ECE 448/528 - Guide to System Setup and Workflow

Prepared by Jia Wang, Aug. 2014
Updated by Jia Wang, Sep. 2014
Updated by Jia Wang, Aug. 2015
Updated by Jia Wang, Aug. 2016
Updated by Jia Wang, Aug. 2017
Adapted by Jia Wang, Dec. 2019
Updated by Jia Wang, Jan. 2021
Updated by Jia Wang, Jan. 2022
Updated by Won-Jae Yi, Jan. 2023

DISCLAIMER

Please read this disclaimer completely before continuing. The software packages may ask you to disable your anti-virus software during installation -- you will not be protected from viruses in such a situation. Use this document at your own risk. Illinois Tech disclaims any liability for any actions or results of the following documentation.

Since a network failure may interrupt your operation, please save your data often.

Table of Contents

I. Overview

Welcome to ECE448/528!

This document presents a guide to the tools that are essential for our software development process. As we focus on the system and the workflow that utilize these tools to complete our projects, you must follow the links provided to learn more about them to work more effectively in this course.

Please be advised that this is a long document, but you have to read and follow it word-by-word to survive the course. Failure to do so may lead to ZERO for your project grades, especially for the emphasized paragraphs like this one.

We follow the current industrial practice to utilize software development tools and processes that integrate well with the Linux Operating System (OS). To save you time setting up such a system by yourself, we provide a Virtual Machine (VM) containing a full-featured Linux installation with all the necessary tools that can run on most OSes. Moreover, an ECE Unix System (Uranus.ece.iit.edu) is used to provide access to a Git-based code repository to store and submit your project codes.

That being said, you will need to get familiar with Linux and Git at the beginning of the course. **Here are the tutorials you should read and follow. You will NOT survive the course if you are not familiar with these tools.**

1. Basic Linux usage: Chapters 1 to 8 from Linux Tutorial.
2. Basic Git usage: Chapters 1 and 2 from Pro Git.

Be patient! You will be able to work comfortably with them by the end of the semester and for the years to come.

Please be advised that running software development tools smoothly in general (and in particular our VM) will require you to use a reasonably powerful computer (desktop or laptop). **Please make sure your** computer **is able to run VirtualBox. We strongly recommend a recent computer with solid-state drive(s) as well as at least 16 GB RAM and a quad-core processor.** You may need to adjust our VM setting if your configuration does not meet the recommended one.

Feel free to email me any questions you may have. You may also find resources online to help you, in particular from Stack Overflow and YouTube after searching from Google, etc. A recommended way to learn is to follow the next few sections to access either the VM or the ECE Unix System and to practice them there.

As a summary of the whole document, here are the most important hints.

1. You are expected to work on the projects on a regular basis. Commit and push your code at least once at the end of the day. **Lost of code due to computer/VM crashes just before the deadline is NOT an excuse for late project submissions, unless there is a recent push from you.**
2. Your Illinois Tech account is used to connect to the IIT VPN. Your ECE account, issued from ECE Department, is only used to access the ECE Unix System (Uranus.ece.iit.edu). It is your responsibility to maintain these accounts as well as your Internet connection and to let us know promptly of any related issues. **Not being able to access the ECE Unix System just before the deadline (e.g. because you forget your password or your Internet connection is not available) is NOT an excuse for late project submissions, unless there is a recent push from you.**
3. You should utilize the provided grading programs to access your project grading reports and to correct any errors to earn full points BEFORE the deadlines. **Re-submission and re-grading are NOT allowed after the deadlines.**
4. **Do NOT upgrade/update your VM and any software within.** Turn off your VM using the system tray in its top-right corner to avoid disk corruption. You can always import the VM to start from scratch again, assuming your code has been committed and pushed to the central Git repository on a timely basis.
5. Support for using multiple copies of the VM (e.g. one on your laptop and one on your desktop) and using your own IDE are not provided in this course. **Any issue resulting from so is NOT an excuse for late project submissions.** Use them at your own risk if you are very familiar with system administration and development tools.

