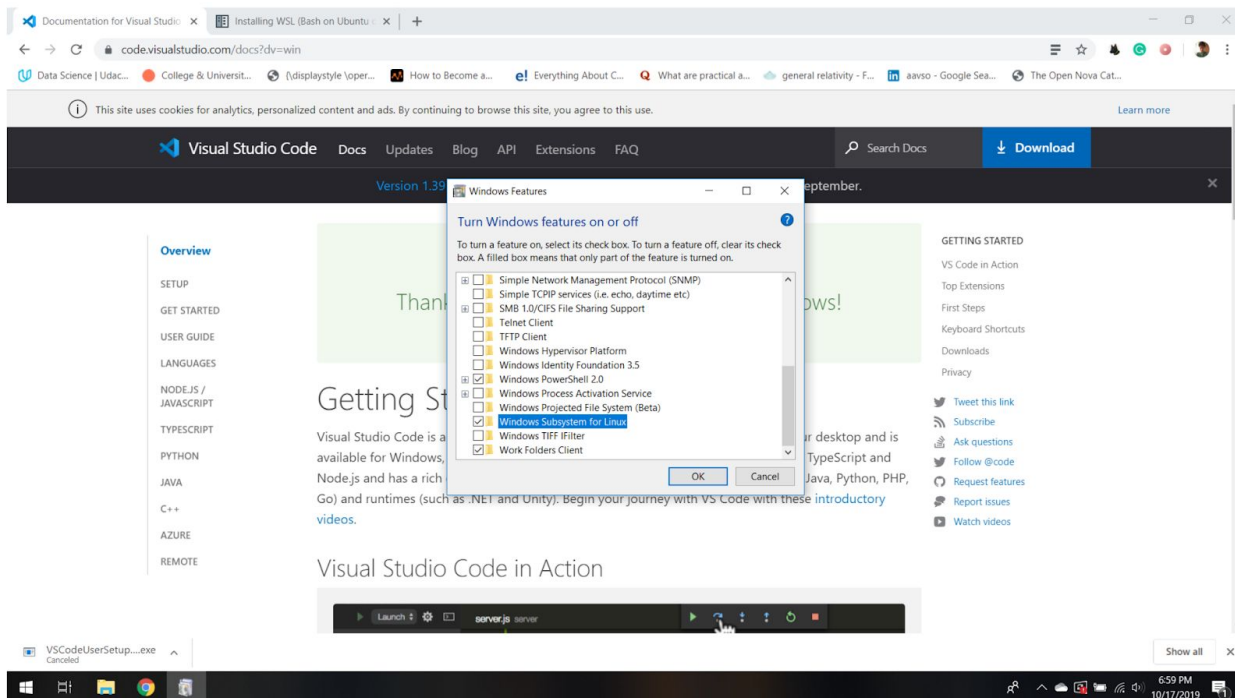
**Step 6**: Search for the Microsoft store and click on it.
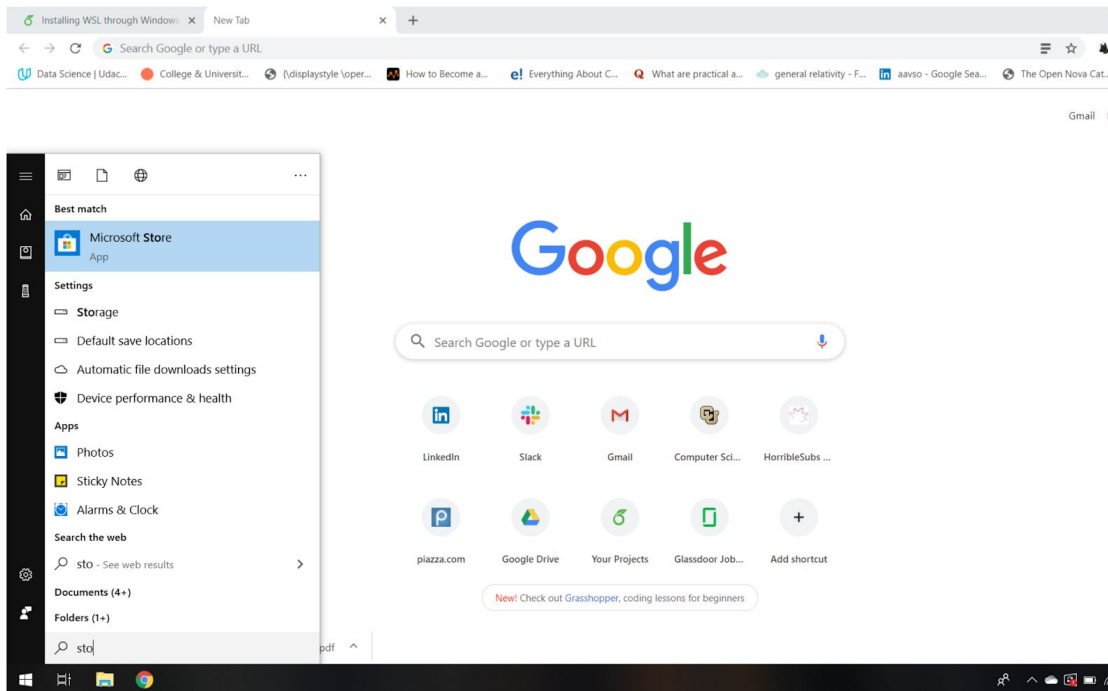


