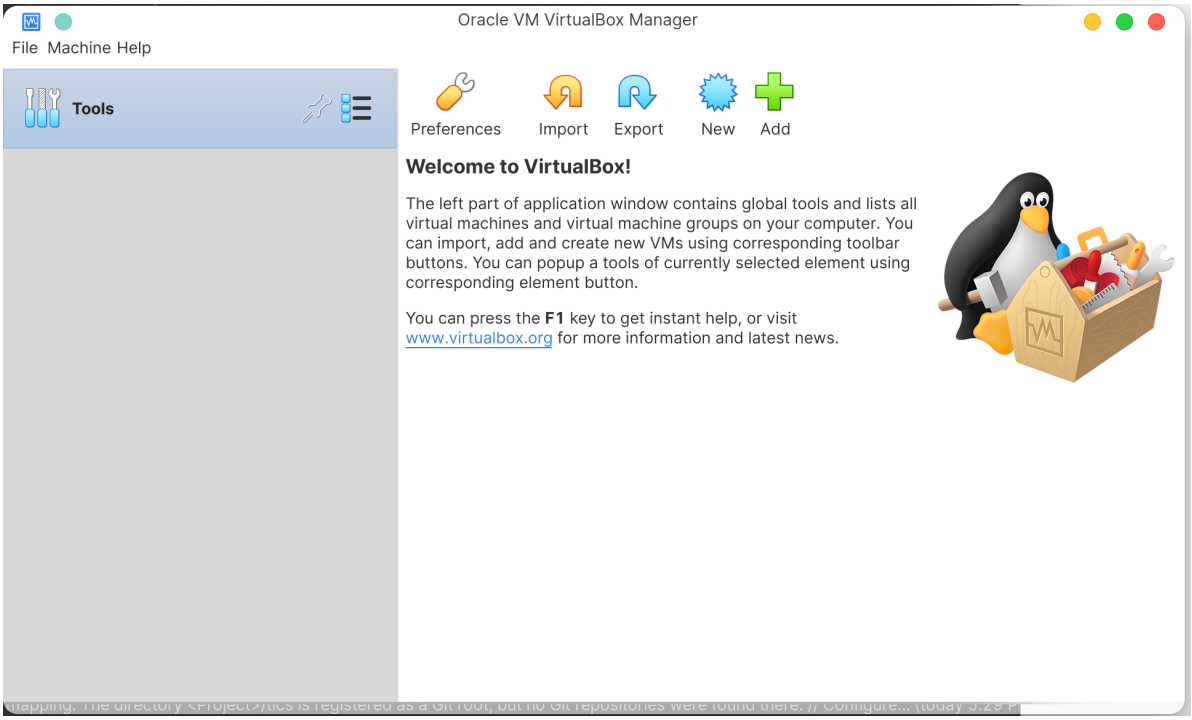
VM and Toolchain

Virtual Machine

Import the VM

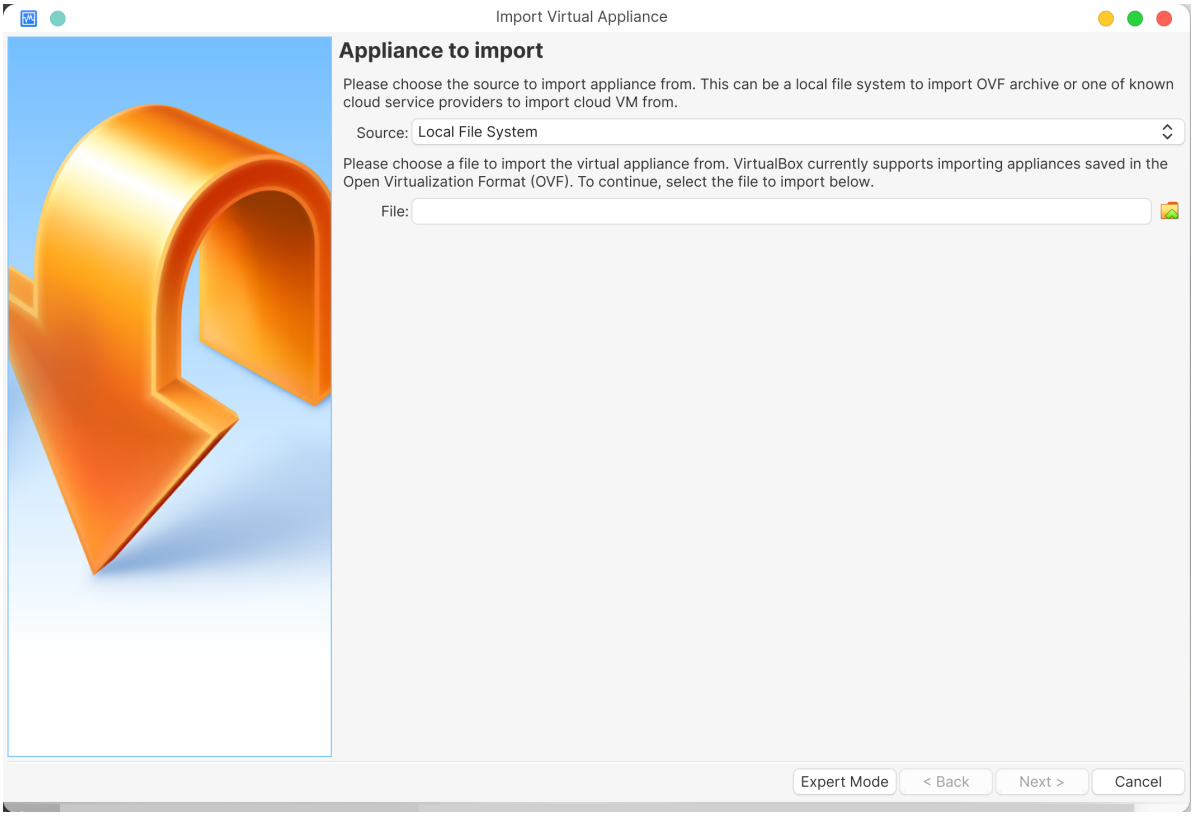
You will need to install VirtualBox 6.1 to run the virtual machine, which is available at https://www.virtualbox.org/wiki/Downloads.

> **Notice**
>
> our virtual machine image is in standard OCI format; which can also be recognized by VMWare. If you are more keen to VMWare's products, you can have a try on yourself, but VM extensions will not be ready out of box.

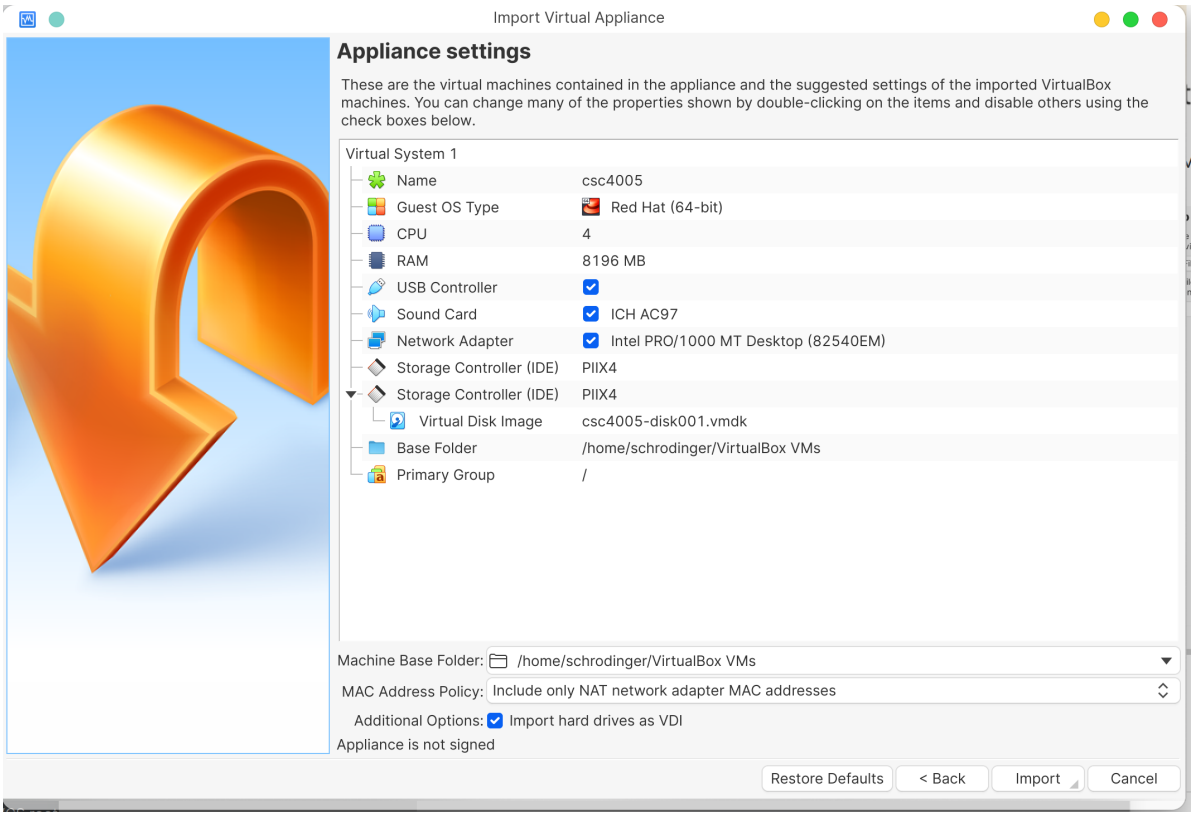


To begin with, click 

Then choose the downloaded OVA file in the prompted window:



Click `Next` and then you can adjust the parameters in the following step:



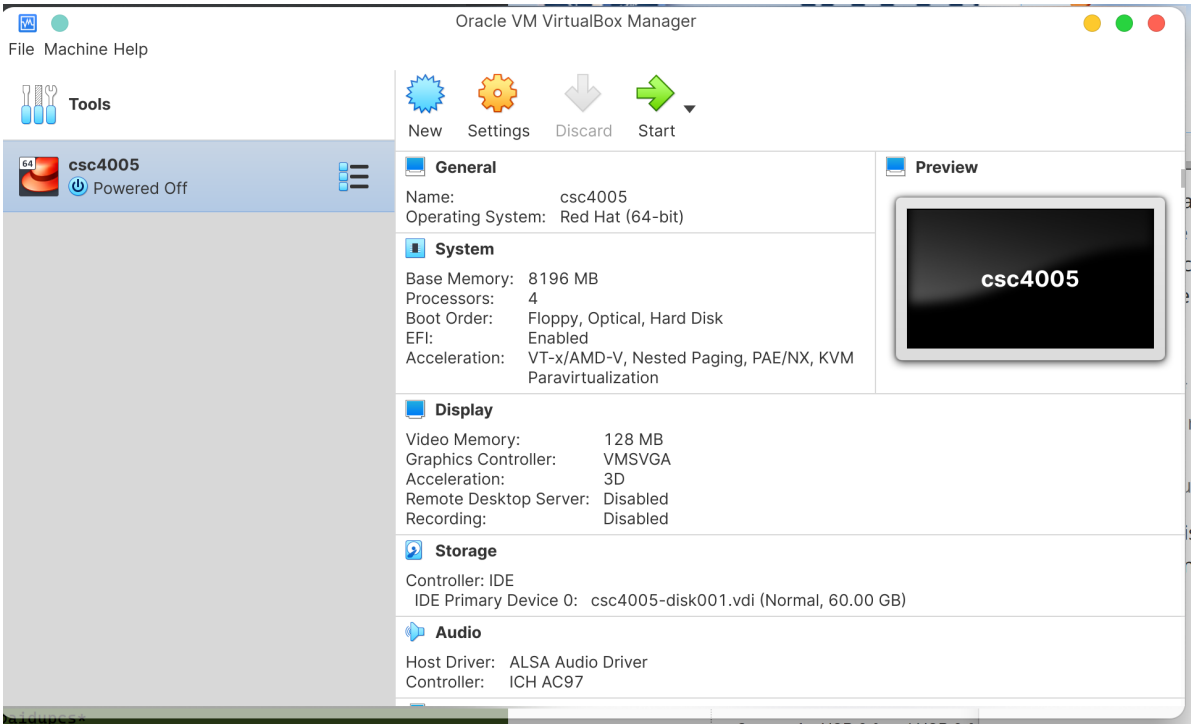
To test your programs locally and speed up the build of your programs in the VM in the future, you are **recommended** to create an instance with 4 CPU cores and 8 GiB RAM. But this configuration **may not** be available depending on you host machine. If it is the case, you can tweak the parameters to an acceptable range.

