EE538: Computing Principles for Electrical Engineers

Discussion 1: Environment Setting

University of Southern California
05/19 Summer 2022
Instructor: Arash Saifhashemi
TA and Mentors: Zixu Wang, Aditi Bodhankar

Introduction

Aditi Bodhankar
Mentor
bodhanka@usc.edu

Zixu (Leon) Wang
Mentor
zixuwang@usc.edu

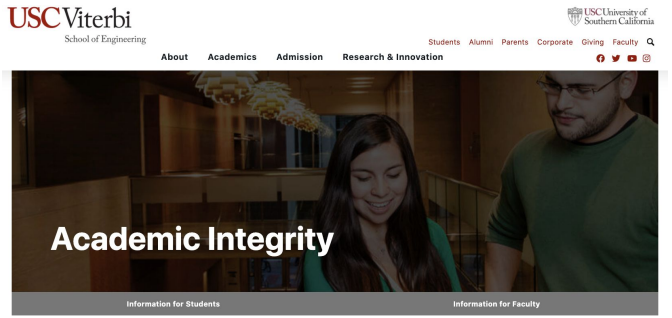
- Office Hours:
  - 2 hours for each Mentor in person
  - Will post to Piazza

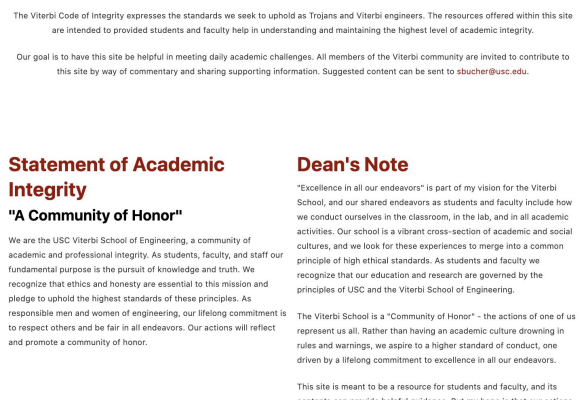
The most important thing - Academic Integrity

- University Policy
  - https://sjacs.usc.edu/students/academic-integrity/
  - https://viterbischool.usc.edu/academic-integrity/

We hope you could...

- Be responsible for yourself!
- Learn some useful knowledge!





Mask Policy: It is not mandatory but feel free to keep your mask in class

- University Policy
  - https://www.provost.usc.edu/covid-19/spring-2022-faculty-faq/
  - https://sjacs.usc.edu/disruptive-behavior/

We hope you could...

- Please stay home if you feel sick. We'll help you catch up.

Course Platform

- Piazza - https://piazza.com/class/l33ln1082u84cd?cid=1
  - All the lecture material will be posted here!
  - **Homework assignment will be released here!**
- GitHub classroom
  - **Submit your homework through GitHub before the deadline!**
- Blackboard/DEN
  - **Your scores will be posted here!**
- DEN
  - TBA

Outline

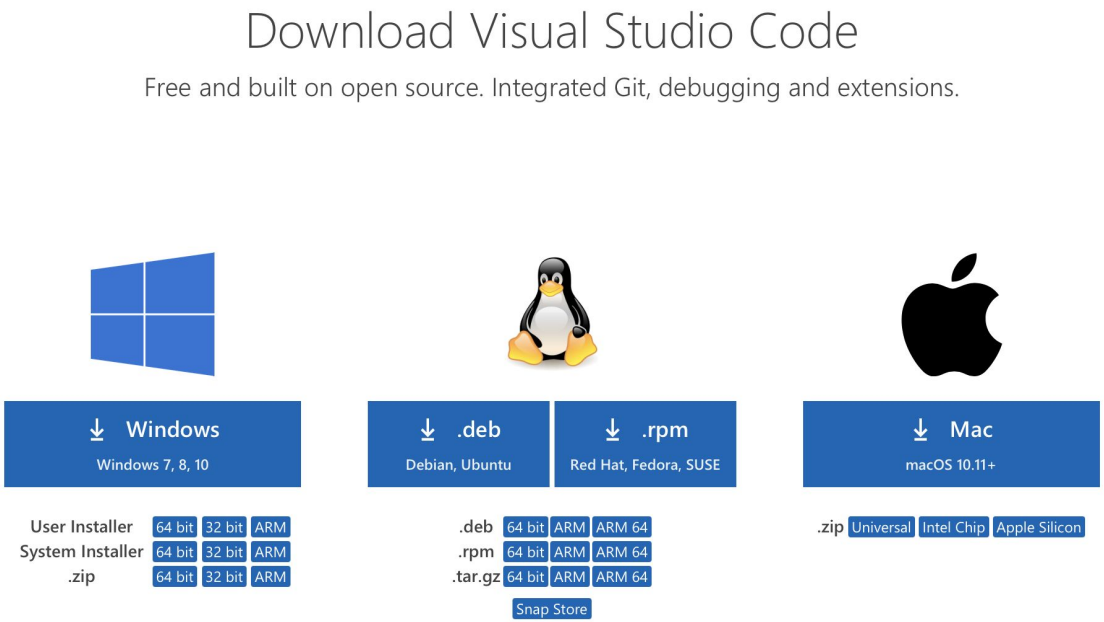
- Introduction
- Terminal Basics
  - Operating System
  - Terminal Commands
- GitHub Usage
  - GitHub Account
  - Git Command
- Bazel
- Environment Setup
  - Bazel Installation for Linux / Mac
  - Install Ubuntu on VirtualBox
- Programming Practice

Operating System

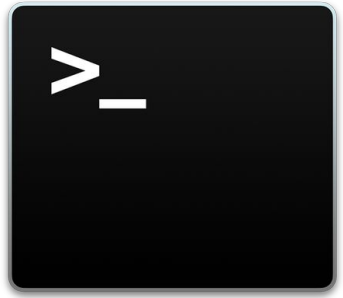
- For Linux (Ubuntu) and Mac users
  - You're all set!
- For Windows users, it is recommended to have a unix-like shell environment
  - Install a Linux system on your computer.
  - **Use a Virtual Machine (VM)!**
    - Install VirtualBox https://www.virtualbox.org/wiki/Downloads
    - Install Ubuntu system https://ubuntu.com/

Visual Studio Code

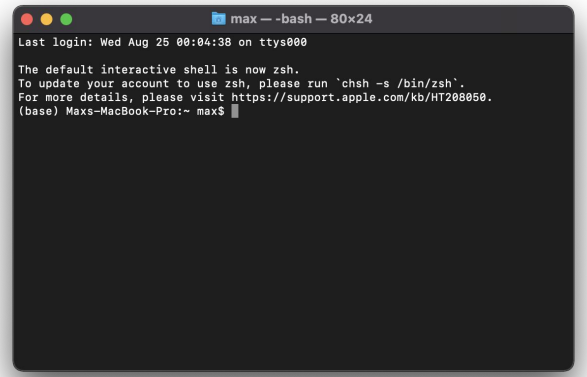
- IDE(Integrated development environment) help you code faster!
- Please go to https://code.visualstudio.com/Download and download VSC



What is Terminal or CLI ?



- Terminal, or the command-line interface (CLI)
  - processes commands to a computer program in the form of lines of text
- CLI used to be the only way to accomplish anything on a computer!
- In this class, we would use CLI as a means of manipulating the computer and get it to perform tasks instead of using a mouse to get things done!

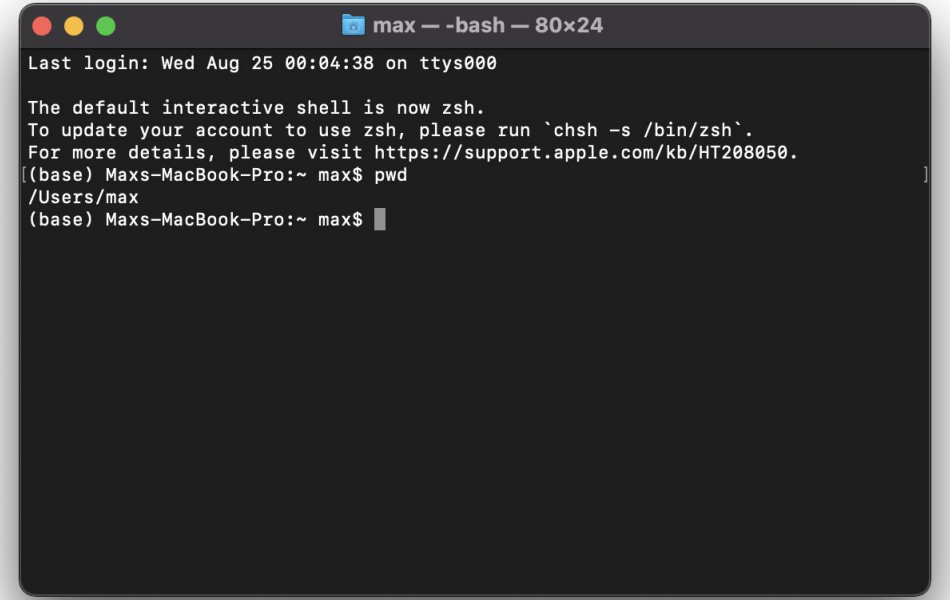


Terminal Commands:

0. Print working directory

    Command: **pwd**

    Purpose: **P**rint **W**orking **D**irectory



Terminal Commands:

1. Listing directory

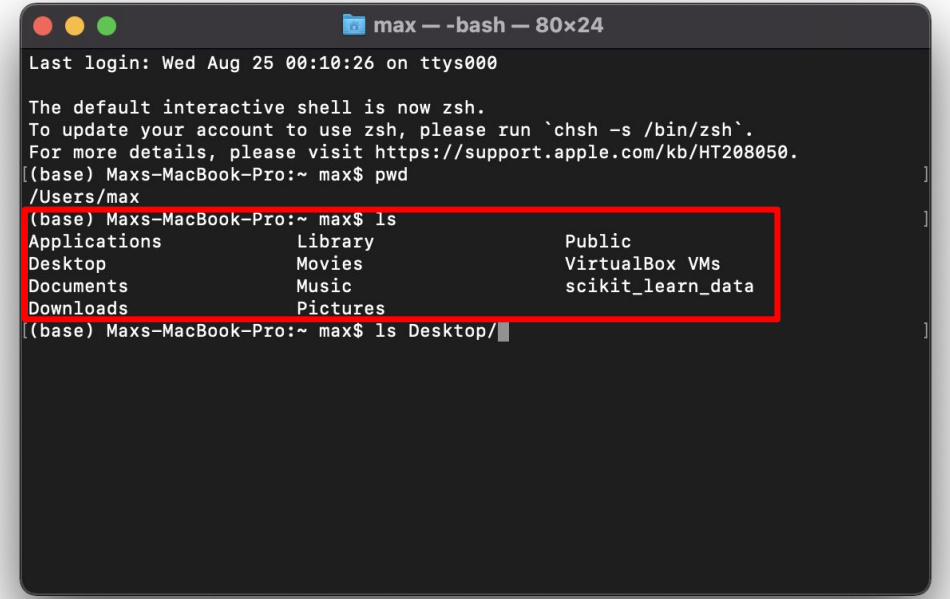
    Command: **ls**

    Purpose: **Lis**t the contents

    of a directory.

    Syntax: **ls “path-to-directory”**

    Example**: ls Desktop**



Terminal Commands:

2. Change directory

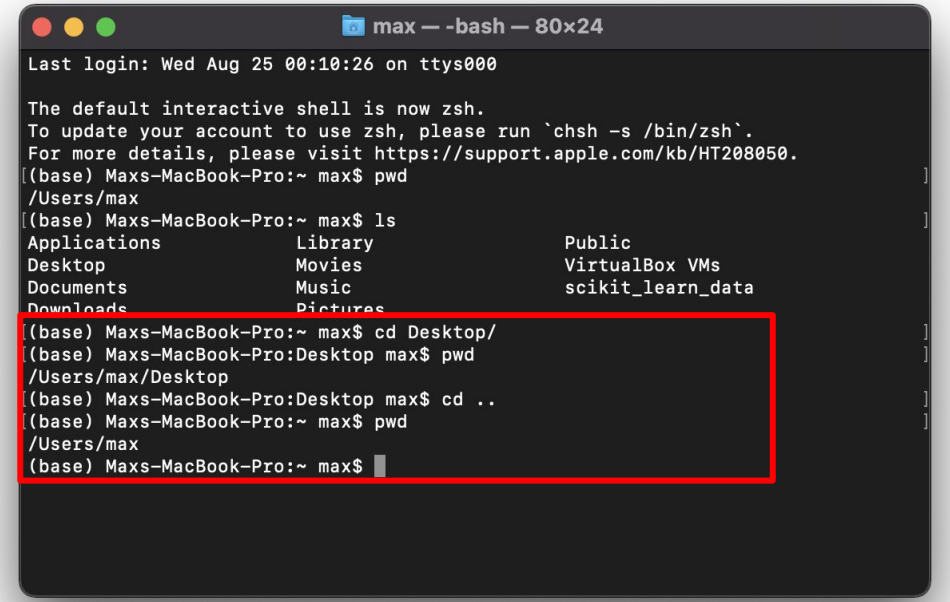
Command: **cd**

Purpose: **C**hange the **D**irectory of the command line path.

