Basic CI/CD for Python projects with Docker and Jenkins

Yen Hoang
Jan 5, 2019 · 6 min read

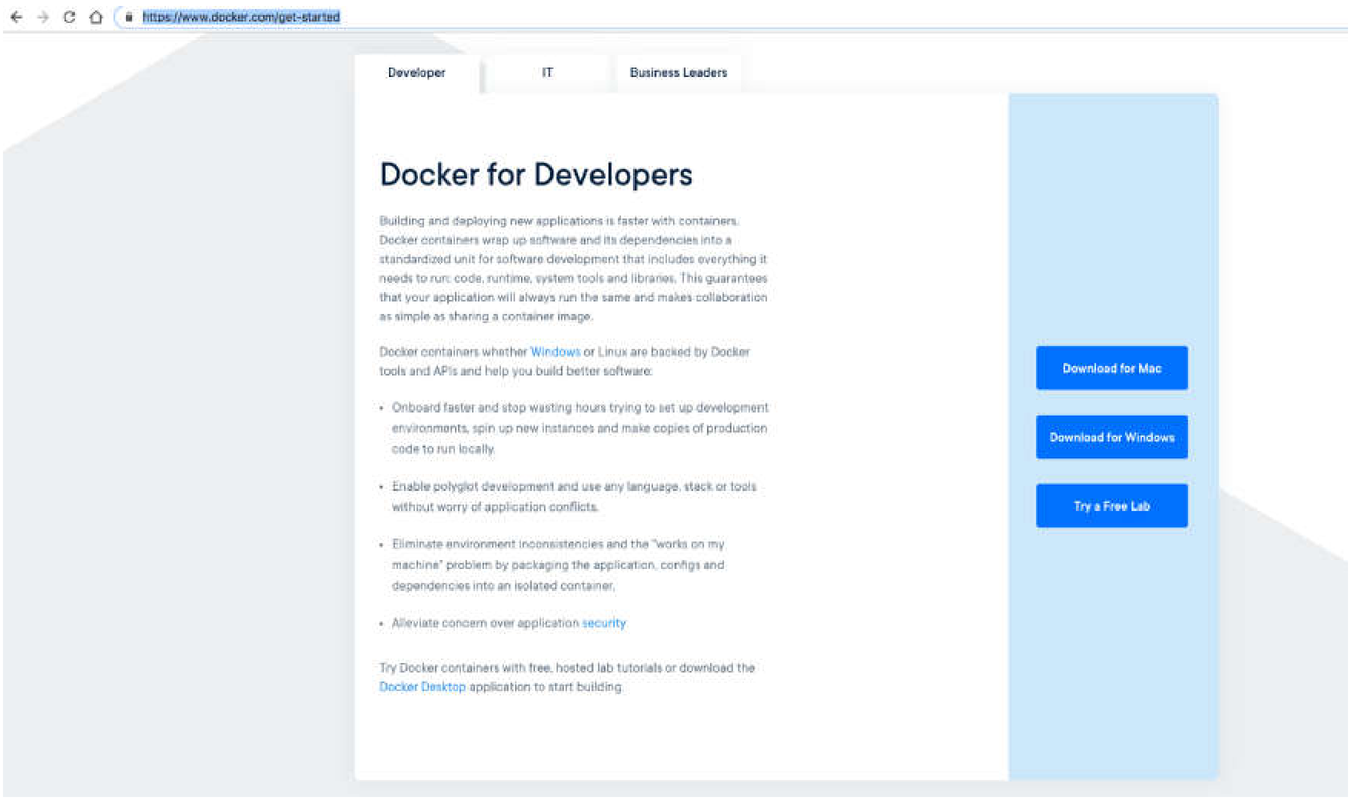
You are a newbie in Docker and Jenkins? You find it's too complicated to set up continuous integration for a Python project with Docker and Jenkins? Don't worry, you are in the right place. This step-by-step guide is firstly written for me (a newbie as well) and I hope that it is useful to you somehow when it comes to continuous integration.

Note: This guide is applied to MacOS only (sorry for that) but to Windows, everything should behave similarly.

Part I: Set up Docker

1. Download and install Docker from the Docker site

_ Go to https://www.docker.com/get-started



Docker for Developers

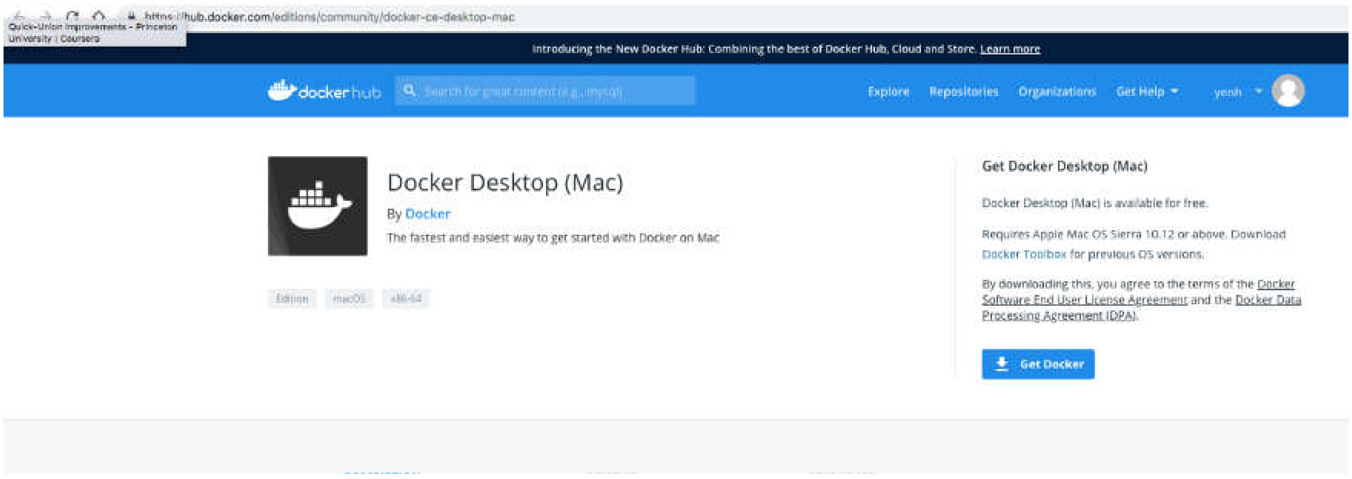
Building and deploying new applications is faster with containers. Docker containers wrap up software and its dependencies into a standardized unit for software development that includes everything it needs to run: code, runtime, system tools and libraries. This guarantees that your application will always run the same and makes collaboration as simple as sharing a container image.

Docker containers whether Windows or Linux are backed by Docker tools and APIs and help you build better software:

- Onboard faster and stop wasting hours trying to set up development environments, spin up new instances and make copies of production code to run locally.
- Enable polyglot development and use any language, stack or tools without worry of application conflicts.
- Eliminate environment inconsistencies and the "works on my machine" problem by packaging the application, configs and dependencies into an isolated container.
- Alleviate concern over application security

Try Docker containers with free, hosted lab tutorials or download the Docker Desktop application to start building.

_ Choose the version compatible with your OS (you will be required to signup/login to download it).


https://hub.docker.com/editions/community/docker-ce-desktop-mac
Quick-Union Improvements - Princeton University | Coursera
docker hub
Explore
Repositories
Organizations
Get Help
yenh
Edition
macOS
x86-64
Get Docker

Introducing the New Docker Hub: Combining the best of Docker Hub, Cloud and Store. Learn more

Docker Desktop (Mac)

By Docker

The fastest and easiest way to get started with Docker on Mac

Get Docker Desktop (Mac)

Docker Desktop (Mac) is available for free.

Requires Apple Mac OS Sierra 10.12 or above. Download Docker Toolbox for previous OS versions.

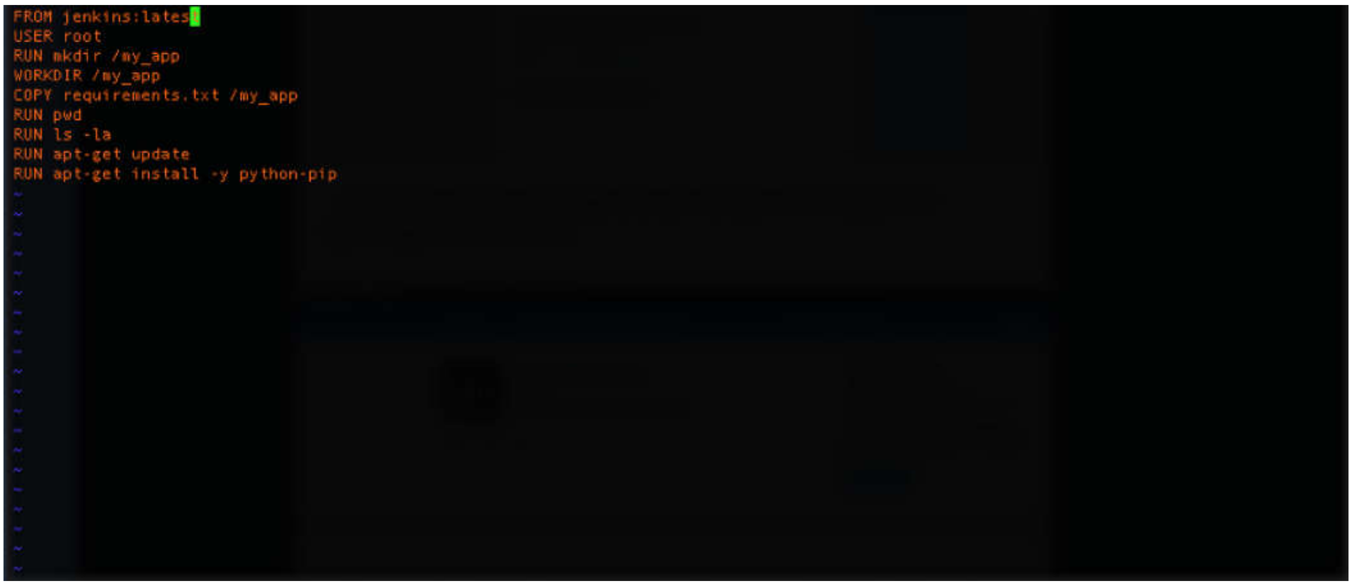
By downloading this, you agree to the terms of the Docker Software End User License Agreement and the Docker Data Processing Agreement (DPA).

_ Once you have Docker installed and running, open a terminal window and type docker –version to make sure that everything is set up appropriately.