II. Remote Access to ECE Unix System

A. VM Setup

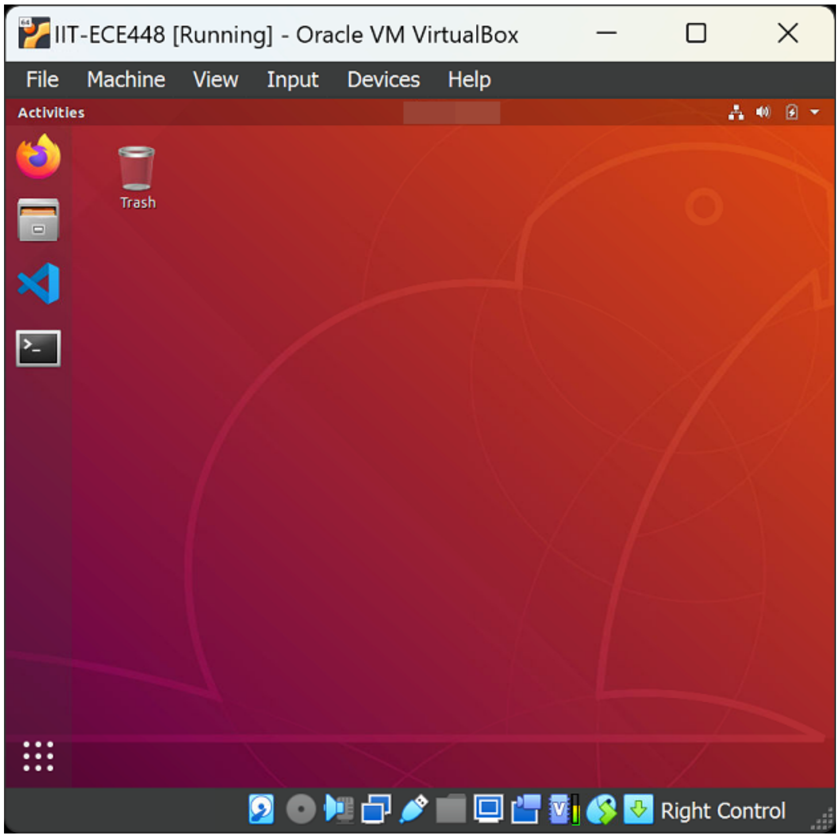
To use our VM, first install the most recent version of VirtualBox on your computer. You will need to turn on hardware virtualization if you haven't done so already.

Download our VM Appliance IIT-ECE448.ova and double-click to install/import. When asked to configure the size of the base memory, you should put it lower than your computer's available RAM, so that your main OS does not become low on memory. For the number of CPU cores, I would recommend putting 2 or 4 cores depending on your computer hardware configuration. You may go higher if you would like. Lastly, verify the integrity if needed. Note that all these steps may take a while to finish so please be patient.

- MD5: a84a5c6380c6018d4c235eae5e1a4653
- SHA-256: e0bf0bee0d592f0707edc4221769db0011cfd1928f2c8f5fd3429e6117a21bac

Start the VM. After a while, it will display the login screen. Log in here to access the Graphical User Interface (GUI) below using username 'ubuntu' and password 'ece448'.

At first, the VM window size would be too small for you to work on. Feel free to adjust the VirtualBox window and the VM window will automatically adjust the size for you.


IIT-ECE448 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Trash
Right Control

The GUI should be intuitive to use. In particular, the 4 large icons on the left allow quick access to the 4 most used applications for this course: FireFox (web browser), File Browser, VS Code, and Terminal. **If the GUI locks itself by displaying a clock, you may simply press any key to unlock it. (like unlocking a lockscreen of your phone)**

If by any chance the VM stops working in the future, you may install/import the VM again and follow the next few subsections of this guideline to re-setup central and local Git repositories on this VM. **You'll be able to recover all the code you have pushed to the central Git repository, though all the code you have NOT pushed to the central Git repository will be lost. Thus, it is VERY important for you to read and follow Section IV.B Git Workflow.**

B. Connecting to IIT VPN

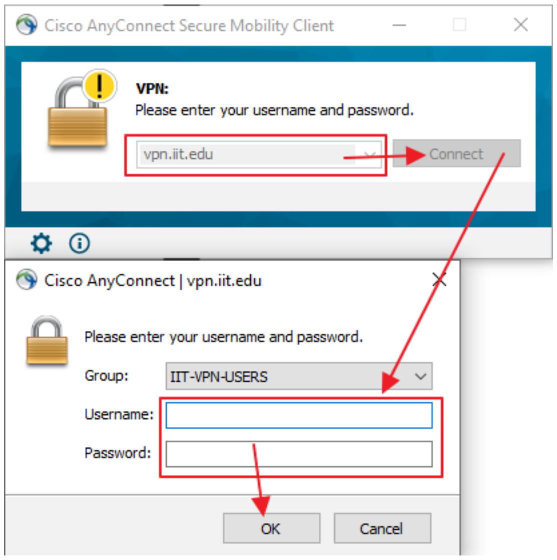
To access the ECE Unix System from outside of the Illinois Tech campus, you will need to connect to IIT-VPN first. If you do not have access to IIT-VPN, you will not be able to connect to the ECE Unix System outside of the Illinois Tech campus network. Please first familiarize yourself with IIT VPN/Remote Access Procedure and Policy.

There are two methods that you can connect to IIT-VPN: using your main OS to connect to IIT-VPN, or using your VM to connect to IIT-VPN. If you choose to connect to IIT-VPN from your main OS, your entire connection will be connected to IIT-VPN. If you choose to connect to IIT-VPN from your VM, only the VM's network connection will be connected to IIT-VPN. It is up to you how the best preferred method to connect to the VPN.