Syntax: **cd “path-to-directory”**

Example**: cd Desktop**

cd .. (move up one directory)



Terminal Commands:

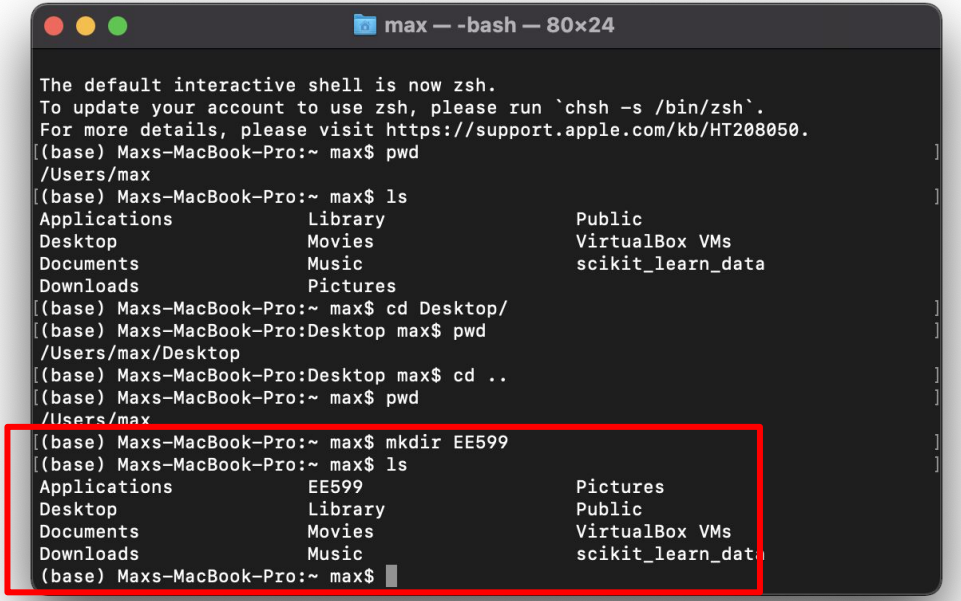
2. Create directory

Command: **mkdir**

Purpose: **Mak**e a new **Dir**ectory

Syntax: **mkdir “path”**

Example**: mkdir EE538**



Terminal Commands:

4. Copy a file to another directory

Command: cp

Syntax: cp "*filename*" "*path-to-directory*"

5. Move a file

Command: mv

Syntax: mv "*filename*" "*path-to-directory*”

6. Remove a file

Command: rm

Syntax: rm "*filename*”

7. Remove a directory

Command: rm -r

Syntax: rm -r "*path-to-directory*”

8. Execute commands with root privileges

Command: sudo

Syntax: sudo "command"

9. List actively running computer processes

Command: top

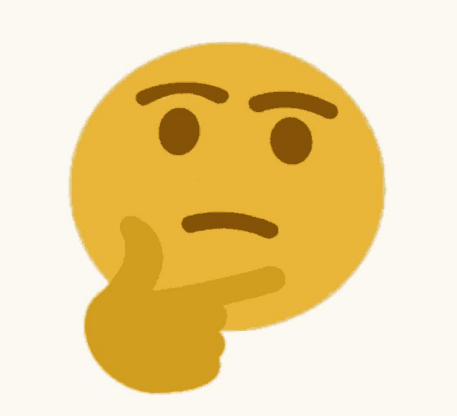
Questions?

1. **How to copy contents of one directory to another?**

   cp -r “source directory” “target directory”

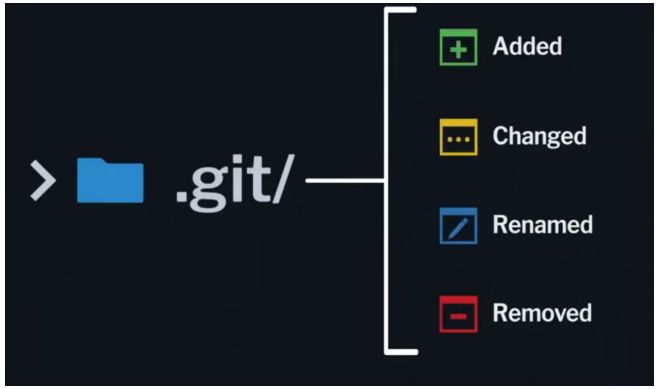
2. **How to clear the terminal?**

   clear



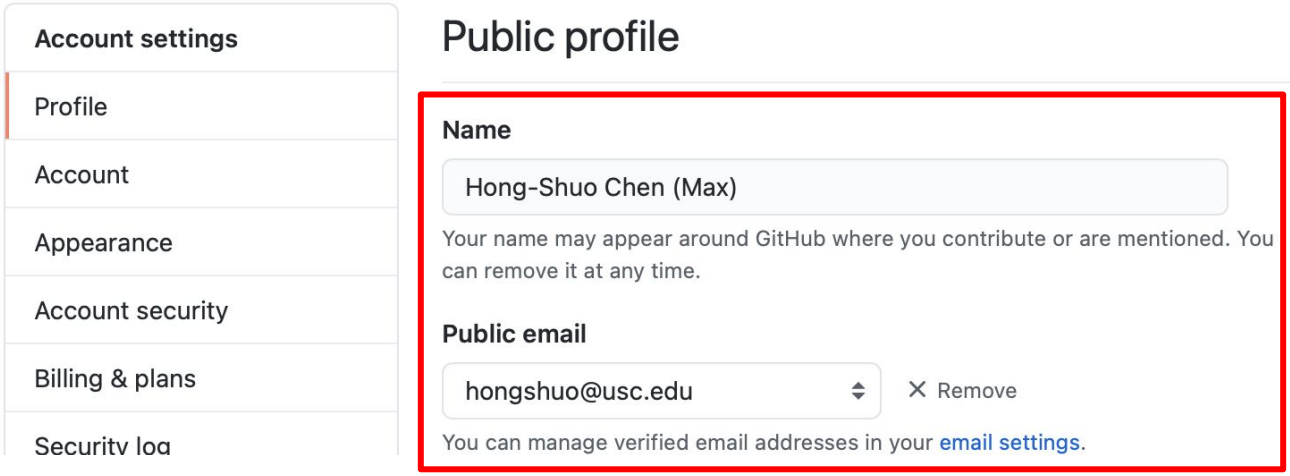
GitHub Usage

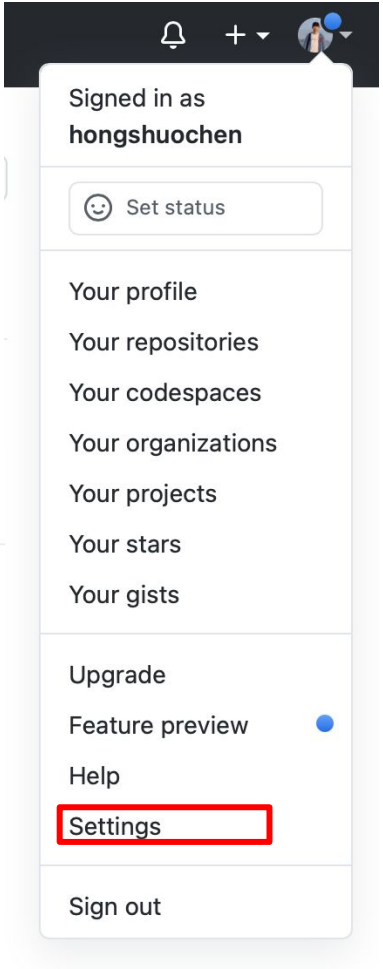
- Distributed version-control system
- Tracks changes you make to your files over the time
- Cooperate with others on the same project



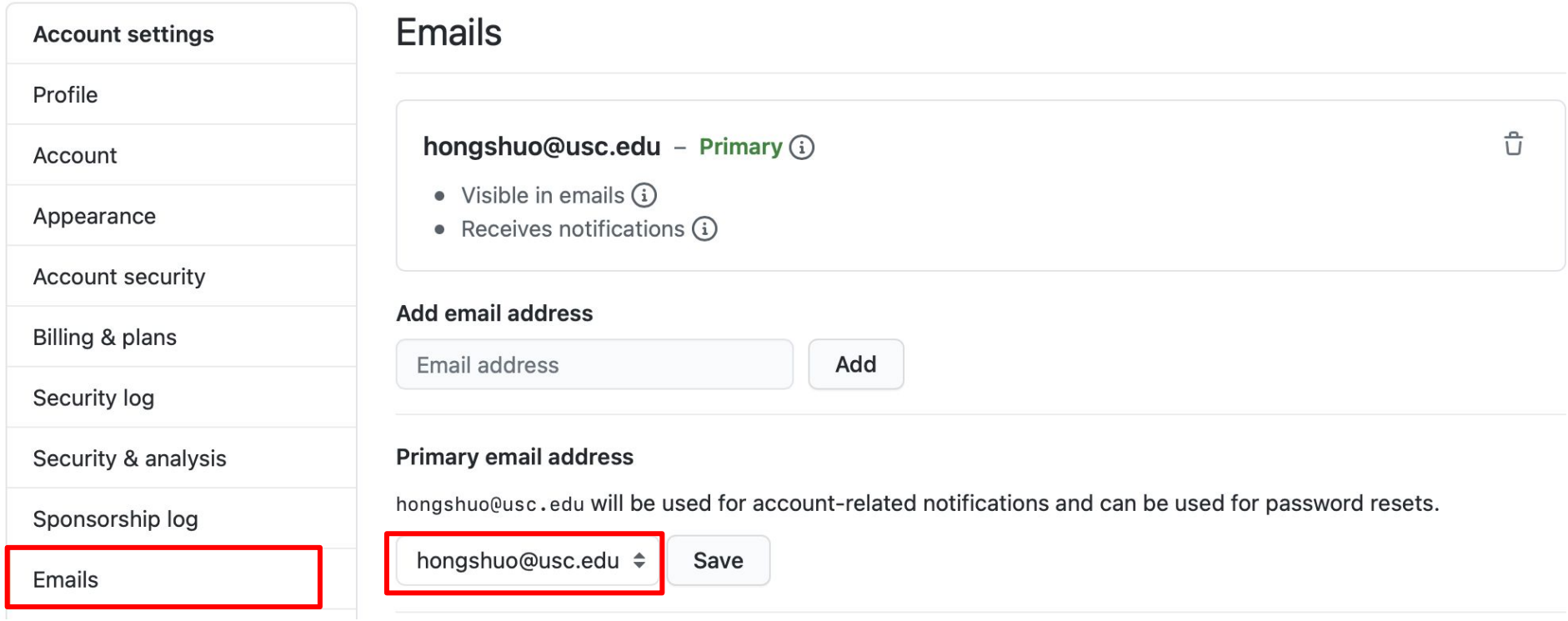
GitHub Account

- Please go to https://github.com, register a github account
- Set your real name, **usc email address** as primary public email address