**Step 7**: Search for Ubuntu inside Microsoft Store and Click on Install
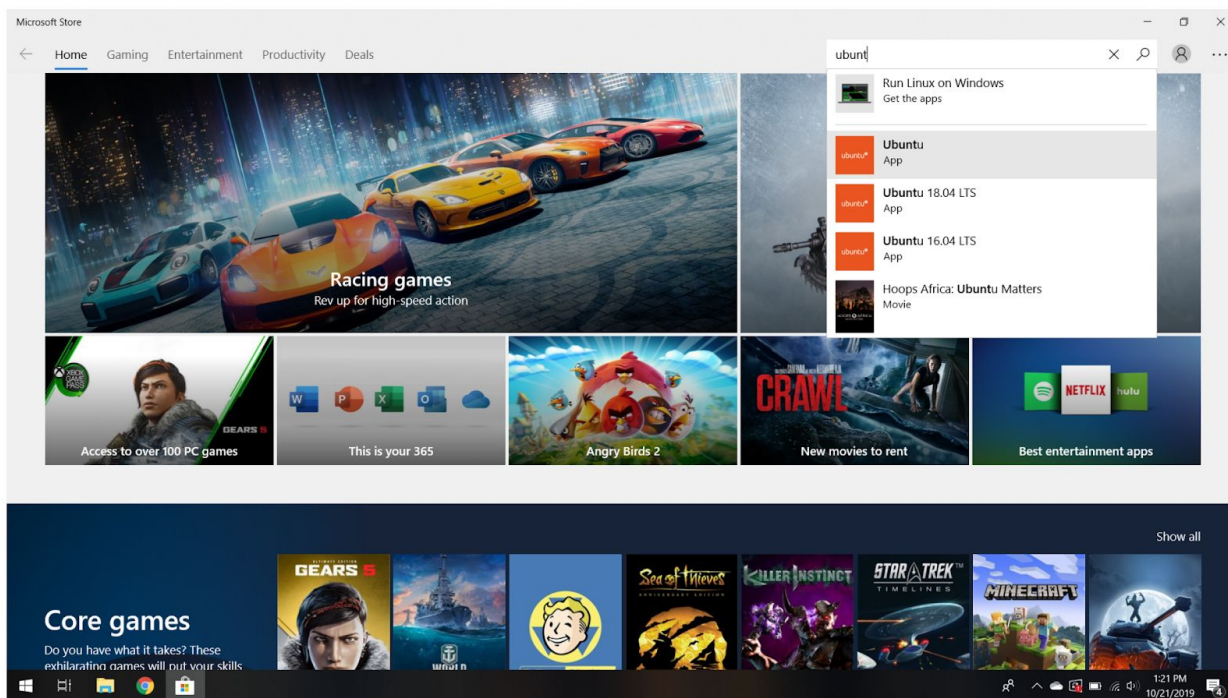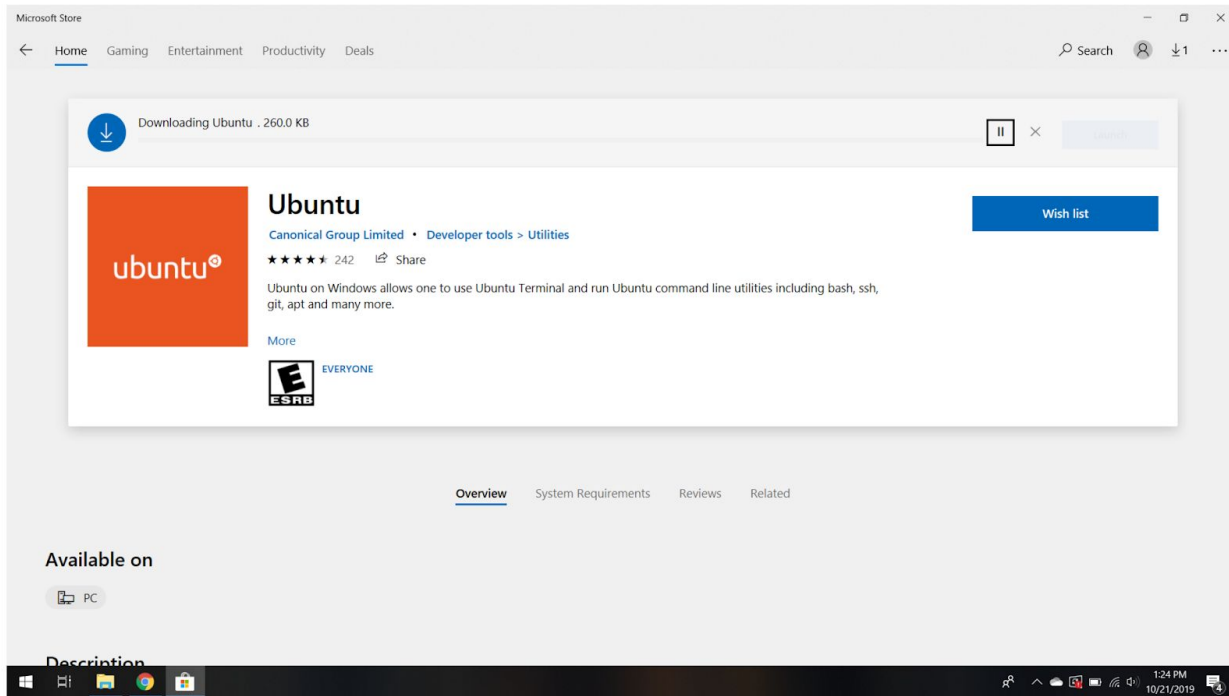
**Step 8**: Open Ubuntu in the start menu and let it finish installing.