> **Notice**

> It is okay even if you created the VM with very limited resource. The final place to run benchmarks of your programs is on a public server. It is just that you may suffer slow build or you may not exam or debug your program on your own computer under a parallel environment.

After you get a satisfactory configuration, click `import`. The image is quite large, it may take several minutes before the importing progress finishes.

If everything goes smoothly, you should be available to see the following window after import:



> **Notice**
>
> VT-x/AMD-V is an important acceleration for virtual machines, but most motherboards disable the feature on default. If you suffer from a poor performance, please consider enabling the feature (For AMD CPUs, it is sometimes called SVM; or if your motherboard has Chinese support, it may show something like `处理器虚拟化`). To do so, you may need to enter the BIOS menu which can be **dangerous**. Please make sure you understand the meaning of each switch before you make any adjustment.

Basic Setup

Click `Start` and then you can enter the virtual machine.

> **Notice:**
>
> Sometimes (e.g. the VM gets stuck) you may find that your mouse and keyboard is captured by the VM but you really want to quit it. In such cases, `Right Ctrl` can be helpful. (Sadly, I do not have a mac so I don't know the corresponding key for Apple users, so you on your own. LoL. but the tip should be normally shown on the lower right corner)
>
> 
>

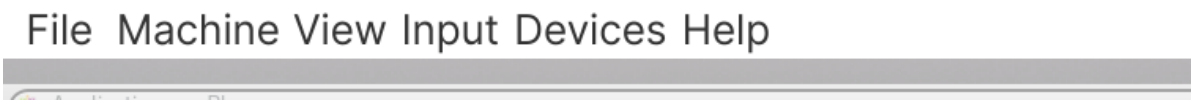
The password and username are both `csc4005`.

Once the VM is successfully booted, you will see:

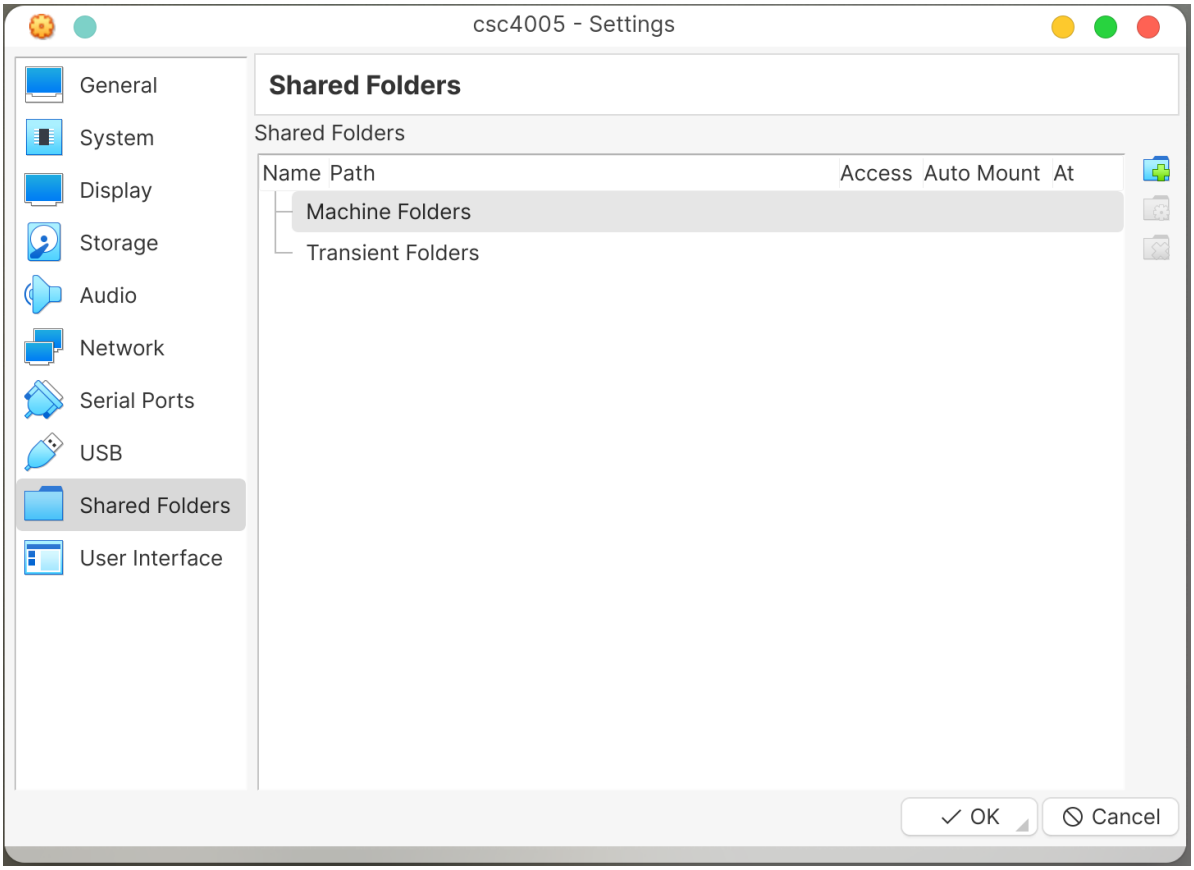
Based on your personal need and preference, you can do the following customization:

Create a Shared Folder

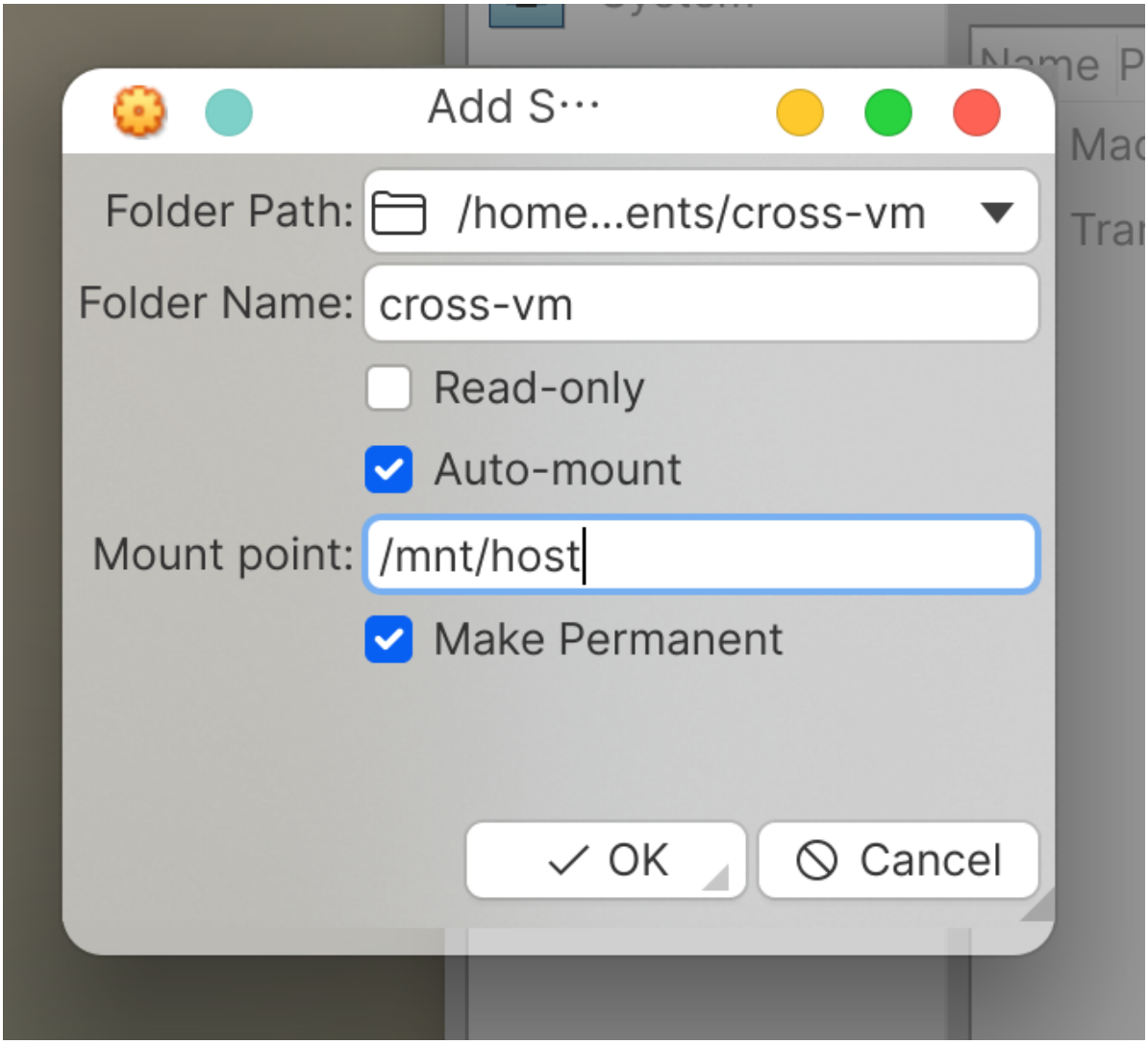
You can create a shared folder if you want to make it more easier to transfer files between the VM and the host.



Click `Devices` and then choose `Shared Folers` in the drop-down menu,



In the prompt dialog, click the button  to add the folder.



It is recommended to choose `Auto-mount` and set the Mount point to something like `/mnt/XXX`.

> **Notice**:
>
> Choose `Make Permanent` to keep the folder on each startup of the VM; or the settings would be cleared once the VM is shutdown.

After that, you should be able to transfer the files through `/mnt/XXX`.

> **Notice:**
>
> If you are experiencing the following problem:
>
> ```
> [csc4005@10 ~]$ cd /mnt/host/
> bash: cd: /mnt/host/: Permission denied
> [csc4005@10 ~]$
> ```
>
> please open an terminal and then type
>
> `sudo usermod -aG csc4005 vboxsf`
>
> **DO NOT FORGET** `-a` **FLAG**, otherwise you will trim yourself from all other groups.
>
> After that, logout the account and login again (or you can directly reboot); the problem should disappear.