2. Build a customized docker image of Jenkins with *dockerfile*

_ Create a folder with a name of your choice, mine is *python_docker_jenkins*, and change to the folder.

_ In *python_docker_jenkins* folder, create a file named *dockerfile or Dockerfile* both work (without extension) with content as below:



```
FROM jenkins:latest
USER root
RUN mkdir /my_app
WORKDIR /my_app
COPY requirements.txt /my_app
RUN pwd
RUN ls -la
RUN apt-get update
RUN apt-get install -y python-pip
```

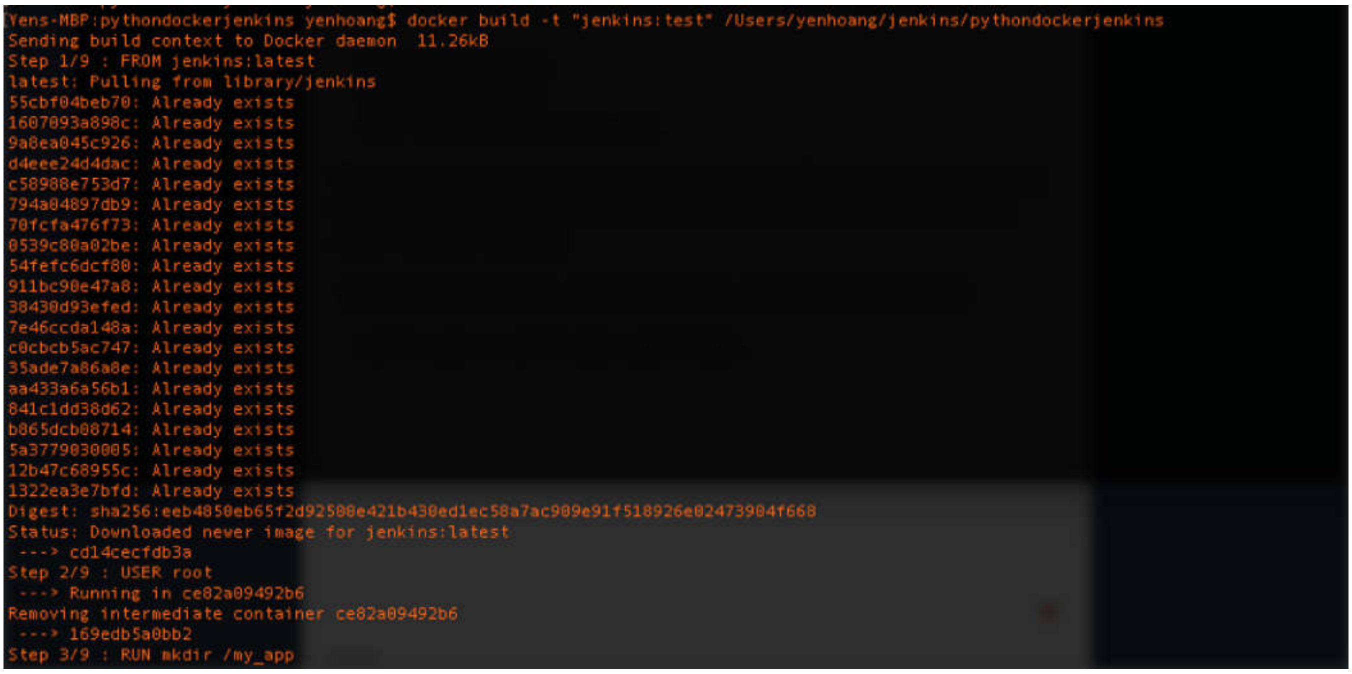
(let me explain a little bit: when we build the image, it will clone the Jenkins image, install pip, and set the user to root). You might question 3 lines:

```
RUN mkdir /my_app
WORKDIR /my_app
COPY requirements.txt /my_app
```

They are used to create a *my_app* folder and copy file *requirements.txt* to that folder (because this is a testing project so I want to separate it from source code in the repo — github).

_ Build the docker image using the following command in the terminal:

```
docker build -t “jenkins:test” path/to/repo
```

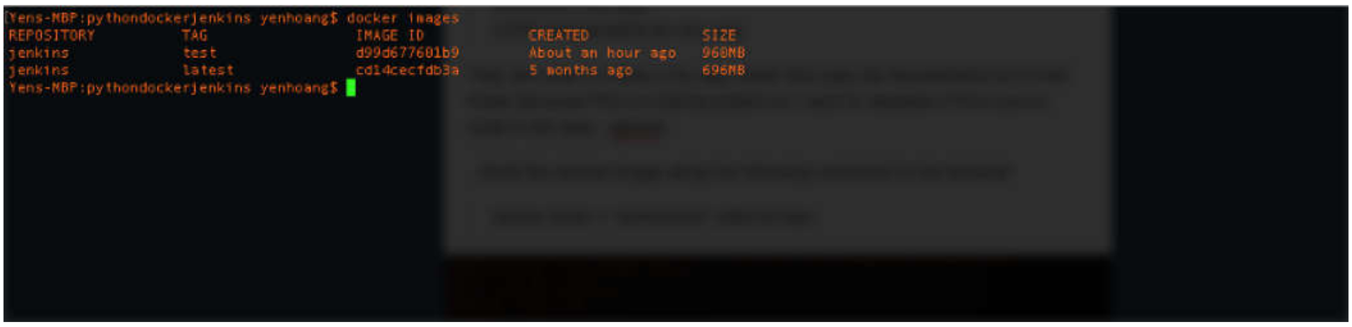


```
Yens-MBP:pythondockerjenkins yenhoang$ docker build -t "jenkins:test" /Users/yenhoang/jenkins/pythondockerjenkins
Sending build context to Docker daemon  11.26kB
Step 1/9 : FROM jenkins:latest
latest: Pulling from library/jenkins
55cbf04beb70: Already exists
1607093a898c: Already exists
9a8ea045c926: Already exists
d4eee24d4dac: Already exists
c58988e753d7: Already exists
794a04897db9: Already exists
70fcfa476f73: Already exists
0539c80a02be: Already exists
54fefc6dcf80: Already exists
911bc90e47a8: Already exists
38430d93efed: Already exists
7e46ccda148a: Already exists
c0cbcb5ac747: Already exists
35ade7a86a8e: Already exists
aa433a6a56b1: Already exists
841c1dd38d62: Already exists
b865dcb08714: Already exists
5a3779030005: Already exists
12b47c68955c: Already exists
1322ea3e7bfd: Already exists
Digest: sha256:eeb4850eb65f2d92500e421b430ed1ec58a7ac909e91f518926e02473904f668
Status: Downloaded newer image for jenkins:latest
 ---> cd14cecfdb3a
Step 2/9 : USER root
 ---> Running in ce82a09492b6
Removing intermediate container ce82a09492b6
 ---> 169edb5a0bb2
Step 3/9 : RUN mkdir /my_app
```

Docker will look for the *dockerfile* in the given path, and build that image. In this example, its name is *jenkins* with the tag *test* (you can change test to a name of your choice).

3. Run the docker image

_ Check the jenkins imaged newly created:

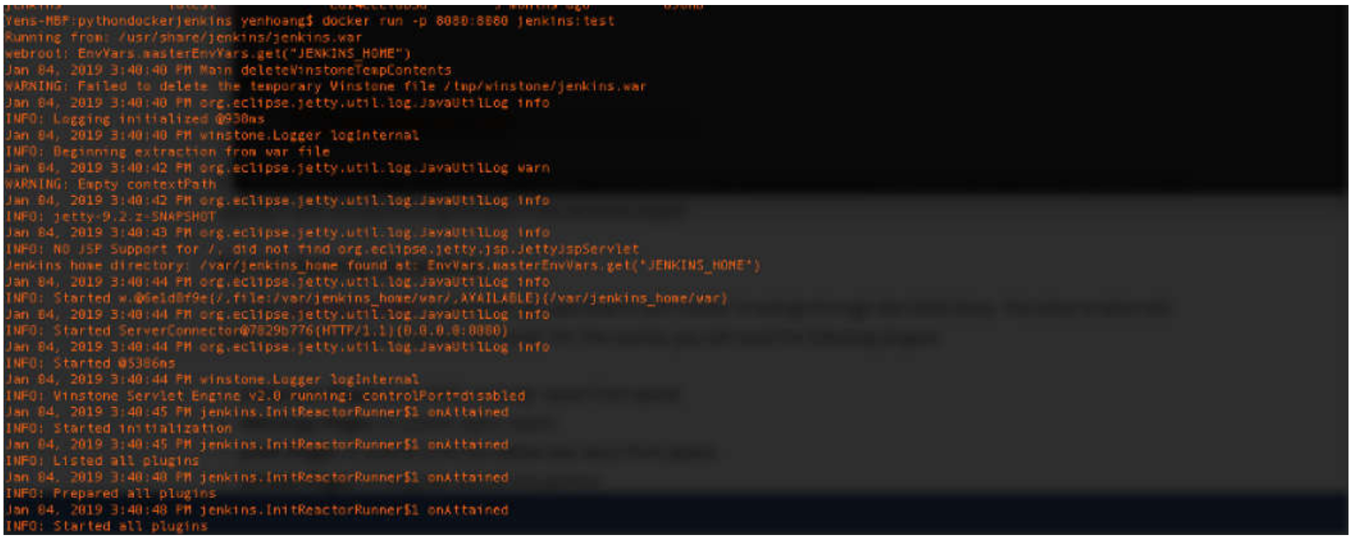


```
Yens-MBP:pythondockerjenkins yenhoang$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
jenkins             test                d99d677601b9        About an hour ago   960MB
jenkins             latest              cd14cecfdb3a        5 months ago        696MB
Yens-MBP:pythondockerjenkins yenhoang$
```

You will see 2 images in which the image with ID = cd14cecfdb3a is the standard one pulled from Docker Hub (parent), the other is our customised one (is a node of the parent).