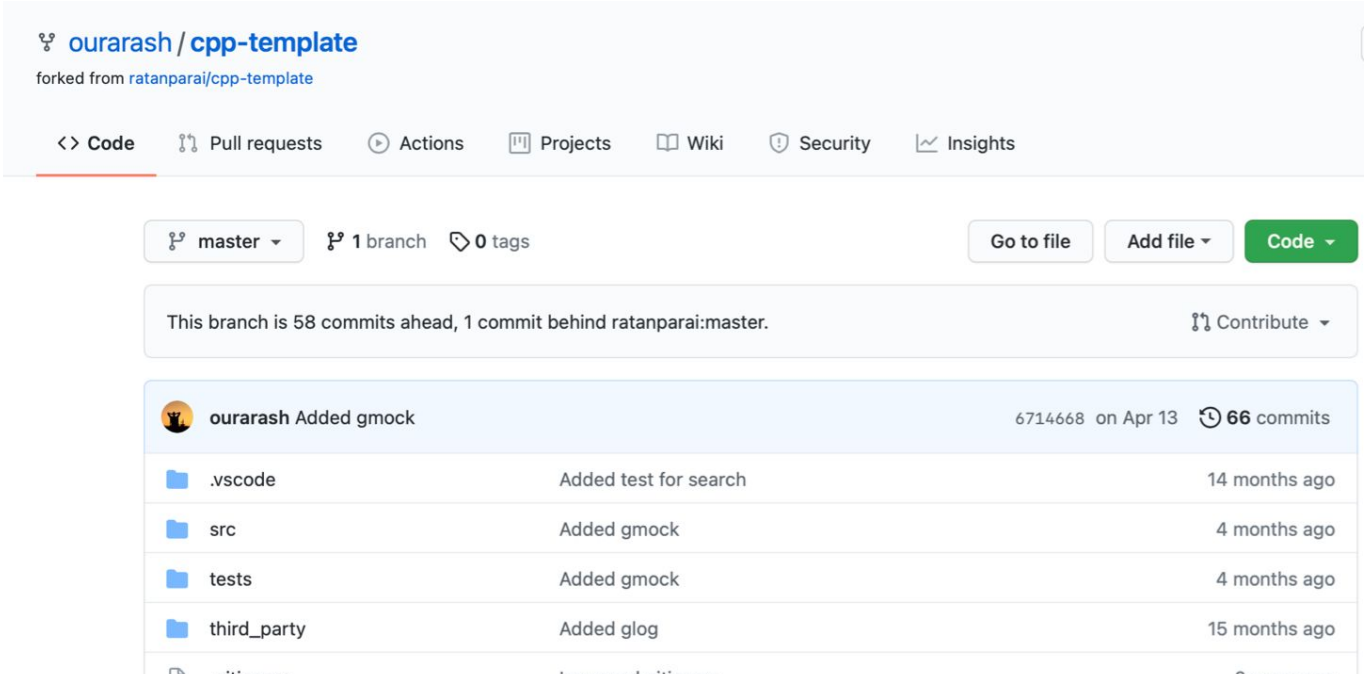


GitHub Account

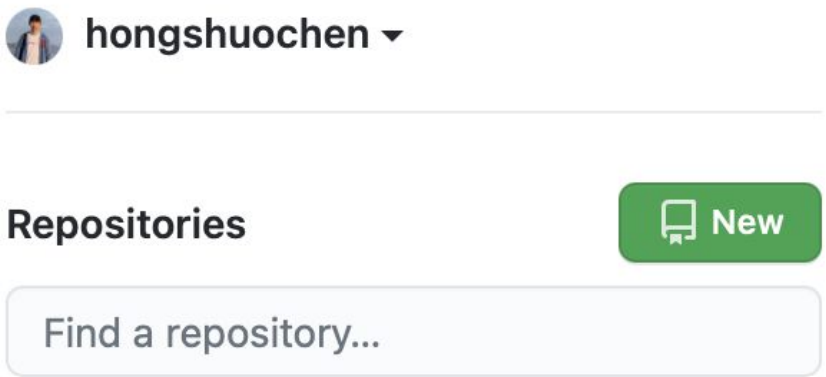


Clone a repository

- git clone https://github.com/ourarash/cpp-template



Create a New Repository



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Repository template
Start your repository with a template repository's contents.

No template

Owner * **Repository name** *

hongshuochen / EE538

Great repository names are short and memorable. Need inspiration? How about vigilant-journey?

Description (optional)

Repository for EE538

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

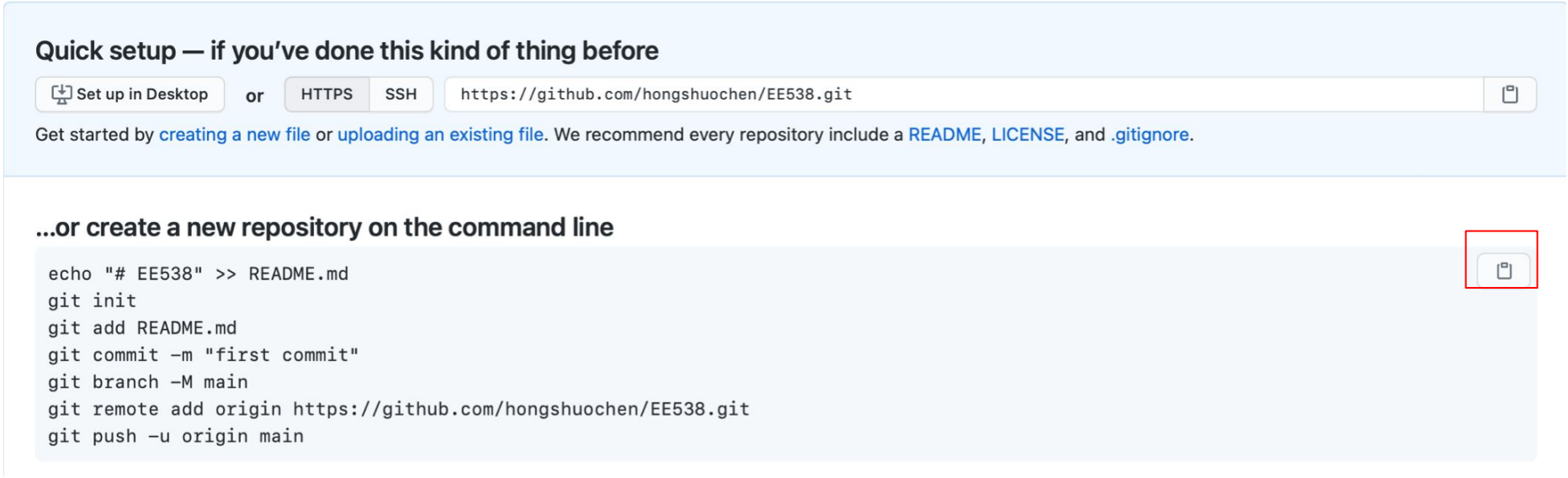
Add a README file
This is where you can write a long description for your project. Learn more.

Add .gitignore
Choose which files not to track from a list of templates. Learn more.

Choose a license
A license tells others what they can and can't do with your code. Learn more.

Create repository

Create a New Repository



Copy and paste to Terminal under EE538 directory!

Markdown basics

- Basics - [Markdown-guide]

  - Add summary and details for the codes
  - Insert both code and text for referencing

```
Inline `code` has `back-ticks around` it.
```

Inline `code` has `back-ticks around` it.

Footnotes

Footnotes aren't part of the core Markdown spec, but they supported by GFM.

```
Here is a simple footnote[^1].

A footnote can also have multiple lines[^2].

You can also use words, to fit your writing style more closely[^note].

[^1]: My reference.
[^2]: Every new line should be prefixed with 2 spaces.
  This allows you to have a footnote with multiple lines.
[^note]:
    Named footnotes will still render with numbers instead of the text but allow easier identification
    This footnote also has been made with a different syntax using 4 spaces for new lines.
```

- Reference any materials used

Basic operation and workflow

- Clone the repository or Create the repository
- Make changes to files!
- Add your changes
  - **git add “filename”** (e.g. git add test.cc)
  - **git add “directory”** (e.g. git add src)
  - **git add .** (add all changes in the current folder)
- Commit changes with a message, like creating a snapshot of your repository
  - **git commit -m “update files”**
- Push local changes to GitHub
  - **git push**
- Pull changes from GitHub
  - **git pull**

Questions?

1. Use which command to download the repository from GitHub?

   ANS: git clone

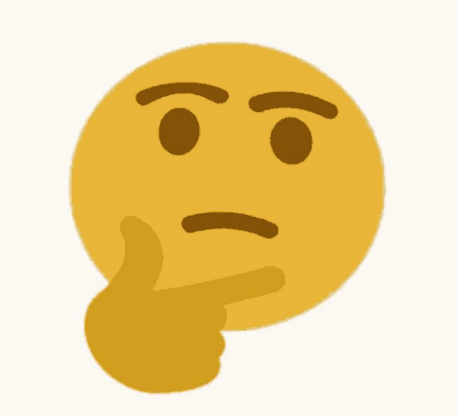
2. Use which two commands to create a version your current changes?

   ANS: git add

   git commit

3. Use which command to upload your changes to GitHub?

   ANS: git push



Bazel

- Bazel is a tool to build and test your code, similar to “Make”
- Build and test software of any size, quickly and reliably

Trusted by industry leaders

Bazel Installation on MacOS

- https://docs.bazel.build/versions/4.2.2/install-os-x.html#install-on-mac-os-x-homebrew

Installing using Homebrew

Step 1: Install Homebrew on macOS

Install Homebrew (a one-time step):

```
/bin/bash -c "$(curl -fsSL \
https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

Step 2: Install Bazel via Homebrew

Install the Bazel package via Homebrew as follows:

```
$ brew install bazel
```

All set! You can confirm Bazel is installed successfully by running the following command:

```
$ bazel --version
```

Once installed, you can upgrade to a newer version of Bazel using the following command:

```
$ brew upgrade bazel
```

Bazel Installation on Ubuntu

- https://docs.bazel.build/versions/4.2.0/install-ubuntu.html

Using Bazel's apt repository

Step 1: Add Bazel distribution URI as a package source

Note: This is a one-time setup step.

```
sudo apt install curl gnupg
curl -fsSL https://bazel.build/bazel-release.pub.gpg | gpg --dearmor > bazel.gpg
sudo mv bazel.gpg /etc/apt/trusted.gpg.d/
echo "deb [arch=amd64] https://storage.googleapis.com/bazel-apt stable jdk1.8" | sudo t
ee /etc/apt/sources.list.d/bazel.list
```

The component name “jdk1.8” is kept for legacy reasons only and doesn’t relate to supported or included JDK versions anymore. In the past, when Bazel did not yet bundle a private JRE, we had two release versions, one compatible with JDK 7 and one with JDK 8. However, since we dropped Java 7 support and started bundling a private runtime, Bazel releases are Java version agnostic. Changing the “jdk1.8” component name would break existing users of the repo though.

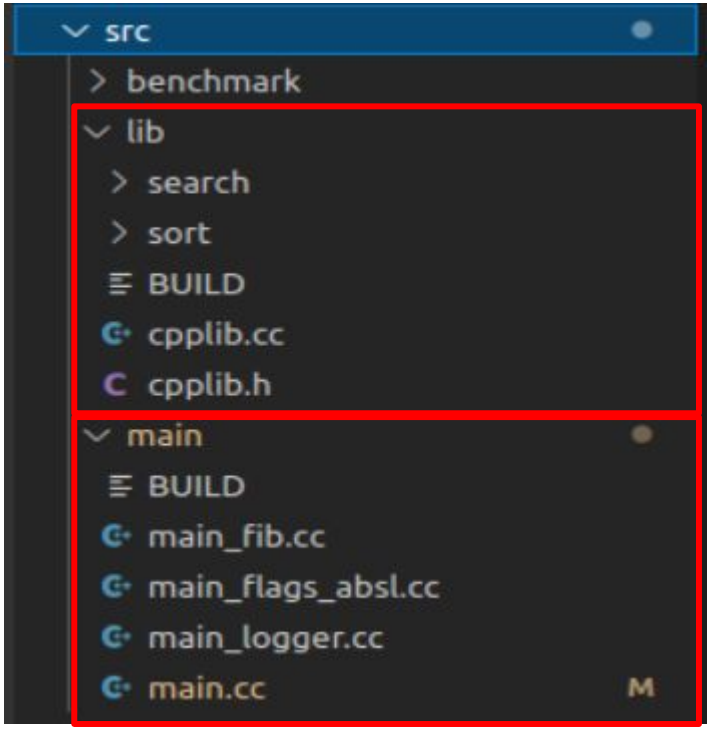
Step 2: Install and update Bazel

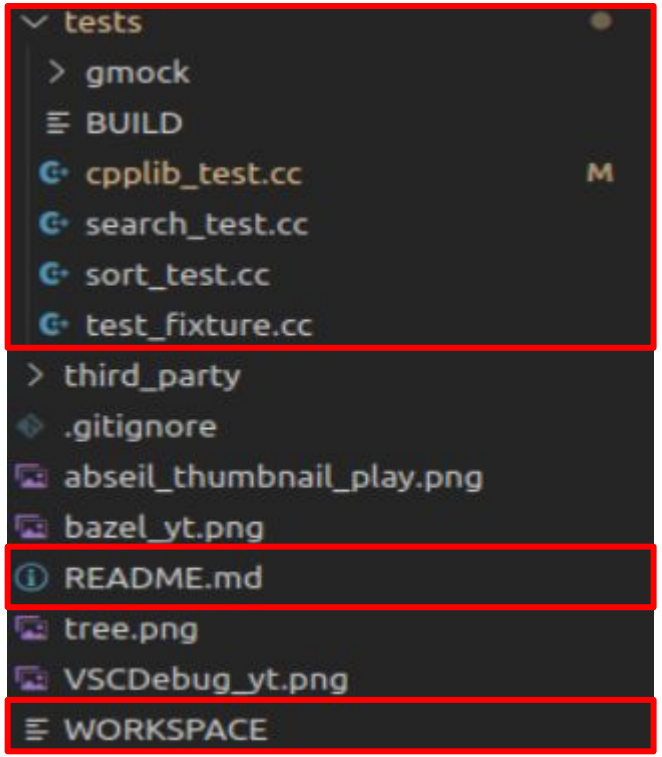
```
sudo apt update && sudo apt install bazel
```

Once installed, you can upgrade to a newer version of Bazel as part of your normal system updates:

```
sudo apt update && sudo apt full-upgrade
```

Bazel project structure





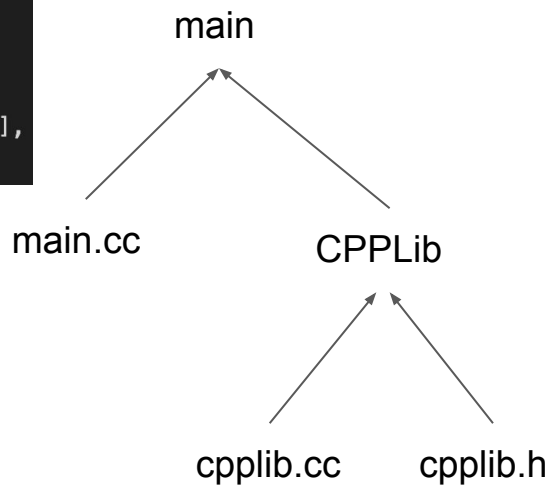
Understand BUILD file

- BUILD file is telling Bazel how to compile codes and headers of this project.
  - We need to use cc_binary, cc_library, cc_test
- Example on: /src/main/BUILD
- The BUILD file looks like:

```
cc_binary(
    name = "main",
    srcs = ["main.cc"],
    deps = ["//src/lib:CPPLib"],
)
```

```
cc_library(
    name = "CPPLib",
    srcs = ["cpplib.cc"],
    hdrs = ["cpplib.h"],
    visibility = ["//visibility:public"],
)
```

  - Bazel will then compile and build the executable binary
  - To run the binary file, run `bazel run src/main:main`



src/main/main.cc

```cpp
#include <iostream>

#include "src/lib/cpplib.h"

int main() {
  // Print Hello world!
  CPPLib s;
  std::cout << s.PrintHelloWorld() << std::endl;

  // std::cout << s.fib(6) << std::endl;
  // std::cout << s.fib(4) << std::endl;
  // std::cout << s.fib(8) << std::endl;
  return 0;
}
```

Bazel Commands

- Run
  - **bazel run src/main:main**

```
ee538@ee538-VirtualBox:~/Desktop/cpp-template$ bazel run src/main:main
INFO: Analyzed target //src/main:main (0 packages loaded, 0 targets configured).
INFO: Found 1 target...
Target //src/main:main up-to-date:
  bazel-bin/src/main/main
INFO: Elapsed time: 0.131s, Critical Path: 0.01s
INFO: 1 process: 1 internal.
INFO: Build completed successfully, 1 total action
INFO: Build completed successfully, 1 total action
**** Hello World ****
```

Test

GoogleTest

- Refer to this video:
  - https://youtu.be/0wMNtl2xDT0
  - https://www.youtube.com/watch?v=JJqRlSTQlh4
- A testing tool developed by Google is widely used in many other places.
- Purpose: independently test whether each function or class is working as expected.

In /tests/BUILD file:

```
cc_test(
    name = "cpplib_test",
    srcs = ["cpplib_test.cc"],
    deps = [
        "//src/lib:CPPLib",
        "@com_google_googletest//:gtest_main",
    ],
)
```

- To test, run `bazel test tests/gtest_demo:cpplib_test`

In /tests/cpplib_test.cc file:

```
#include "src/lib/cpplib.h"

#include <map>
#include <vector>

#include "gtest/gtest.h"

TEST(CPPLibTest, ReturnHelloWorld) {
  CPPLib cpplib;
  std::string actual = cpplib.PrintHelloWorld();
  std::string expected = "**** Hello Wold ****";
  EXPECT_EQ(expected, actual);
}
```

GoogleTest

- Find more assertions on this link
- https://google.github.io/googletest/reference/assertions.html
- Basic Assertions

These assertions do basic true/false condition testing.

Fatal assertion	Nonfatal assertion	Verifies
`ASSERT_TRUE(condition);`	`EXPECT_TRUE(condition);`	`condition` is true
`ASSERT_FALSE(condition);`	`EXPECT_FALSE(condition);`	`condition` is false

```
TEST(CPPLibTest, ReturnHelloWorld) {
  CPPLib cpplib;
  std::string actual = cpplib.PrintHelloWorld();
  std::string expected = "**** Hello Wold ****";
  EXPECT_EQ(expected, actual);
}
```

- Binary Assertions

Fatal assertion	Nonfatal assertion	Verifies
`ASSERT_EQ(val1, val2);`	`EXPECT_EQ(val1, val2);`	`val1 == val2`
`ASSERT_NE(val1, val2);`	`EXPECT_NE(val1, val2);`	`val1 != val2`
`ASSERT_LT(val1, val2);`	`EXPECT_LT(val1, val2);`	`val1 < val2`
`ASSERT_LE(val1, val2);`	`EXPECT_LE(val1, val2);`	`val1 <= val2`
`ASSERT_GT(val1, val2);`	`EXPECT_GT(val1, val2);`	`val1 > val2`
`ASSERT_GE(val1, val2);`	`EXPECT_GE(val1, val2);`	`val1 >= val2`

Bazel Commands

- Test
  - **`bazel test tests/gtest_demo:cpplib_test`**

```
ee538@ee538-VirtualBox:~/Desktop/cpp-template$ bazel test tests:cpplib_test
...
INFO: Analyzed target //tests:cpplib_test (11 packages loaded, 316 targets configured).
INFO: Found 1 test target...
Target //tests:cpplib_test up-to-date:
  bazel-bin/tests/cpplib_test
INFO: Elapsed time: 15.922s, Critical Path: 1.84s
INFO: 29 processes: 10 internal, 19 linux-sandbox.
INFO: Build completed successfully, 29 total actions
//tests:cpplib_test                                                    PASSED in 0.0s