**Step 9**: Make sure you give it an appropriate username and password of your choice.
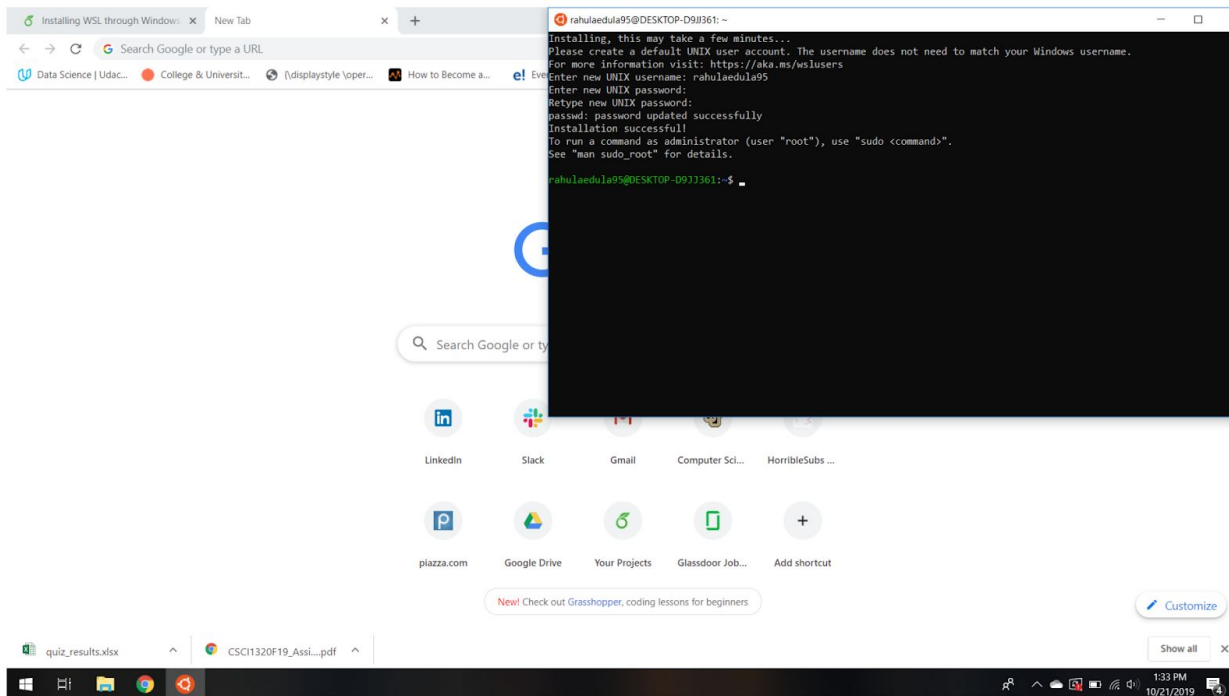
**Step 10**: Now start Visual Studio Code again to add the final touches. Press on ctrl+shift+p to get a drop down and search for Terminal: Select Default Shell. Choose the WSL Shell as your default shell.
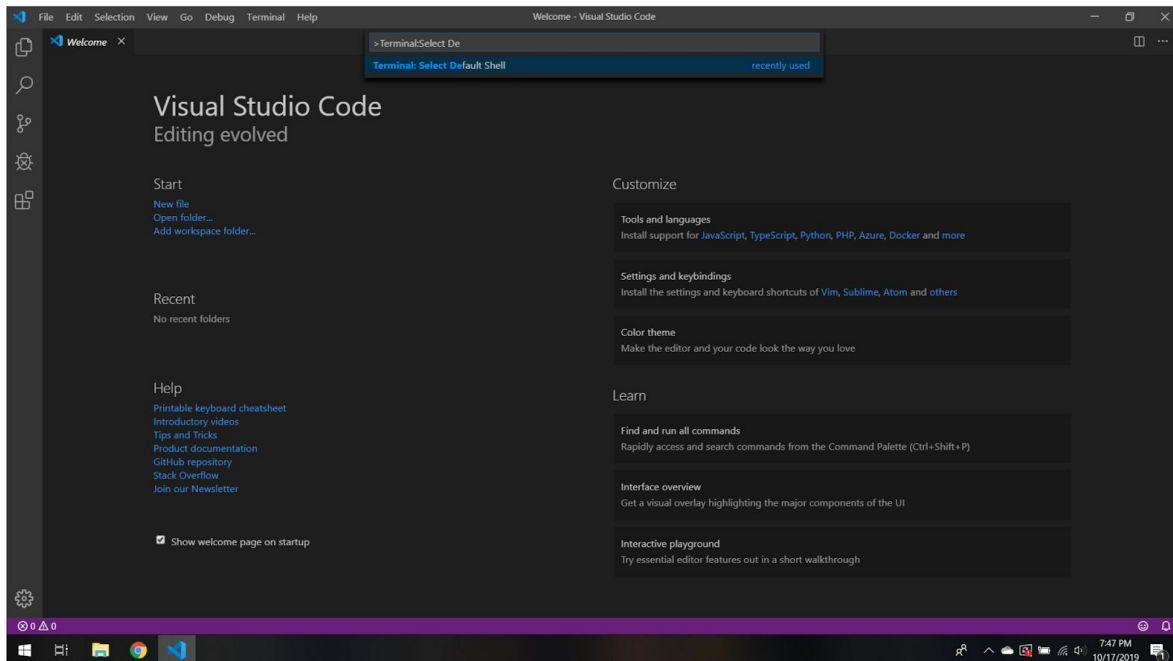