Setup SSH

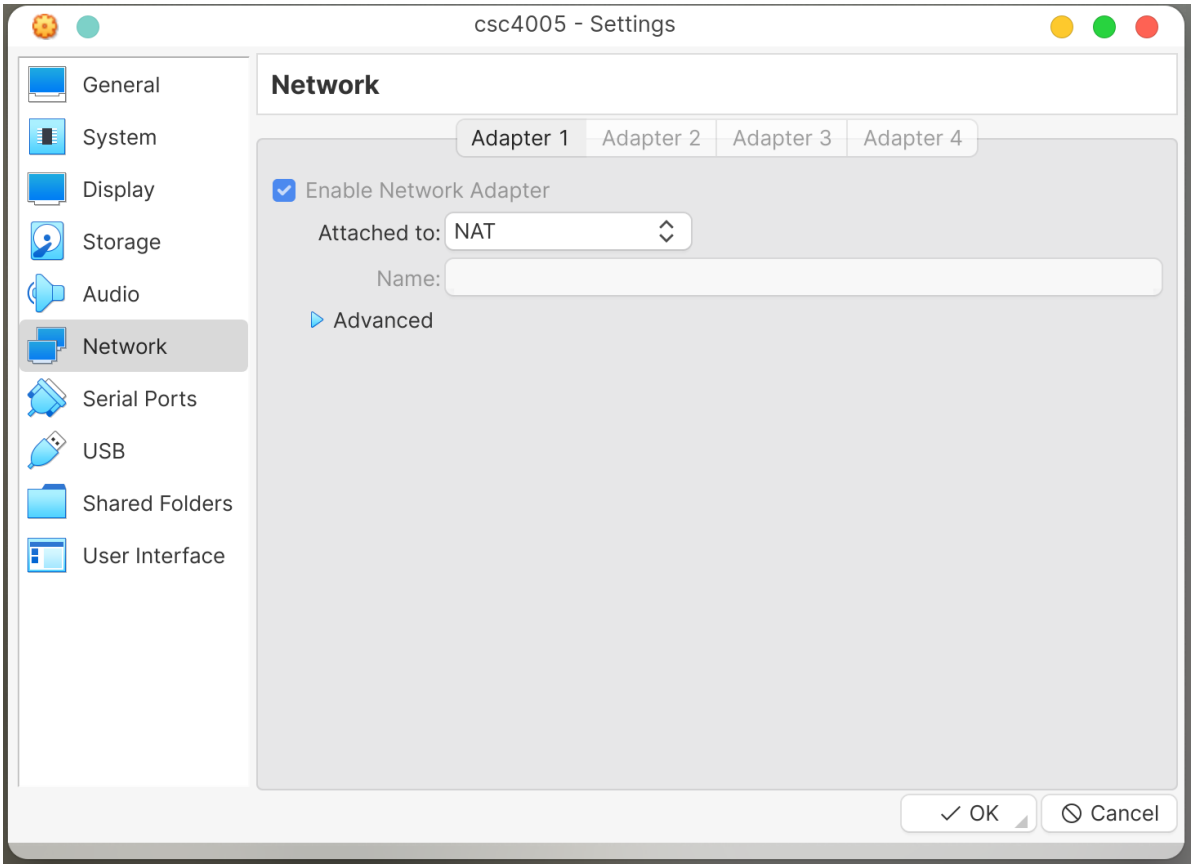
You may feel it annoying to operate in the VM; by setting up SSH, you can have an extra access via SSH tunnels.

Open the terminal and type

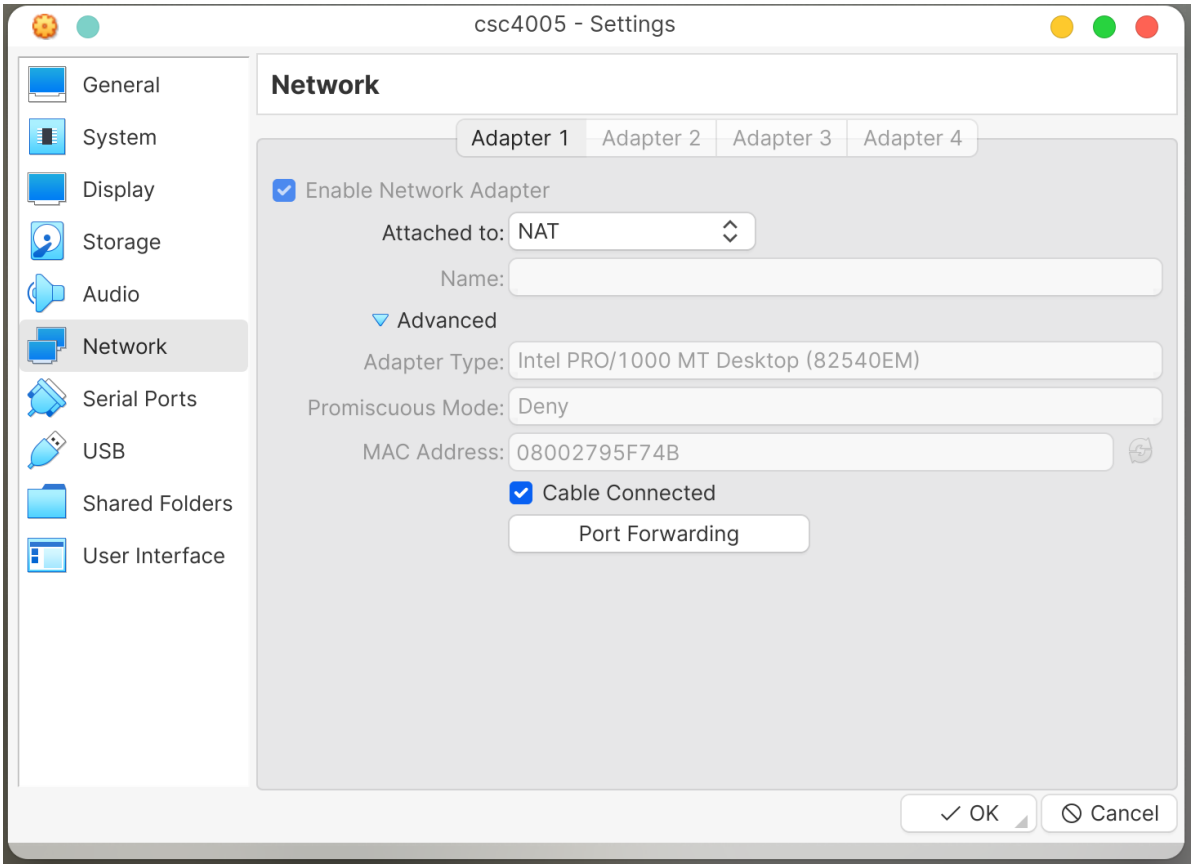
```
systemctl enable sshd --now
```

to make sure the SSH server is running.

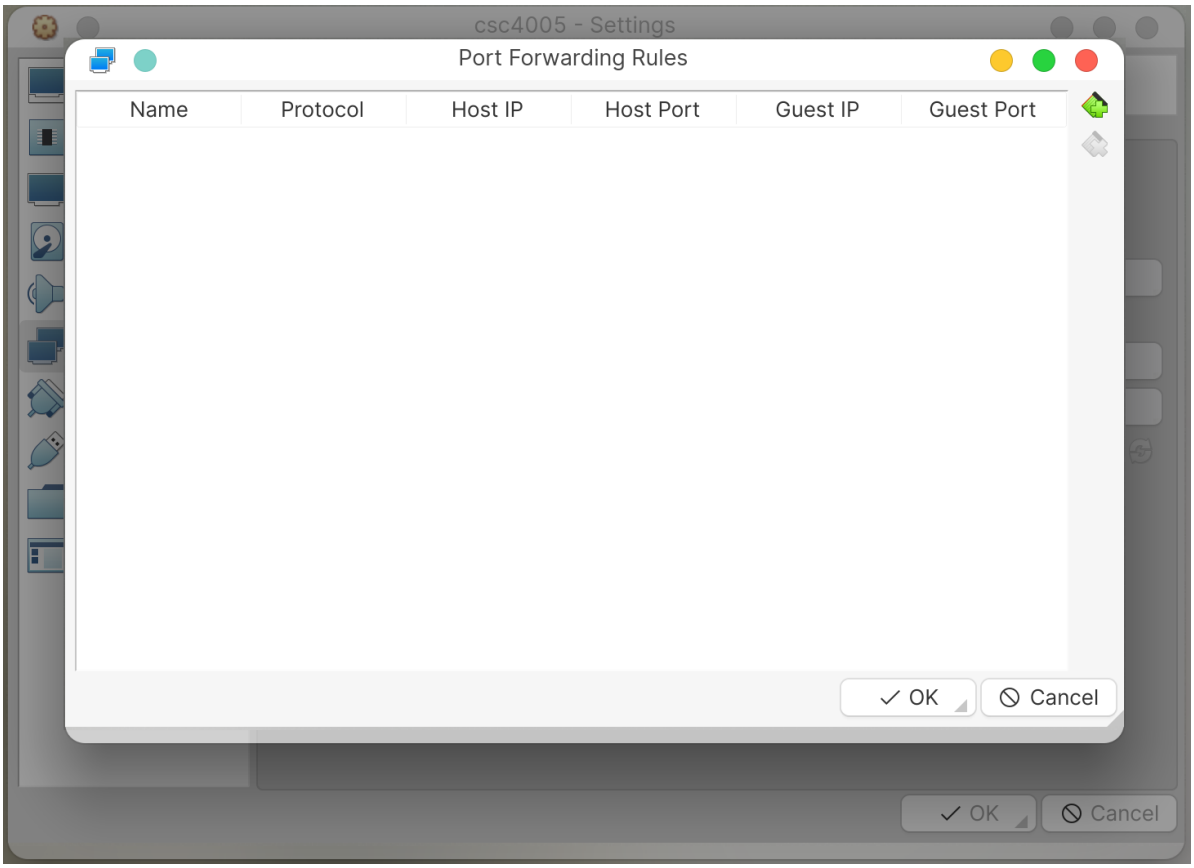
Again, click `Devices`. Then navigate to `Network > Network Settings`.



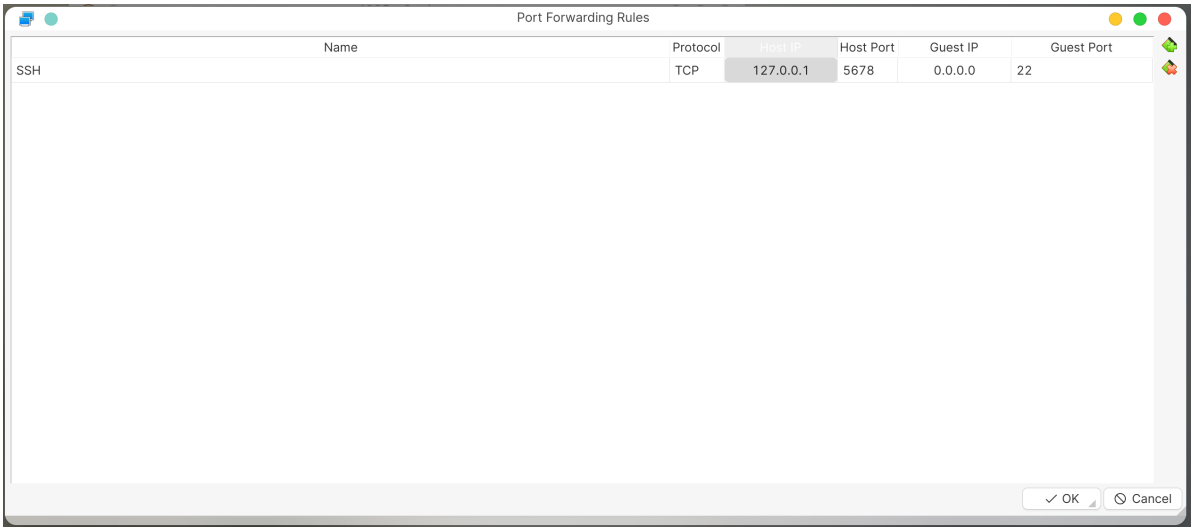
Expand `Advanced`:



Then click `Port Forwarding`



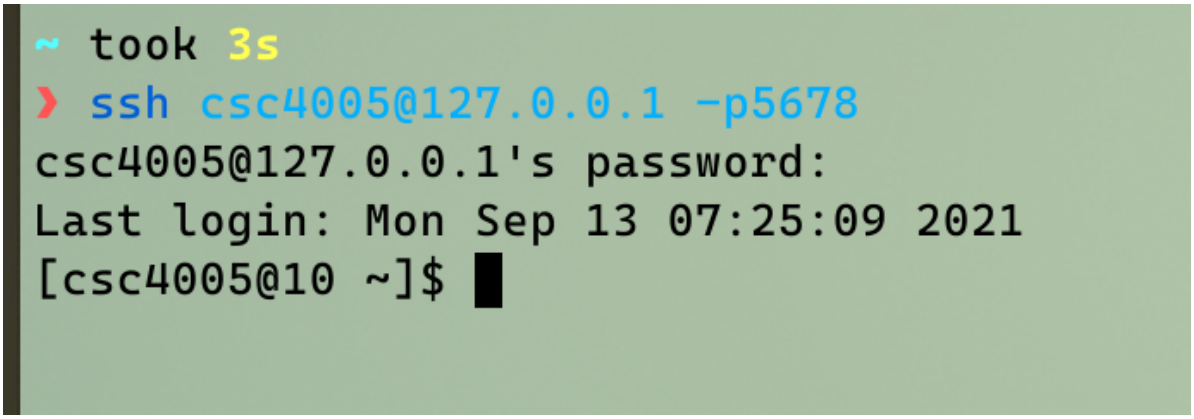
Click 



Fill in a record as it is shown in the image.

