

**LAB 5****CI/CD PIPELINE USING JENKINS, GITHUB AND DOCKER**

Fullname: Trương Đặng Trúc Lâm

Student ID: B2111933

- Note: screenshots need to be clear and good-looking; submissions must be in PDF format.

**1. Manually dockerize a Flask project****1.1. Deploy a Flask application**

- Create a sample Flask application:

```
$mkdir cicd_tutorial ; cd cicd_tutorial
```

```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial x
b2111933@b2111933-virtualbox:~$ mkdir cicd_tutorial
b2111933@b2111933-virtualbox:~$ cd cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

Create directory **cicd\_tutorial** & change the working directory to **cicd\_tutorial** directory

```
$nano flask_docker.py
```

```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial x
b2111933@b2111933-virtualbox:~/cicd_tutorial$ nano flask_docker.py
```

```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial x
GNU nano 6.2 flask_docker.py
```

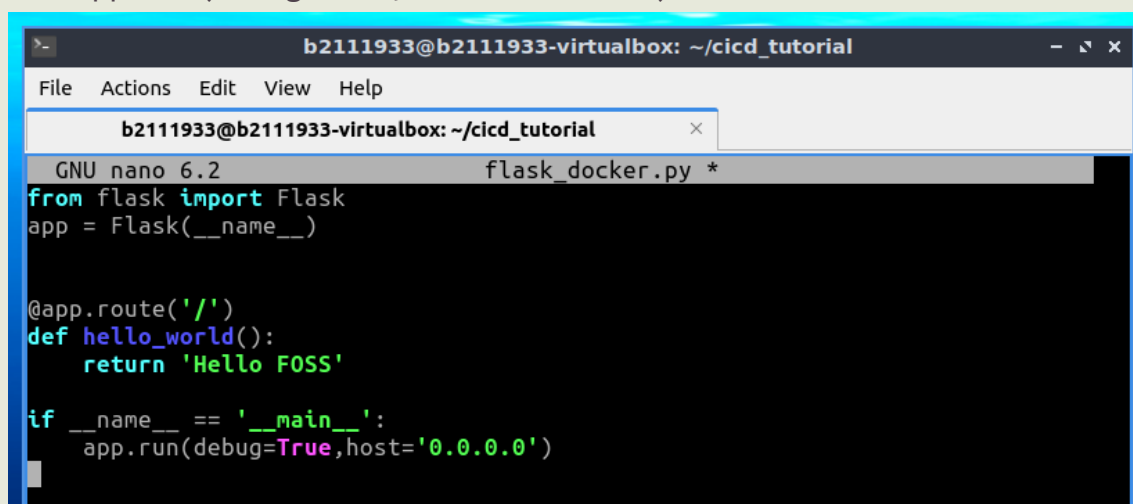
Create file **flask\_docker.py** with **nano**

**flask\_docker.py**

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello FOSS'

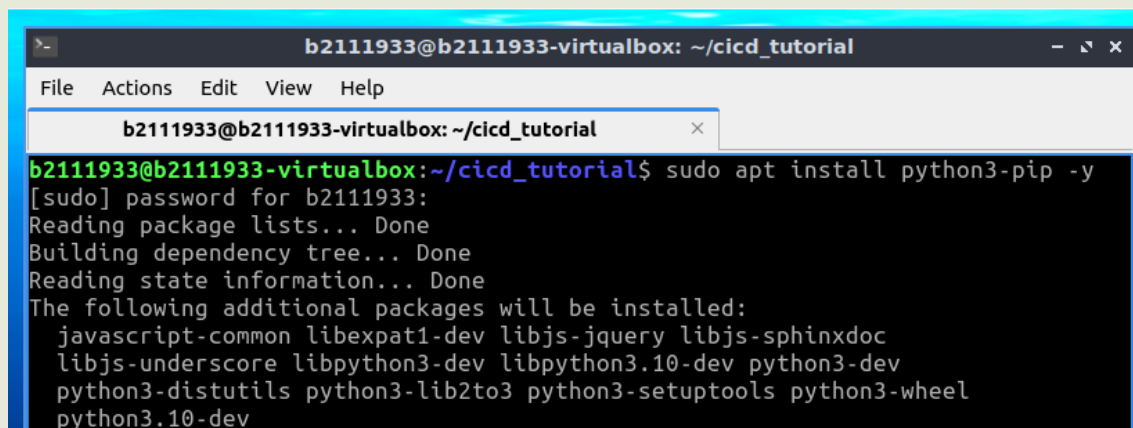
if __name__ == '__main__':
    app.run(debug=True,host='0.0.0.0')
```

A screenshot of a terminal window titled 'b2111933@b2111933-virtualbox: ~/cicd\_tutorial'. The window shows the GNU nano 6.2 editor editing the file 'flask\_docker.py'. The code displayed is the same as the one shown in the previous block: a Flask application with a single route '/' that returns 'Hello FOSS', and a main block that runs the app on 0.0.0.0 with debug mode enabled.

The contents of file **flask\_docker.py** is a simple website that runs on the IP address **0.0.0.0** . When we get access to this website, it will display “**Hello FOSS**”

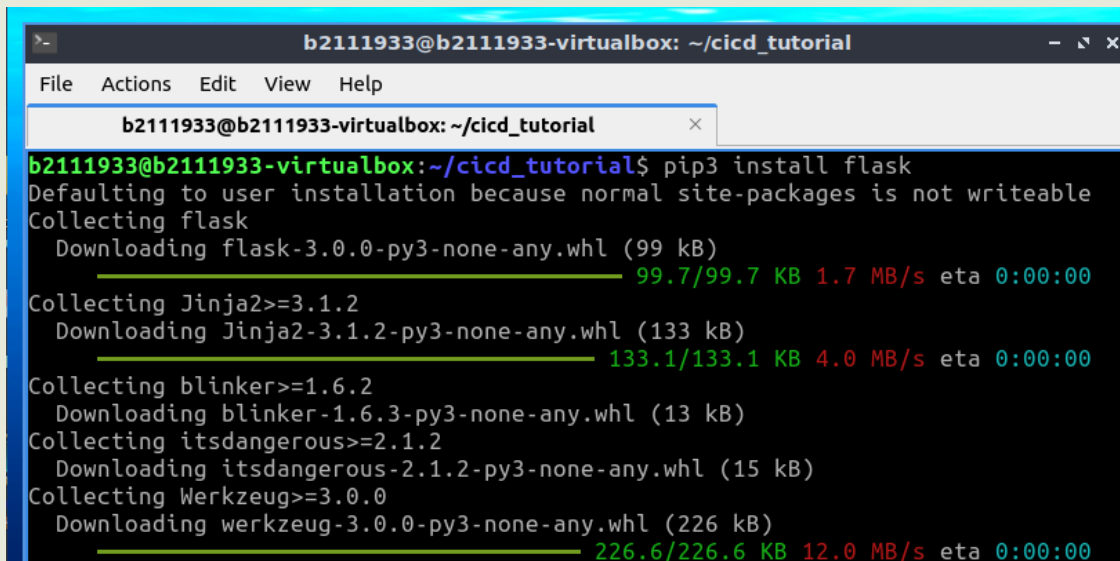
- Install pip (package installer for Python), and then the Flask framework

```
$sudo apt install python3-pip -y
```

A screenshot of a terminal window titled 'b2111933@b2111933-virtualbox: ~/cicd\_tutorial'. The terminal shows the command 'sudo apt install python3-pip -y' being executed. The output shows the password prompt, package lists being read, dependency tree being built, and state information being read. It then lists additional packages that will be installed: javascript-common, libexpat1-dev, libjs-jquery, libjs-sphinxdoc, libjs-underscore, libpython3-dev, libpython3.10-dev, python3-dev, python3-distutils, python3-lib2to3, python3-setuptools, python3-wheel, and python3.10-dev.