**Step 11**: We can now click on New Terminal from the drop down menu on the top of Visual Studio Code. This will open a new terminal on the bottom of your screen.
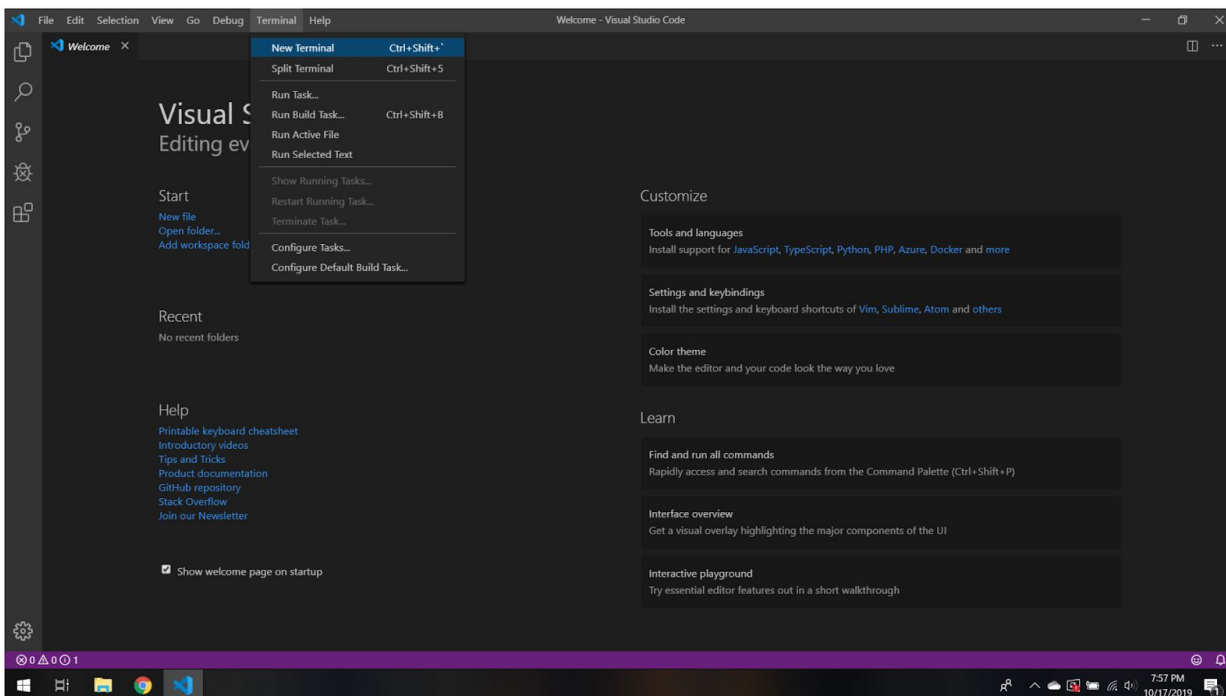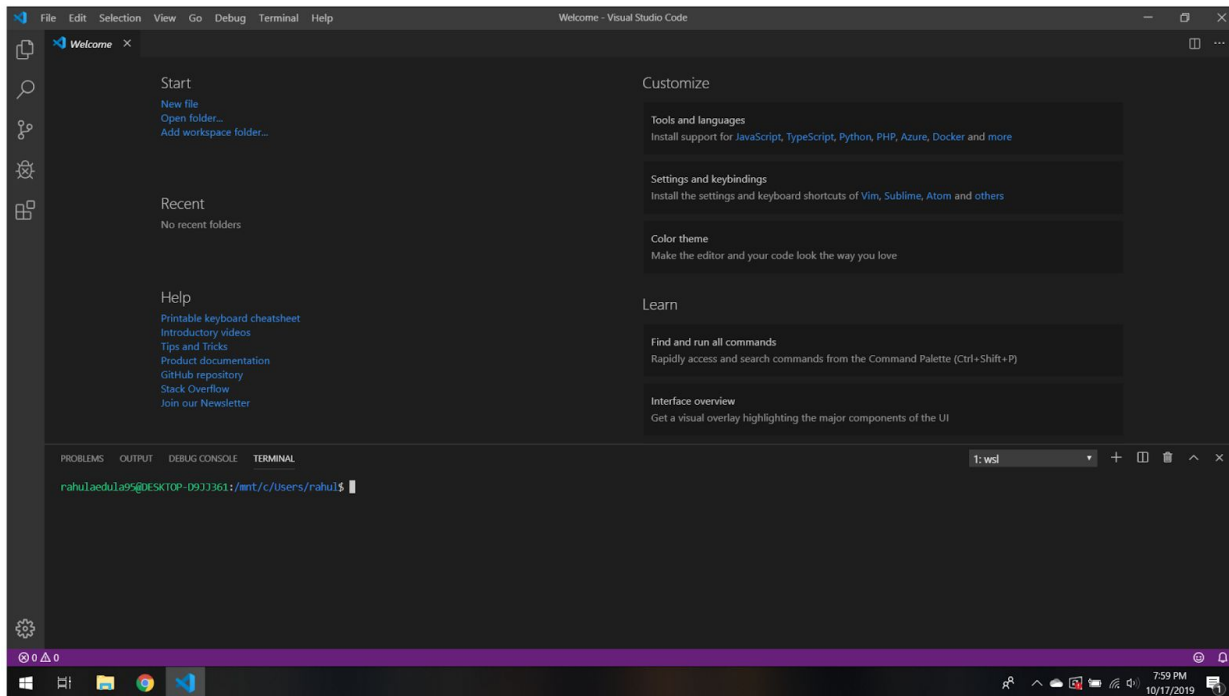
**Step 12**: We need to run an update so that we can use the C++ compiler. In the terminal window, run the command sudo apt-get update and hit Enter. It may take several minutes to complete.
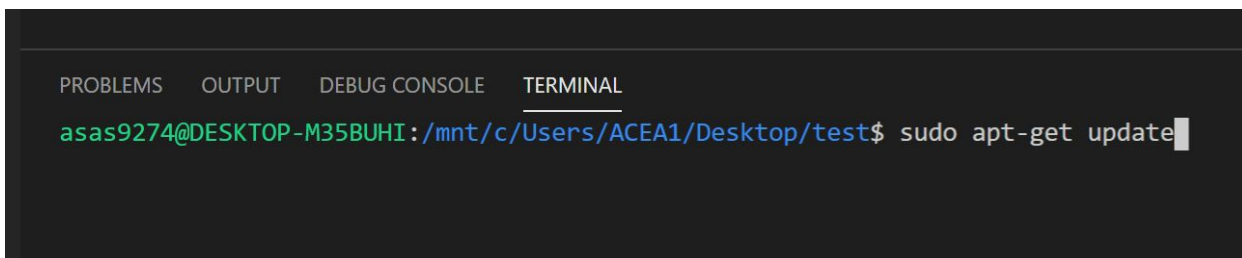