> **Notice:**
>
> Make sure that on your host, there is no other program listening on port 5678; otherwise, you need to set to another port here.

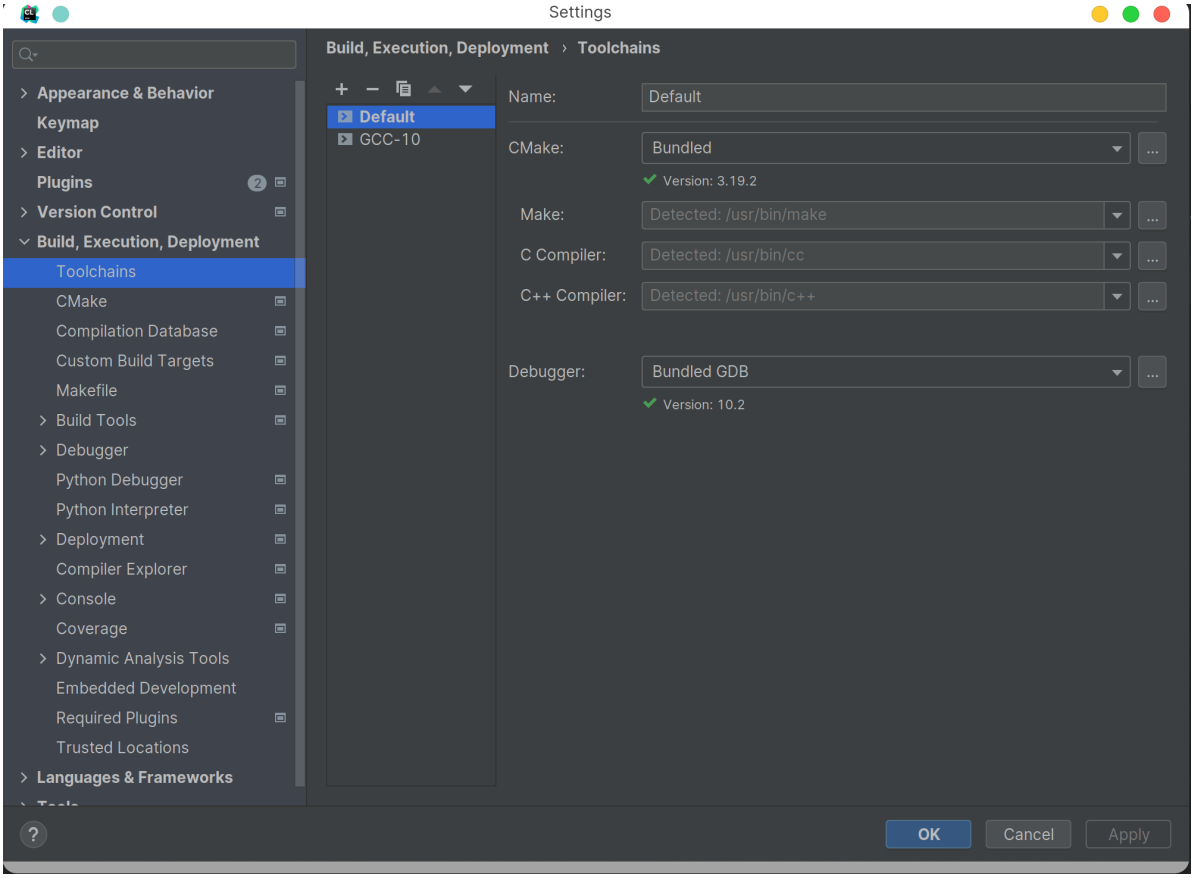
Outside the VM, in any terminal you like, you can now access to the VM in the following way:



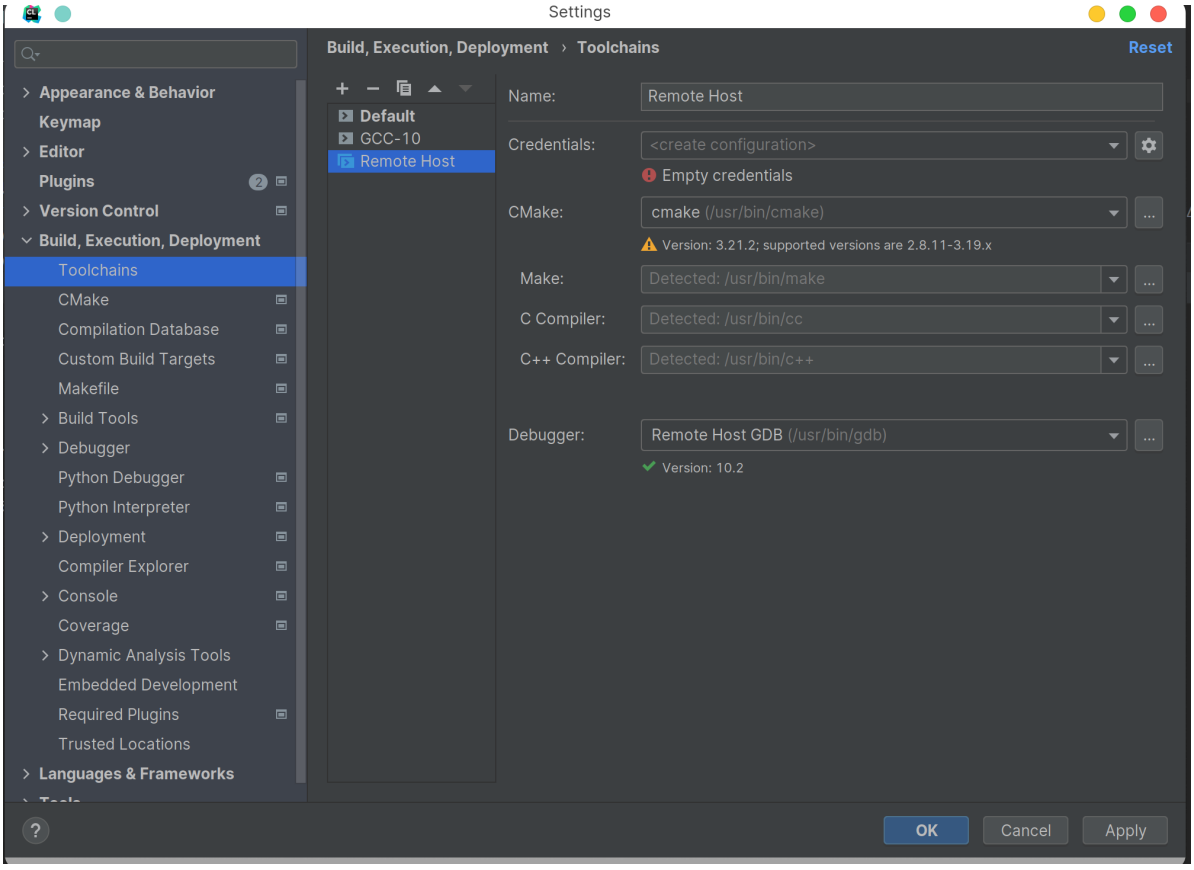
Setup CLion to Use VM Directly

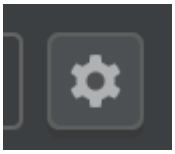
> **Notice:**
>
> To setup CLion, please make sure that SSH is properly set up.

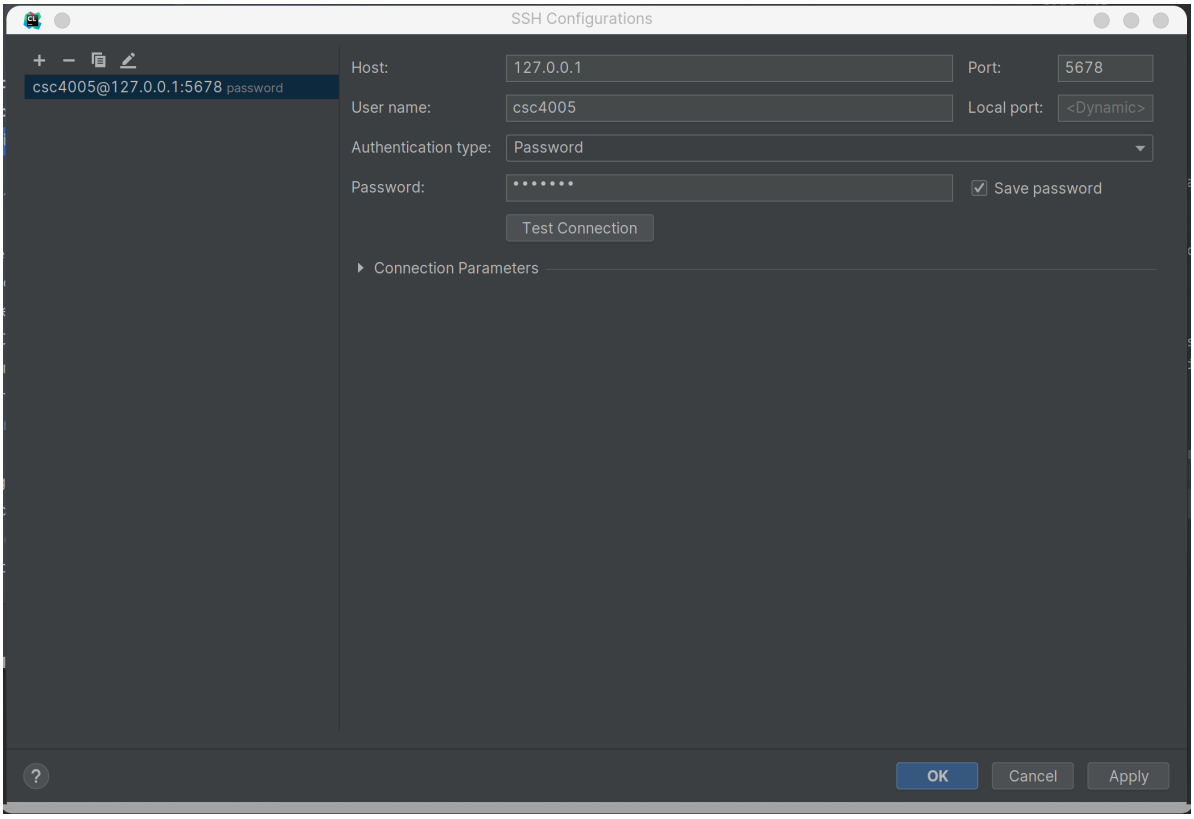
Navigate to `File>Settings` in CLion, then switch to tag `Build,Execution,Deployment>Toolchains`