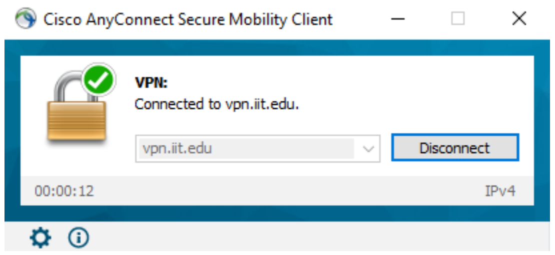
i) Connecting to IIT-VPN from your main OS

Visit https://ots.iit.edu/network-infrastructure/vpn-remote-access and follow instructions according to your computing environment including downloading the IIT VPN software. **Please contact Support Desk directly at supportdesk@iit.edu for further assistance**

Once installed, execute Cisco AnyConnect from your computer. In the address field, enter *vpn.iit.edu*. When prompted for Username and Password, enter your IIT Hawk username (without @hawk.iit.edu) and password. It will perform an update in the first login process.


Cisco AnyConnect Secure Mobility Client
VPN:
Please enter your username and password.
vpn.iit.edu
Connect
Cisco AnyConnect | vpn.iit.edu
Please enter your username and password.
Group:
IIT-VPN-USERS
Username:
Password:
OK
Cancel

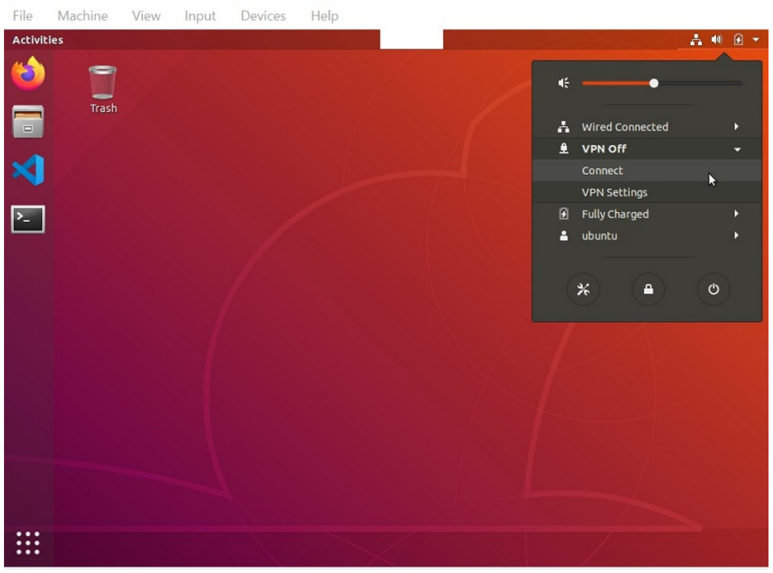
If the connection is successful, you'll be able to connect to the ECE Unix System.


Cisco AnyConnect Secure Mobility Client
VPN:
Connected to vpn.iit.edu.
vpn.iit.edu
Disconnect
00:00:12
IPv4

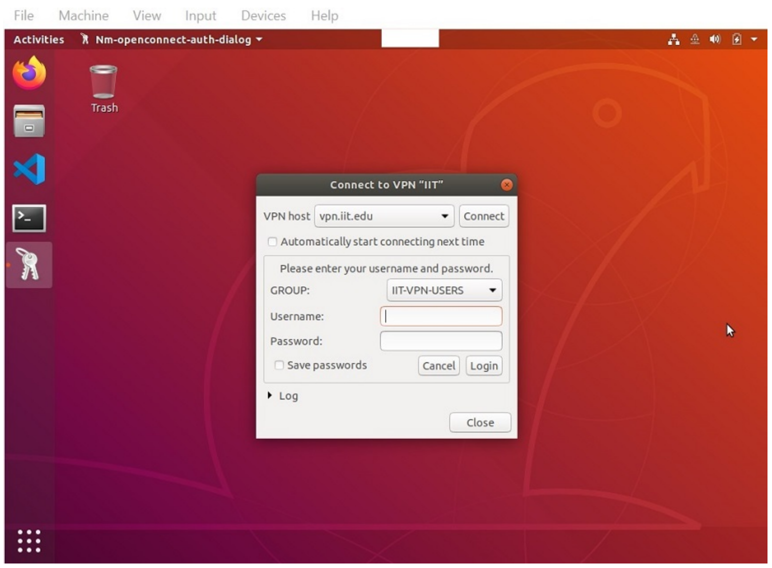
In any case the program alerts you that the login does not work, please open up a ticket at OTS SupportDesk by emailing supportdesk@iit.edu. OTS will help your ID to work for VPN.