Install **pip** (package installer for Python)

```
$pip3 install flask
```

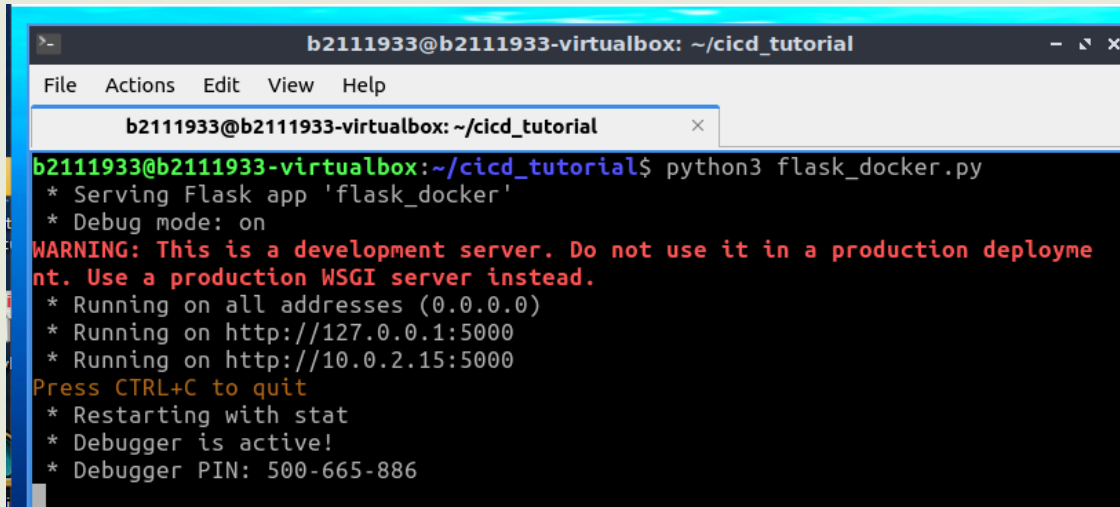


```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ pip3 install flask
Defaulting to user installation because normal site-packages is not writeable
Collecting flask
  Downloading flask-3.0.0-py3-none-any.whl (99 kB)
    _____ 99.7/99.7 KB 1.7 MB/s eta 0:00:00
Collecting Jinja2>=3.1.2
  Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
    _____ 133.1/133.1 KB 4.0 MB/s eta 0:00:00
Collecting blinker>=1.6.2
  Downloading blinker-1.6.3-py3-none-any.whl (13 kB)
Collecting itsdangerous>=2.1.2
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting Werkzeug>=3.0.0
  Downloading werkzeug-3.0.0-py3-none-any.whl (226 kB)
    _____ 226.6/226.6 KB 12.0 MB/s eta 0:00:00
```

Use **pip** to install the **Flask** framework (a necessary framework for our website)

- We can test it out by running:

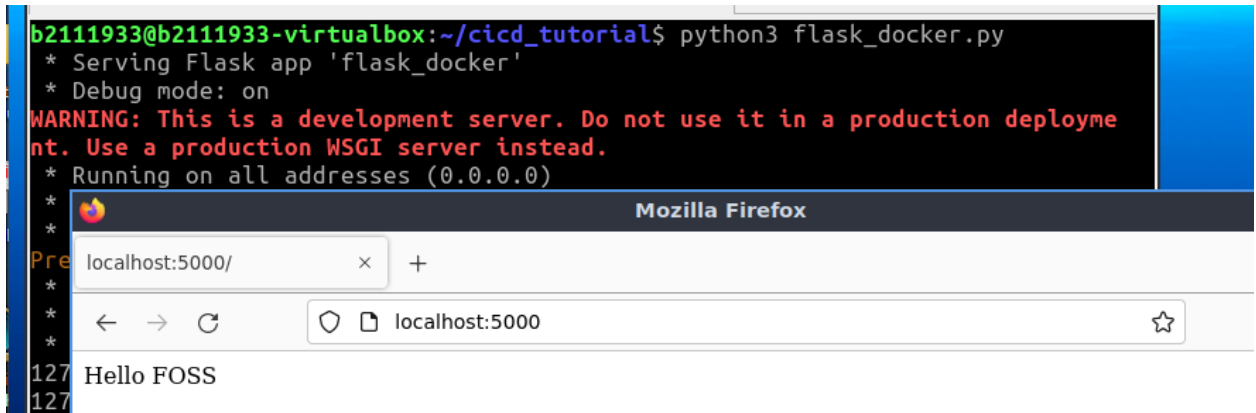
```
$python3 flask_docker.py
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 135-043-124
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ python3 flask_docker.py
* Serving Flask app 'flask_docker'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://10.0.2.15:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 500-665-886
```

The website is running on port **5000** for every IP address of my VM

- Access the application from a browser (<http://localhost:5000>), (take a screenshot)

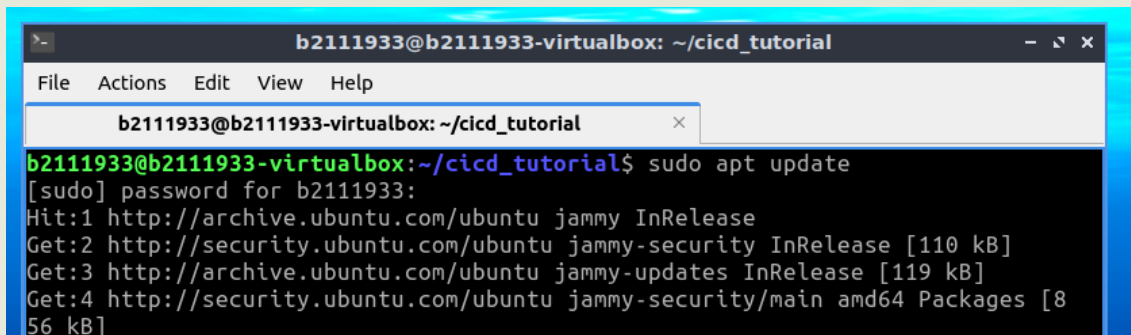


Access the website from **Mozilla Firefox** (<http://localhost:5000>)

## 1.2. Dockerize a Flask application using Dockerfile

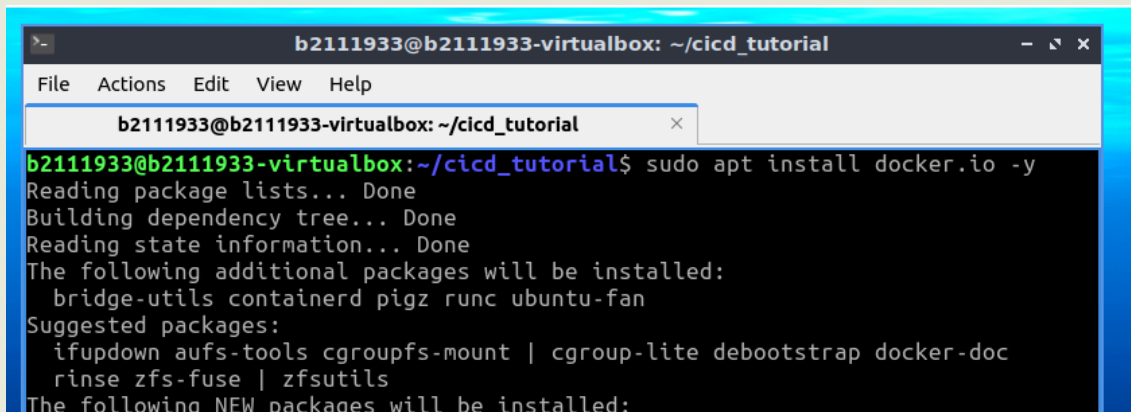
- Update the apt package index and install Docker

```
$sudo apt update
```



Update the apt package index

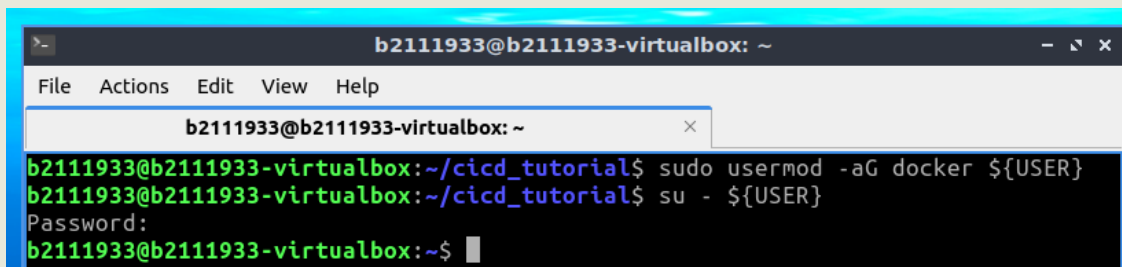
```
$sudo apt install docker.io -y
```



Install **docker.io**

- Add current user to the docker group:

```
$sudo usermod -aG docker ${USER}
$su - ${USER}
```

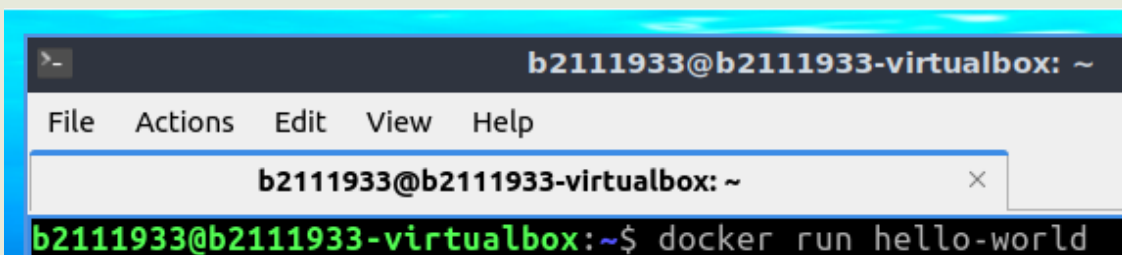
A terminal window titled 'b2111933@b2111933-virtualbox: ~' with a menu bar (File, Actions, Edit, View, Help). The terminal shows the following commands and output:

```
b2111933@b2111933-virtualbox:~/cicd_tutorial$ sudo usermod -aG docker ${USER}
b2111933@b2111933-virtualbox:~/cicd_tutorial$ su - ${USER}
Password:
b2111933@b2111933-virtualbox:~$
```