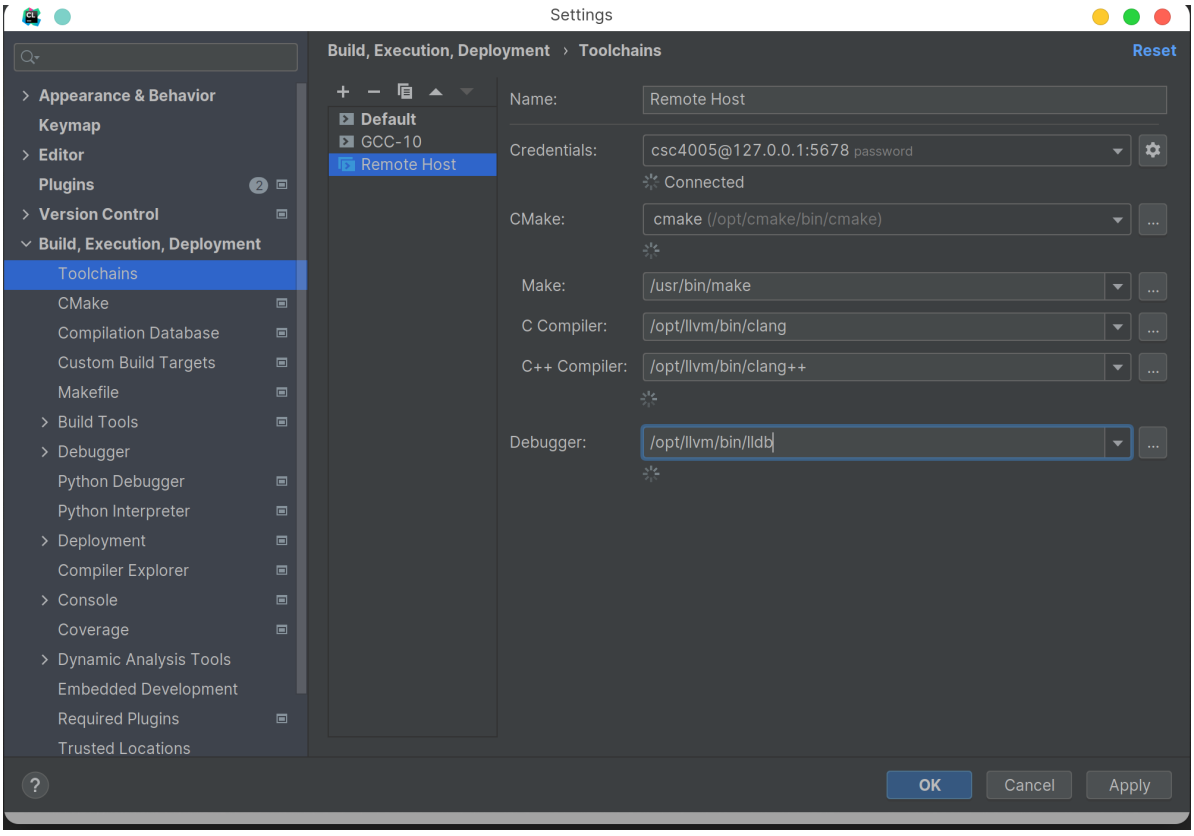
Click that  button, and choose `Remote Host`



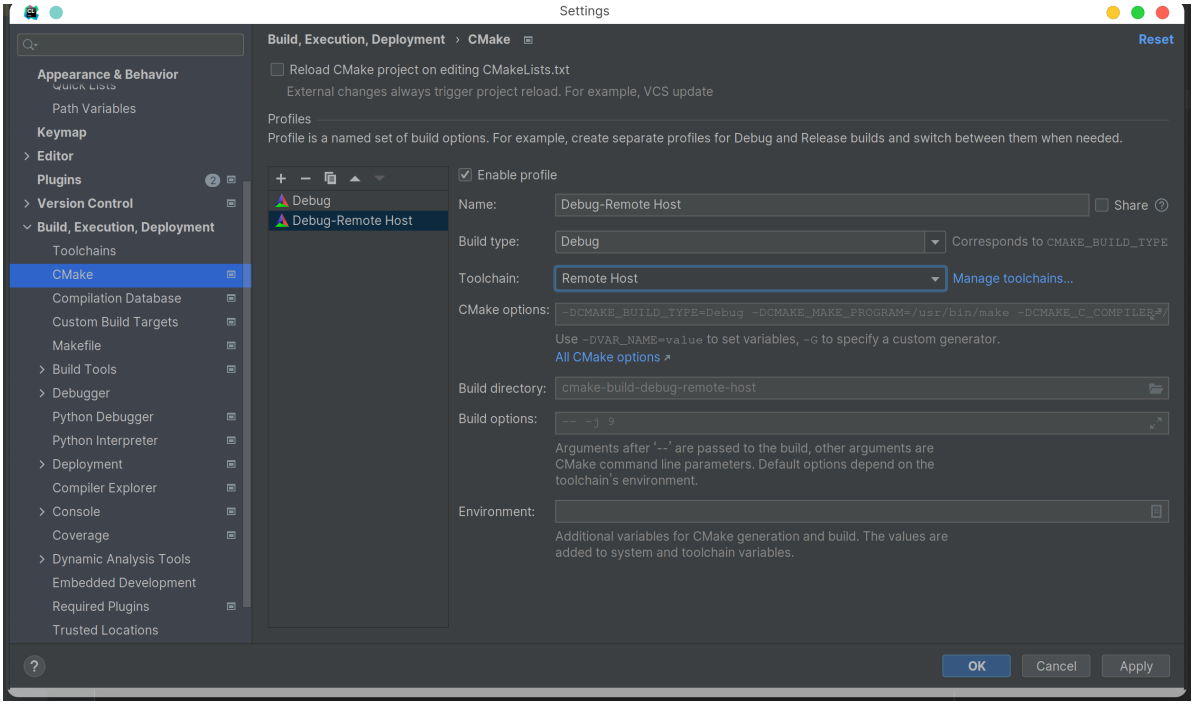
Click  and then add your credential as the following:



Then return to the previous window, and setup as the following:

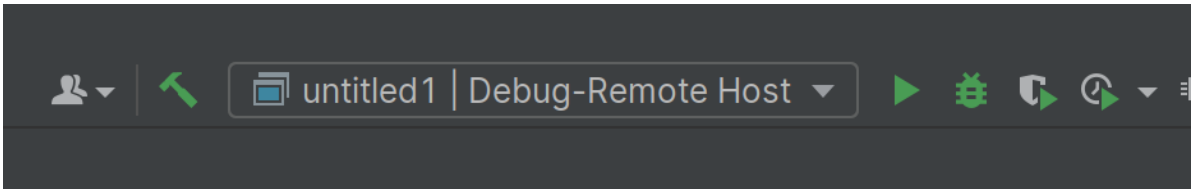


Then navigate to `Build,Execution,Deployment>CMake` and add a profile as the following

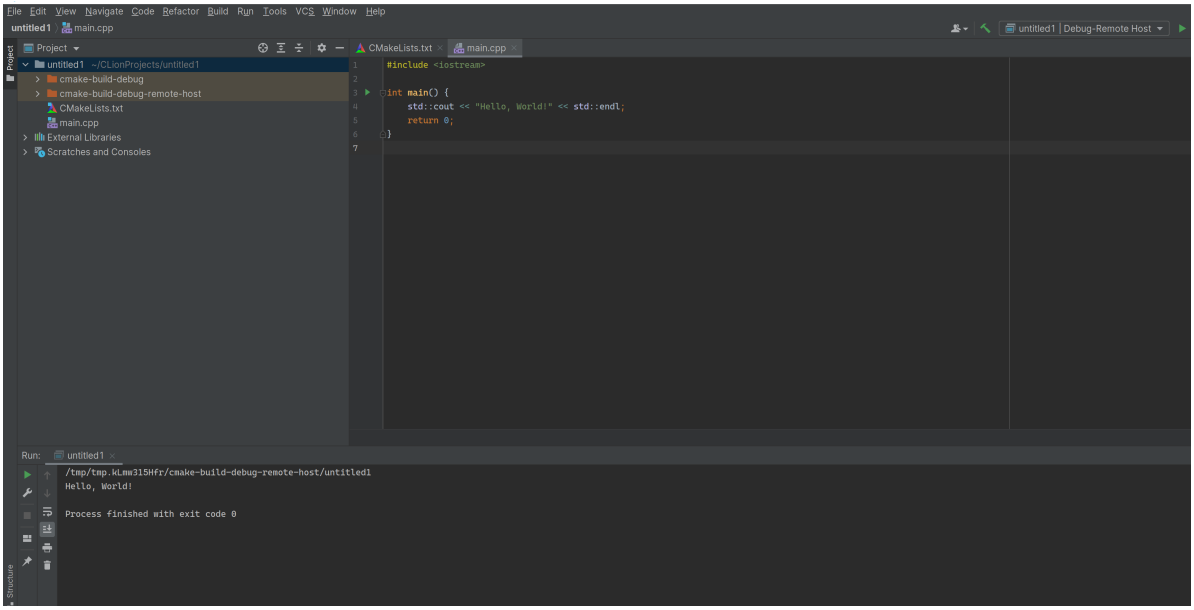


> **Notice:**
>
> You may also want to add other ones with `build type` Release and Relwithdebinfo; these ones will be compiled with high optimization; so that you can analysis the real performance of your programs.

Open a new project and switch to Remote Host toolchain using the drop-down menu on the upper right corner:



Then you can build and see the outputs as normal:



> **Notice:**
>
> You clion may not support `lldb 12`. To work around the debugging issue, you can install gdb on your VM via:

> ```
> sudo yum install -y devtoolset-10-gdb
> ```
>
> Then change the debugger to
>
> ```
> /opt/rh/devtoolset-10/root/usr/bin/gdb
> ```
>
> **It is not recommended to use the default gdb on the OS. It is too old.**