ii) Connecting to IIT-VPN from your VM

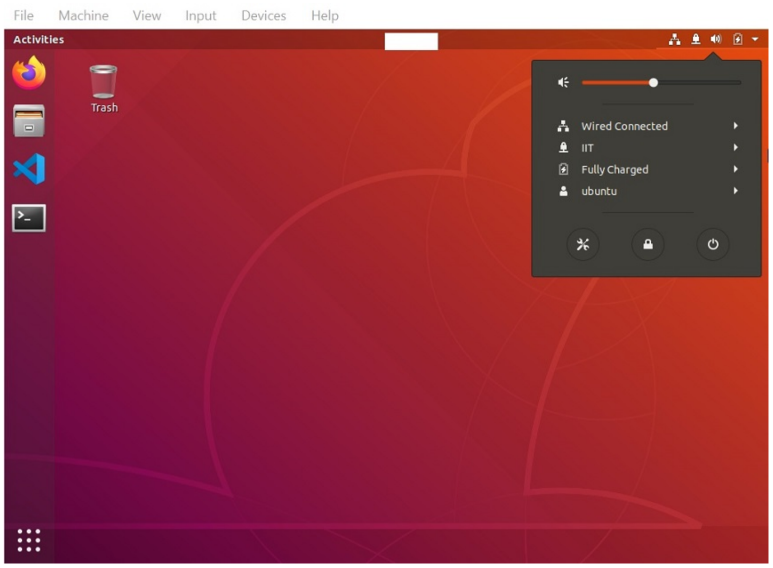
Click the top-right corner of the VM to bring up the system tray. VPN is currently off as indicated. Click "Connect" under "VPN Off".



Now you should be able to see the login dialog box. Click the "Connect" button to connect to "vpn.iit.edu". Make sure the GROUP is IIT-VPN-USERS. Input your **Illinois Tech username and password** and then click the "Login" button. **This is your Illinios Tech account to access your Illinois Tech email, myIIT, Blackboard, etc.** Do NOT use your ECE account here.



After a while, the IIT-VPN should be connected as confirmed from the system tray.

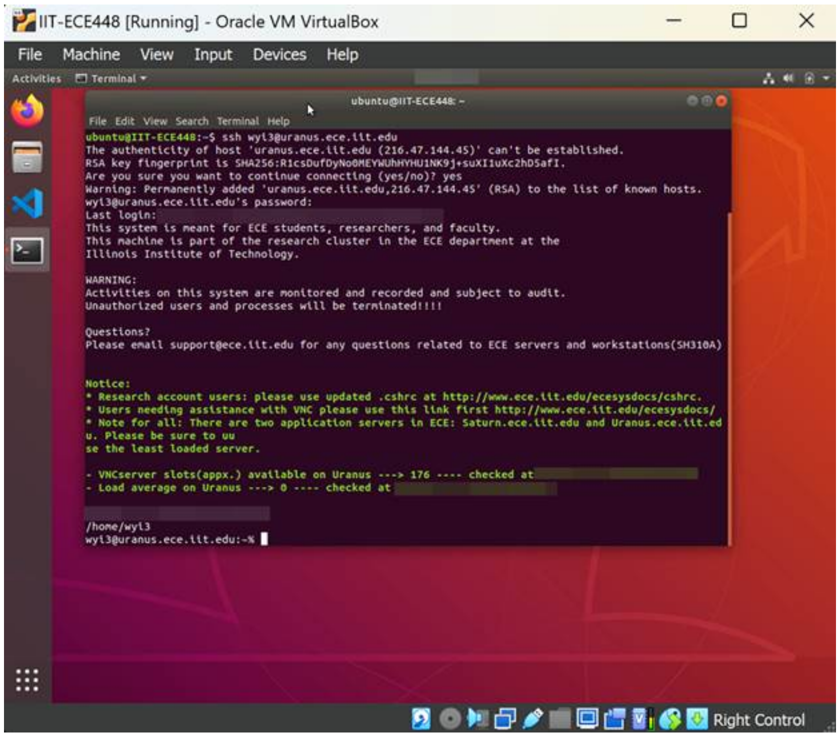


C. Remote Access Using SSH

Your ECE account, which is different than your Illinois account, will be created at the beginning of the semester for you to access the ECE Unix System. Your ECE account information including your ECE username and password will be sent to you via a separate email.

Once you have received the email containing your ECE account information, you may access the ECE Unix System to work on your project from our provided VM assuming the Internet connection is available.

Make sure the IIT-VPN is connected. Open a Terminal window and execute the command **ssh username@uranus.ece.iit.edu** from there. Answer "yes" when asked to verify RSA key fingerprint. Wait a while until it prompts you for your password. **Type your given default password and then hit enter ("*" will not be displayed).** Don't forget to use your ECE username instead of mine.



```
ubuntu@IIT-ECE448:~$ ssh wyi3@uranus.ece.iit.edu
The authenticity of host 'uranus.ece.iit.edu (216.47.144.45)' can't be established.
RSA key fingerprint is SHA256:R1csDufDyNo0MEYWUhHYHU1NK9j+suXI1uXc2hD5afI.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'uranus.ece.iit.edu,216.47.144.45' (RSA) to the list of known hosts.
wyi3@uranus.ece.iit.edu's password:
Last login: [redacted]
This system is meant for ECE students, researchers, and faculty.
This machine is part of the research cluster in the ECE department at the
Illinois Institute of Technology.

WARNING:
Activities on this system are monitored and recorded and subject to audit.
Unauthorized users and processes will be terminated!!!!

Questions?
Please email support@ece.iit.edu for any questions related to ECE servers and workstations(SH310A)


Notice:
* Research account users: please use updated .cshrc at http://www.ece.iit.edu/ecesysdocs/cshrc.
* Users needing assistance with VNC please use this link first http://www.ece.iit.edu/ecesysdocs/
* Note for all: There are two application servers in ECE: Saturn.ece.iit.edu and Uranus.ece.iit.ed
u. Please be sure to uu
se the least loaded server.

- VNCserver slots(appx.) available on Uranus ---> 176 ---- checked at [redacted]
- Load average on Uranus ---> 0 ---- checked at [redacted]

[redacted]
/home/wyi3
wyi3@uranus.ece.iit.edu:~%
```