_ Run the command:

```
docker run -p 8080:8080 jenkins:test
```



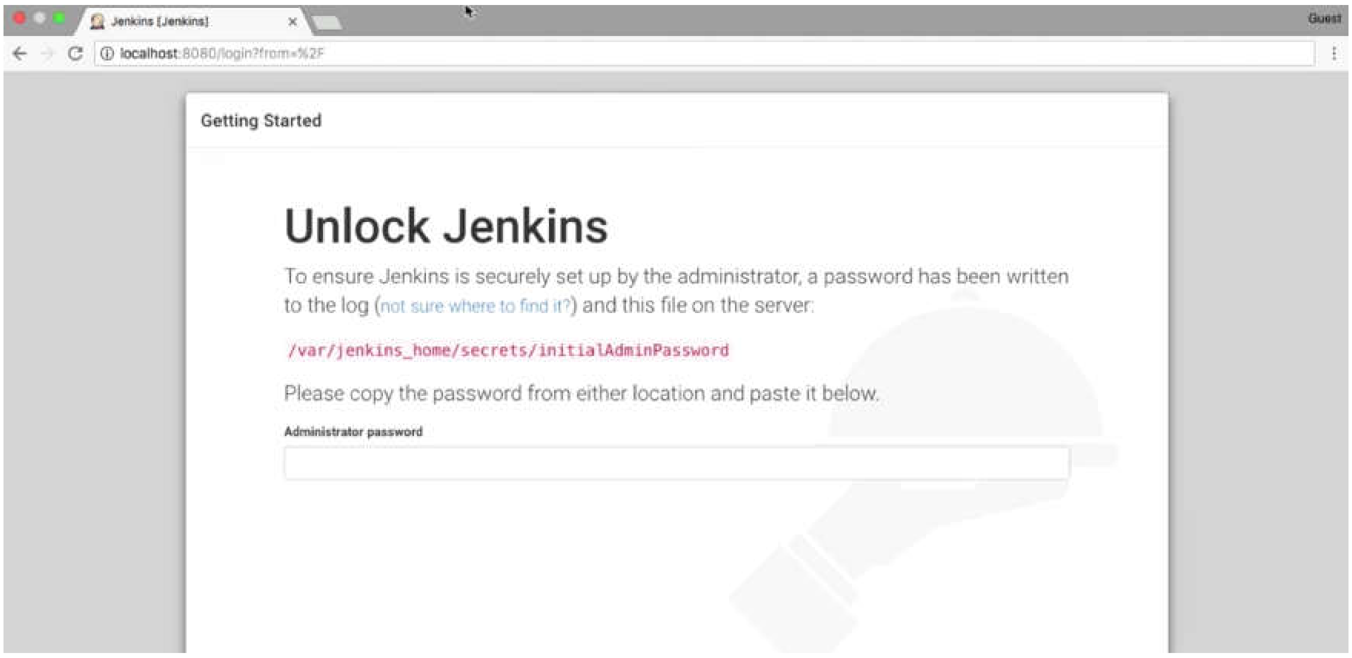
```
Yens-MBP:pythondockerjenkins yenhoang$ docker run -p 8080:8080 jenkins:test
Running from: /usr/share/jenkins/jenkins.war
webroot: EnvVars.masterEnvVars.get("JENKINS_HOME")
Jan 04, 2019 3:40:40 PM Main deleteWinstoneTempContents
WARNING: Failed to delete the temporary Winstone file /tmp/winstone/jenkins.war
Jan 04, 2019 3:40:40 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: Logging initialized @938ms
Jan 04, 2019 3:40:40 PM winstone.Logger logInternal
INFO: Beginning extraction from war file
Jan 04, 2019 3:40:42 PM org.eclipse.jetty.util.log.JavaUtilLog warn
WARNING: Empty contextPath
Jan 04, 2019 3:40:42 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: jetty-9.2.z-SNAPSHOT
Jan 04, 2019 3:40:43 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: NO JSP Support for /, did not find org.eclipse.jetty.jsp.JettyJspServlet
Jenkins home directory: /var/jenkins_home found at: EnvVars.masterEnvVars.get("JENKINS_HOME")
Jan 04, 2019 3:40:44 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: Started w.@6e1d0f9e{/,file:/var/jenkins_home/war/,AVAILABLE}{/var/jenkins_home/war}
Jan 04, 2019 3:40:44 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: Started ServerConnector@7829b776{HTTP/1.1}{0.0.0.0:8080}
Jan 04, 2019 3:40:44 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: Started @5386ms
Jan 04, 2019 3:40:44 PM winstone.Logger logInternal
INFO: Winstone Servlet Engine v2.0 running: controlPort=disabled
Jan 04, 2019 3:40:45 PM jenkins.InitReactorRunner$1 onAttained
INFO: Started initialization
Jan 04, 2019 3:40:45 PM jenkins.InitReactorRunner$1 onAttained
INFO: Listed all plugins
Jan 04, 2019 3:40:48 PM jenkins.InitReactorRunner$1 onAttained
INFO: Prepared all plugins
Jan 04, 2019 3:40:48 PM jenkins.InitReactorRunner$1 onAttained
INFO: Started all plugins
```

_ Go to the port you specified in a browser window- in this example, localhost:8080 Here, you will need to setup Jenkins after signing in with the password generated in the terminal output.

Part II: Set up Jenkins

1. Getting started with Jenkins

_ Go to localhost:8080, you will see passwords prompted as below:



Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

/var/jenkins_home/secrets/initialAdminPassword

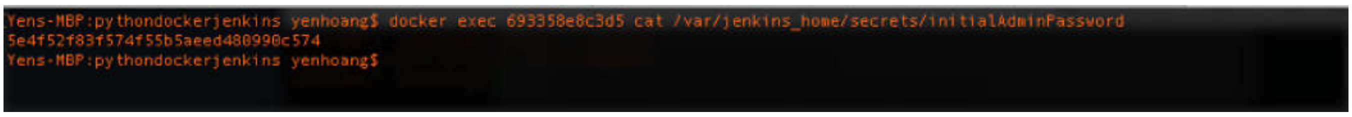
Please copy the password from either location and paste it below.

Administrator password

_ New Jenkins installations are locked down by default, and you need to get that password to log in. But if Jenkins is running on the docker container rather than directly on your system how can you get the initial admin password? The easiest way is to run the following command:

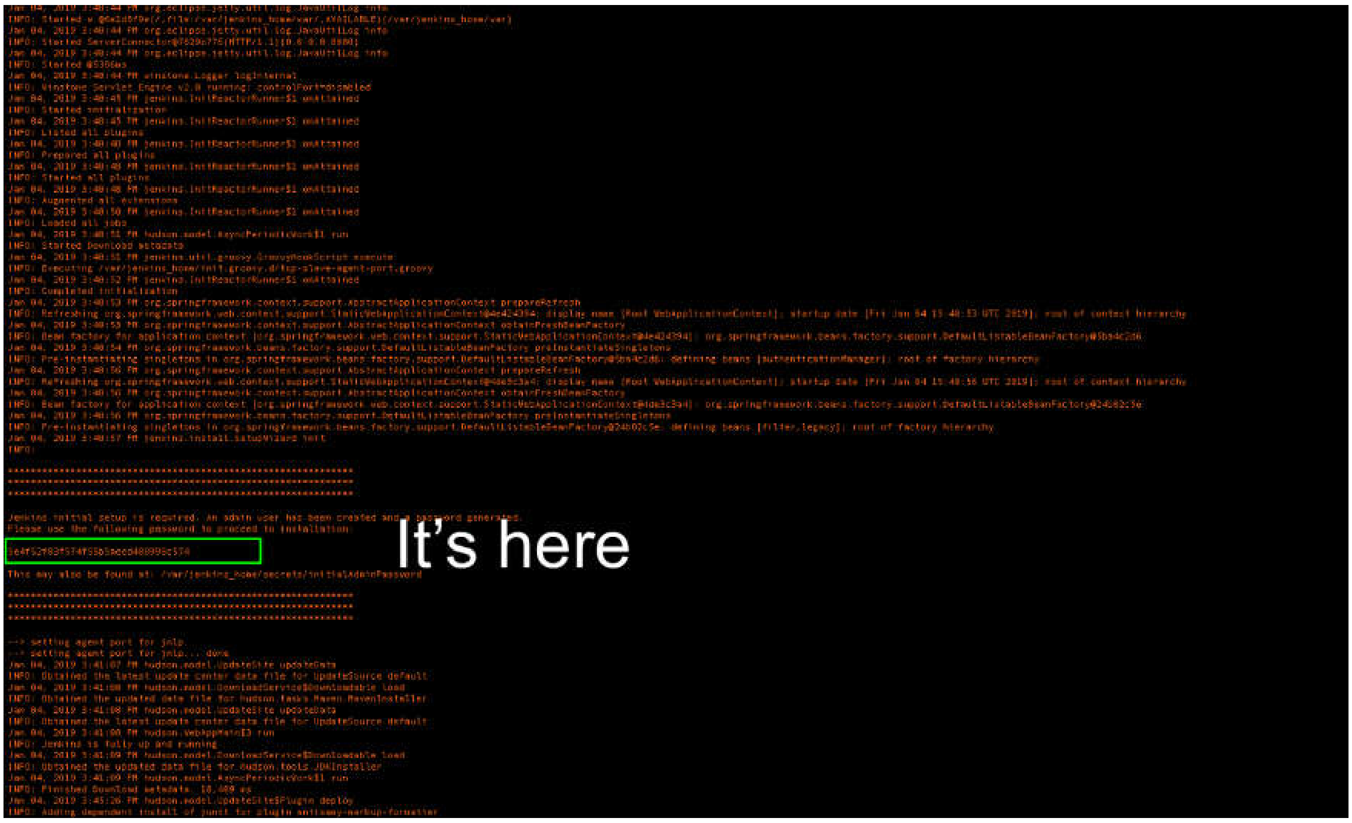
```
docker exec 693358e8c3d5 cat /var/jenkins_home/secrets/initialAdminPassword
```

(*693358e8c3d5* is my jenkins container ID, get yours by running the command: docker container ls)