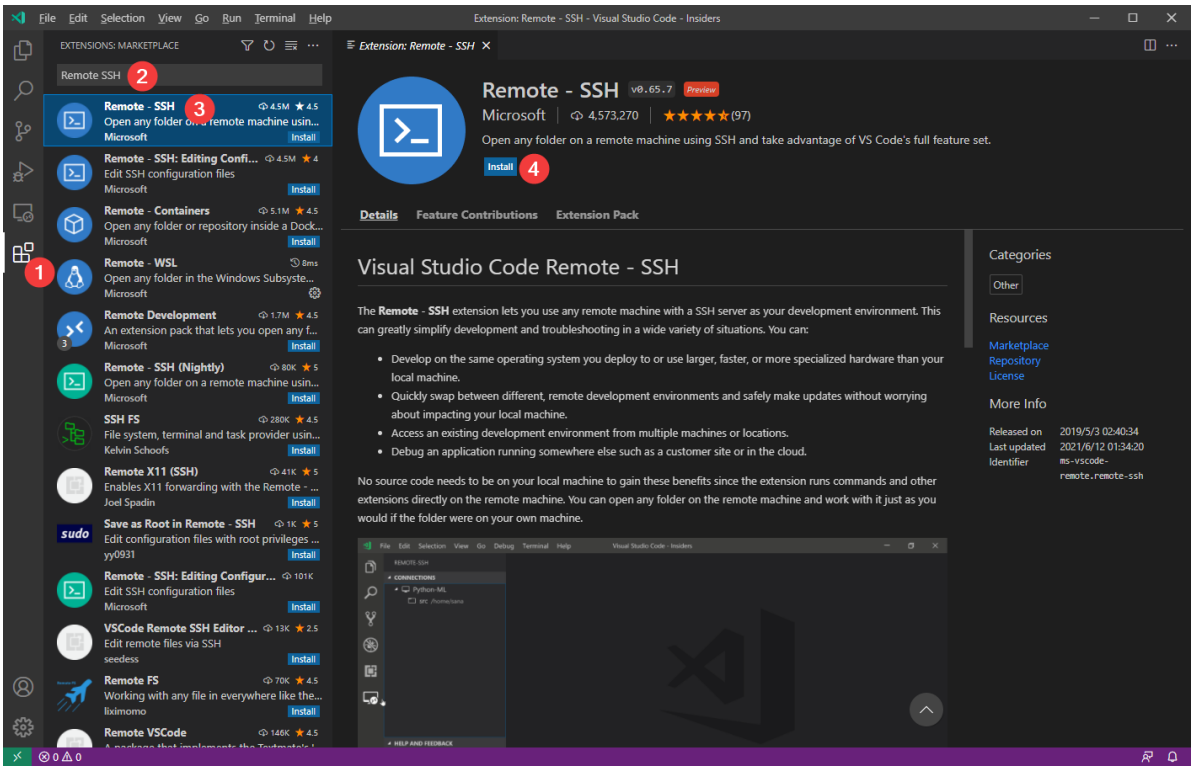
Setup VSCode to use VM directly

> **Notice:**
>
> To setup VSCode, please make sure that SSH is properly set up.

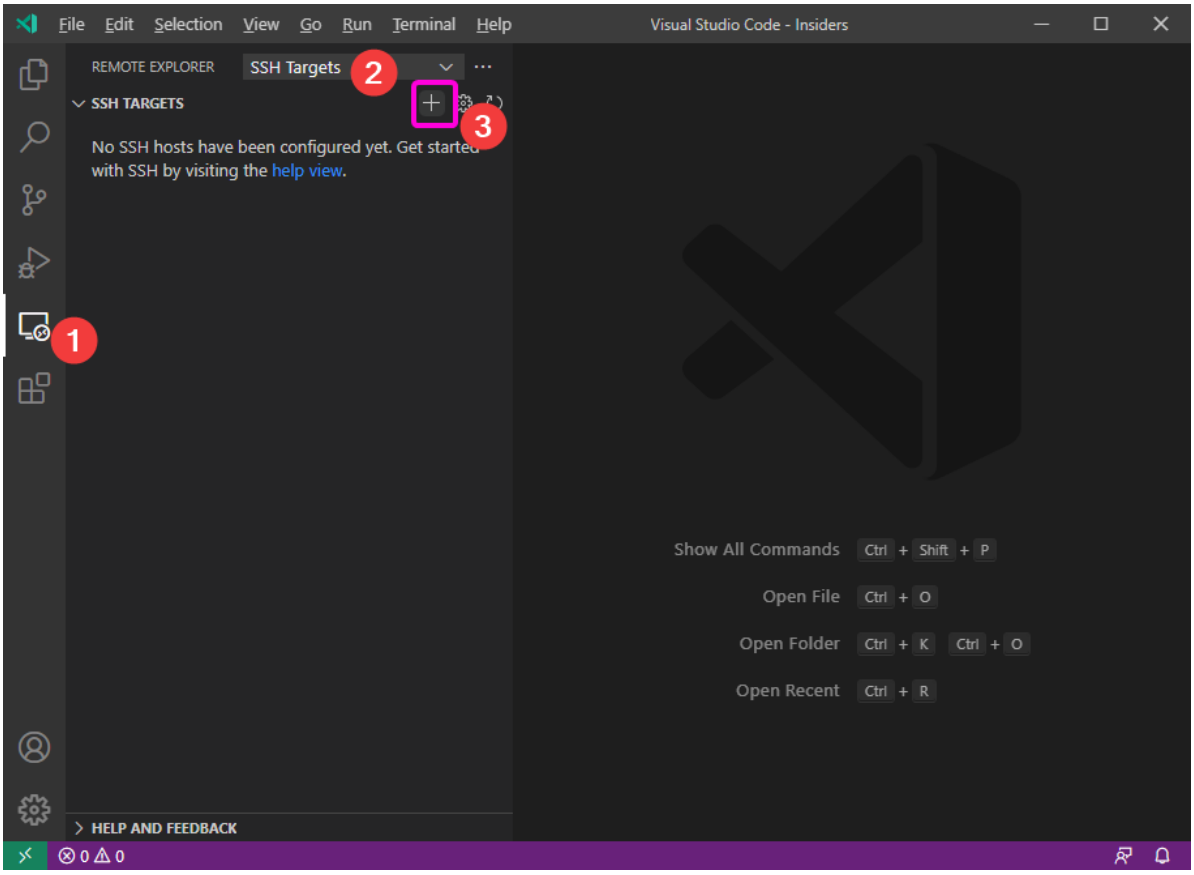
Visual Studio Code (VSCode) is a useful IDE for developers.

To setup the VM for use in VSCode, please refer to the following steps:

- Select "Extensions" at the side bar.
- Input "Remote SSH" in the search area.
- Select "Remote - SSH".
- Click "Install".

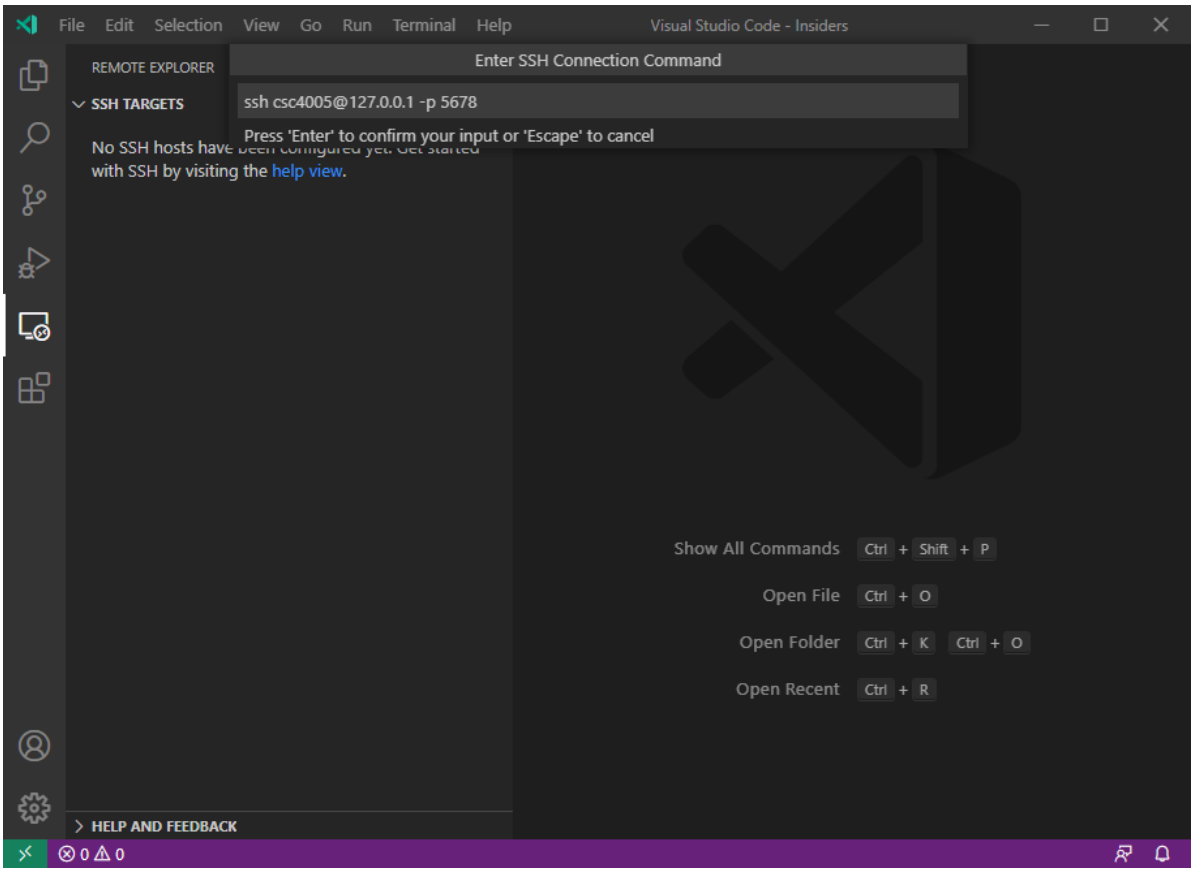


- Select "Remote Explorer" at the side bar.
- Select "SSH Targets" from the top dropdown.
- Move the mouse to the "SSH TARGETS" column, and buttons will show.
- Click the plus button.

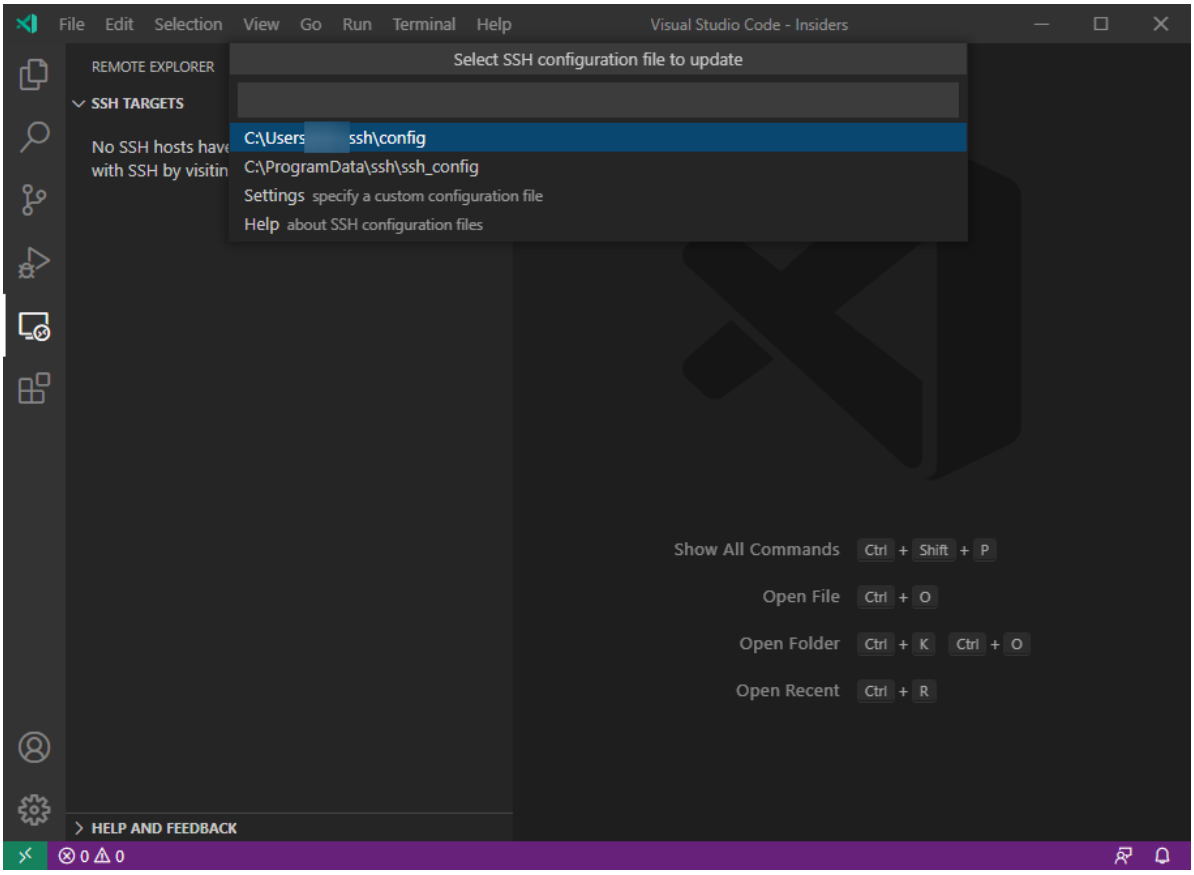


- Input "ssh csc4005@127.0.0.1 -p 5678".