You are now connected to Uranus ECE Server and can execute commands. **The very first command you should execute is *passwd* which allows you to reset your ECE password.**

```
wyi3@uranus:/home/wyi3% passwd
Changing password for user wyi3.
Enter login(LDAP) password:
```

III. Git Repository Setup

A. Key Setup for Central Git Repository

All your source code should go to our central Git repository at the end of the day. This practice provides a lot of benefits while requiring a minimum effort from you, some being listed below,

- Your code won't be lost if your VM, laptop, etc. fails.
- We can easily access your code to solve any issue and for grading.

For security reasons, although our central Git repository is also located within the ECE Unix System, it relies on Public Key Infrastructure (PKI) for authentication and authorization. A pair of public/private keys need to be created for use in our central Git repository.

Click the web browser icon in our VM GUI to start the web browser and open this page. Right click setup_ece448_uranus and choose 'Save Link As...' to save it to your home directory. Click the Terminal icon in our VM GUI to start a Terminal. You should be in your home directory. Use **chmod +x setup_ece448_uranus** to mark the file as executable, and then use **ls -l setup_ece448_uranus** to make sure the command to setup the keys exist.

```
ubuntu@IIT-ECE448:~$ chmod +x setup_ece448_uranus
ubuntu@IIT-ECE448:~$ ls -l setup_ece448_uranus
-rwxrwxr-x 1 ubuntu ubuntu   387 Aug 12 00:25 setup_ece448_uranus
```

Make sure the IIT-VPN is connected. Now, execute **setup_ece448_uranus** with your ECE account username (replace wyi3 with your ECE account username). Input the password of your ECE account when prompted.

```
ubuntu@IIT-ECE448:~$ ./setup_ece448_uranus wyi3
Generating public/private rsa key pair.
Your identification has been saved in /home/ubuntu/.ssh/id_rsa.
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub.
The key fingerprint is:
******** wyi3@ece448
wyi3@uranus.ece.iit.edu's password:
Locating record for wyi3 ...
Receiving key for wyi3 ...
ssh-rsa ********
Updating system for wyi3 ...
Your key wyi3.pub has been updated.
Initialize repository for wyi3 ...
You may clone your repository as below now.
  git clone ece448@uranus.ece.iit.edu:wyi3 iot_ece448
```

You might see a slightly different output as I have hidden my key and the command may take a while to complete.

Here are a few notes regarding your keys.

- Only one public/private key pair is supported. **You are discouraged from using multiple VMs (e.g. one on your laptop and another on your desktop).**
- If you already have a public/private key pair, you may use them by answering 'no' when being asked 'Overwrite (y/n)?'.
- Your public key is stored in the file '~/.ssh/id_rsa.pub' and your private key is stored in the file '~/.ssh/id_rsa', where '~' is a shortcut to your home directory.
- It is safe to pass the public key to other people but you should keep the private key yourself.
- Ideally, you only need to run the above step once at the beginning of the semester.
- If by any chance you feel your private key is stolen, you should run the step again to replace your previous key pair (answering 'yes' when being asked 'Overwrite (y/n)?').

B. Local Git Repository Setup

Git is a distributed version control system where a repository is mirrored locally in addition to being stored on a server (e.g., in our central Git repository). Thus, it is necessary for you to setup your local Git repository before you can start working on the projects. As we would also like to provide you with an initial project package containing useful files, we request you to setup your local Git repository by cloning from our central Git repository.