Add **current user** to the **docker group**, then login again the current user  
(no need to restart the VM or log out)

- Check whether you can access and download images from Docker Hub

```
$docker run hello-world
```

A terminal window titled 'b2111933@b2111933-virtualbox: ~' with a menu bar (File, Actions, Edit, View, Help). The terminal shows the command:

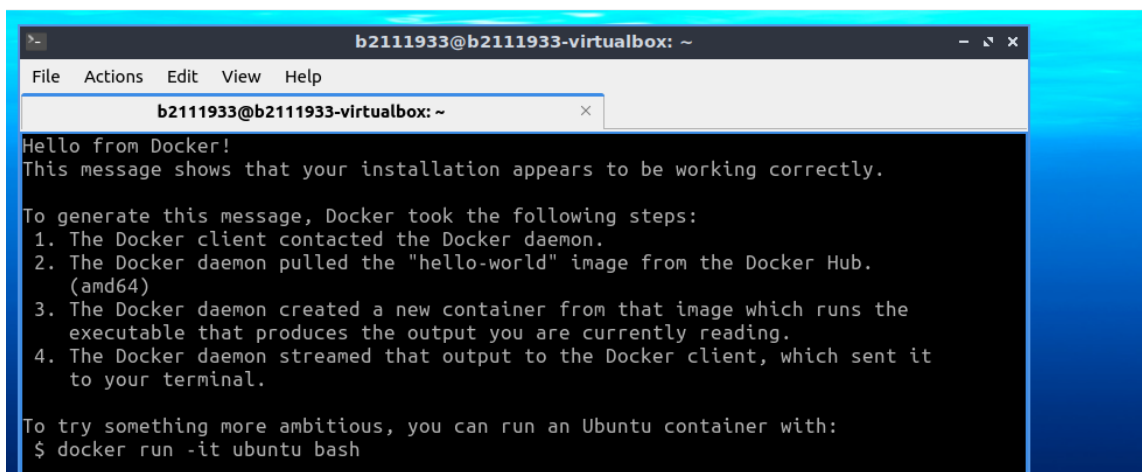
```
b2111933@b2111933-virtualbox:~$ docker run hello-world
```

Pull the image from **Docker** then execute it

The output will indicate that Docker is working correctly:

Hello from Docker!

This message shows that your installation appears to be working correctly.

A terminal window titled 'b2111933@b2111933-virtualbox: ~' with a menu bar (File, Actions, Edit, View, Help). The terminal shows the output of the 'docker run hello-world' command:

```
Hello from Docker!
This message shows that your installation appears to be working correctly.

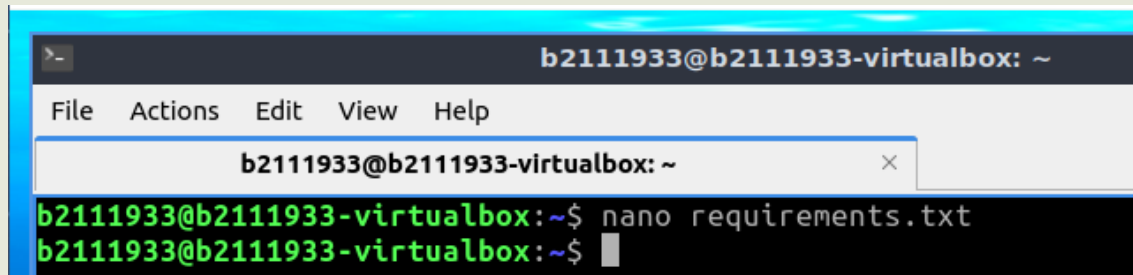
To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash
```

We can see that **Docker** is working correctly

- Create a requirements.txt file

```
$nano requirements.txt
```

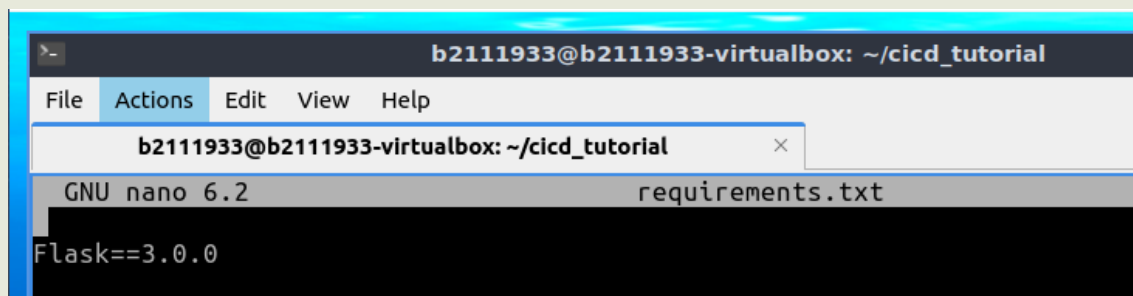
A terminal window titled 'b2111933@b2111933-virtualbox: ~' with a menu bar (File, Actions, Edit, View, Help) and a tab titled 'b2111933@b2111933-virtualbox: ~'. The terminal shows the command 'nano requirements.txt' being entered and executed, with a cursor on the next line.

```
b2111933@b2111933-virtualbox: ~  
File Actions Edit View Help  
b2111933@b2111933-virtualbox: ~  
b2111933@b2111933-virtualbox:~$ nano requirements.txt  
b2111933@b2111933-virtualbox:~$
```

Create a **requirements.txt** file with **nano**

requirements.txt

```
Flask==2.2.2
```

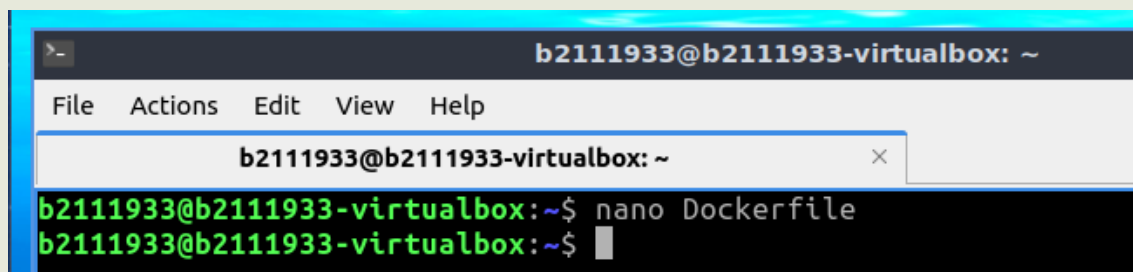
A terminal window titled 'b2111933@b2111933-virtualbox: ~/cicd\_tutorial' with a menu bar (File, Actions, Edit, View, Help) and a tab titled 'b2111933@b2111933-virtualbox: ~/cicd\_tutorial'. The terminal shows the contents of 'requirements.txt' being displayed, including the text 'Flask==3.0.0'.

```
b2111933@b2111933-virtualbox: ~/cicd_tutorial  
File Actions Edit View Help  
b2111933@b2111933-virtualbox: ~/cicd_tutorial  
GNU nano 6.2 requirements.txt  
Flask==3.0.0
```

The contents of file including the **Flask framework version 3.0.0**  
(Flask framework version 2.2.2 is not supported anymore)

- Create a Dockerfile file

```
$nano Dockerfile
```

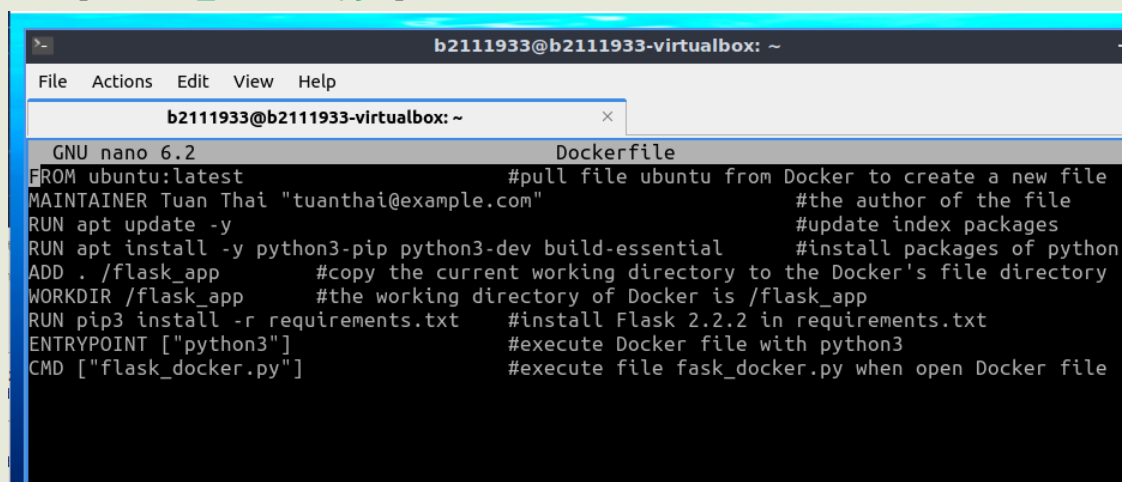
A terminal window titled 'b2111933@b2111933-virtualbox: ~' with a menu bar (File, Actions, Edit, View, Help) and a tab titled 'b2111933@b2111933-virtualbox: ~'. The terminal shows the command 'nano Dockerfile' being entered and executed, with a cursor on the next line.

```
b2111933@b2111933-virtualbox: ~  
File Actions Edit View Help  
b2111933@b2111933-virtualbox: ~  
b2111933@b2111933-virtualbox:~$ nano Dockerfile  
b2111933@b2111933-virtualbox:~$
```