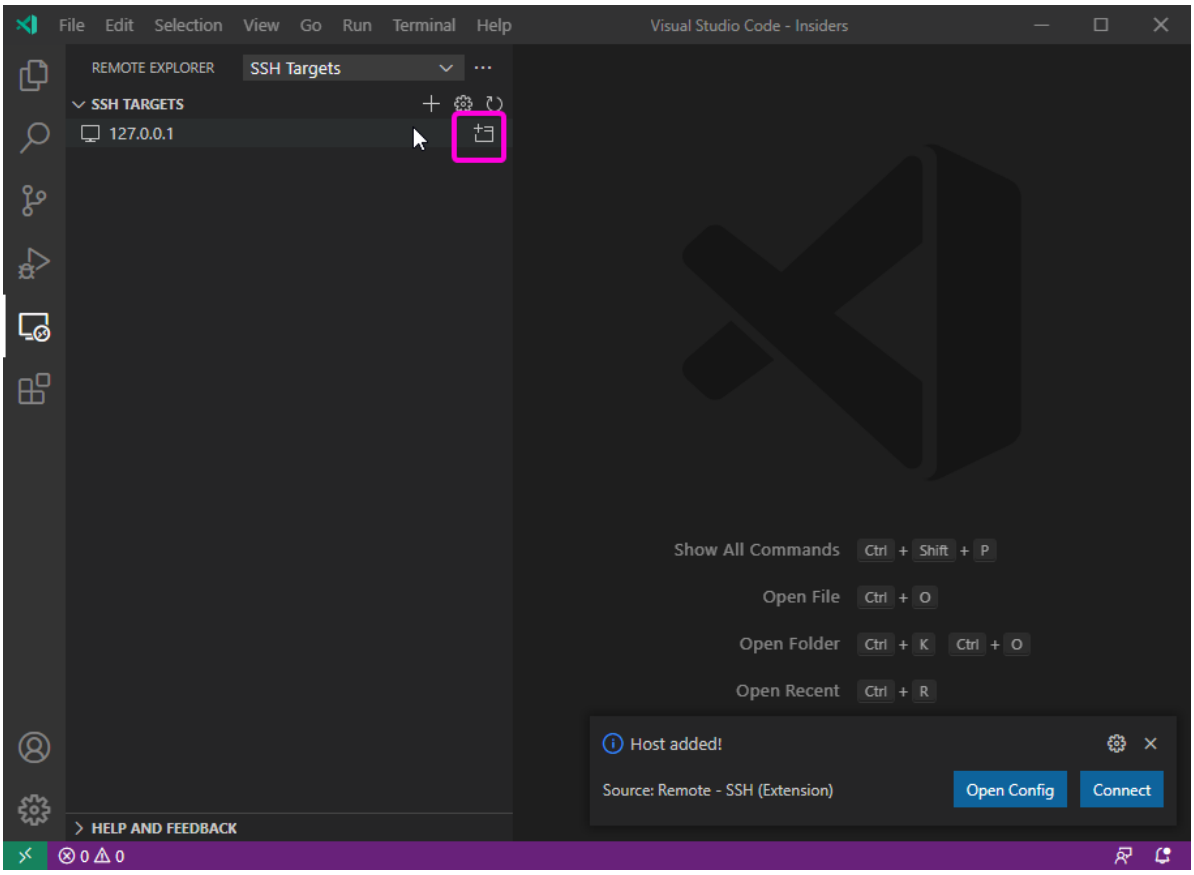
  (Note that the port "5678" might be changed according to your previous port choice.)



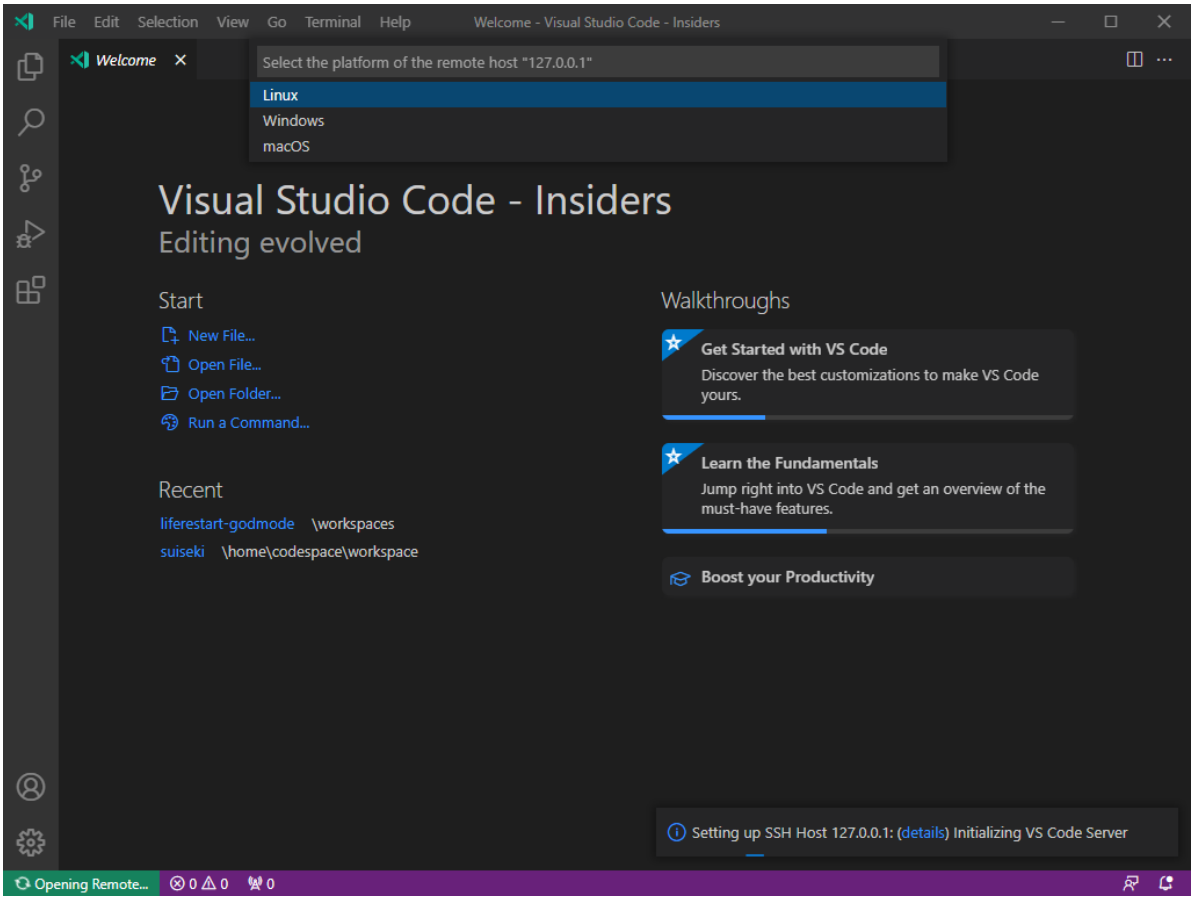
- Choose any of the two is okay.



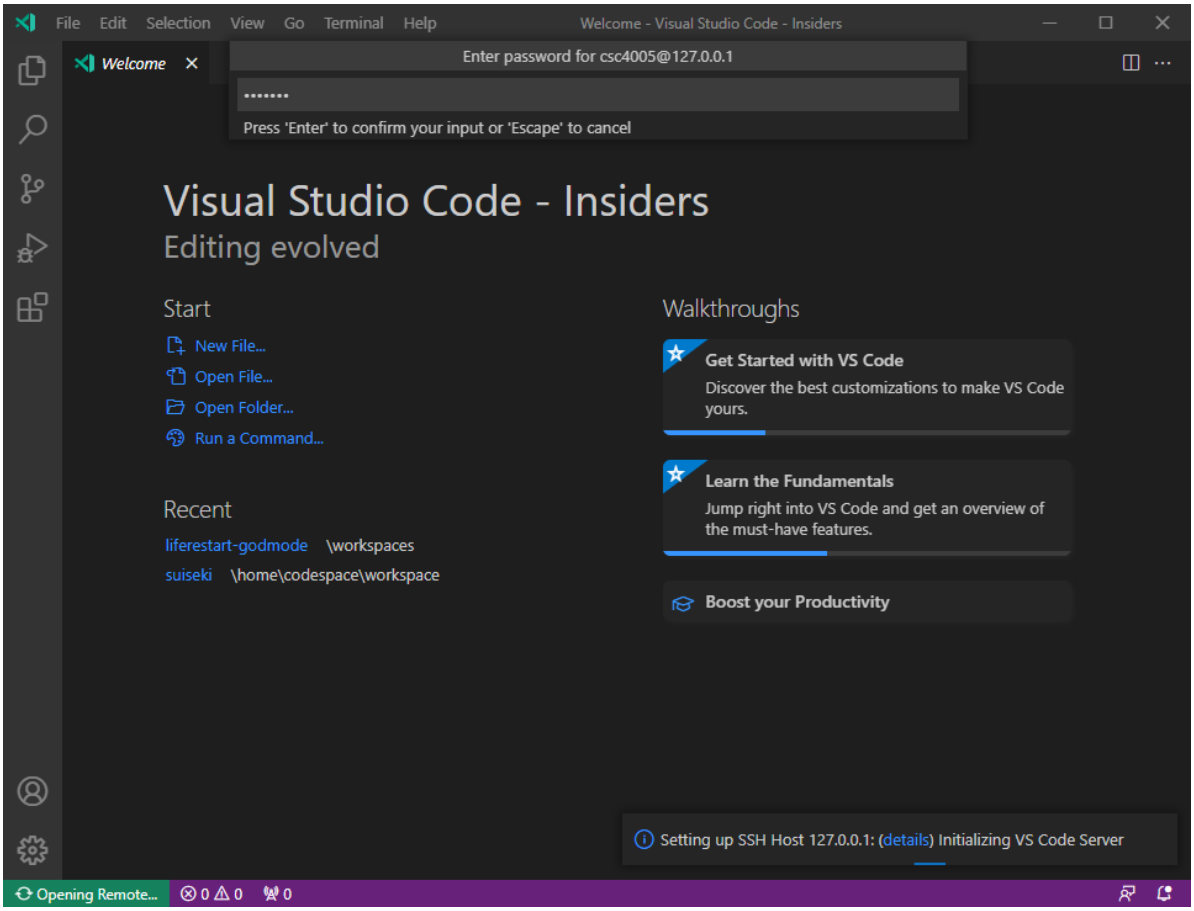
- Move the pointer to the line of "127.0.0.1".
- Then click the button called "Connect to Host in New Window".