As you may have noticed at the end of the previous subsection, the command to be used is **git clone ece448@uranus.ece.iit.edu:*username* iot_ece448**. Make sure the IIT-VPN is connected.

```
ubuntu@IIT-ECE448:~$ git clone ece448@uranus.ece.iit.edu:wyi3 iot_ece448
Cloning into 'iot_ece448'...
remote: Counting objects: 50, done.
remote: Compressing objects: 100% (43/43), done.
remote: Total 50 (delta 5), reused 0 (delta 0)
Receiving objects: 100% (50/50), 16.83 KiB | 957.00 KiB/s, done.
Resolving deltas: 100% (5/5), done.
```

!!!!STOP TYPING ANY PASSWORD!!!! if you were asked for the password of ece448@uranus.ece.iit.edu as shown below, otherwise our servers may block your access for a few hours. This is prompted if you have forgotten to replace my username 'wyi3' with yours. Please repeat Section III.B Key Setup for Central Git Repository first and try to clone again. If you are still asked for the password, contact the course instructor for help.

```
ubuntu@IIT-ECE448:~$ git clone ece448@uranus.ece.iit.edu:wyi3 iot_ece448
Cloning into 'iot_ece448'...
ece448@uranus.ece.iit.edu's password:
```

You can then verify that you downloaded source codes from our central Git repository.

```
ubuntu@IIT-ECE448:~$ cd iot_ece448
ubuntu@IIT-ECE448:~/iot_ece448$ ls
build.gradle    iot_ece448.code-workspace  simConfigEx.json  src
hubConfig.json  public                     simConfig.json
```

Here are a few notes regarding cloning:

- Ideally, you only need to clone once at the beginning of the semester.
- If by any chance you feel your local repository 'iot_ece448' is corrupted, you should first **mv** it to a different directory as it may contain files that haven't been sent to our central Git repository yet. Then you could clone again.

C. Java and Gradle

Most of our code will be written in Java and we use a build tool named Gradle to manage all aspects of Java development including third-party libraries, build and test, as well as grading. While Java and Gradle are already installed in the VM, it is a good time to verify that they work properly. Execute **gradle** from 'iot_ece448'. Note that it takes a while for Gradle to download Java libraries for the first time before it will build and test your projects.

```
ubuntu@IIT-ECE448:~/iot_ece448$ gradle
Starting a Gradle Daemon (subsequent builds will be faster)
Download ********
> Task :test

ece448.iot_sim.PlugSimTests > testSwitchOn FAILED
    java.lang.AssertionError at PlugSimTests.java:22

2 tests completed, 1 failed


FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':test'.
> There were failing tests. See the report at: file:///home/ubuntu/iot_ece448/build/reports/tests/test/index.html

* Try:
Run with --stacktrace option to get the stack trace. Run with --info or --debug option to get more log output. Run with --

* Get more help at https://help.gradle.org

BUILD FAILED in 52s
5 actionable tasks: 4 executed, 1 up-to-date
```

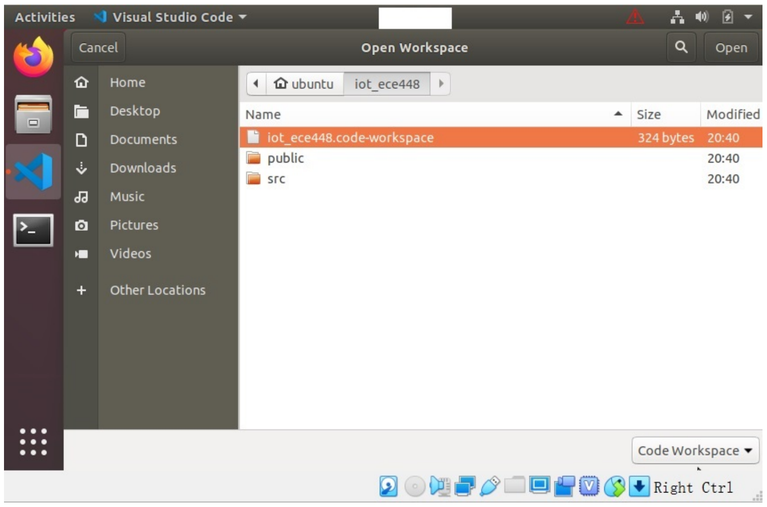
Read the messages and you will find that the test fails: the messages even tell where the failed test case is located. You will need to modify the Java code to pass the test before your project could be graded.

IV. VS Code and Git Work Flow

A. VS Code

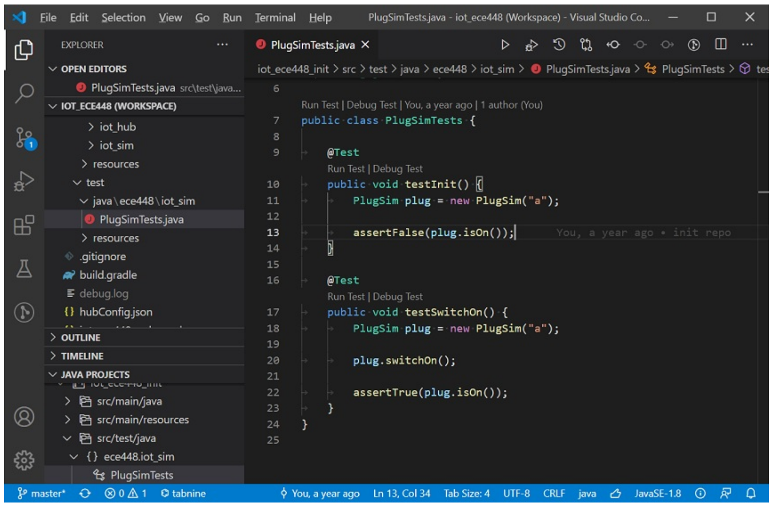
Visual Studio Code (VS Code) is a code editor that can be extended into a powerful IDE by third-party extensions. Our VM comes with VS Code and necessary extensions to support Java and web application development. Feel free to add other extensions to support your development as necessary.