Create a **Dockerfile** file with **nano**

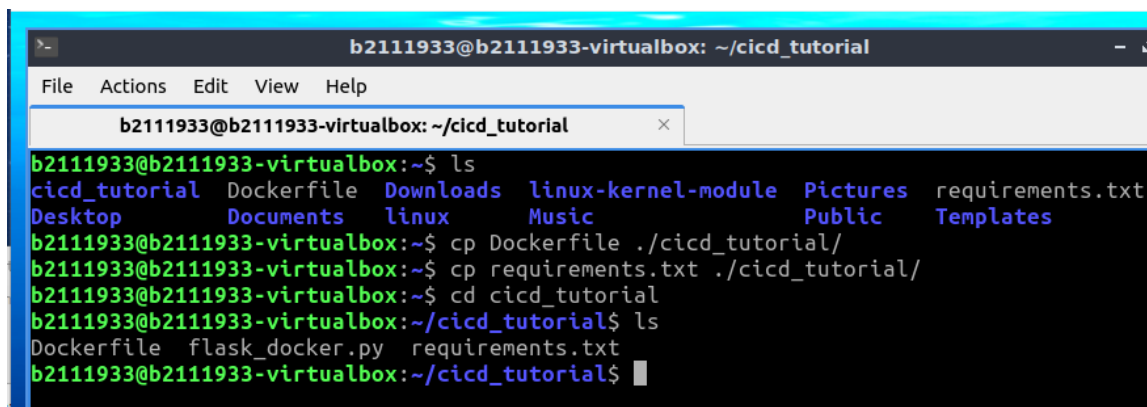
## Dockerfile

```
FROM ubuntu:latest
MAINTAINER Tuan Thai "tuanthai@example.com"
RUN apt update -y
RUN apt install -y python3-pip python3-dev build-essential
ADD . /flask_app
WORKDIR /flask_app
RUN pip3 install -r requirements.txt
ENTRYPOINT ["python3"]
CMD ["flask_docker.py"]
```

A screenshot of a terminal window titled "b2111933@b2111933-virtualbox: ~". The window shows the content of a file named "Dockerfile" using the GNU nano 6.2 editor. The file content is the same as the Dockerfile code block above. The terminal window has a menu bar with "File", "Actions", "Edit", "View", and "Help". The status bar at the bottom shows "b2111933@b2111933-virtualbox: ~".

The file will automatically do works on **Docker**

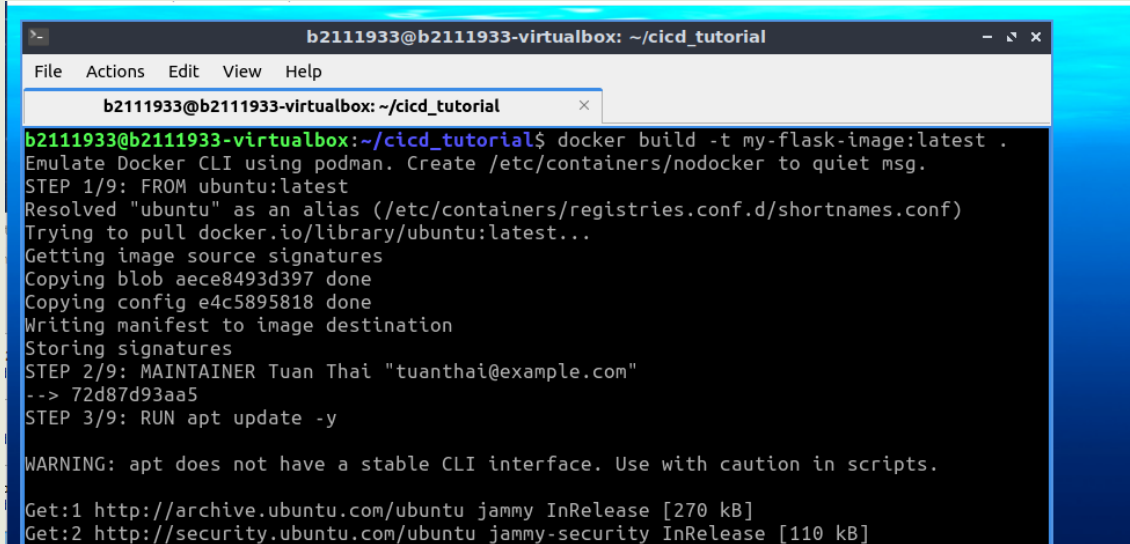
```
cp Dockerfile ./cicd_tutorial/
cp requirements.txt ./cicd_tutorial/
```

A screenshot of a terminal window titled "b2111933@b2111933-virtualbox: ~/cicd\_tutorial". The window shows the execution of several commands to copy files into the "cicd\_tutorial" directory. The commands and their outputs are: "ls" showing the current directory contents; "cp Dockerfile ./cicd\_tutorial/" copying the Dockerfile; "cp requirements.txt ./cicd\_tutorial/" copying the requirements.txt file; "cd cicd\_tutorial" changing the directory; and "ls" showing the updated directory contents including "Dockerfile", "flask\_docker.py", and "requirements.txt". The terminal window has a menu bar with "File", "Actions", "Edit", "View", and "Help". The status bar at the bottom shows "b2111933@b2111933-virtualbox: ~/cicd\_tutorial".

Copy those files to **Docker** file directory

- Create a Docker image whose name is "my-flask-image:latest", using the Dockerfile

```
$docker build -t my-flask-image:latest .
```



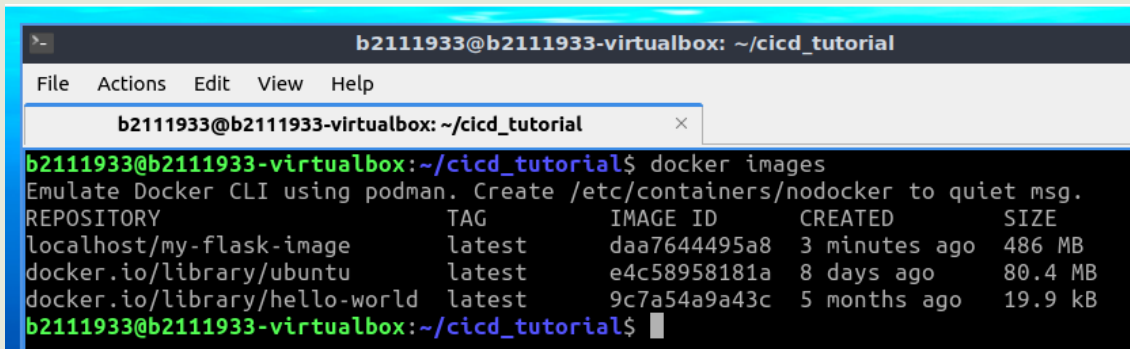
```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ docker build -t my-flask-image:latest .
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
STEP 1/9: FROM ubuntu:latest
Resolved "ubuntu" as an alias (/etc/containers/registries.conf.d/shortnames.conf)
Trying to pull docker.io/library/ubuntu:latest...
Getting image source signatures
Copying blob aece8493d397 done
Copying config e4c5895818 done
Writing manifest to image destination
Storing signatures
STEP 2/9: MAINTAINER Tuan Thai "tuanthai@example.com"
--> 72d87d93aa5
STEP 3/9: RUN apt update -y
WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
Get:1 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
```

```
STEP 8/9: ENTRYPOINT ["python3"]
--> 634e16b5ddf
STEP 9/9: CMD ["flask_docker.py"]
COMMIT my-flask-image:latest
--> daa7644495a
Successfully tagged localhost/my-flask-image:latest
daa7644495a80b17baf3b374083f3433cefc540923eb067663083030e02c9b28
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

Create a **Docker image** with the name is **my-flask-image:latest**, using the **Dockerfile**

- Then see if your image is in Docker (take a screenshot)

```
$docker images
```



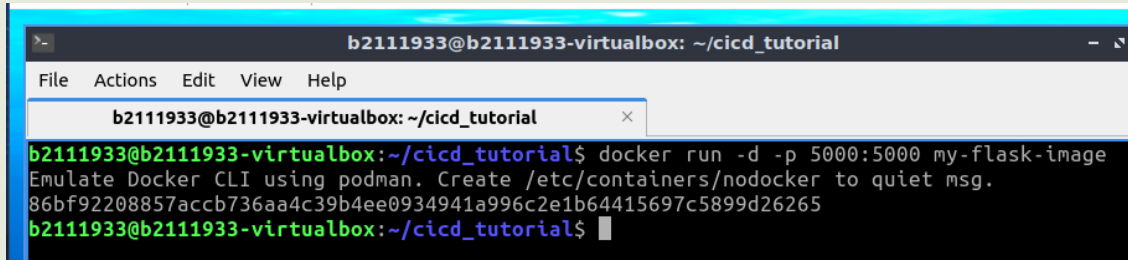
```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ docker images
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
localhost/my-flask-image  latest     daa7644495a8  3 minutes ago  486 MB
docker.io/library/ubuntu  latest     e4c58958181a  8 days ago   80.4 MB
docker.io/library/hello-world latest     9c7a54a9a43c  5 months ago  19.9 kB
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