**Step 13**: Install updates to Ubuntu with
"**sudo apt update && sudo apt upgrade**"
Type "y" and hit enter to accept changes.

**Step 14**: We need to install g++.  In the terminal window, run the command
"**sudo apt install gcc && sudo apt install g++**"
and hit Enter. It may take several minutes to complete. Once it finishes, restart VS Code.
Type "y" and hit enter when prompted.
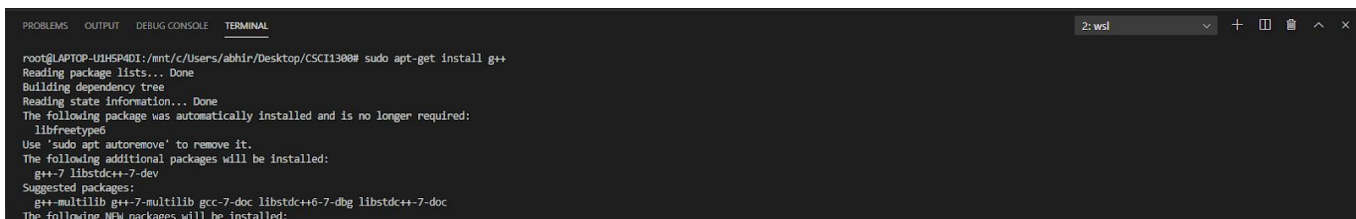


Well done! Go to the section Compiling Code In Terminal (For Both Mac & Windows) (next page) to compile code on your computer.