Executed 1 out of 1 test: 1 test passes.
INFO: Build completed successfully, 29 total actions
```

Install Ubuntu on VM

Step 1: Download and Install VirtualBox

https://www.virtualbox.org/wiki/Downloads

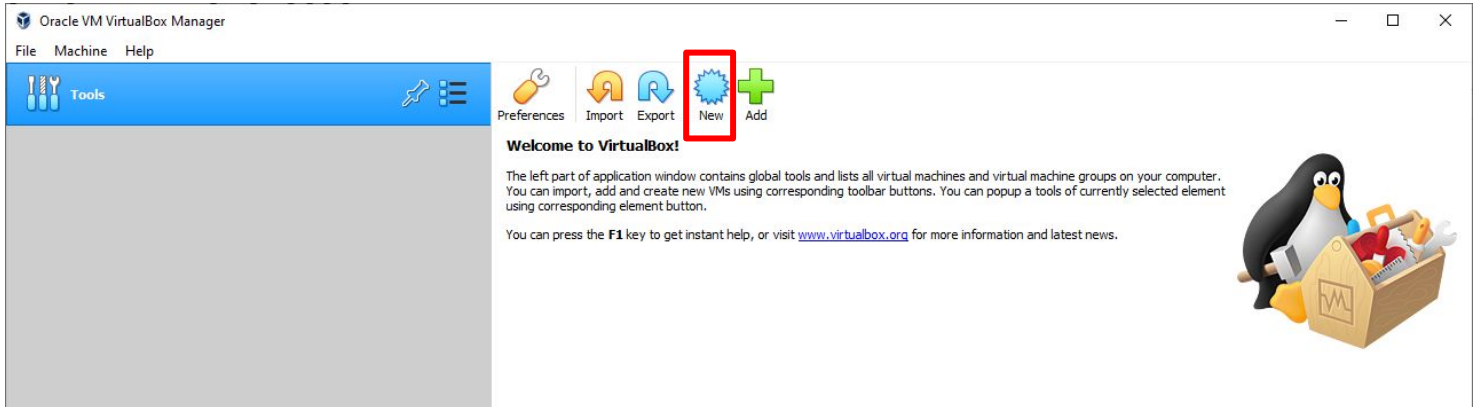
Install Ubuntu on VM

Step 2 Download Ubuntu

https://ubuntu.com/#download

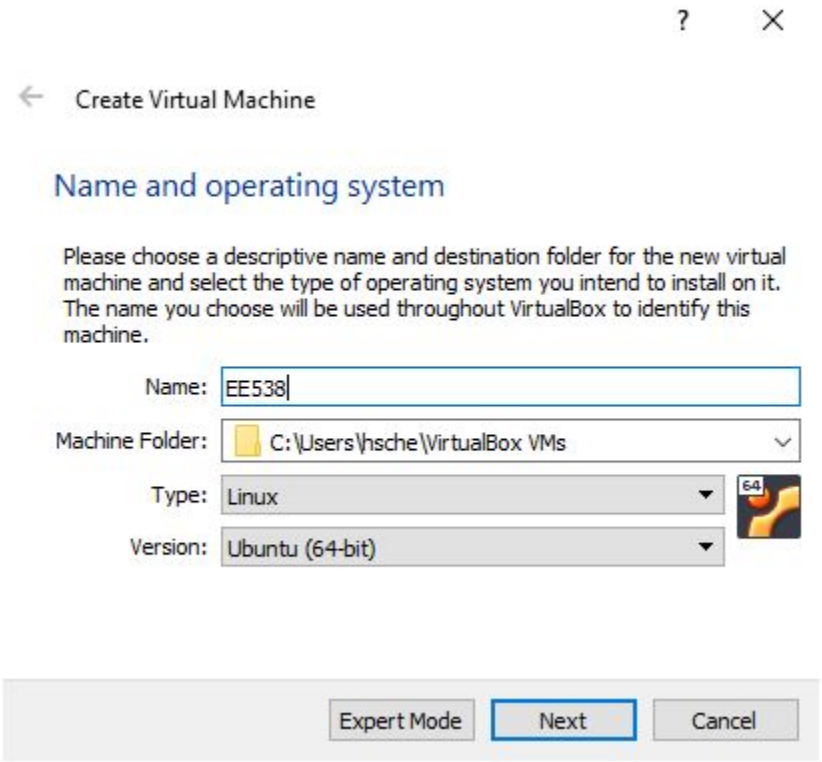
Install Ubuntu on VM

Step 3 Open VirtualBox and press New



Install Ubuntu on VM

Step 4



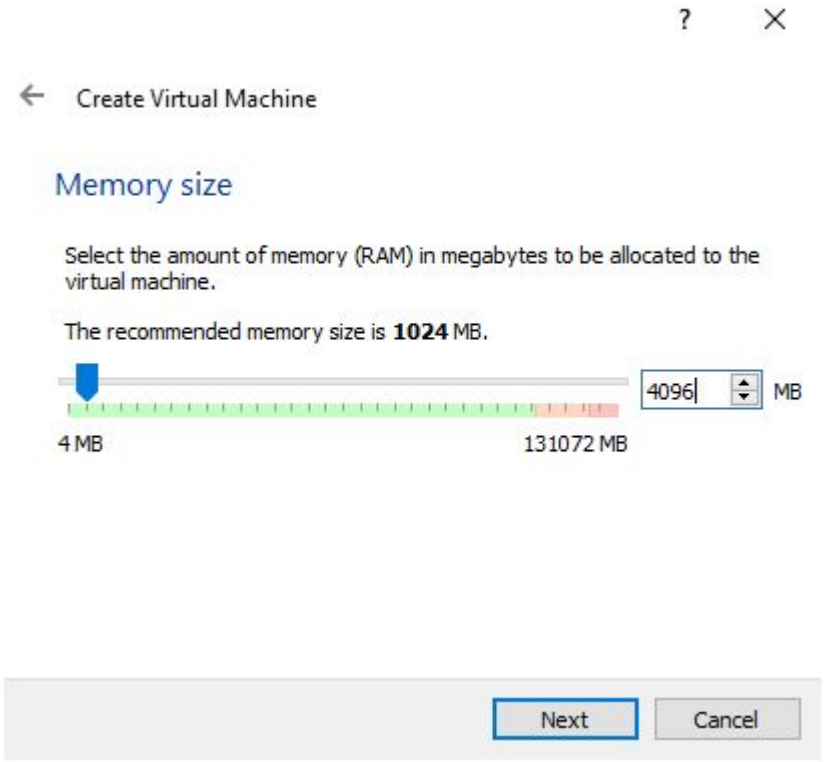
Install Ubuntu on VM

Step 5 Larger Memory is Faster

Memory should not exceed

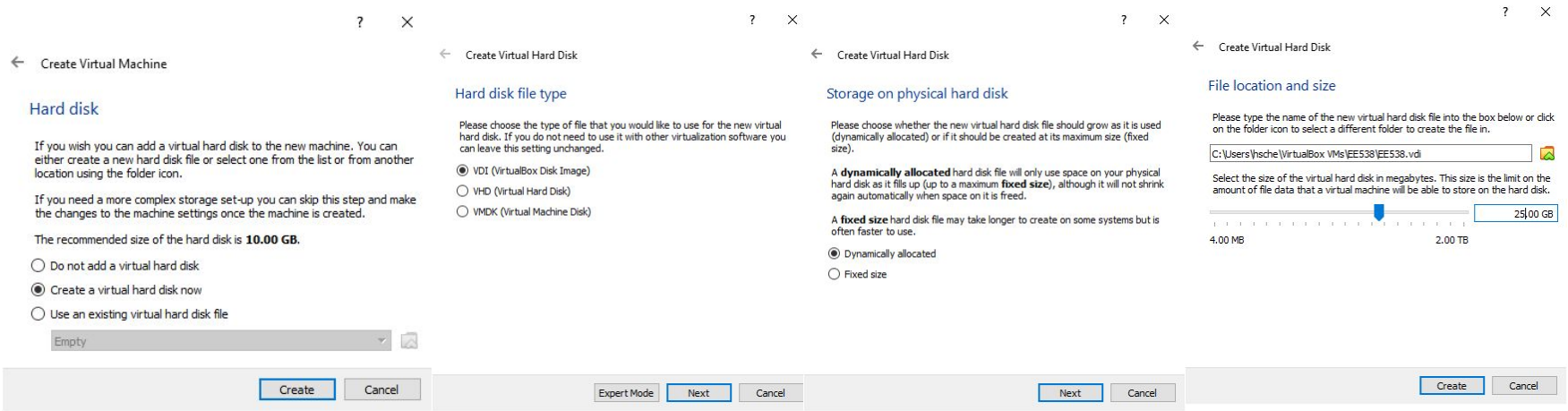
1/3 of the total amount of memory of

your host machine!

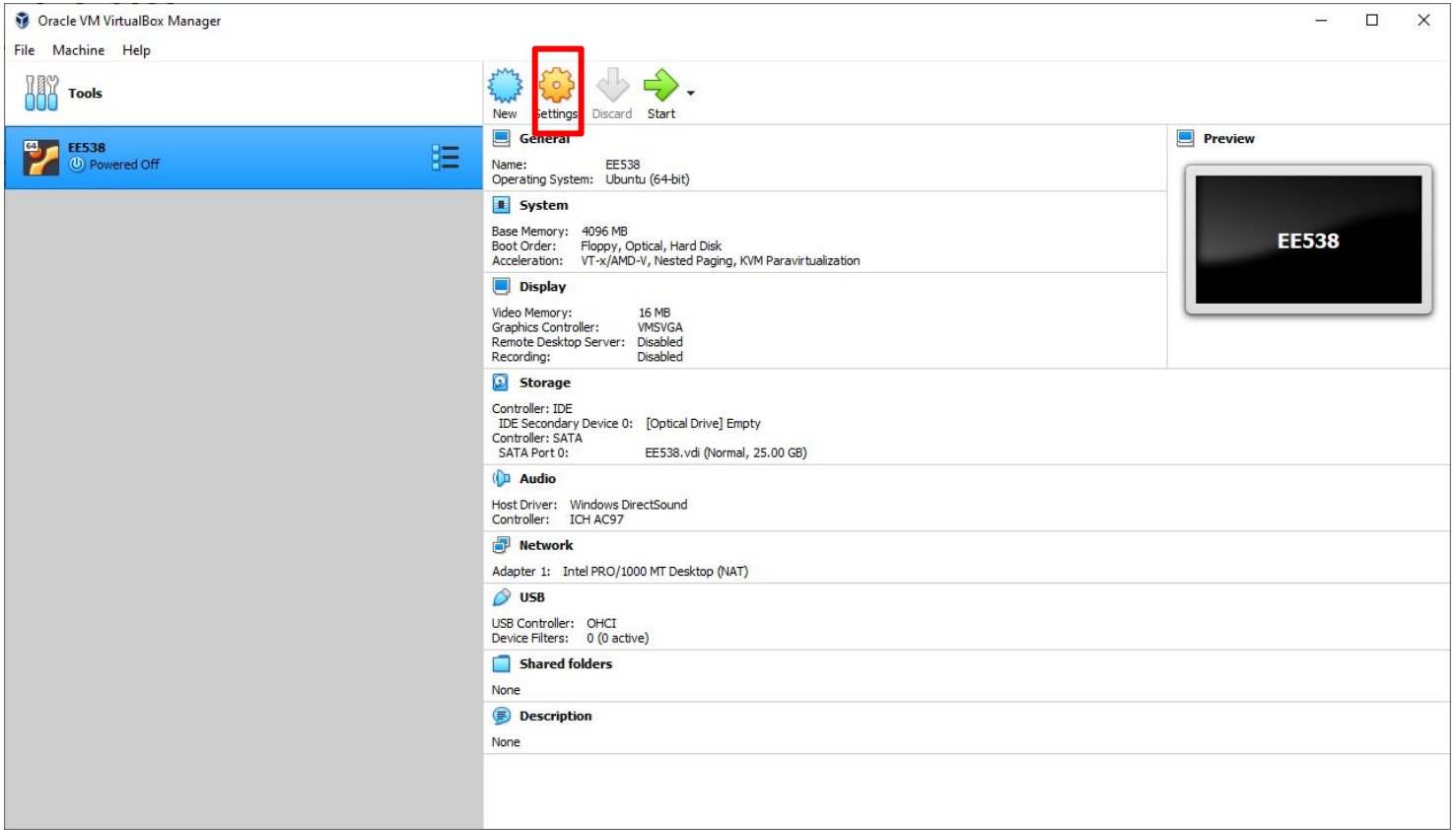


Install Ubuntu on VM

Step 6 Disk size: 25 GB



Install Ubuntu on VM



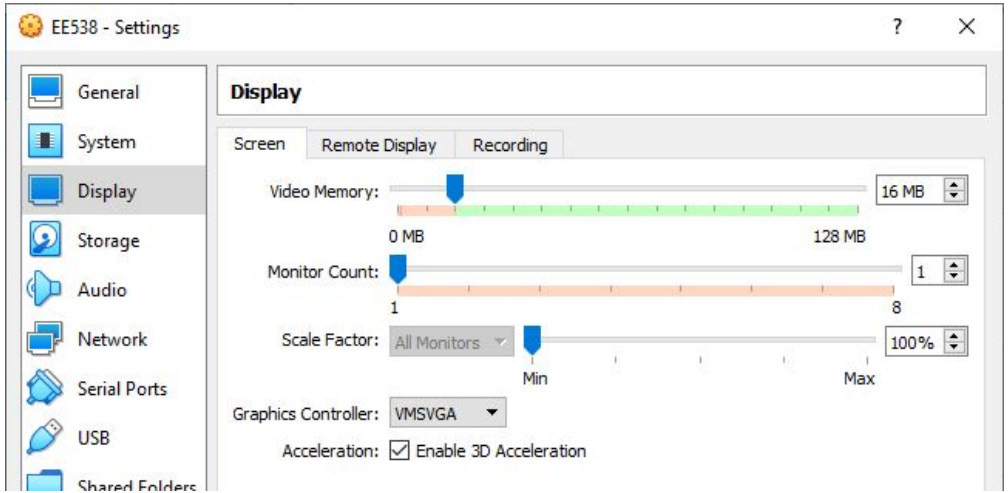
Install Ubuntu on VM

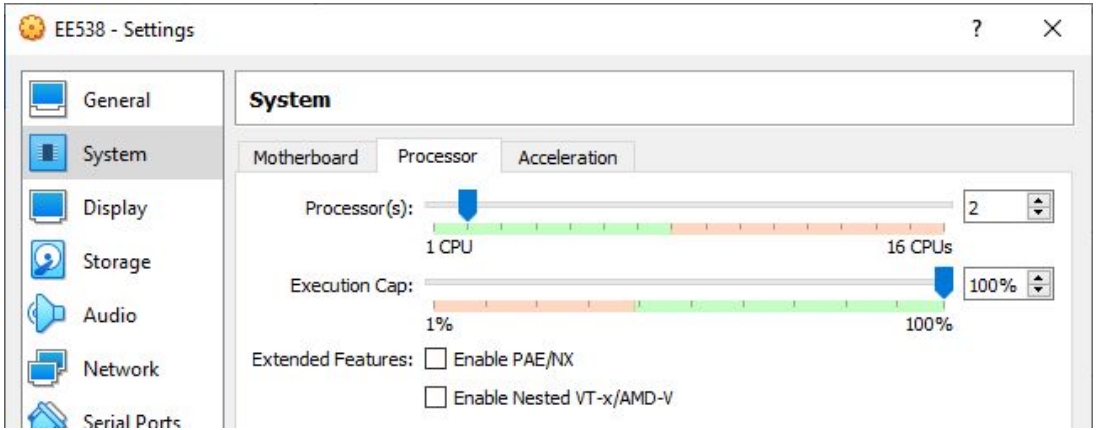
Step 7

Enable 3D acceleration

Make sure the VM

has at least 2 CPU cores allocated.



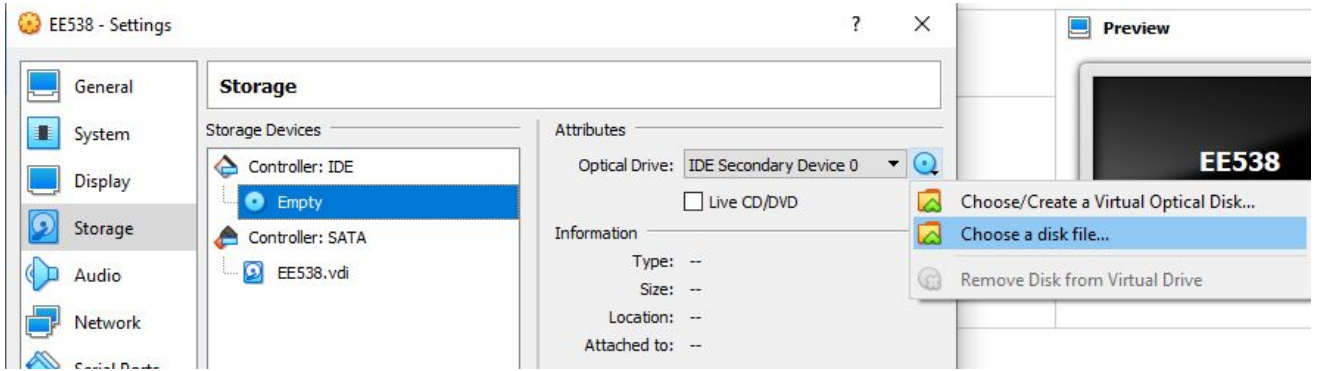


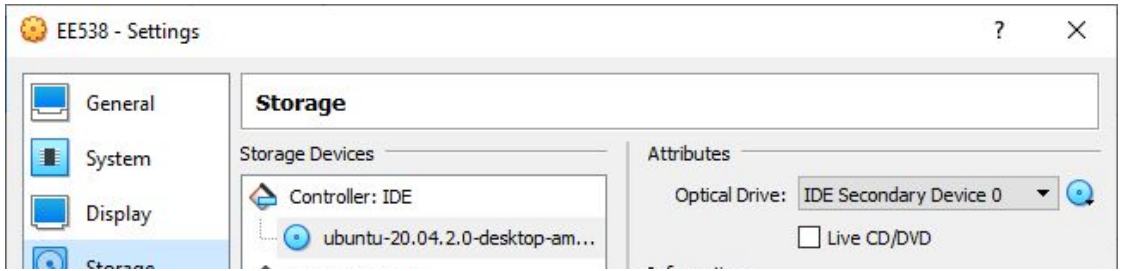
Install Ubuntu on VM

Step 8

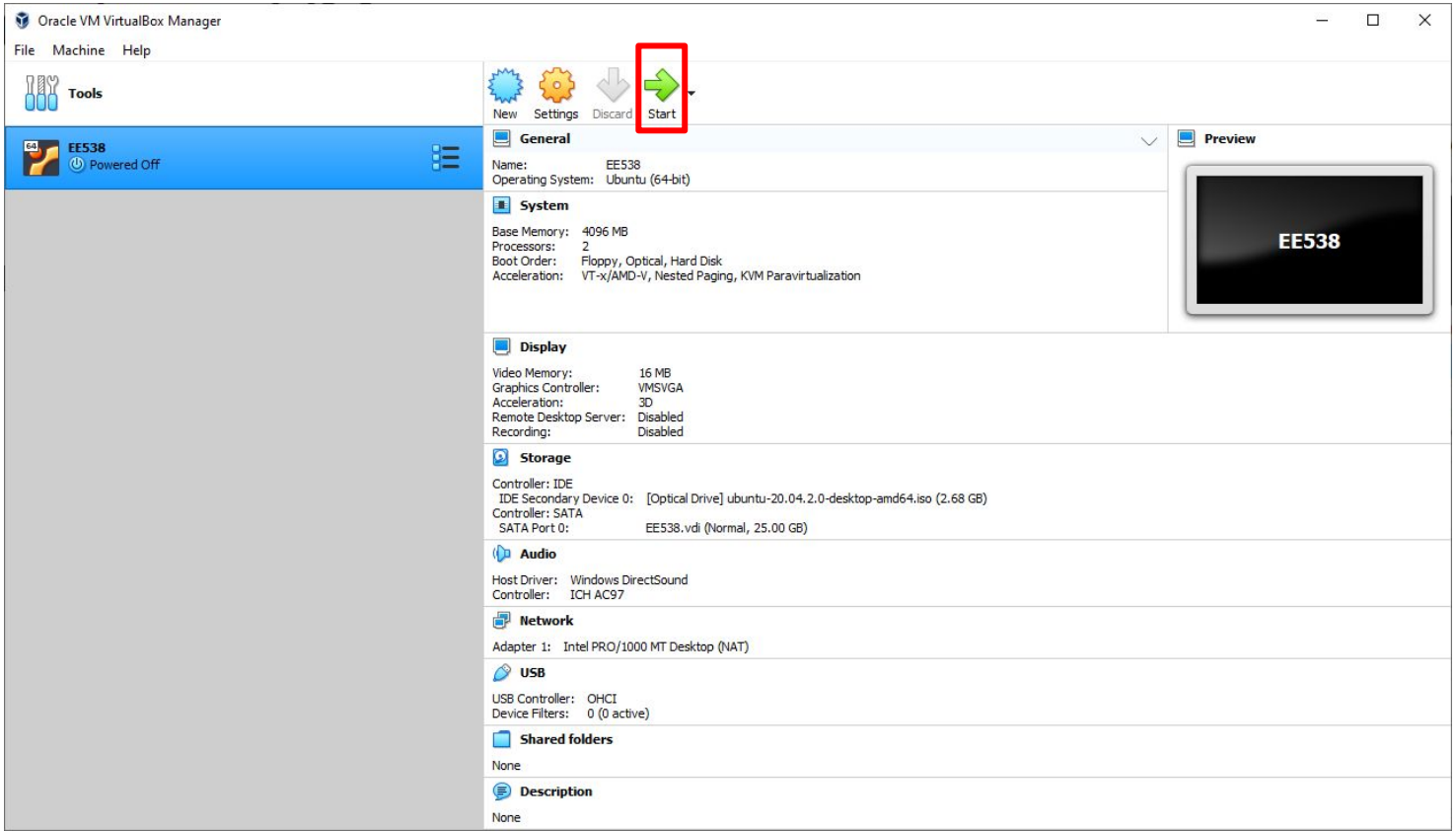
Choose Ubuntu disk file

that just downloaded





Install Ubuntu on VM



EE538 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Aug 25 09:23

Install

Welcome

- Asturianu
- Bahasa Indonesia
- Bosanski
- Català
- Čeština
- Cymraeg
- Dansk
- Deutsch
- Eesti
- English
- Español
- Esperanto
- Euskara
- Français
- Gaeilge
- Galego
- Hrvatski
- Íslenska

Try Ubuntu

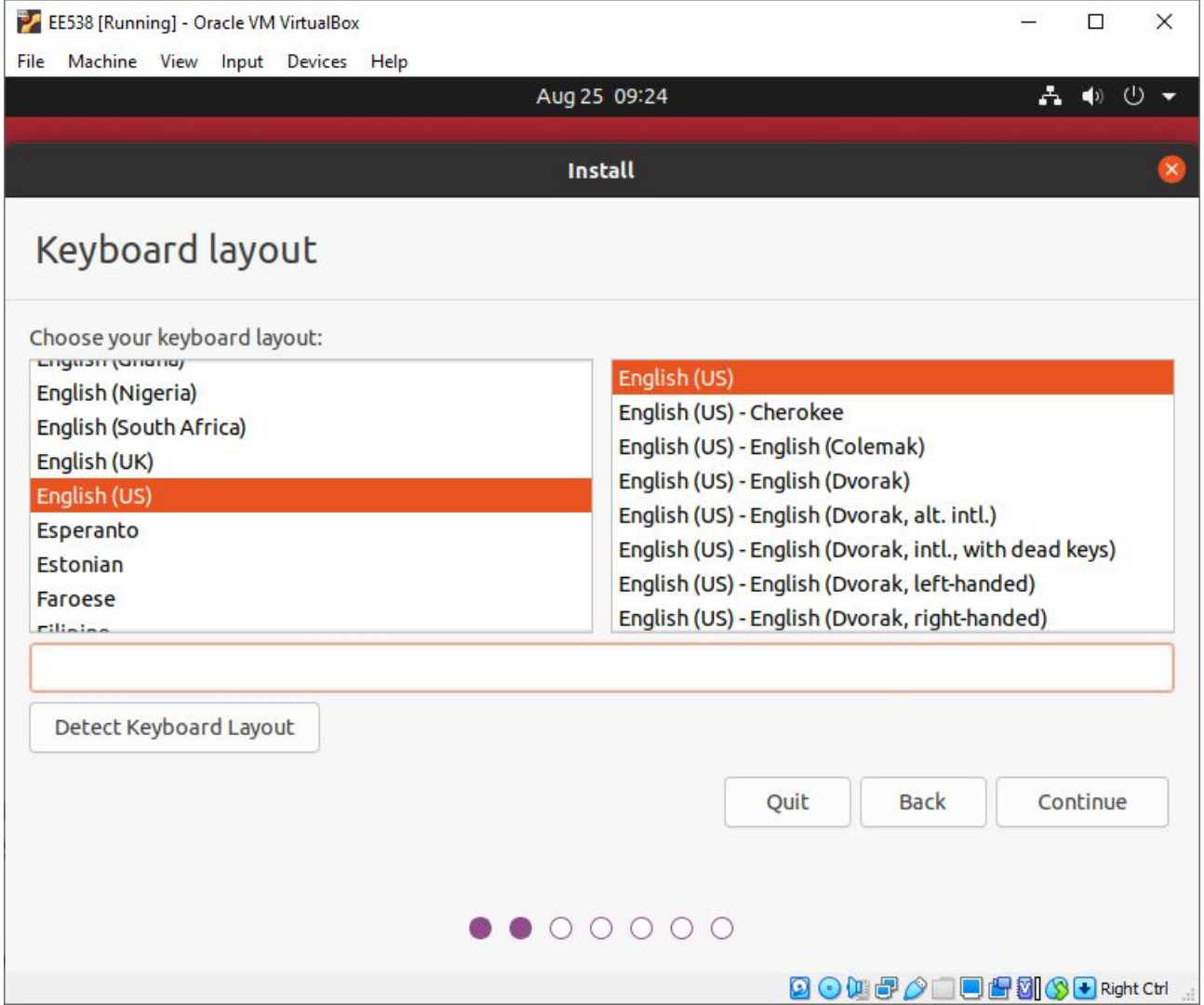
Install Ubuntu

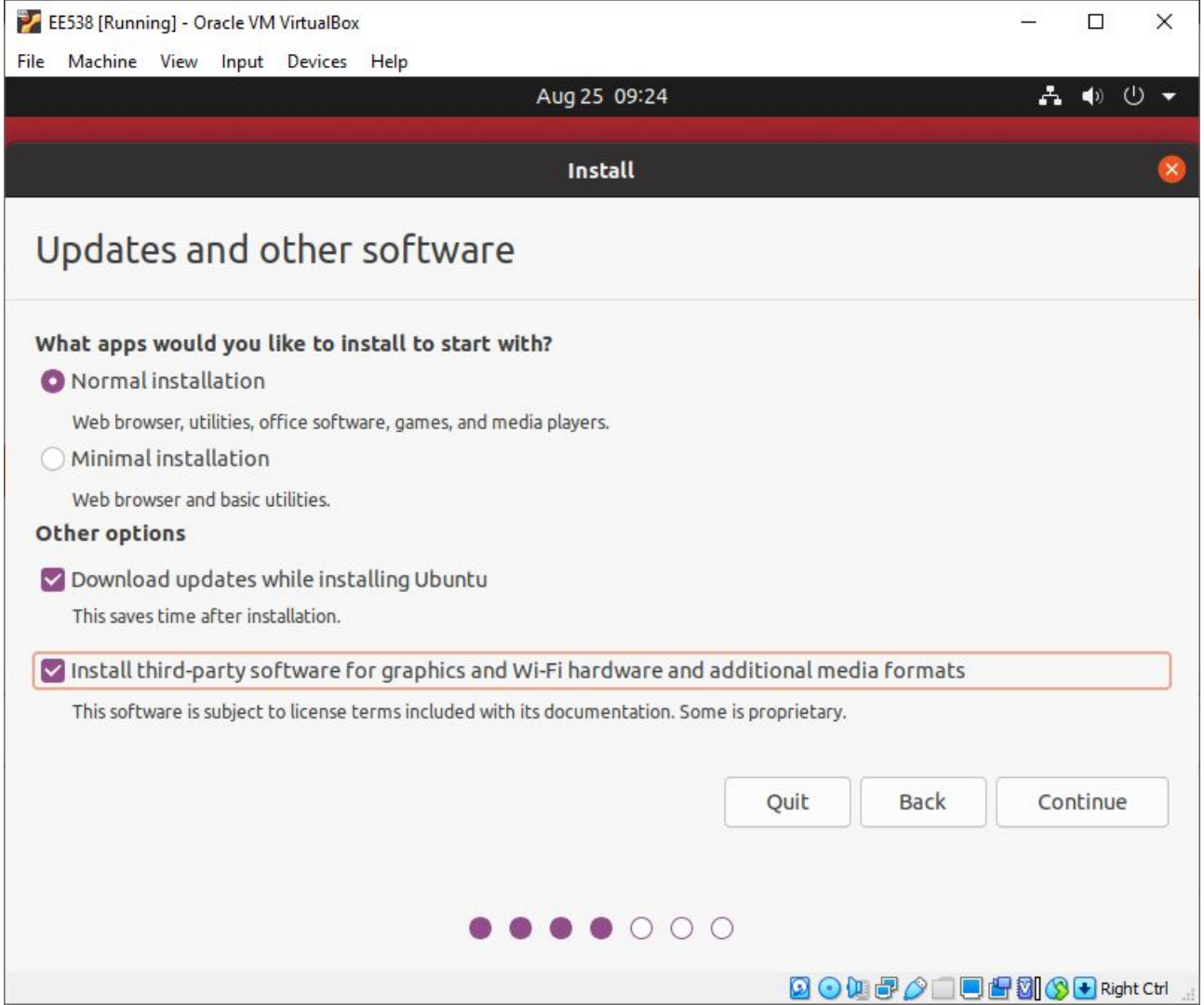