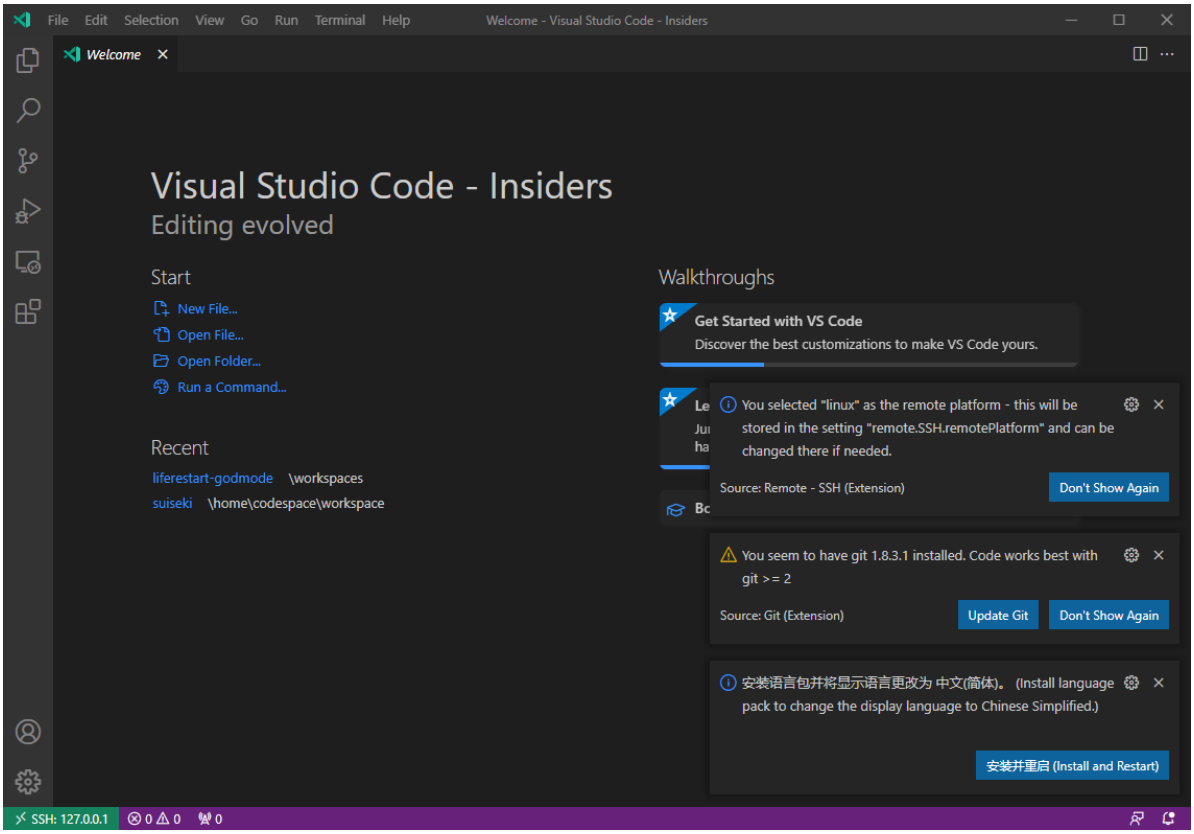
- A new window will open.
- Choose "Linux" as the platform when asked to.



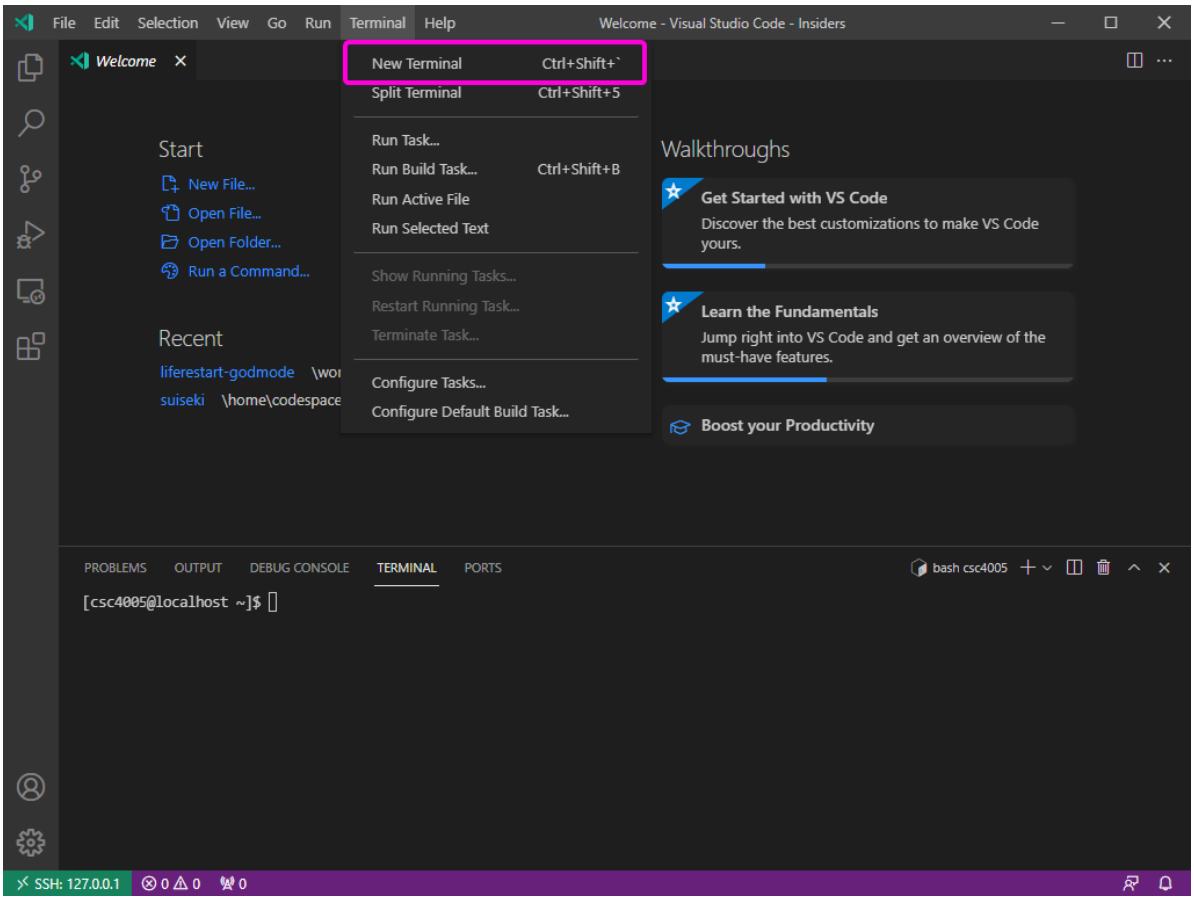
- Input the password when prompted.



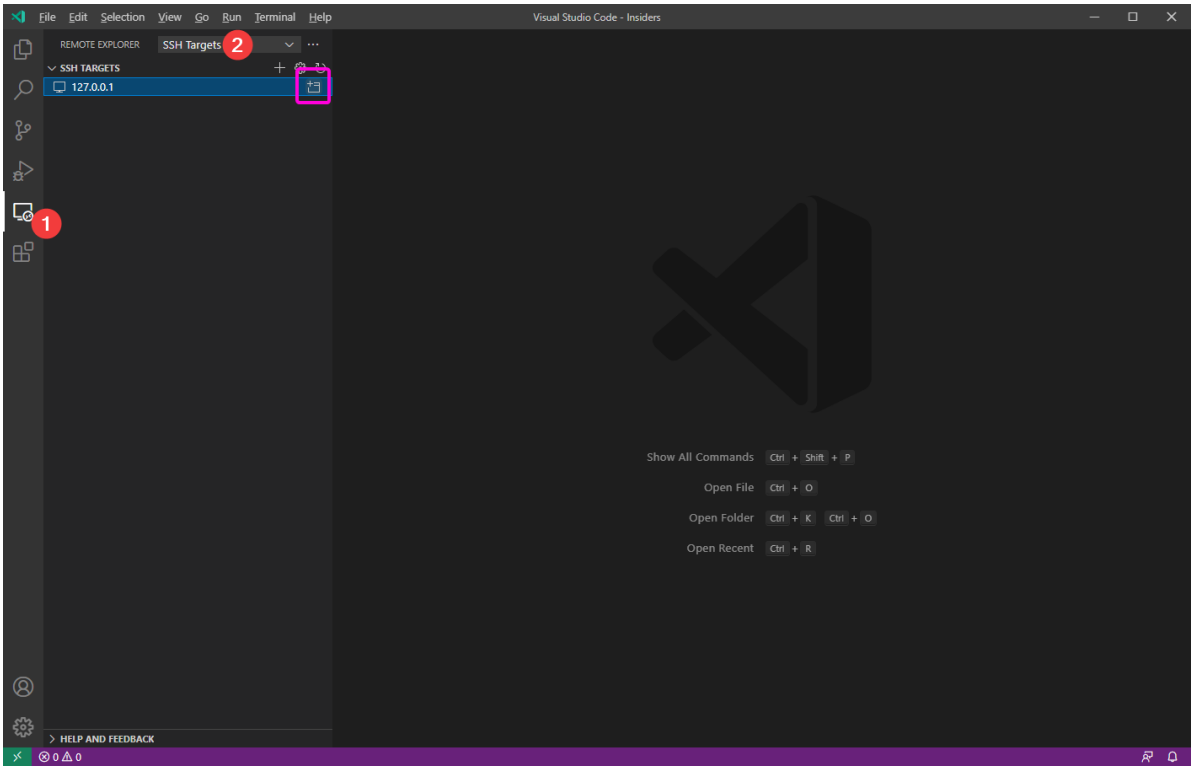
- You may see some prompts. They can be disregarded for now.
- A notation will be displayed left bottom on the screen, saying "SSH: 127.0.0.1", indicating that you are connecting to a remote host.



- Select "Terminal" > "New Terminal" to open a terminal at the bottom.



- When you want to connect to the host again, find the host from "Remote Explorer", "SSH Targets", and click the button right after the "127.0.0.1" line.



Toolchain

Our toolchains include

- Clang/Clang++ 12.0.1
- CMake `3.21.1`
- Open MPI 4.1.1
- CUDA 11.4

> **Notice:**
>
> Our clang/clang++ will use `libc++`, `ld.lld`, `compiler-rt`, `compiler-rt`, `libunwind`, `openmp` on default.

> **Notice:**
>
> Clang has full sanitizer support.

> **Notice:**
>
> **SM_35** is the suggested runtime for CUDA.

GUI

We will use `IMGUI + OpenGL2 + SDL2` for GUI library.

To setup the dependency, enter the following in the VM:

```
sudo yum install -y freetype-devel SDL2-devel mesa-libGL-devel
```

> **Notice:**
>
> These libraries will be ready out of box on the server. Sorry for the inconvenience that you have to install them on yourself in the VM.

Template project is at `https://github.com/SchrodingerZhu/csc4005-imgui/tree/opengl2`

You can clone it with

```
$ git clone https://github.com/SchrodingerZhu/csc4005-imgui -bopengl2 --recursive
```

Then you can build and run the GUI example with:

```
ssh -Y csc4005@127.0.0.1 -p5678
cd <path-to-project>
mkdir build
cd build
cmake .. -DCMAKE_BUILD_TYPE=Release
make
./csc4005_imgui
```

> **Notice:**
>
> The `-Y` flag is needed to forward display via SSH, this requires the local machine must have a X11 server. While this is also applicable to windows via something like https://sourceforge.net/projects/vcxsrv/ and to macOS via project like https://www.xquartz.org/ it requires you to install them first.
>
> If you do not bother installing them, you can always open the program within the VM display.
>
> More use cases like using `xvfb` to create a headless X11 environment will be taught in the tutorials.