# Compiling Code In Terminal (For Both Mac & Windows)

When you open VSCode at the bottom of the screen you should see a terminal.
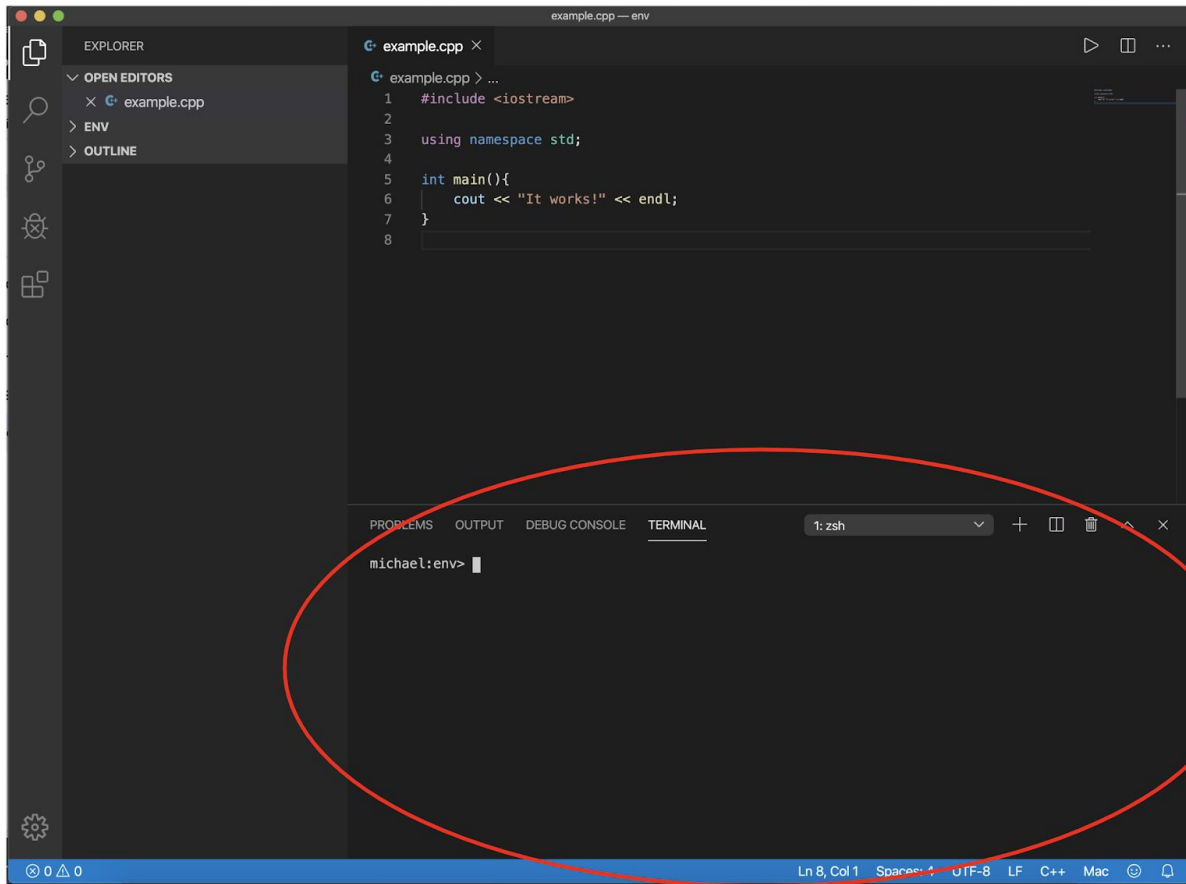


In order to compile and run your code you first need to navigate to where your code is saved through terminal. You do this using the **cd** (change dirrectory) command (see the hmwk0 pdf on moodle if you need an explanation on how to do this). If you have a folder open in VSCode by default the terminal will open to the location of that folder.

For instance on my computer I have a folder I made called **env** open in VSCode so terminal automatically opened to. You can visualize your location using **pwd** (print working directory).
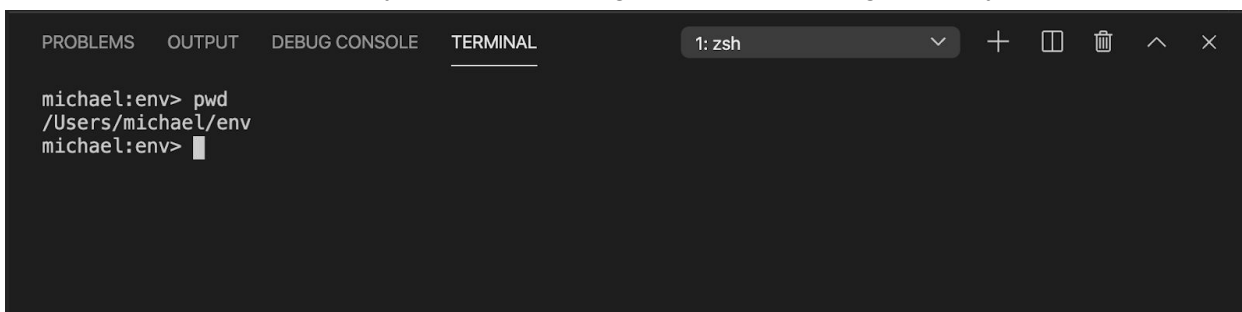
But let's say the code I want to run is located in a subdirectory called **hw0**, I can navigate to hw0 using **cd.**

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                    1: zsh            ∨   +   ⊡   🗑   ∧   ✕

```
michael:env> cd hw0
michael:hw0> ▊
```

Now let's double check the files we are looking for are actual there by using the **ls** (list) command.

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                    1: zsh            ∨   +   ⊡   🗑   ∧   ✕

```
michael:env> cd hw0
michael:hw0> ls
example.cpp
michael:hw0> ▊
```

Now to compile our code we use the program g++ in the following command.

**g++ -std=c++11 example.cpp**

g**++** is the compiler program
**-std=c++11** specifies the version of C++ we want to use
**example.cpp** is the file we want to compile

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                    1: zsh            ∨   +   ⊡   🗑   ∧   ✕

```
michael:hw0> g++ —std=c++11 example.cpp
michael:hw0> ▊
```

This command creates a file named **a.out** which is the compiled version of the code in example.cpp. Now to exceture our code we run a.out by typing the following and hitting enter.

**./a.out**

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                    1: zsh            ∨   +   ⊡   🗑   ∧   ✕

```
michael:hw0> g++ —std=c++11 example.cpp
michael:hw0> ./a.out
It works!
michael:hw0> ▊
```

And that is how you compile and run a C++ program from terminal.