You can try Ubuntu without making any changes to your computer, directly from this CD.

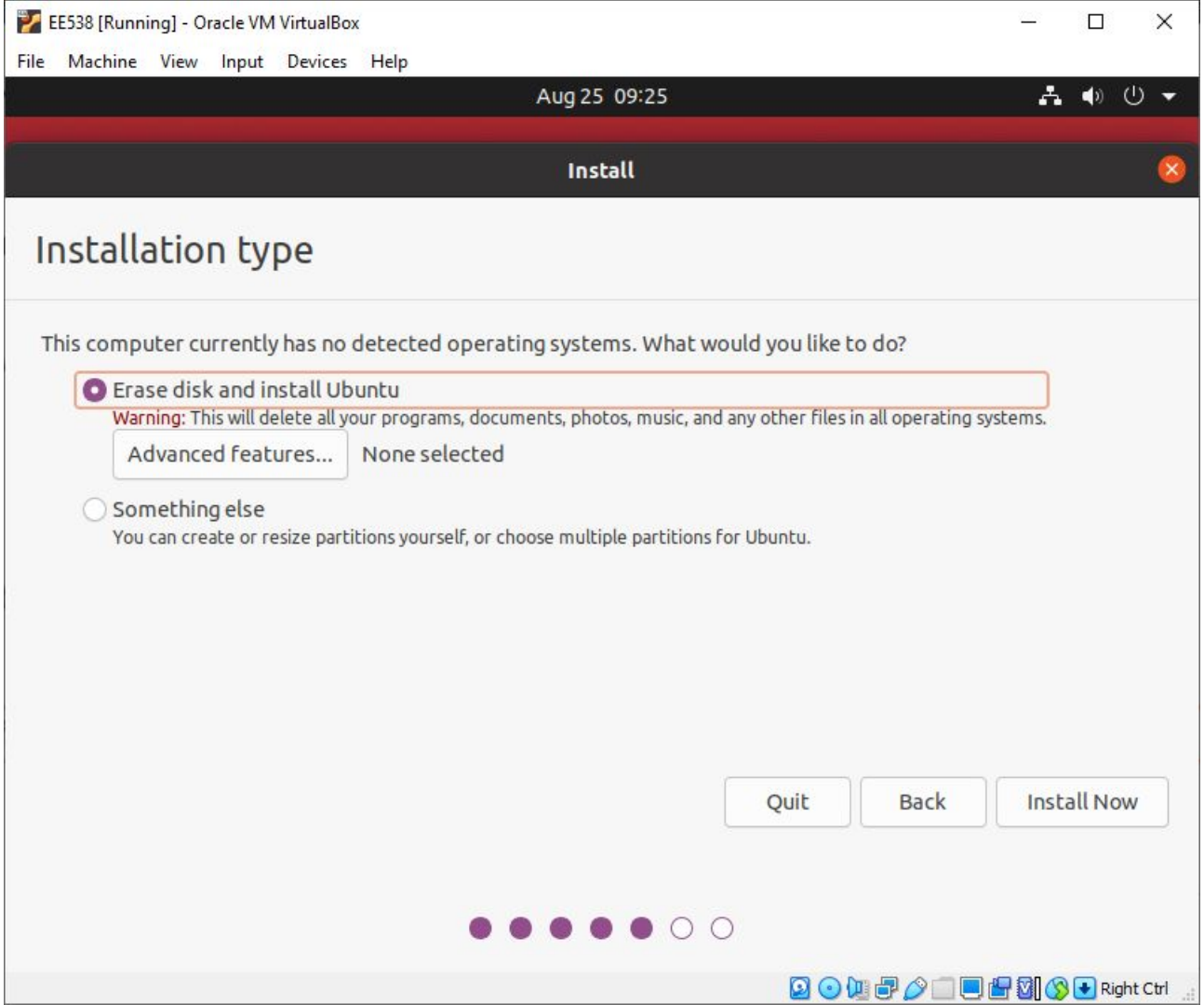
Or if you're ready, you can install Ubuntu alongside (or instead of) your current operating system. This shouldn't take too long.

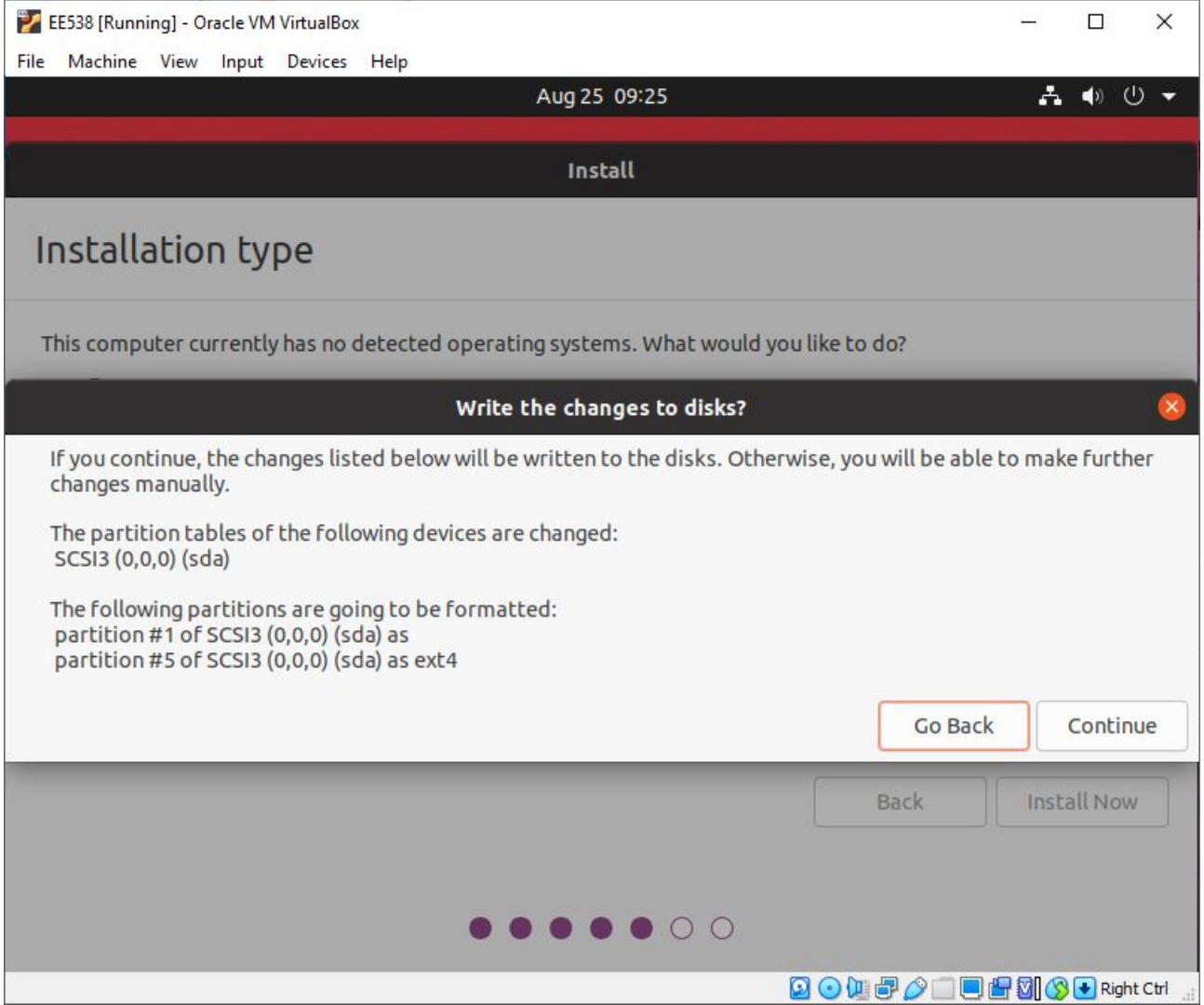
You may wish to read the release notes.

Right Ctrl


EE538 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 25 09:24
Install
Keyboard layout
Choose your keyboard layout:
English (Nigeria)
English (South Africa)
English (UK)
English (US)
Esperanto
Estonian
Faroese
English (US)
English (US) - Cherokee
English (US) - English (Colemak)
English (US) - English (Dvorak)
English (US) - English (Dvorak, alt. intl.)
English (US) - English (Dvorak, intl., with dead keys)
English (US) - English (Dvorak, left-handed)
English (US) - English (Dvorak, right-handed)
Detect Keyboard Layout
Quit
Back
Continue
Right Ctrl


EE538 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 25 09:24
Install
Updates and other software
What apps would you like to install to start with?
Normal installation
Web browser, utilities, office software, games, and media players.
Minimal installation
Web browser and basic utilities.
Other options
Download updates while installing Ubuntu
This saves time after installation.
Install third-party software for graphics and Wi-Fi hardware and additional media formats
This software is subject to license terms included with its documentation. Some is proprietary.
Quit
Back
Continue
Right Ctrl


EE538 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 25 09:25
Install
Installation type