Take a look at the image in **Docker**



- Run your image (take a screenshot)

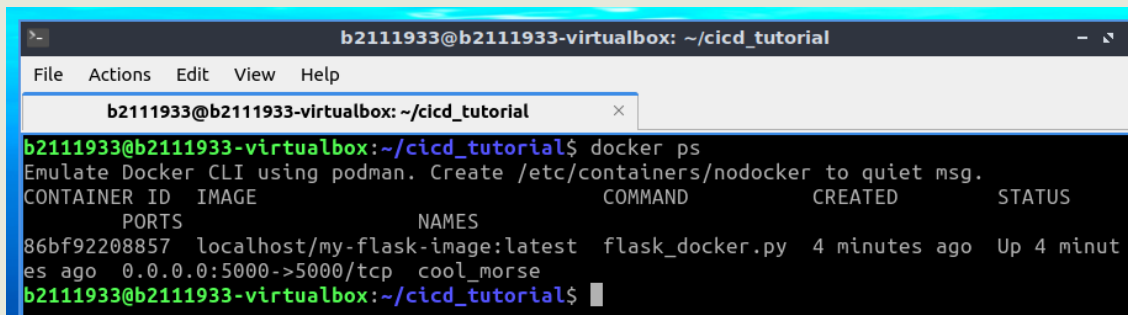
```
$docker run -d -p 5000:5000 my-flask-image
```



A terminal window titled 'b2111933@b2111933-virtualbox: ~/cicd\_tutorial'. The command 'docker run -d -p 5000:5000 my-flask-image' has been executed. The output shows a long container ID: '86bf92208857accb736aa4c39b4ee0934941a996c2e1b64415697c5899d26265'. The prompt returns to the user.

Run the image (port 5000 of the image will run on port 5000 of the VM)

```
$docker ps
```

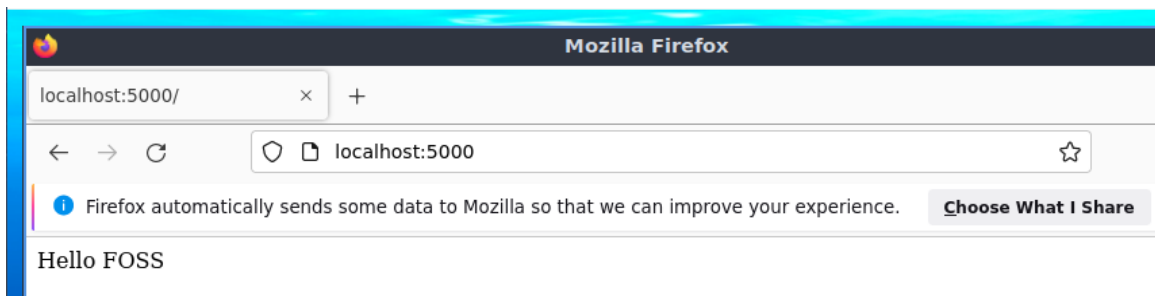


A terminal window titled 'b2111933@b2111933-virtualbox: ~/cicd\_tutorial'. The command 'docker ps' has been executed. The output shows a table of running containers:

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS
86bf92208857	localhost/my-flask-image:latest	flask_docker.py	4 minutes ago	Up 4 minutes	

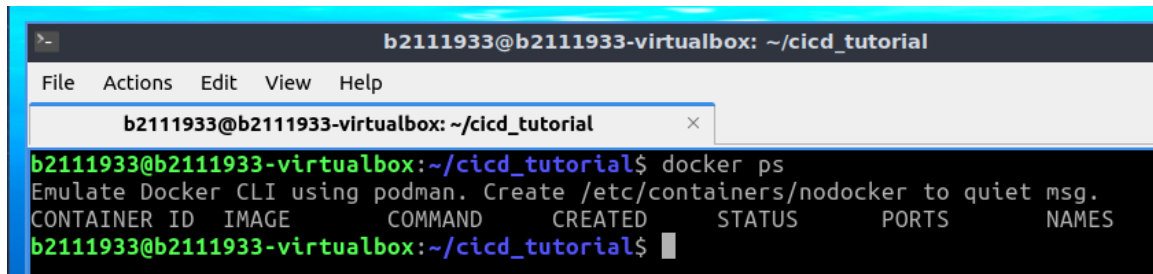
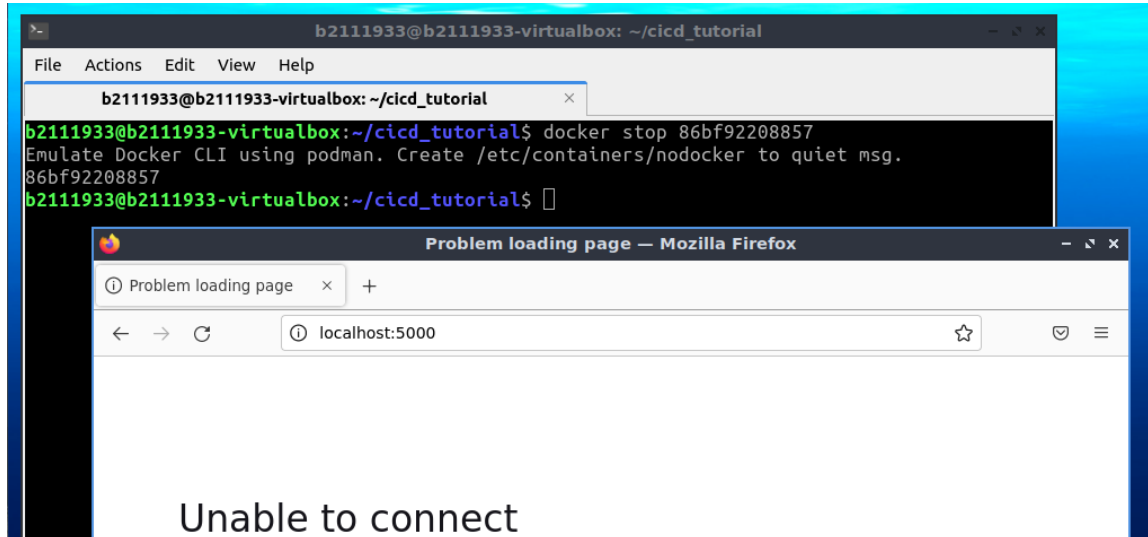
List running containers

- Access the application from a browser (<http://localhost:5000>)



Access the Docker file from Mozilla Firefox <http://localhost:5000>

Let's try to stop the container:

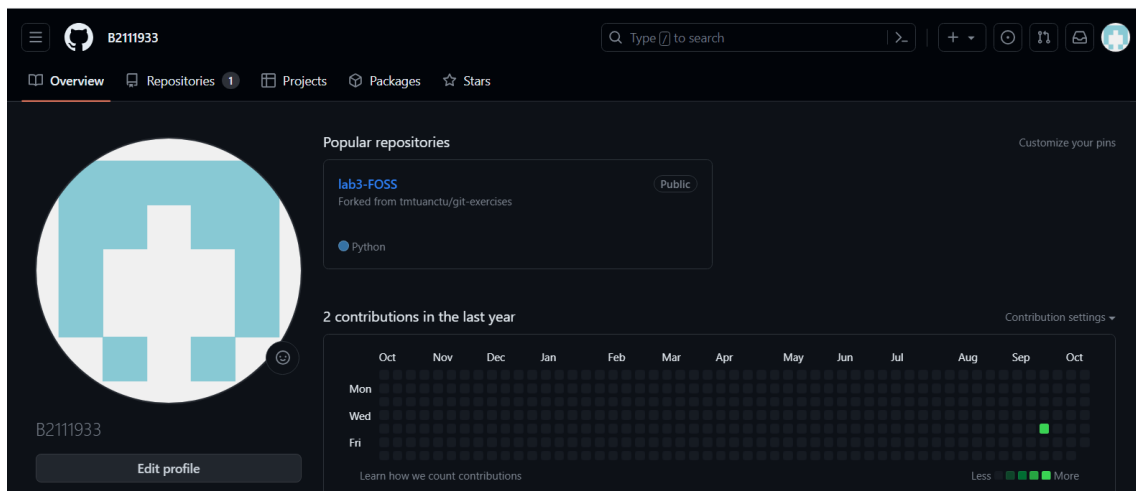


After stop the container

## 2. Automatically dockerize a Flask project using Jenkins

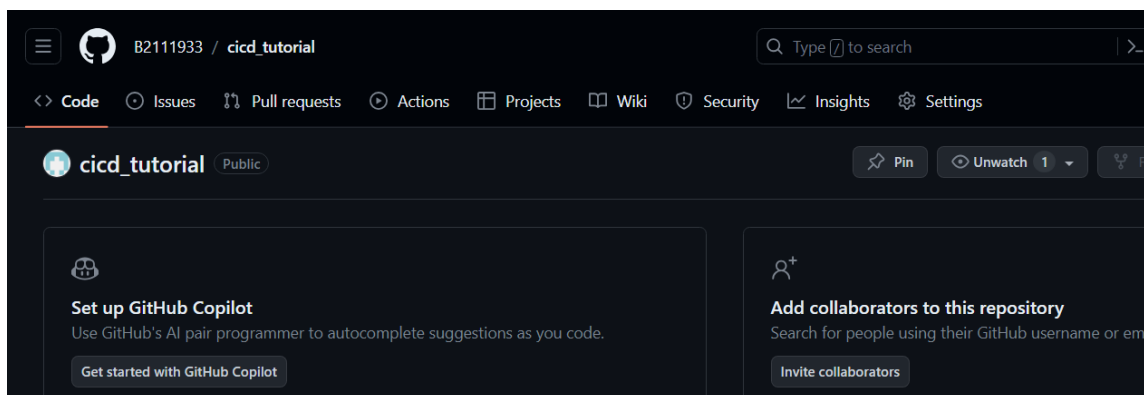
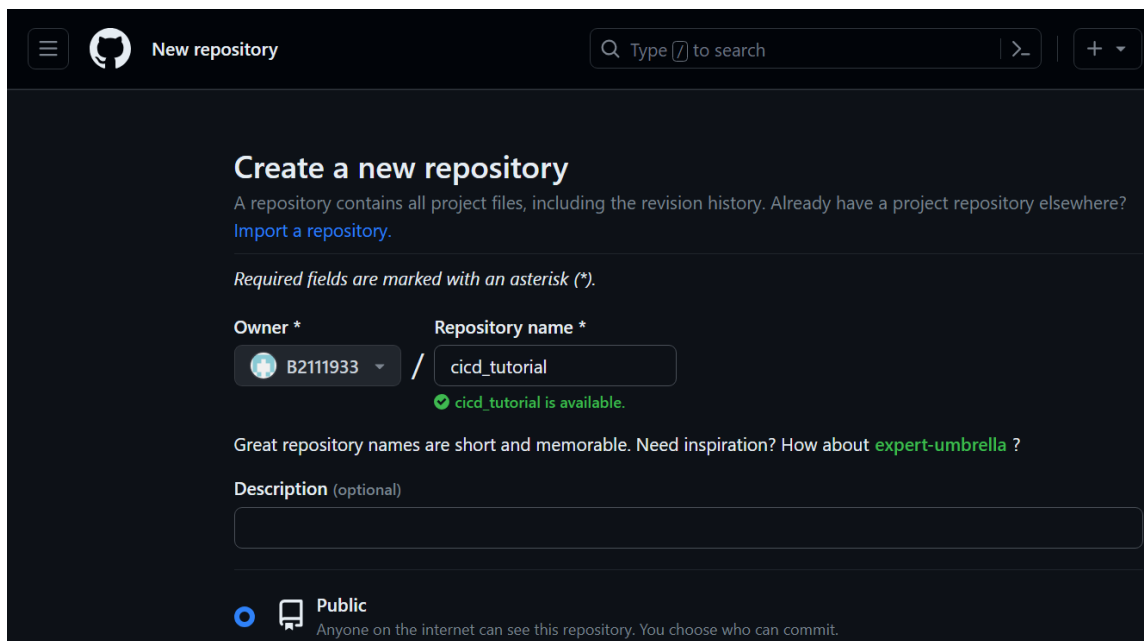
### 2.1. Push your code to a Github repository

- Create an account (or login) to GitHub at <https://github.com>



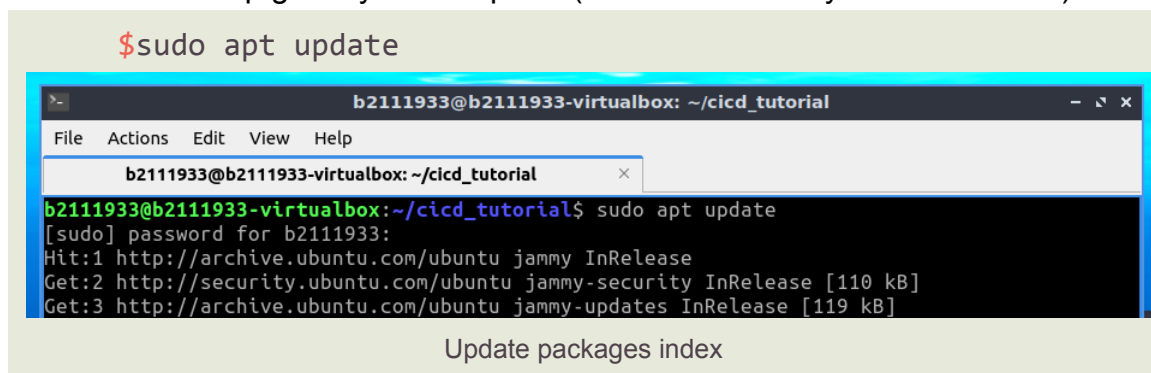
My **GitHub** account

- Create a new repository, name it as "cicd\_tutorial". Get the repository URL (for example: [https://github.com/TuanThai/cicd\\_tutorial.git](https://github.com/TuanThai/cicd_tutorial.git))

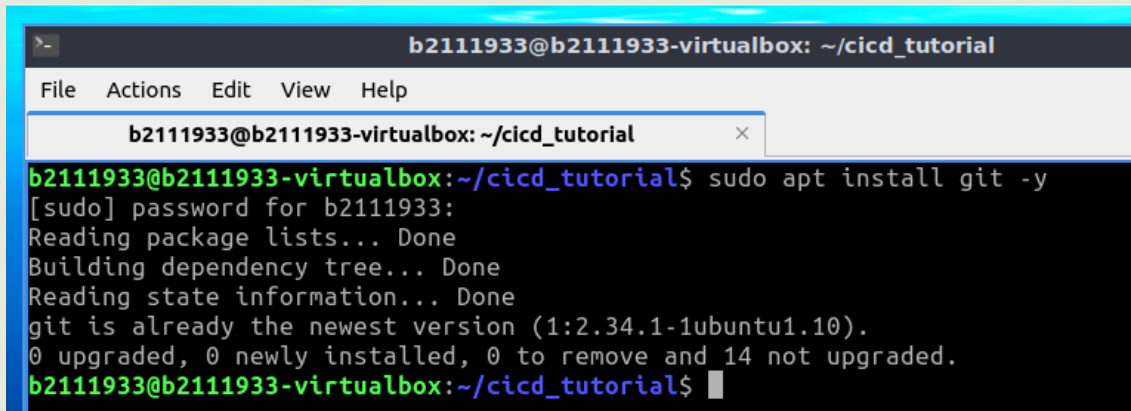


Create a new repository named "cicd\_tutorial". URL: [https://github.com/B2111933/cicd\\_tutorial](https://github.com/B2111933/cicd_tutorial)

- Install and setup git on your computer (remember to set your name/email)



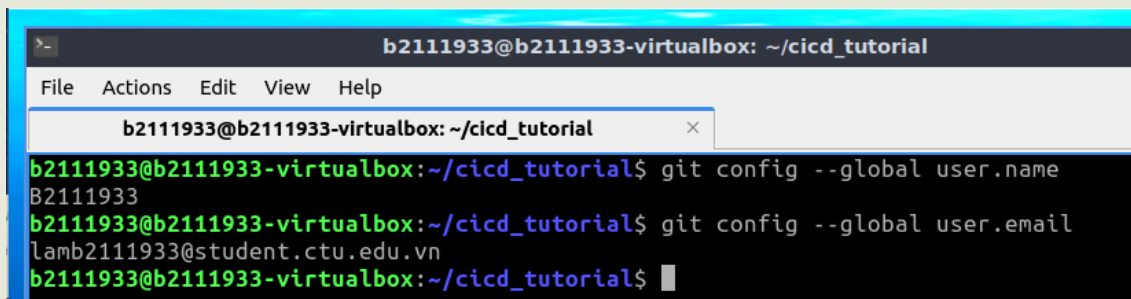
```
$sudo apt install git -y
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ sudo apt install git -y
[sudo] password for b2111933:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.34.1-1ubuntu1.10).
0 upgraded, 0 newly installed, 0 to remove and 14 not upgraded.
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

git is already installed in lab 3

```
$git config --global user.name "Firstname Lastname"
$git config --global user.email "example@ctu.edu.vn"
```

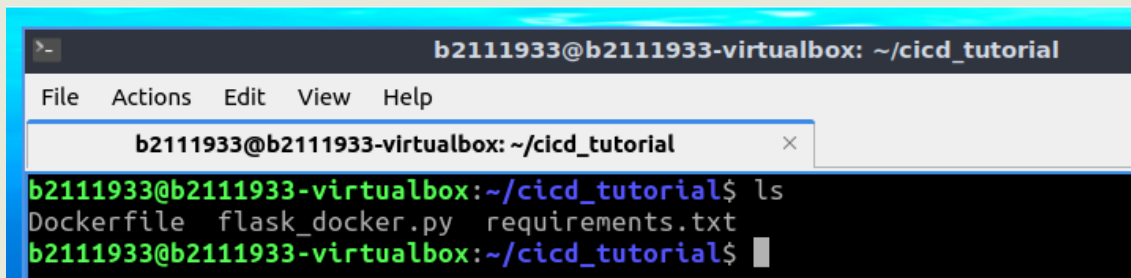