Start VS Code. Click 'File' and 'Open Workspace...'. Then, navigate to the directory 'iot_ece448' and select 'iot_ece448.code-workspace' to open our project workspace.



Name	Size	Modified
iot_ece448.code-workspace	324 bytes	20:40
public		20:40
src		20:40

The newly added 'JAVA PROJECTS' in the left panel indicates that the project is initiated successfully. You may now navigate to the failed test case and try to reason why it fails. Feel free to play with the test case by clicking 'Run Test' or 'Debug test' right above line 17.



```
 6
   Run Test | Debug Test | You, a year ago | 1 author (You)
 7 public class PlugSimTests {
 8
 9     @Test
       Run Test | Debug Test
10     public void testInit() {
11         PlugSim plug = new PlugSim("a");
12
13         assertFalse(plug.isOn());        You, a year ago • init repo
14     }
15
16     @Test
       Run Test | Debug Test
17     public void testSwitchOn() {
18         PlugSim plug = new PlugSim("a");
19
20         plug.switchOn();
21
22         assertTrue(plug.isOn());
23     }
24 }
25
```

If you prefer to use terminals within VS Code instead of as GUI applications, you may do so by clicking 'Terminal' and then 'New Terminal' from the menu.

B. Git Workflow

A workflow is a series of steps that you follow to complete certain tasks. As these steps may need to be executed at different frequencies and be combined differently for various stages of that task, you must not just follow them blindly - try to understand their purposes to use them effectively. If there is any issue, you need to read the output messages carefully and follow their instructions.

Here is a typical Git workflow that you can iterate multiple times a day.

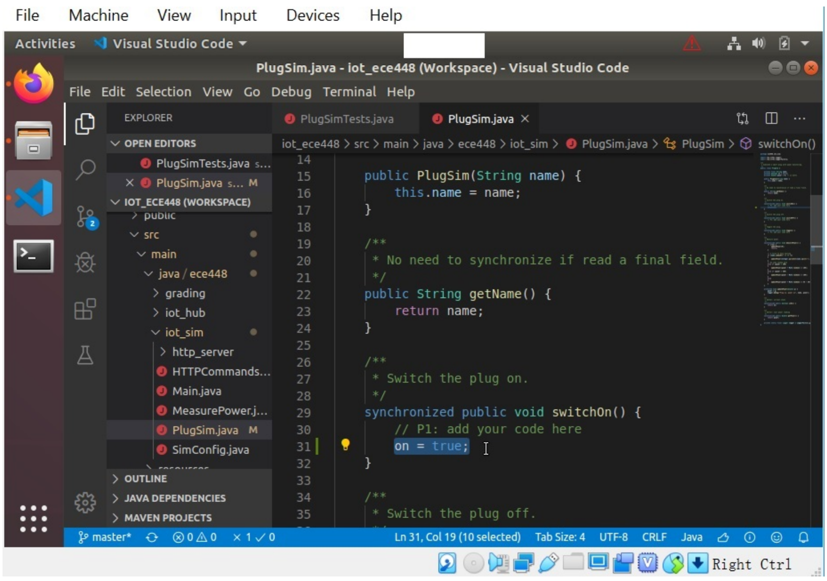
1. Pull changes from the central Git repository. (git pull)
2. Modify and add source files and test cases. (git add)
3. Build and test your project locally. (gradle)
4. Commit changes locally and push them to the central Git repository. (git commit)

Details are as follows:

Before you begin to work on your local repository every day, you should pull changes from our central Git repository first. This is necessary as we may update our central Git repository occasionally. To pull the changes, make sure the IIT-VPN is connected and use the command **git pull**. Depending on whether there are any changes, your output message could be different.

```
ubuntu@IIT-ECE448:~/iot_ece448$ git pull
Already up to date.
```

Now, you probably have figured out why the test case fails and decided to modify the function 'ece448.iot_sim.PlugSim.switchOn' so the test case may pass.