This computer currently has no detected operating systems. What would you like to do?
Erase disk and install Ubuntu
Warning: This will delete all your programs, documents, photos, music, and any other files in all operating systems.
Advanced features...
None selected
Something else
You can create or resize partitions yourself, or choose multiple partitions for Ubuntu.
Quit
Back
Install Now
Right Ctrl


EE538 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 25 09:25
Install
Installation type
This computer currently has no detected operating systems. What would you like to do?
Write the changes to disks?
If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.
The partition tables of the following devices are changed:
SCSI3 (0,0,0) (sda)
The following partitions are going to be formatted:
partition #1 of SCSI3 (0,0,0) (sda) as
partition #5 of SCSI3 (0,0,0) (sda) as ext4
Go Back
Continue
Back
Install Now
Right Ctrl

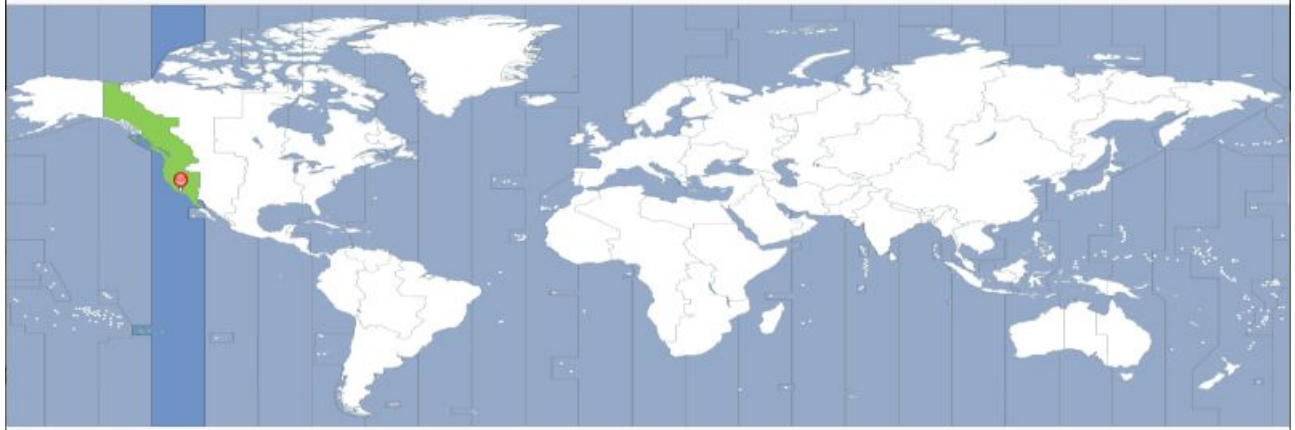
EE538 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Aug 25 09:25

Install

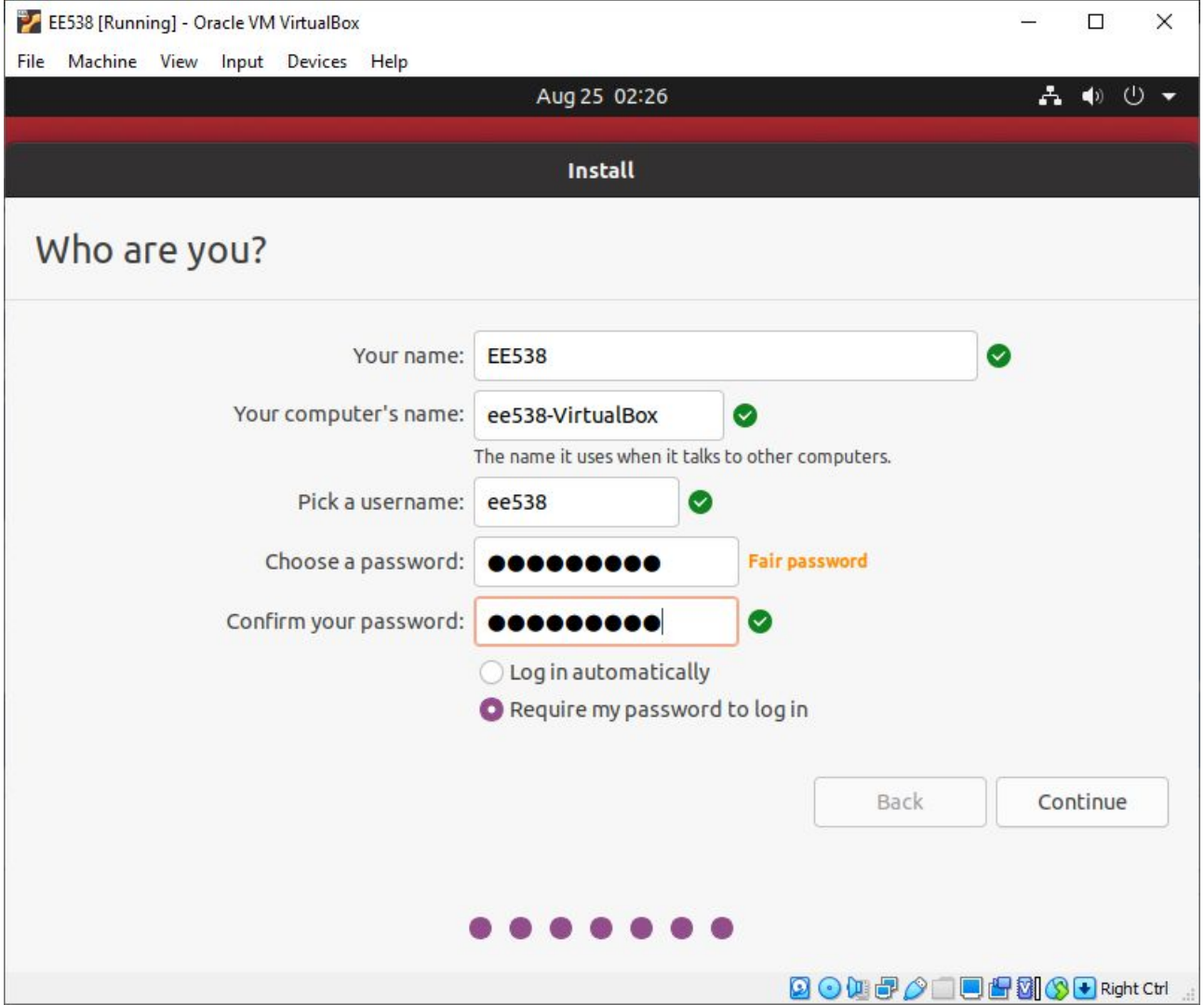
Where are you?

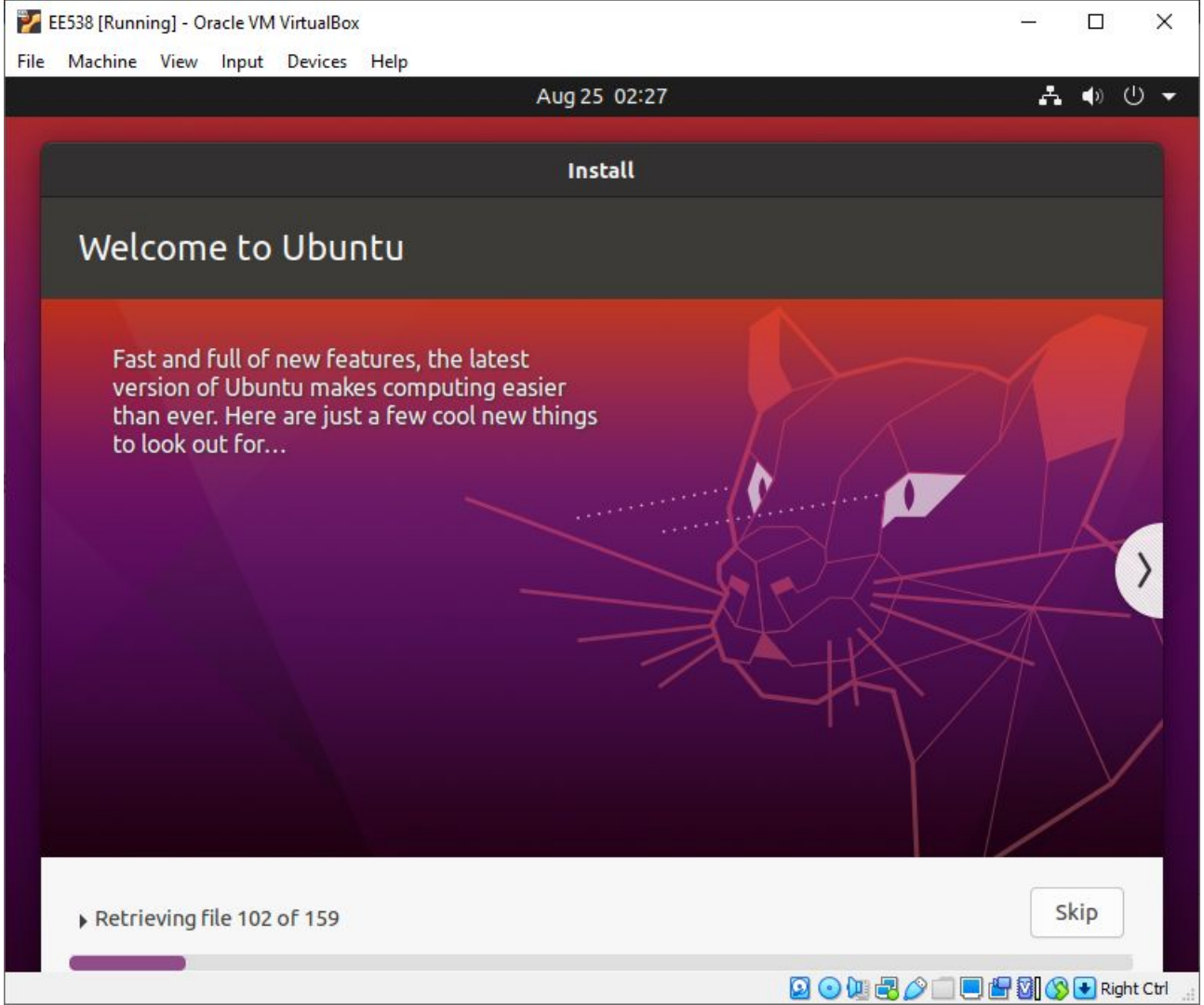


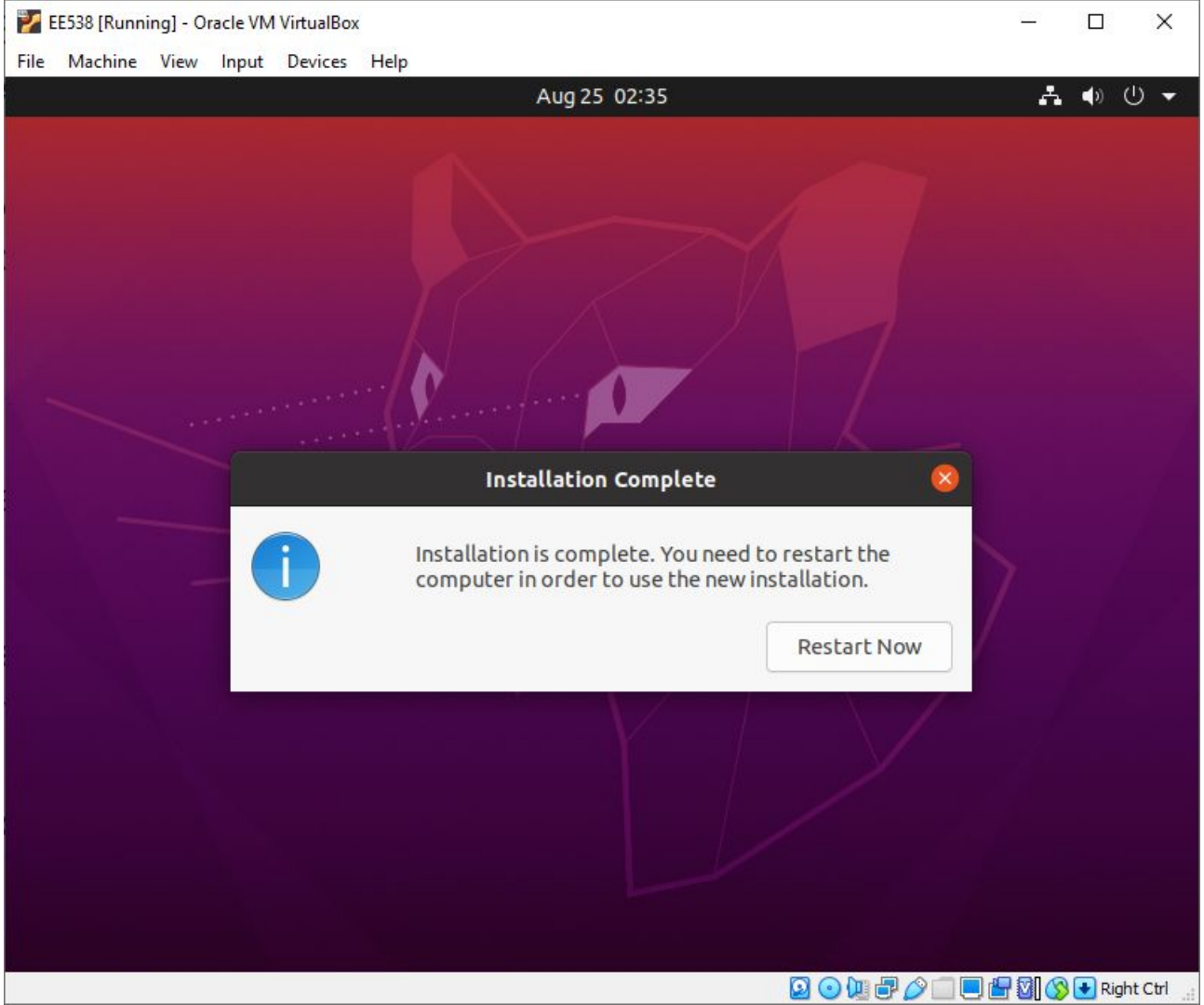
Los Angeles

Back Continue

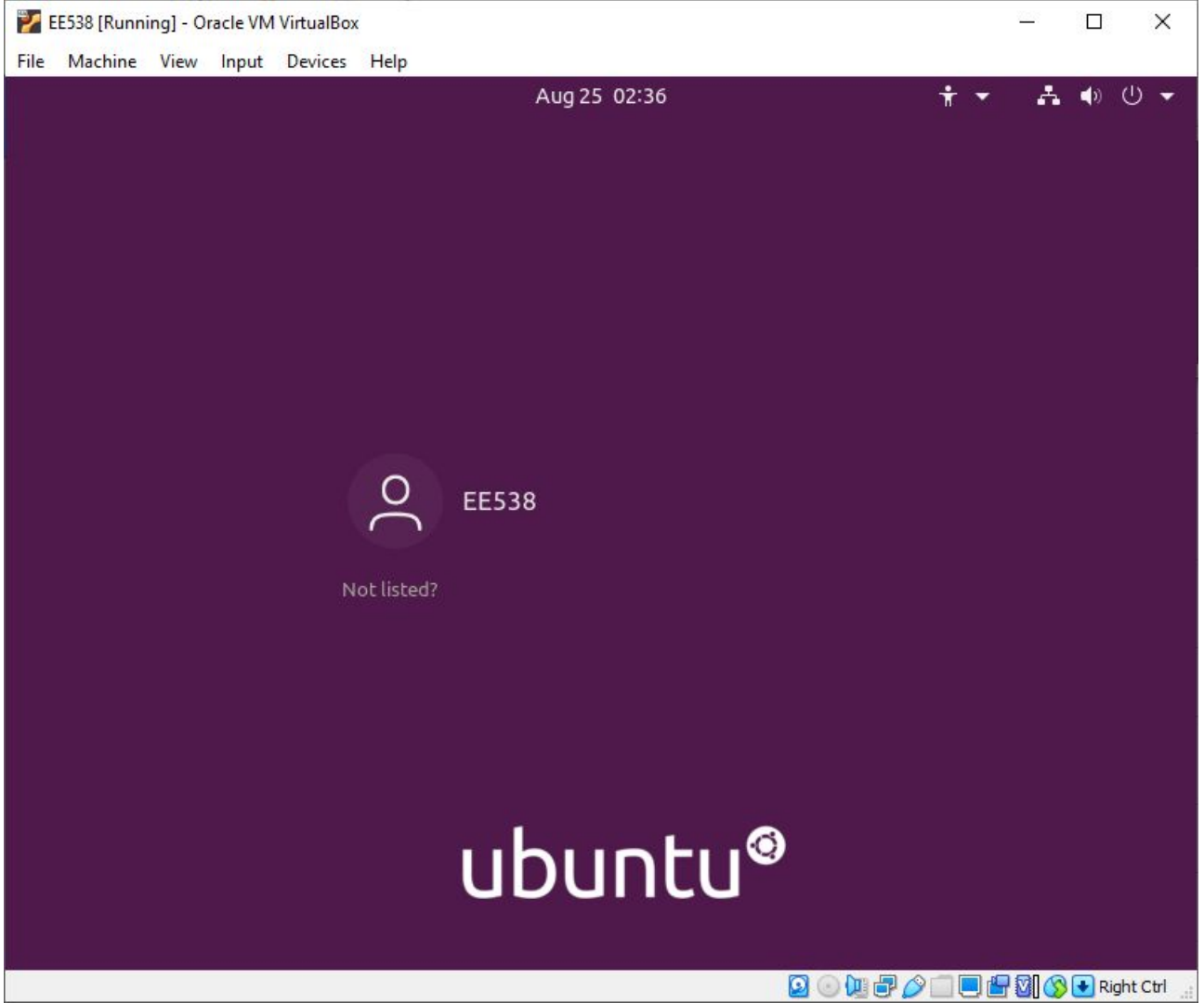
Right Ctrl


EE538 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 25 02:26
Install
Who are you?
Your name: EE538
Your computer's name: ee538-VirtualBox
The name it uses when it talks to other computers.
Pick a username: ee538
Choose a password: ●●●●●●●●●
Fair password
Confirm your password: ●●●●●●●●●
Log in automatically
Require my password to log in
Back
Continue
Right Ctrl