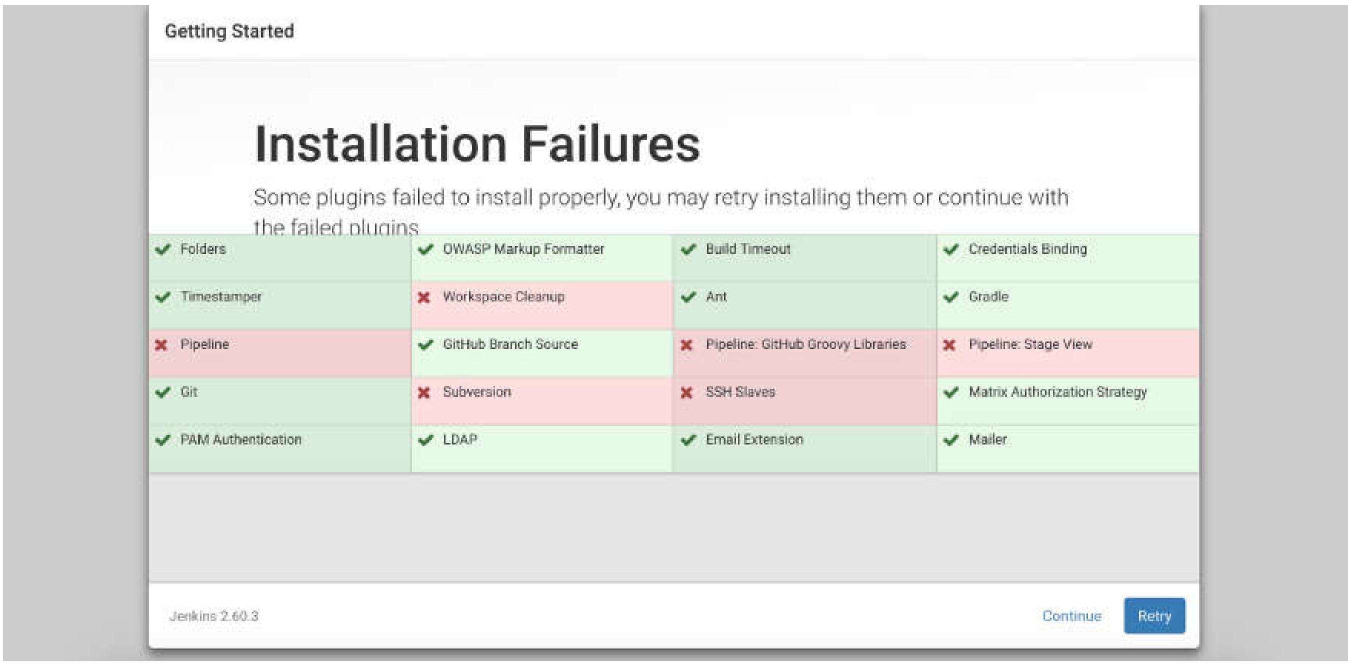
```
Yens-MBP:pythondockerjenkins yenhoang$ docker exec 693358e8c3d5 cat /var/jenkins_home/secrets/initialAdminPassword
5e4f52f83f574f55b5aeed480990c574
Yens-MBP:pythondockerjenkins yenhoang$
```

(another way is to find in the log when running the docker image)



```
Jan 04, 2019 3:40:44 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: Started w.@6e2d5f0e{/,file:///var/jenkins_home/war/,AVAILABLE}{/var/jenkins_home/war}
Jan 04, 2019 3:40:44 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: Started ServerConnector@7629c775{HTTP/1.1}{0.0.0.0:8080}
Jan 04, 2019 3:40:44 PM org.eclipse.jetty.util.log.JavaUtilLog info
INFO: Started @5356ms
Jan 04, 2019 3:40:44 PM winstone.Logger logInternal
INFO: Winstone Servlet Engine v2.0 running: controlPort=disabled
Jan 04, 2019 3:40:45 PM jenkins.InitReactorRunner$1 onAttained
INFO: Started initialization
Jan 04, 2019 3:40:45 PM jenkins.InitReactorRunner$1 onAttained
INFO: Listed all plugins
Jan 04, 2019 3:40:48 PM jenkins.InitReactorRunner$1 onAttained
INFO: Prepared all plugins
Jan 04, 2019 3:40:48 PM jenkins.InitReactorRunner$1 onAttained
INFO: Started all plugins
Jan 04, 2019 3:40:48 PM jenkins.InitReactorRunner$1 onAttained
INFO: Augmented all extensions
Jan 04, 2019 3:40:50 PM jenkins.InitReactorRunner$1 onAttained
INFO: Loaded all jobs
Jan 04, 2019 3:40:51 PM hudson.model.AsyncPeriodicWork$1 run
INFO: Started Download metadata
Jan 04, 2019 3:40:51 PM jenkins.util.groovy.GroovyHookScript execute
INFO: Executing /var/jenkins_home/init.groovy.d/tcp-slave-agent-port.groovy
Jan 04, 2019 3:40:52 PM jenkins.InitReactorRunner$1 onAttained
INFO: Completed initialization
Jan 04, 2019 3:40:53 PM org.springframework.context.support.AbstractApplicationContext prepareRefresh
INFO: Refreshing org.springframework.web.context.support.StaticWebApplicationContext@4e424394: display name [Root WebApplicationContext]; startup date [Fri Jan 04 15:40:53 UTC 2019]; root of context hierarchy
Jan 04, 2019 3:40:53 PM org.springframework.context.support.AbstractApplicationContext obtainFreshBeanFactory
INFO: Bean factory for application context [org.springframework.web.context.support.StaticWebApplicationContext@4e424394]: org.springframework.beans.factory.support.DefaultListableBeanFactory@5ba4c2d6
Jan 04, 2019 3:40:54 PM org.springframework.beans.factory.support.DefaultListableBeanFactory preInstantiateSingletons
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@5ba4c2d6: defining beans [authenticationManager]; root of factory hierarchy
Jan 04, 2019 3:40:56 PM org.springframework.context.support.AbstractApplicationContext prepareRefresh
INFO: Refreshing org.springframework.web.context.support.StaticWebApplicationContext@4de5c3a4: display name [Root WebApplicationContext]; startup date [Fri Jan 04 15:40:56 UTC 2019]; root of context hierarchy
Jan 04, 2019 3:40:56 PM org.springframework.context.support.AbstractApplicationContext obtainFreshBeanFactory
INFO: Bean factory for application context [org.springframework.web.context.support.StaticWebApplicationContext@4de5c3a4]: org.springframework.beans.factory.support.DefaultListableBeanFactory@24b02c5e
Jan 04, 2019 3:40:56 PM org.springframework.beans.factory.support.DefaultListableBeanFactory preInstantiateSingletons
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@24b02c5e: defining beans [filter,legacy]; root of factory hierarchy
Jan 04, 2019 3:40:57 PM jenkins.install.SetupWizard init
INFO:

*************************************************************
*************************************************************
*************************************************************

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

5e4f52f83f574f55b5aeed480990c574

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*************************************************************
*************************************************************
*************************************************************

--> setting agent port for jnlp
--> setting agent port for jnlp... done
Jan 04, 2019 3:41:07 PM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Jan 04, 2019 3:41:08 PM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
Jan 04, 2019 3:41:08 PM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Jan 04, 2019 3:41:08 PM hudson.WebAppMain$3 run
INFO: Jenkins is fully up and running
Jan 04, 2019 3:41:09 PM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tools.JDKInstaller
Jan 04, 2019 3:41:09 PM hudson.model.AsyncPeriodicWork$1 run
INFO: Finished Download metadata. 18,409 ms
Jan 04, 2019 3:45:26 PM hudson.model.UpdateSite$Plugin deploy
INFO: Adding dependent install of junit for plugin antisamy-markup-formatter
```

_ Copy the string then open localhost:8080 in your browser. In the browser, enter the admin password and click continue. Then click install suggested plugins. This will complete the Jenkins by installing the plugins that are most commonly used by Jenkins. Depending on the speed of your laptop and your internet connection, this may take several minutes to complete.


Getting Started

Installation Failures

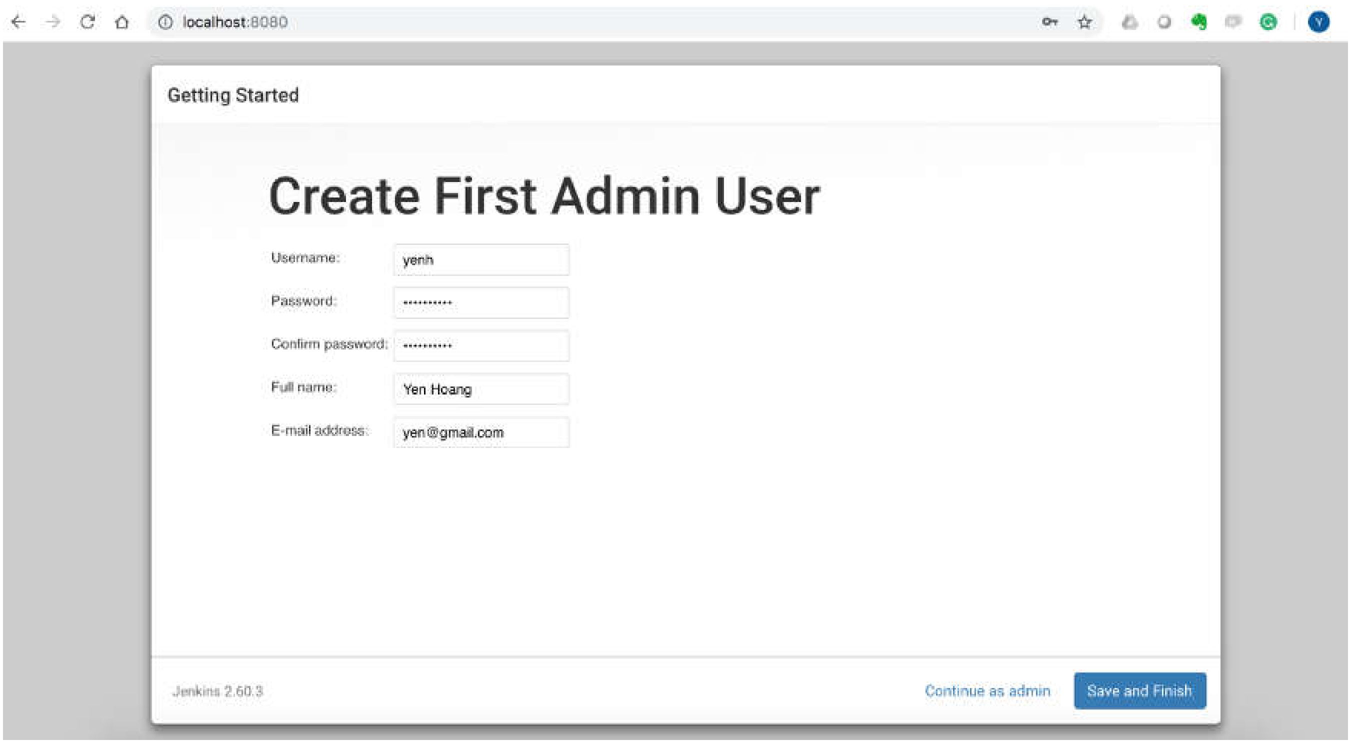
Some plugins failed to install properly, you may retry installing them or continue with the failed plugins

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ Timestamper	✗ Workspace Cleanup	✓ Ant	✓ Gradle
✗ Pipeline	✓ GitHub Branch Source	✗ Pipeline: GitHub Groovy Libraries	✗ Pipeline: Stage View
✓ Git	✗ Subversion	✗ SSH Slaves	✓ Matrix Authorization Strategy
✓ PAM Authentication	✓ LDAP	✓ Email Extension	✓ Mailer

Jenkins 2.60.3

Unfortunately, not everything is green (aka successful). The tricky part is that the failures are due to this Jenkins version of Docker is outdated. I will show you how to resolve this issue later, now just click **Continue** to move on.