```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git config --global user.name
B2111933
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git config --global user.email
lamb2111933@student.ctu.edu.vn
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

The username and email was configured in lab 3 too

- Initialize git, commit and push your flask project files to Github

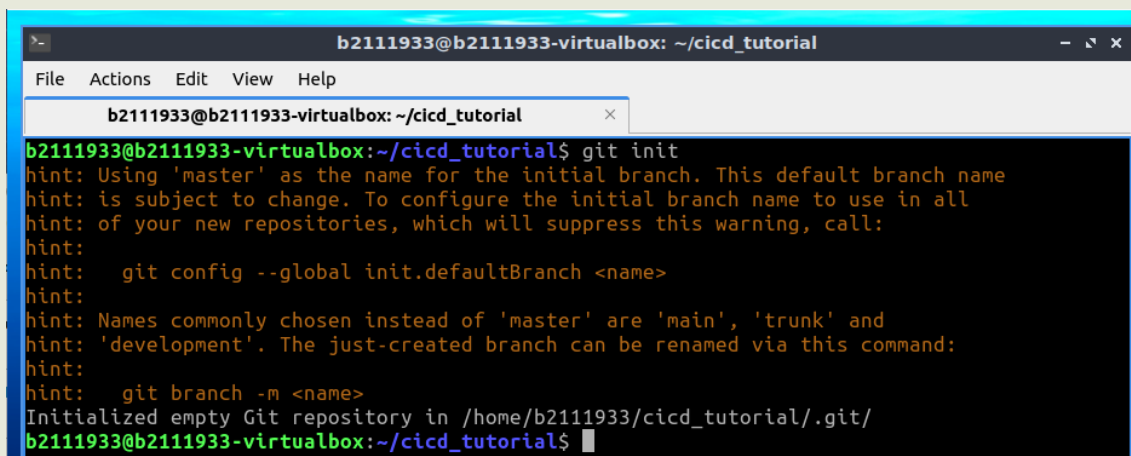
```
$mv ~/cicd_tutorial
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ ls
Dockerfile flask_docker.py requirements.txt
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

We are already in ~/cicd\_tutorial

```
$git init
```

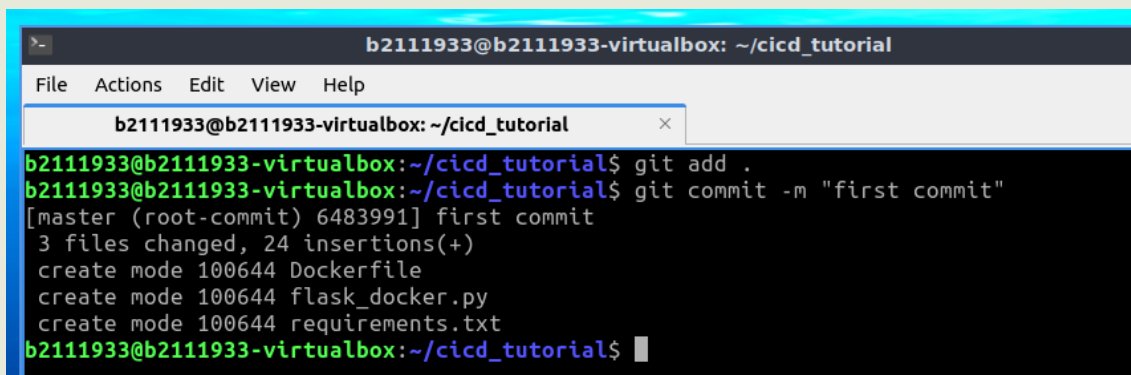


The screenshot shows a terminal window titled "b2111933@b2111933-virtualbox: ~/cicd\_tutorial". The terminal displays the command `git init` and its output. The output includes several hints from Git about the default branch name and how to configure it. It also shows that an empty Git repository has been initialized in the directory `/home/b2111933/cicd_tutorial/.git/`.

```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:   git config --global init.defaultBranch <name>
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:   git branch -m <name>
Initialized empty Git repository in /home/b2111933/cicd_tutorial/.git/
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

Create a new **Git** repository

```
$git add .
$git commit -m "first commit"
```

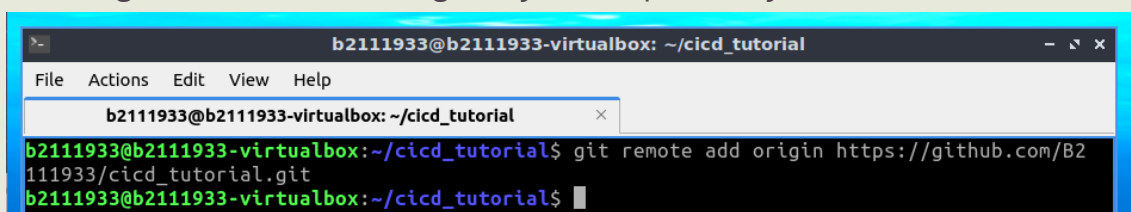


The screenshot shows a terminal window titled "b2111933@b2111933-virtualbox: ~/cicd\_tutorial". The terminal displays the commands `git add .` and `git commit -m "first commit"`. The output shows that the commit was successful, with a message indicating that 3 files were changed (24 insertions) and the files `Dockerfile`, `flask_docker.py`, and `requirements.txt` were created.

```
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git add .
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git commit -m "first commit"
[master (root-commit) 6483991] first commit
3 files changed, 24 insertions(+)
create mode 100644 Dockerfile
create mode 100644 flask_docker.py
create mode 100644 requirements.txt
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

Add and commit all files from the current working repository

```
$git remote add origin <your repository URL>
```

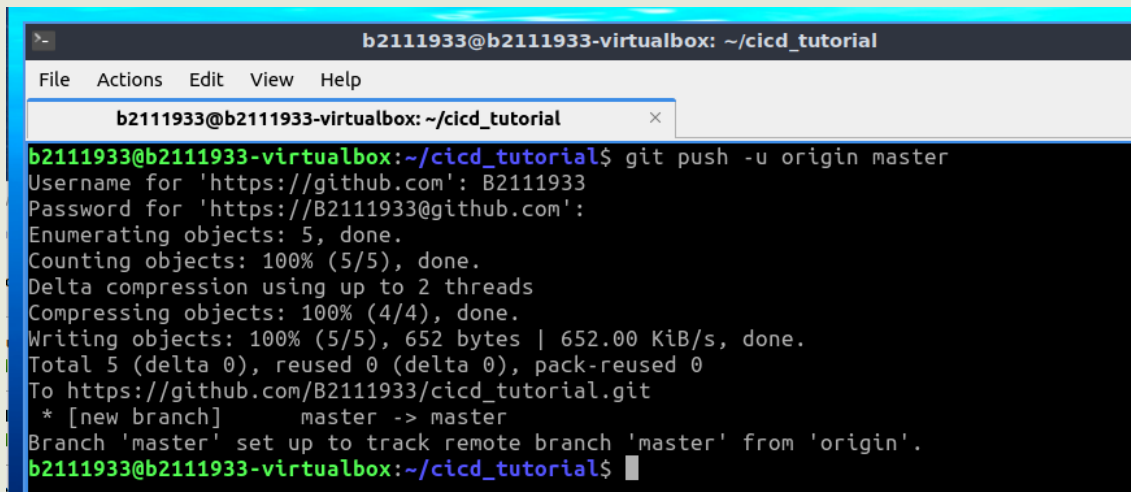


The screenshot shows a terminal window titled "b2111933@b2111933-virtualbox: ~/cicd\_tutorial". The terminal displays the command `git remote add origin https://github.com/B2111933/cicd_tutorial.git`. The output shows that the remote repository has been added successfully.