# Compiling Code in the UI (Windows & Linux & Mac)

You do not have to manually compile and using the terminal. Instead you can install the Code Runner extension. In the extensions tab search for Code Runner and click install. Code runner allows you to compile and run code by the click of a single button.

**Step 1**: Search for and install the "Code Runner" Plugin



**Step 2**: Select the editor tab of the source file you wish to compile.

**Step 3**: Compile with the "Play" button and observe output in the terminal.



**Note**: Some students have reported that Code Runner does not work on their systems, or works inconsistently. If this is the case for you, verify that you have set up your compiler correctly as above. If you are still having trouble, then you can use the method below.
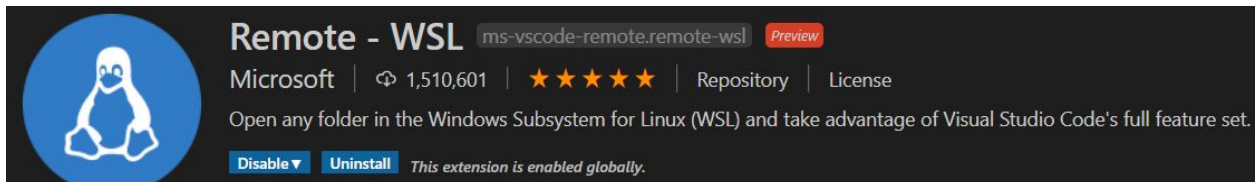
# Compiling Code with a Build Script (For Linux & Windows)

These steps assume that you have completed the VS Code installation steps on Moodle.
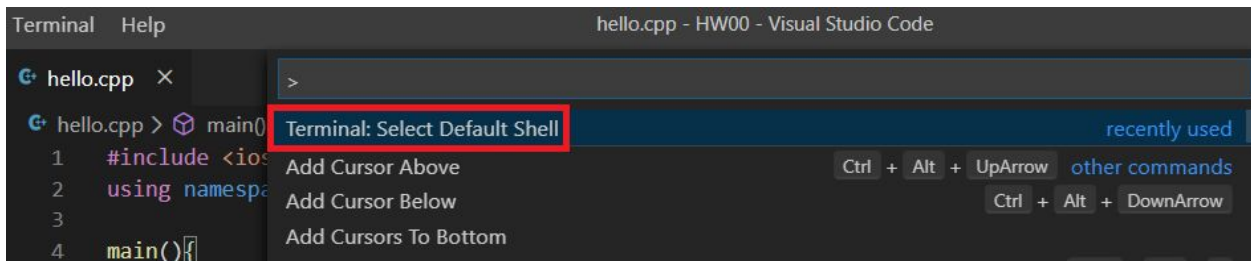
## *Windows*

**Set up:**

1. In VS Code: Make sure the "Remote - WSL" plugin is installed.



2. [ Ctrl ] + [ Shift ] + [ P ]: To open the settings chooser.  Search for "Terminal" if "Terminal: Select Default Shell" is not already at the top.  And select this option



3. Choose "WSL Bash" as the default terminal.

4. Close any open VS Code Windows.

5. In File Explorer, navigate to the directory where your code is.

6. Open a WSL terminal by typing "bash" in the address bar and pressing [ Enter ].
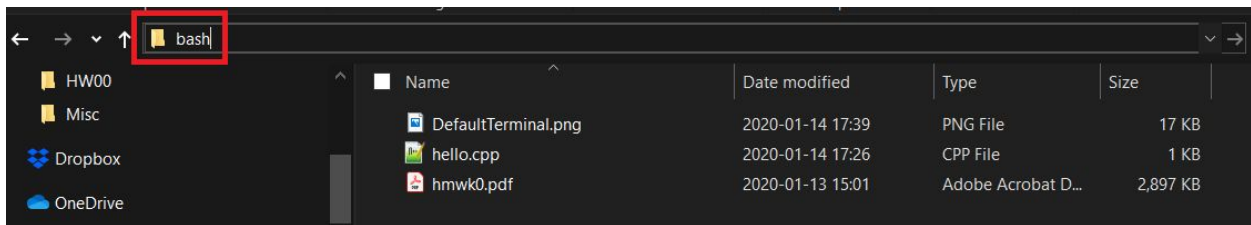