EE538 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 25 02:27
Install
Welcome to Ubuntu
Fast and full of new features, the latest version of Ubuntu makes computing easier than ever. Here are just a few cool new things to look out for…
Retrieving file 102 of 159
Skip
Right Ctrl


EE538 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 25 02:35
Installation Complete
Installation is complete. You need to restart the computer in order to use the new installation.
Restart Now
Right Ctrl

You are done!


EE538 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Aug 25 02:36
EE538
Not listed?
ubuntu
Right Ctrl

Programming Practice - Find Average

- Steps:
  - 1. Declare a function

double FindAverage(std::vector<int> &inputs) in src/lib/cpplib.h

  - 2. Define and implement FindAverage in src/lib/cpplib.cc
  - 3. Use your FindAverage in src/main/main.cc
  - 4. Try ***bazel run src/main:main***
- Tests:
  - 5. Add a test case in tests/cpplib_test.cc
  - 6. Try ***bazel test tests:cpplib_test***

Programming Practice - Find Average

```
class CPPLib {
public:
 // A very first question: Find Average in an vector
 double FindAverage(std::vector< int> &inputs);

};
```

Programming Practice - Find Average

```
double CPPLib::FindAverage(std::vector< int> &inputs) {
 if (inputs.size() == 0) {
   return -1;
 }
 double result = 0;
 for (auto n : inputs) {
   result += n;
 }
 return result / (inputs.size());
}
```

Programming Assignment - Find Average

```
TEST(CPPLibTest, FindAverage) {
 CPPLib cpplib;
 std::vector<int> inputs = {1, 2, 3, 4, 5};
 EXPECT_EQ(cpplib.FindAverage(inputs), 3);
}
```