File Machine View Input Devices Help
Activities Visual Studio Code
PlugSim.java - iot_ece448 (Workspace) - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
EXPLORER
OPEN EDITORS
PlugSimTests.java
PlugSim.java M
IOT_ECE448 (WORKSPACE)
public
src
main
java / ece448
grading
iot_hub
iot_sim
http_server
HTTPCommands...
Main.java
MeasurePower.j...
PlugSim.java M
SimConfig.java
OUTLINE
JAVA DEPENDENCIES
MAVEN PROJECTS
PlugSimTests.java
PlugSim.java
iot_ece448 > src > main > java > ece448 > iot_sim > PlugSim.java > PlugSim > switchOn()
master* 0 0 1 0
Ln 31, Col 19 (10 selected) Tab Size: 4 UTF-8 CRLF Java
Right Ctrl

```java

    public PlugSim(String name) {
        this.name = name;
    }

    /**
     * No need to synchronize if read a final field.
     */
    public String getName() {
        return name;
    }

    /**
     * Switch the plug on.
     */
    synchronized public void switchOn() {
        // P1: add your code here
        on = true;
    }

    /**
     * Switch the plug off.
```

Run **gradle** again to build and test your project. There should be no problem now. Note that Gradle may download additional Java libraries for coverages for the first time.

```
ubuntu@IIT-ECE448:~/iot_ece448$ gradle
Download ********
BUILD SUCCESSFUL in 5s
6 actionable tasks: 6 executed
```

You may locate the test and coverage reports in 'build/reports' and view them by locating and opening 'index.html' using the web browser.

For your convenience, We provide Gradle grading scripts to help you to troubleshoot your code locally in your VM. For example, run **gradle grade_p1** for Project 1 as shown below.

```
ubuntu@IIT-ECE448:~/iot_ece448$ gradle grade_p1
```

```
> Task :grade_p1
2019-12-26 11:11:39,309 INFO GradeP1-testCase00: success
2019-12-26 11:11:39,314 INFO GradeP1-testCase01: success
2019-12-26 11:11:39,314 INFO GradeP1-testCase02: success
2019-12-26 11:11:39,315 INFO GradeP1-testCase03: failed
2019-12-26 11:11:39,315 INFO GradeP1-testCase04: success
2019-12-26 11:11:39,316 INFO GradeP1-testCase05: failed
2019-12-26 11:11:39,316 INFO GradeP1-testCase06: success
2019-12-26 11:11:39,322 INFO GradeP1-testCase07: success
2019-12-26 11:11:39,325 INFO GradeP1-testCase08: failed
2019-12-26 11:11:39,327 INFO GradeP1-testCase09: failed
Local Testing: 6 cases passed
**********************************************************
* You may receive 0 points unless your code tests correctly. *
* Please commit and push your code to the central Git repository.  *
**********************************************************

BUILD SUCCESSFUL in 1s
6 actionable tasks: 1 executed, 5 up-to-date
```

The above output shows that 6 out of the 10 grading cases passed for Project 1 in the VM.

We, the TA and the instructor, will use the same Gradle grading scripts to build, test, and grade your project. However, as emphasized in the above output, **any tests you run in your own VM, passed or not, do NOT count towards your project grades**. In other words, **we will only use the grading reports from our end to decide your project grades**.

It is a good time to commit changes locally and to push them to the central Git repository whenever you feel tired and decide to work on your project on another day, no matter whether all test cases are passed or not.

The Git system may ask you to set up your name and email address if you haven't done so already, which is very useful when multiple people need to work on the same repository. You can set up your name and email address by **git config**.

```
ubuntu@IIT-ECE448:~/iot_ece448$ git config --global user.name "Won-Jae Yi"
ubuntu@IIT-ECE448:~/iot_ece448$ git config --global user.email wyi3@iit.edu
```

If there are any newly added files, you must notify Git of any by **git add**. Otherwise, Git can track the modification to a file that is already in the repository, and you should save the changes to your local repository by **git commit -am** so that different revisions of the same file (e.g., its history, can be recorded).

```
ubuntu@IIT-ECE448:~/iot_ece448$ git commit -am "unit test passed"
[master 3e97106] unit test passed
 1 file changed, 1 insertion(+)
```

Don't forget to include a brief message with the commit, e.g. "unit test passed" in my example, so that later on you can easily recognize what the commit is about.

Please make sure all source files and test cases have been added and committed by git status before proceeding to the next step.

```
ubuntu@IIT-ECE448:~/iot_ece448$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commits.
  (use "git push" to publish your local commits)
```

```
nothing to commit, working tree clean
```

Send your changes to the central Git repository by **git push**. Make sure the IIT VPN is connected.

```
ubuntu@IIT-ECE448:~/iot_ece448$ git push
Counting objects: 8, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 601 bytes | 150.00 KiB/s, done.
Total 8 (delta 4), reused 0 (delta 0)
To uranus.ece.iit.edu:wyi3
   a72245d..3e97106  master -> master
```

It is VERY important for you to commit and push your changes at least once at the end of the day no matter how you feel about your progress during the day and no matter whether your project builds and tests successfully or not.

Overall, you should start working on your project WELL before the deadline and make full use of the grading program to avoid any surprise in your project grades. **It is your responsibility to check your grading reports and to correct any errors BEFORE the deadlines. It is possible for all test cases to pass in your VM but for some or all of them may fail on our end if the grading program has been altered on your VM. Failure to check your grading reports on a timely manner may result in VERY LOW project grades, and is NOT an excuse for late project submissions.**