7. Type "`code .`" in the new terminal and press [ Enter ]. VS Code will initiate a one-time download of packages for interfacing with WSL before starting VS Code.
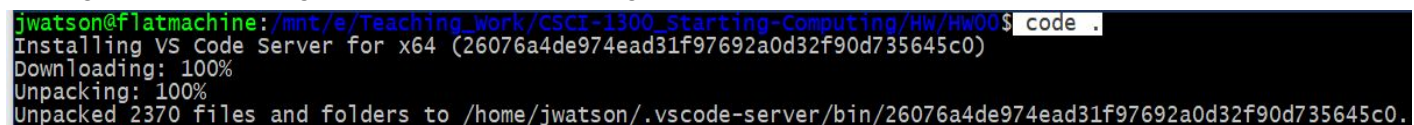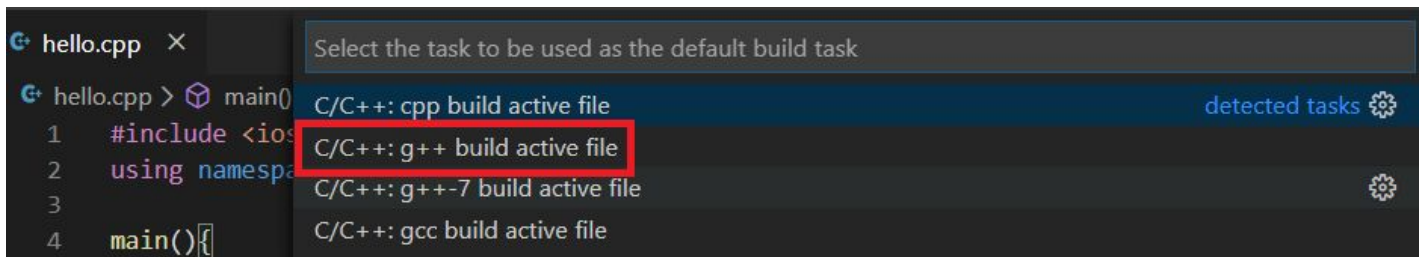
8. Make sure that the "C/C++" plugin has been installed in the local WSL instance of VS Code.
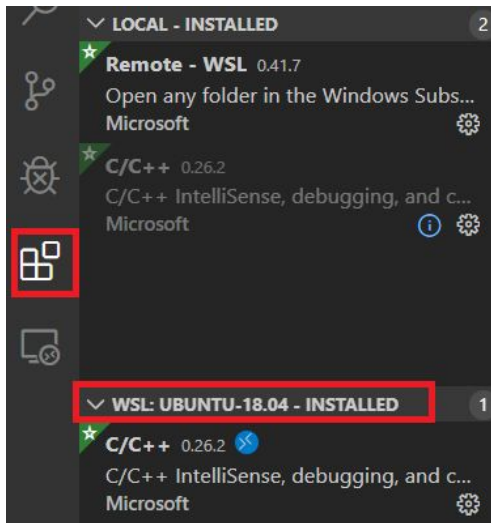


**Per project:**

9. Select "Terminal" → "Configure Default Build Task".  Select the "g++ build active" option. A JSON file will be created.  There is no need to modify the JSON file now.



**Compilation:**

10. Select the editor tab for your source file.

11. Select "Terminal" → "Run Build Task" -or- press [ Ctrl ] + [ Shift ] + [ B ].  If there are no errors, you can press any key to close the terminal.  The default build task creates an executable program with the same name as your source file, but without a file extension.

12. Select "Terminal" → "New Terminal", then type "`./<programName>`" and press ; you should see the program output in the terminal.

## _Linux_

**Set up:**

1.  Make sure that the "C/C++" plugin has been installed in the local WSL instance of VS Code.



**Per project:**

2.  Select "Terminal" → "Configure Default Build Task". Select the "g++ build active" option. A JSON file will be created. There is no need to modify the JSON file now.
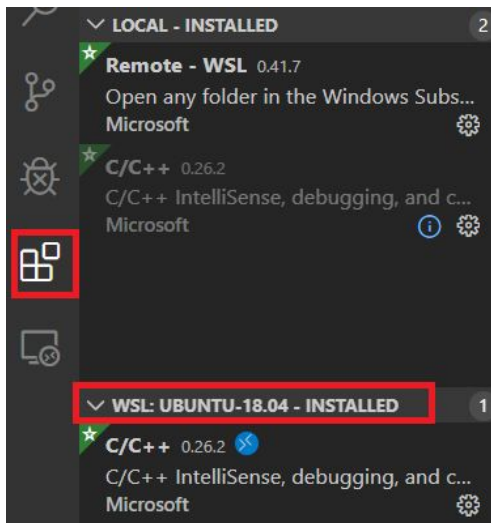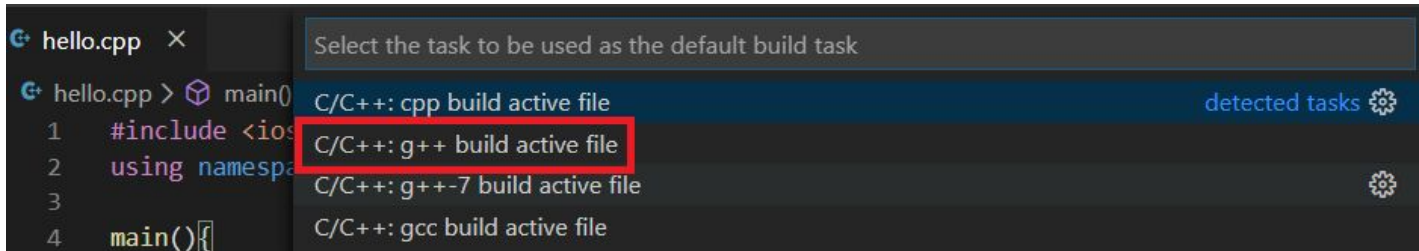


**Compilation:**

3.  Select the editor tab for your source file.

4.  Select "Terminal" → "Run Build Task" -or- press [ Ctrl ] + [ Shift ] + [ B ]. If there are no errors, you can press any key to close the terminal. The default build task creates an executable program with the same name as your source file, but without a file extension.

5.  Select "Terminal" → "New Terminal", then type ".`/<programName>`" and press ; you should see the program output in the terminal.