_ Now you'll be prompted to create your first admin user. At this point, you should create your account, with a username of your choice and a password that's easy to remember but hard for others to guess.


localhost:8080

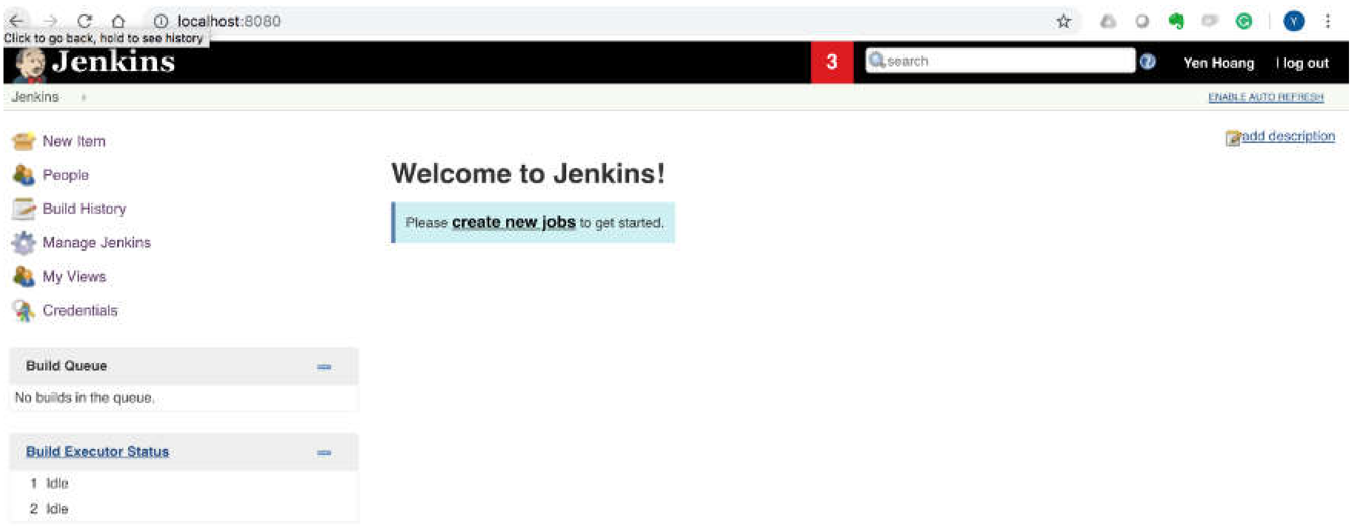
Getting Started

Create First Admin User

Username:	yenh
Password:	••••••••••
Confirm password:	••••••••••
Full name:	Yen Hoang
E-mail address:	yen@gmail.com

Jenkins 2.60.3

_ Upon click **Save and Finish**, you will be navigated to Jenkins Dashboard as you expected huh.

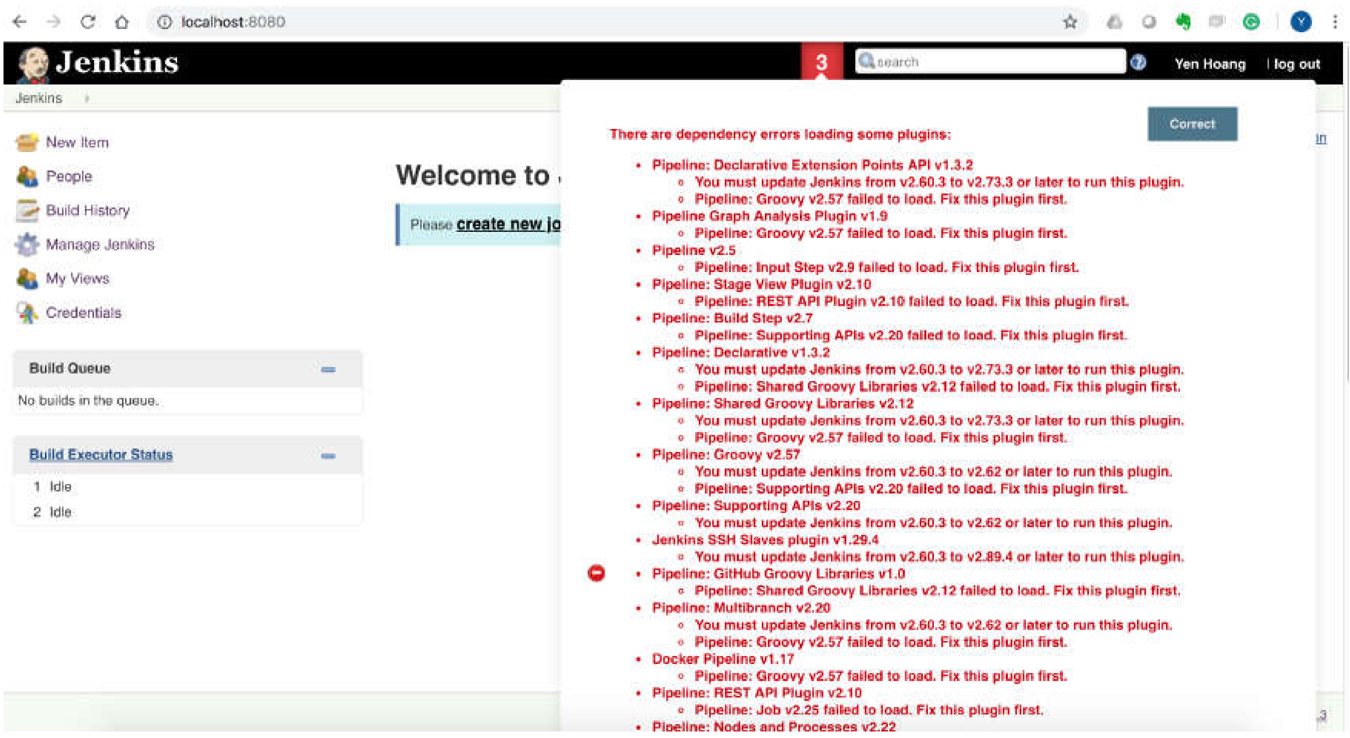


Welcome to Jenkins!

Please **create new jobs** to get started.

No builds in the queue.

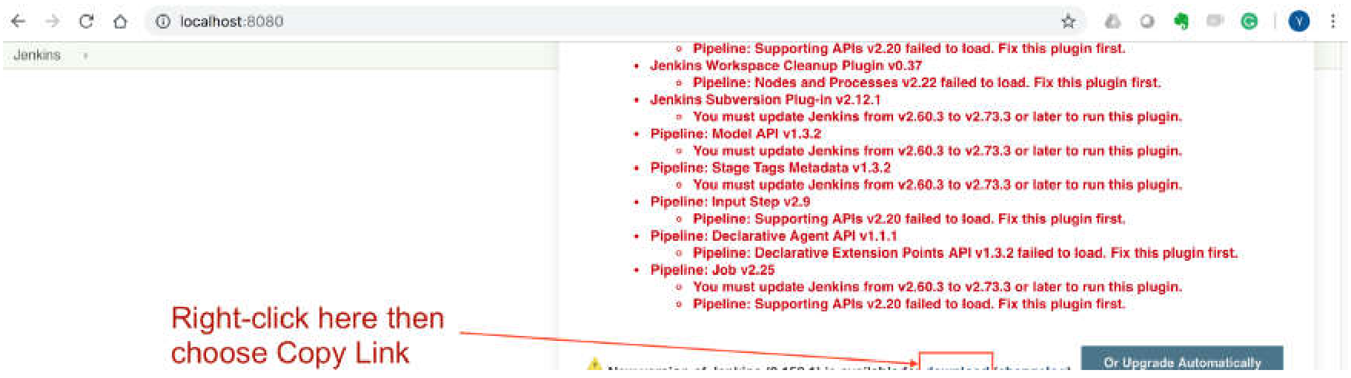
2. Now let's go back to the issue of plugin installation failure above. Click on the red notification on the top you will be given the causes.



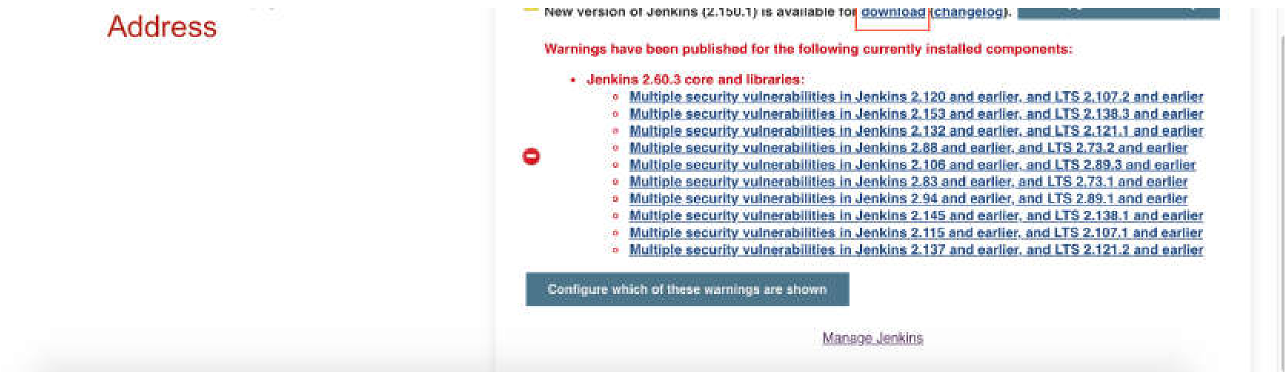
There are dependency errors loading some plugins:

- Pipeline: Declarative Extension Points API v1.3.2
  - You must update Jenkins from v2.60.3 to v2.73.3 or later to run this plugin.
  - Pipeline: Groovy v2.57 failed to load. Fix this plugin first.
- Pipeline Graph Analysis Plugin v1.9
  - Pipeline: Groovy v2.57 failed to load. Fix this plugin first.
- Pipeline v2.5
  - Pipeline: Input Step v2.9 failed to load. Fix this plugin first.
- Pipeline: Stage View Plugin v2.10
  - Pipeline: REST API Plugin v2.10 failed to load. Fix this plugin first.
- Pipeline: Build Step v2.7
  - Pipeline: Supporting APIs v2.20 failed to load. Fix this plugin first.
- Pipeline: Declarative v1.3.2
  - You must update Jenkins from v2.60.3 to v2.73.3 or later to run this plugin.
  - Pipeline: Shared Groovy Libraries v2.12 failed to load. Fix this plugin first.
- Pipeline: Shared Groovy Libraries v2.12
  - You must update Jenkins from v2.60.3 to v2.73.3 or later to run this plugin.
  - Pipeline: Groovy v2.57 failed to load. Fix this plugin first.
- Pipeline: Groovy v2.57
  - You must update Jenkins from v2.60.3 to v2.62 or later to run this plugin.
  - Pipeline: Supporting APIs v2.20 failed to load. Fix this plugin first.
- Pipeline: Supporting APIs v2.20
  - You must update Jenkins from v2.60.3 to v2.62 or later to run this plugin.
- Jenkins SSH Slaves plugin v1.29.4
  - You must update Jenkins from v2.60.3 to v2.89.4 or later to run this plugin.
- Pipeline: GitHub Groovy Libraries v1.0
  - Pipeline: Shared Groovy Libraries v2.12 failed to load. Fix this plugin first.
- Pipeline: Multibranch v2.20
  - You must update Jenkins from v2.60.3 to v2.62 or later to run this plugin.
  - Pipeline: Groovy v2.57 failed to load. Fix this plugin first.
- Docker Pipeline v1.17
  - Pipeline: Groovy v2.57 failed to load. Fix this plugin first.
- Pipeline: REST API Plugin v2.10
  - Pipeline: Job v2.25 failed to load. Fix this plugin first.
- Pipeline: Nodes and Processes v2.22

_ To solve that scroll down the notification then right click on download as below:



- Pipeline: Supporting APIs v2.20 failed to load. Fix this plugin first.
- Jenkins Workspace Cleanup Plugin v0.37
  - Pipeline: Nodes and Processes v2.22 failed to load. Fix this plugin first.
- Jenkins Subversion Plug-in v2.12.1
  - You must update Jenkins from v2.60.3 to v2.73.3 or later to run this plugin.
- Pipeline: Model API v1.3.2
  - You must update Jenkins from v2.60.3 to v2.73.3 or later to run this plugin.
- Pipeline: Stage Tags Metadata v1.3.2
  - You must update Jenkins from v2.60.3 to v2.73.3 or later to run this plugin.
- Pipeline: Input Step v2.9
  - Pipeline: Supporting APIs v2.20 failed to load. Fix this plugin first.
- Pipeline: Declarative Agent API v1.1.1
  - Pipeline: Declarative Extension Points API v1.3.2 failed to load. Fix this plugin first.
- Pipeline: Job v2.25
  - You must update Jenkins from v2.60.3 to v2.73.3 or later to run this plugin.
  - Pipeline: Supporting APIs v2.20 failed to load. Fix this plugin first.

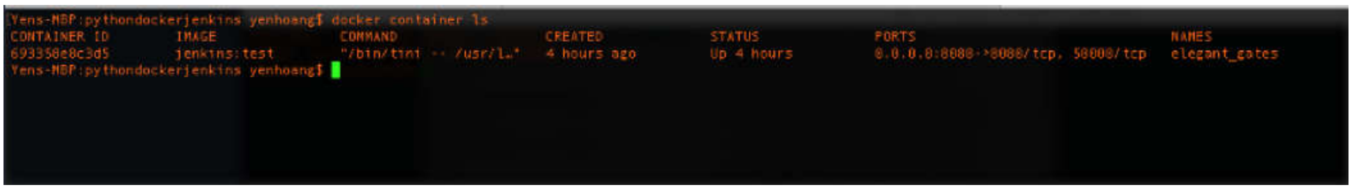

Address

New version of Jenkins (2.150.1) is available for download (changelog).

Warnings have been published for the following currently installed components:

- Jenkins 2.60.3 core and libraries:
  - Multiple security vulnerabilities in Jenkins 2.120 and earlier, and LTS 2.107.2 and earlier
  - Multiple security vulnerabilities in Jenkins 2.153 and earlier, and LTS 2.138.3 and earlier
  - Multiple security vulnerabilities in Jenkins 2.132 and earlier, and LTS 2.121.1 and earlier
  - Multiple security vulnerabilities in Jenkins 2.88 and earlier, and LTS 2.73.2 and earlier
  - Multiple security vulnerabilities in Jenkins 2.106 and earlier, and LTS 2.89.3 and earlier
  - Multiple security vulnerabilities in Jenkins 2.83 and earlier, and LTS 2.73.1 and earlier
  - Multiple security vulnerabilities in Jenkins 2.94 and earlier, and LTS 2.89.1 and earlier
  - Multiple security vulnerabilities in Jenkins 2.145 and earlier, and LTS 2.138.1 and earlier
  - Multiple security vulnerabilities in Jenkins 2.115 and earlier, and LTS 2.107.1 and earlier
  - Multiple security vulnerabilities in Jenkins 2.137 and earlier, and LTS 2.121.2 and earlier

_ Go back to the Terminal, list all containers:


```
Yens-MBP:pythondockerjenkins yenhoang$ docker container ls
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                              NAMES
693358e8c3d5        jenkins:test        "/bin/tini -- /usr/l…"   4 hours ago         Up 4 hours          0.0.0.0:8080->8080/tcp, 50000/tcp   elegant_gates
Yens-MBP:pythondockerjenkins yenhoang$
```

_ Log in to the Jenkins container (I use its ID got from above command):

```
docker container exec -u 0 -it CONTAINER_ID bash
```

_ Download the update with the link copied above:

```
wget http://updates.jenkins-ci.org/download/war/2.150.1/jenkins.war
```

_ Move it to the appropriate place:

```
mv ./jenkins.war /usr/share/jenkins
```

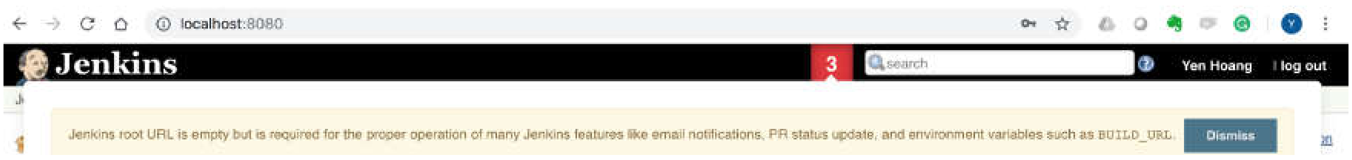
_ Change permission:

```
chown jenkins:jenkins /usr/share/jenkins/jenkins.war
```

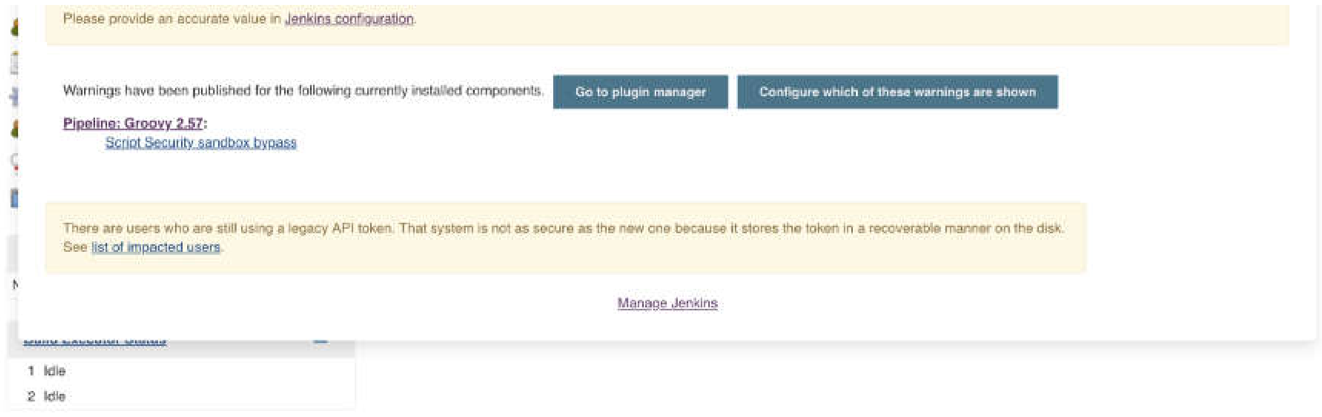
_ Exit container and restart the container:

```
exit

docker container restart CONTAINER_ID
```

_ Go back to localhost:8080, the errors have been resolved.



Jenkins root URL is empty but is required for the proper operation of many Jenkins features like email notifications, PR status update, and environment variables such as BUILD_URL.


Please provide an accurate value in Jenkins configuration.

Warnings have been published for the following currently installed components. Go to plugin manager | Configure which of these warnings are shown

Pipeline: Groovy 2.57:
Script Security sandbox bypass

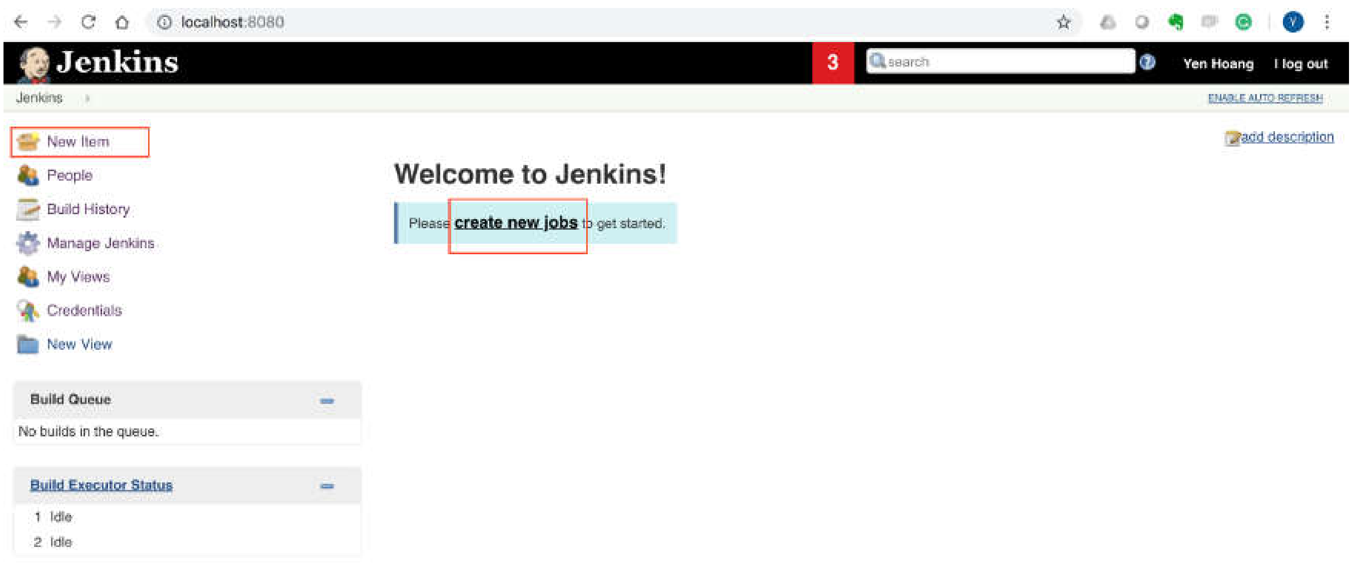
There are users who are still using a legacy API token. That system is not as secure as the new one because it stores the token in a recoverable manner on the disk. See list of impacted users.

Manage Jenkins

1 Idle
2 Idle

3. Create Jenkins Jobs

_ From Jenkins dashboard, click *create new job* or *New Item* to create a job.



Welcome to Jenkins!

Please create new jobs to get started.

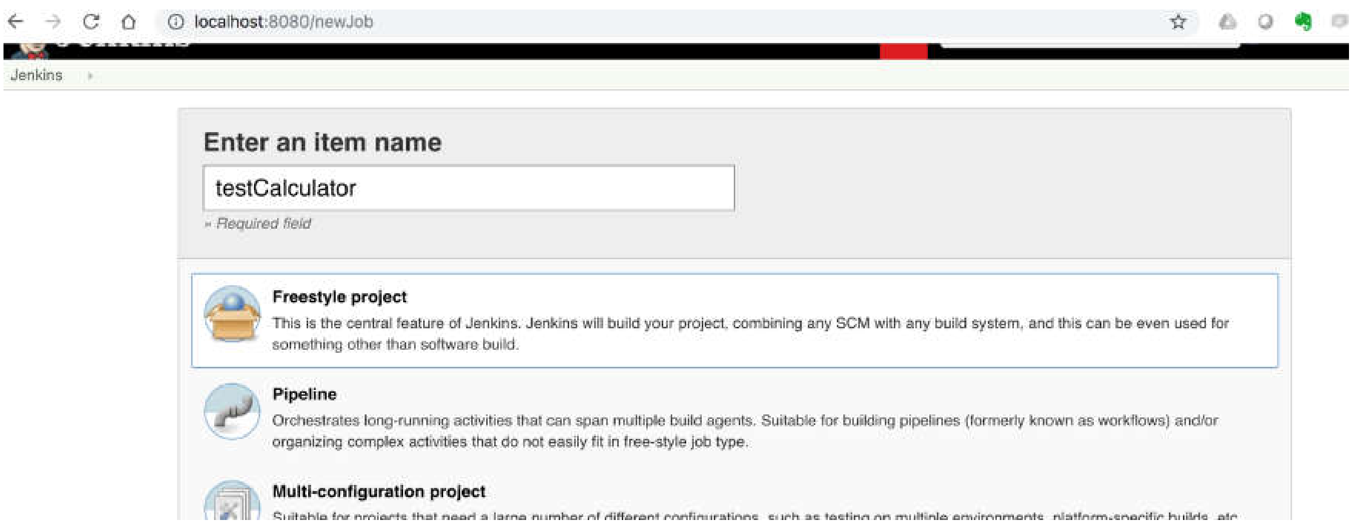
Build Queue

No builds in the queue.

Build Executor Status

1 Idle
2 Idle

_ Enter an item name then choose project type (I choose Freestyle project in this case).



Enter an item name

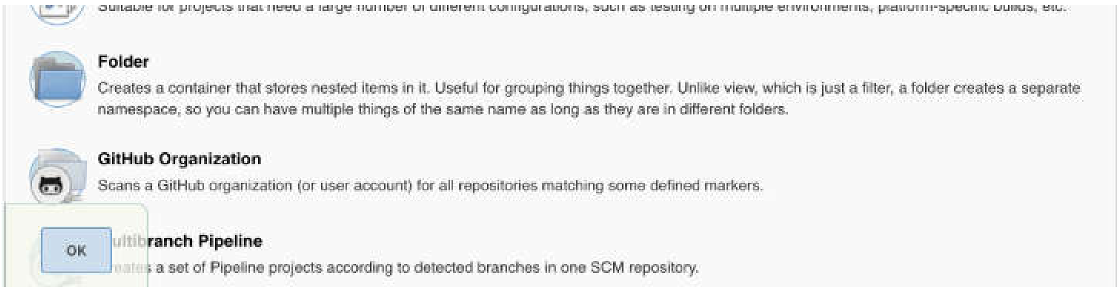
testCalculator

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project


Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

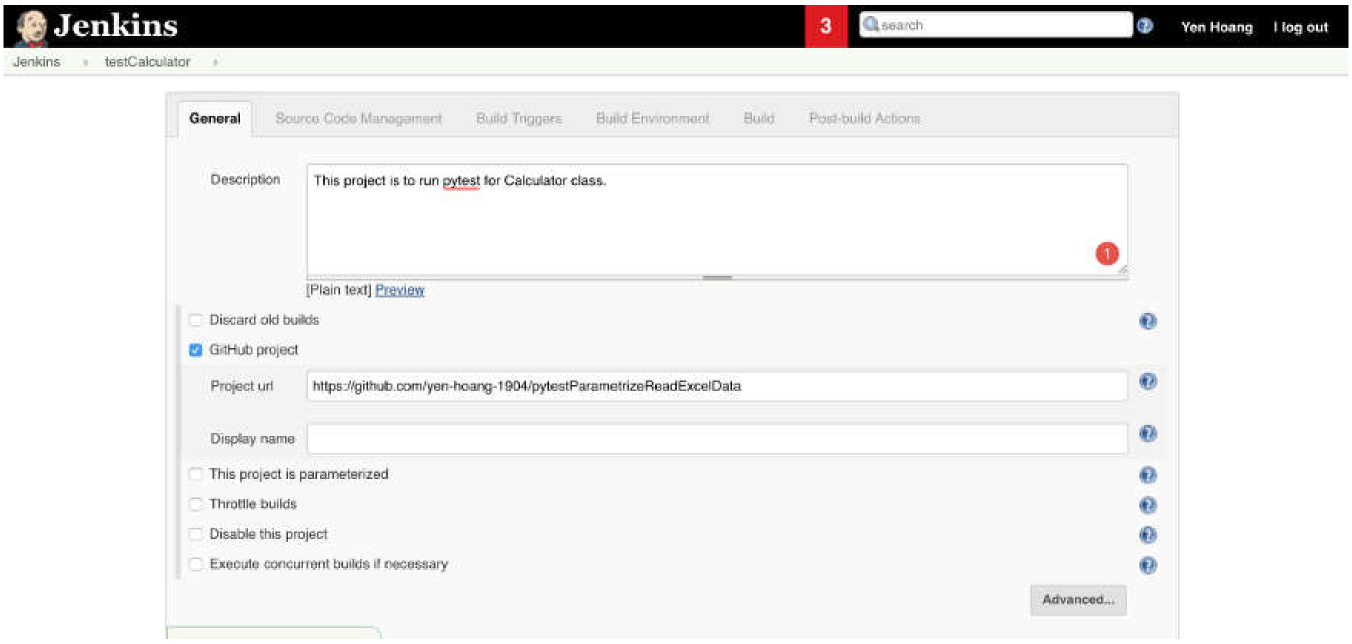
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

OK

ultibranch Pipeline
eates a set of Pipeline projects according to detected branches in one SCM repository.

_ Now we have to config some things to run the test. Under **General**section, provide some description then check the GitHub Project box and enter the URL to your GitHub project.



Description: This project is to run pytest for Calculator class.

[Plain text] Preview

- [ ] Discard old builds
- [x] GitHub project

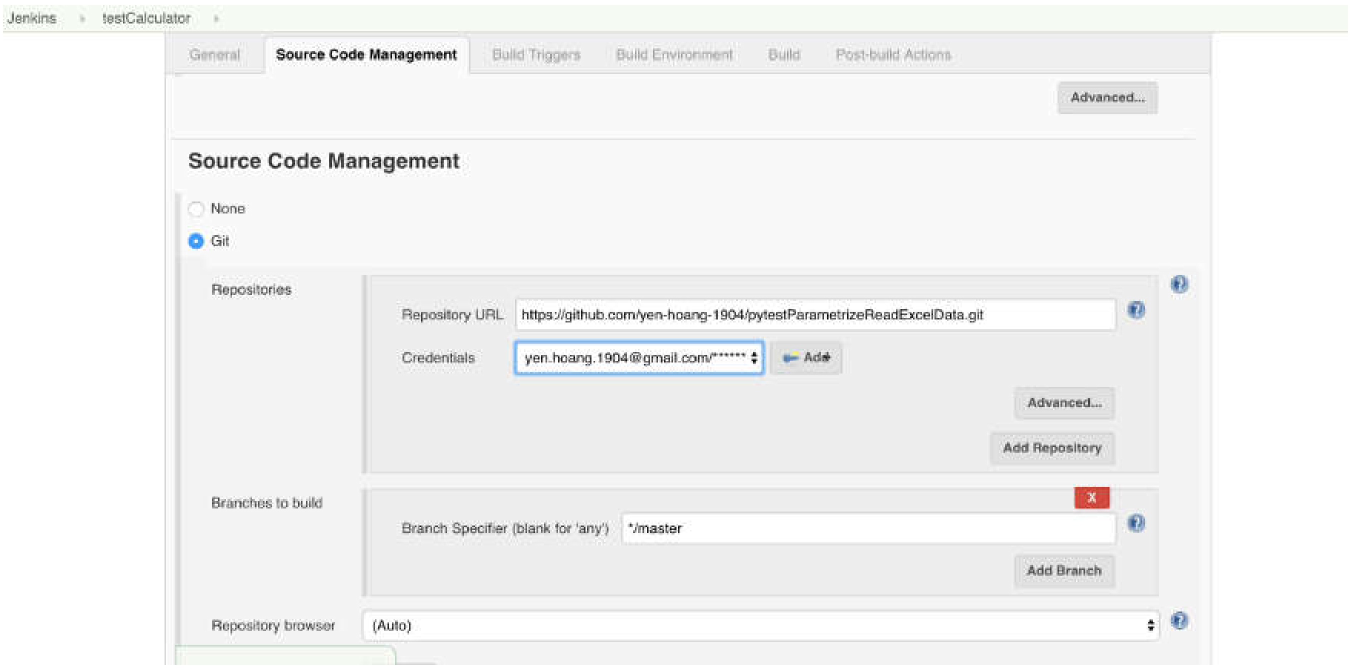
Project url: https://github.com/yen-hoang-1904/pytestParametrizeReadExcelData

Display name:

- [ ] This project is parameterized
- [ ] Throttle builds
- [ ] Disable this project
- [ ] Execute concurrent builds if necessary

Advanced...

_ Under **Source Code Management**, click on Git and add the repository URL again. You will be able to add credentials here as well and specify the branch to build.



Source Code Management

- ( ) None
- (●) Git

Repositories

Repository URL: https://github.com/yen-hoang-1904/pytestParametrizeReadExcelData.git

Credentials: yen.hoang.1904@gmail.com/****** Add

Advanced...

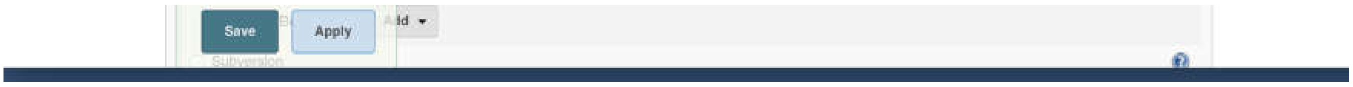
Add Repository

Branches to build

Branch Specifier (blank for 'any'): */master

Add Branch

Repository browser: (Auto)

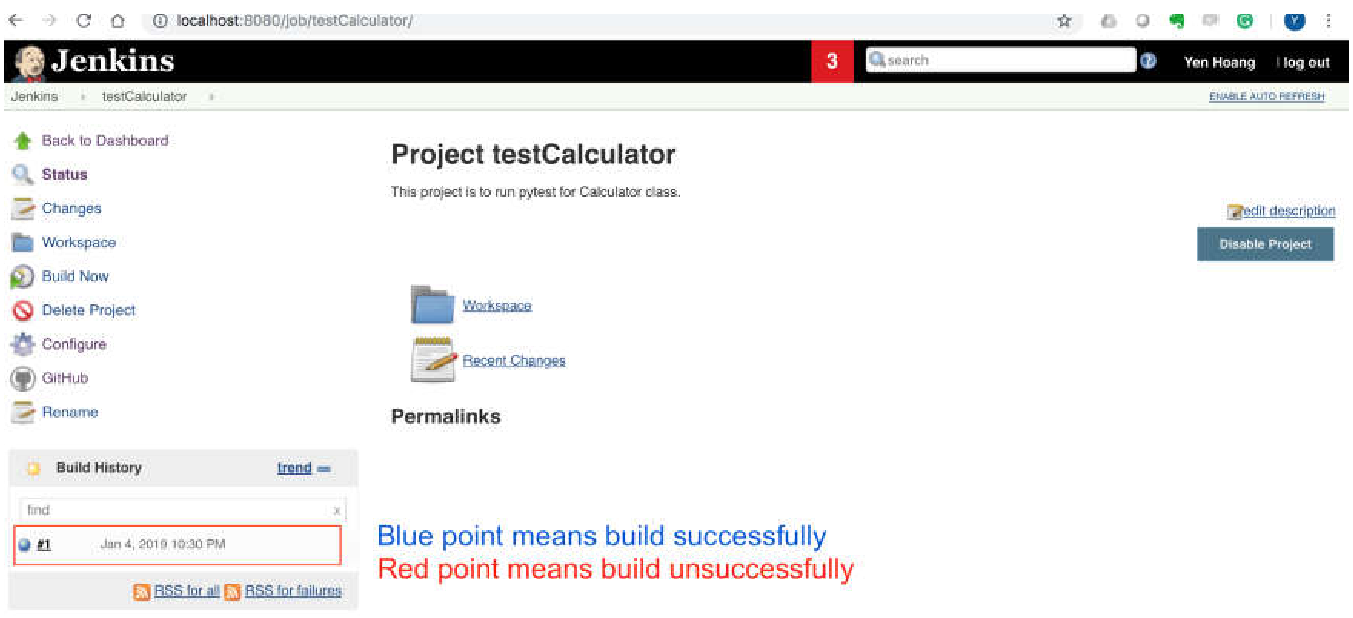


_ Under **Build Triggers**, select Github hook trigger for GITScm polling

_ Under **Build**, click on Add a build step and select Execute Shell. This is a place to put the test commands that you need to execute during the build:

```
echo 'Start installing dependencies'

#!/bin/bash
pip install -r /my_app/requirements.txt

echo 'Start running test cases'
pytest testCalculator.py
```

_ Save and go back to the job then click Build Now on the sidebar.



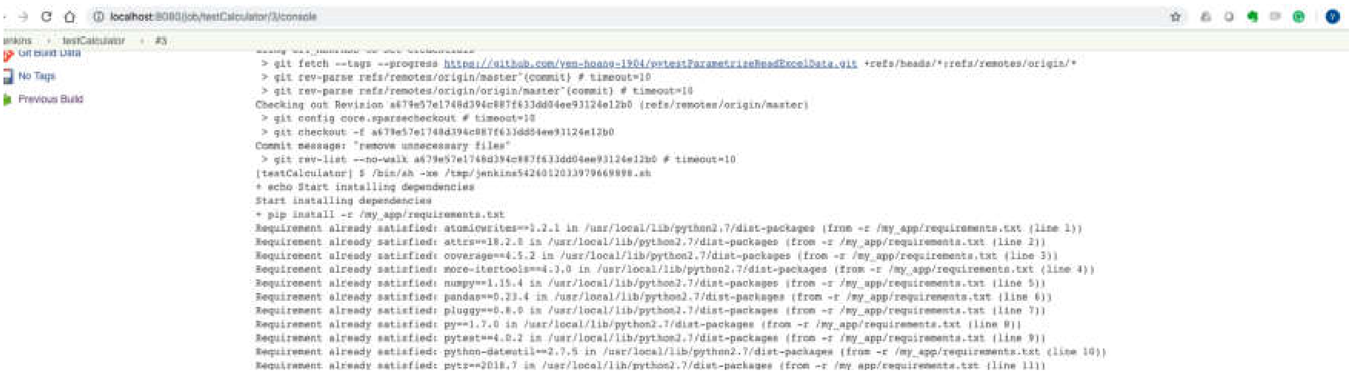
Project testCalculator

This project is to run pytest for Calculator class.

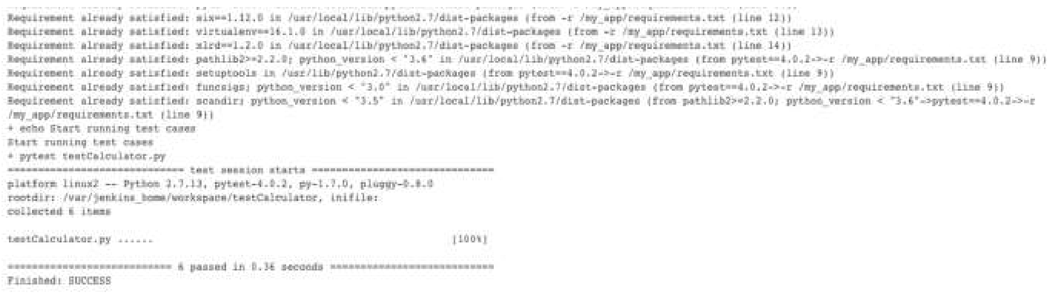
#1 Jan 4, 2019 10:30 PM

Blue point means build successfully
Red point means build unsuccessfully

_ You can click on the blue (or red) point to view Console output:



```
 > git fetch --tags --progress https://github.com/yen-hoang-1904/pytestParametrizeReadExcelData.git +refs/heads/*:refs/remotes/origin/*
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision a679e57e1748d394c887f633dd04ee93124e12b0 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f a679e57e1748d394c887f633dd04ee93124e12b0
Commit message: "remove unnecessary files"
 > git rev-list --no-walk a679e57e1748d394c887f633dd04ee93124e12b0 # timeout=10
[testCalculator] $ /bin/sh -xe /tmp/jenkins5426012033979669898.sh
+ echo Start installing dependencies
Start installing dependencies
+ pip install -r /my_app/requirements.txt
Requirement already satisfied: atomicwrites==1.2.1 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 1))
Requirement already satisfied: attrs==18.2.0 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 2))
Requirement already satisfied: coverage==4.5.2 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 3))
Requirement already satisfied: more-itertools==4.3.0 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 4))
Requirement already satisfied: numpy==1.15.4 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 5))
Requirement already satisfied: pandas==0.23.4 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 6))
Requirement already satisfied: pluggy==0.8.0 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 7))
Requirement already satisfied: py==1.7.0 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 8))
Requirement already satisfied: pytest==4.0.2 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 9))
Requirement already satisfied: python-dateutil==2.7.5 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 10))
Requirement already satisfied: pytz==2018.7 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 11))
```



```
Requirement already satisfied: six==1.12.0 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 12))
Requirement already satisfied: virtualenv==16.1.0 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 13))
Requirement already satisfied: xlrd==1.2.0 in /usr/local/lib/python2.7/dist-packages (from -r /my_app/requirements.txt (line 14))
Requirement already satisfied: pathlib2>=2.2.0; python_version < "3.6" in /usr/local/lib/python2.7/dist-packages (from pytest==4.0.2->-r /my_app/requirements.txt (line 9))
Requirement already satisfied: setuptools in /usr/lib/python2.7/dist-packages (from pytest==4.0.2->-r /my_app/requirements.txt (line 9))
Requirement already satisfied: funcsigs; python_version < "3.0" in /usr/local/lib/python2.7/dist-packages (from pytest==4.0.2->-r /my_app/requirements.txt (line 9))
Requirement already satisfied: scandir; python_version < "3.5" in /usr/local/lib/python2.7/dist-packages (from pathlib2>=2.2.0; python_version < "3.6"->pytest==4.0.2->-r
/my_app/requirements.txt (line 9))
+ echo Start running test cases
Start running test cases
+ pytest testCalculator.py
============================= test session starts ==============================
platform linux2 -- Python 2.7.13, pytest-4.0.2, py-1.7.0, pluggy-0.8.0
rootdir: /var/jenkins_home/workspace/testCalculator, inifile:
collected 6 items

testCalculator.py ......                                                 [100%]

=========================== 6 passed in 0.36 seconds ===========================
Finished: SUCCESS
```

Hurray, you are reaching the end of this guide. There are many other things to learn about continuous integration but these are basically what you need to run a CI/CD for python project with Docker and Jenkins. Hope it can help you!