```
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git remote add origin https://github.com/B2111933/cicd_tutorial.git
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

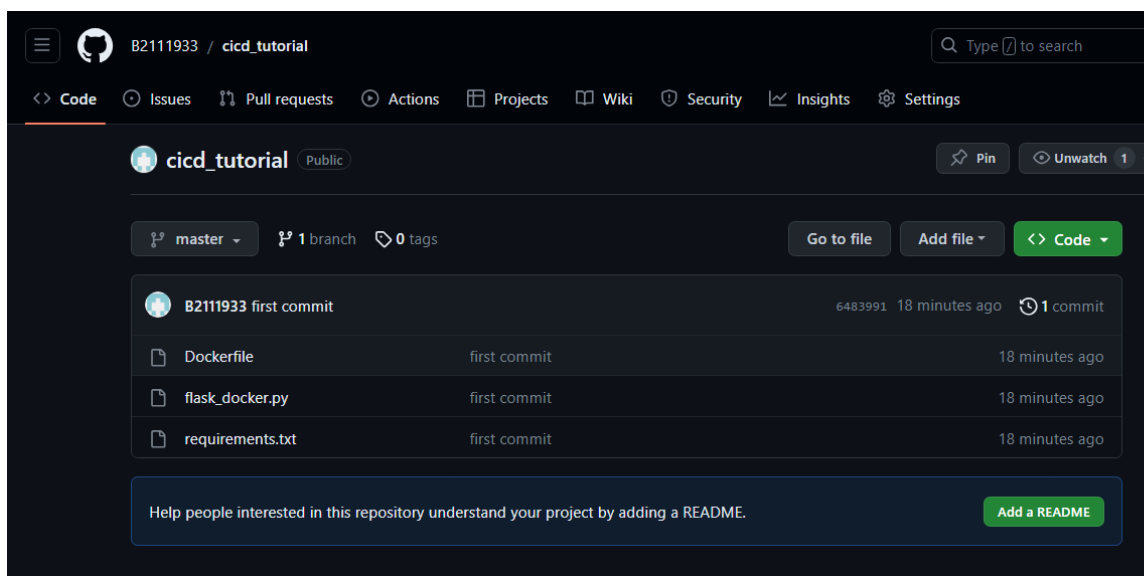
Add the **git** URL: [https://github.com/B2111933/cicd\\_tutorial](https://github.com/B2111933/cicd_tutorial) to our VM

```
$git push -u origin master
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git push -u origin master
Username for 'https://github.com': B2111933
Password for 'https://B2111933@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 652 bytes | 652.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/B2111933/cicd_tutorial.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

Push **flask** project files to **Github**

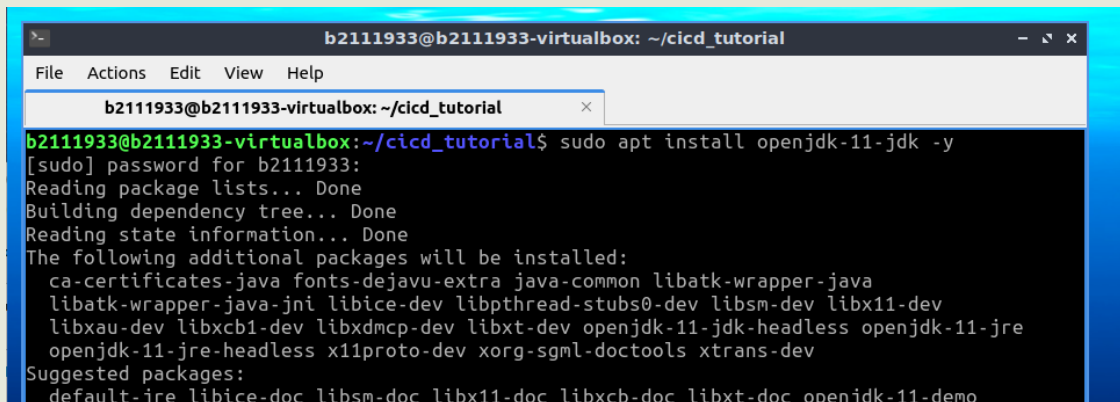


Check the result

## 2.2. Install and configure Jenkins

### - Install Java and Jenkins

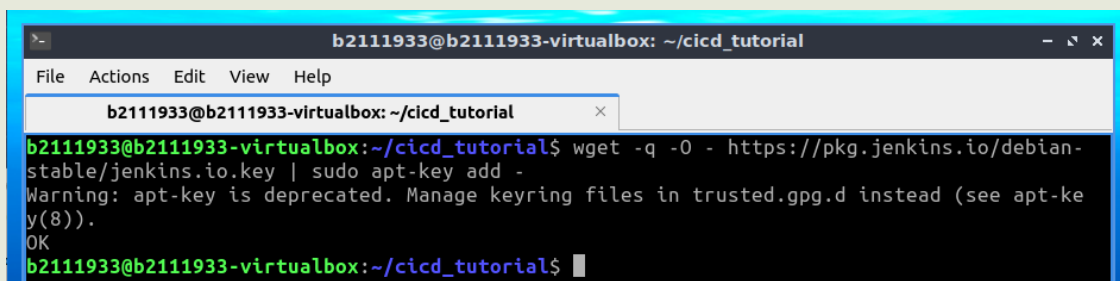
```
$sudo apt install openjdk-11-jdk -y
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ sudo apt install openjdk-11-jdk -y
[sudo] password for b2111933:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java
  libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev libx11-dev
  libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk-headless openjdk-11-jre
  openjdk-11-jre-headless x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
  default-jre libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-11-demo
```

Install **Java 11**

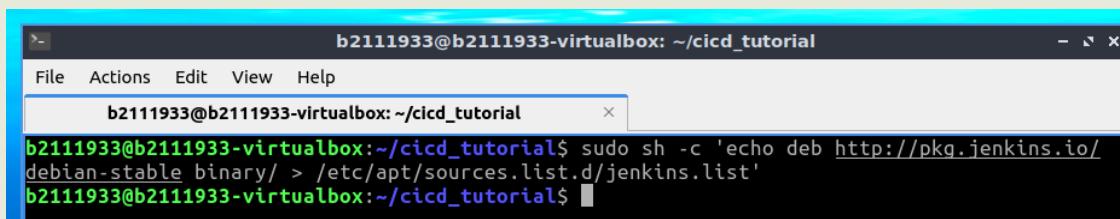
```
$wget -q -O -
https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo
apt-key add -
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ wget -q -O - https://pkg.jenkins.io/debian-
stable/jenkins.io.key | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-ke
y(8)).
OK
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

It's warning **apt-key** is deprecated, but the following part still runs for some reason

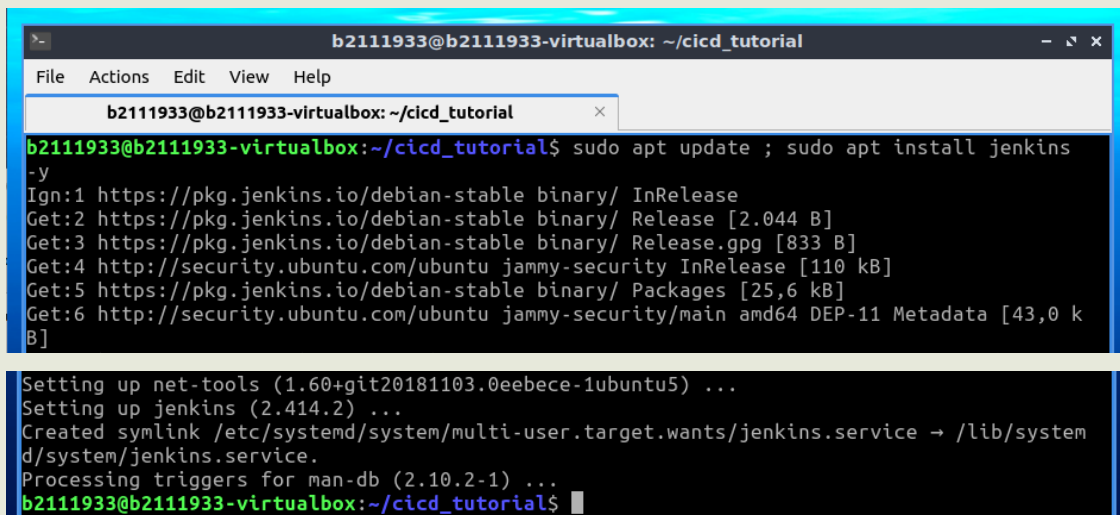
```
$sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable
binary/ > /etc/apt/sources.list.d/jenkins.list'
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
File Actions Edit View Help
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ sudo sh -c 'echo deb http://pkg.jenkins.io/
debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

Update the download link to **apt**

```
$sudo apt update ; sudo apt install jenkins -y
```



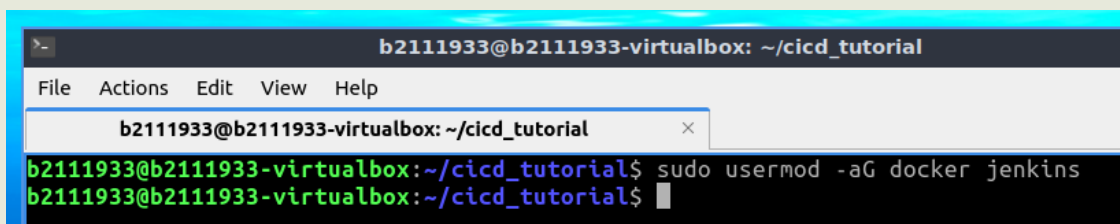
```
b2111933@b2111933-virtualbox: ~/cicd_tutorial$ sudo apt update ; sudo apt install jenkins -y
Ign:1 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:2 https://pkg.jenkins.io/debian-stable binary/ Release [2,044 B]
Get:3 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 https://pkg.jenkins.io/debian-stable binary/ Packages [25,6 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [43,0 kB]

Setting up net-tools (1.60+git20181103.0eebece-1ubuntu5) ...
Setting up jenkins (2.414.2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /lib/systemd/system/jenkins.service.
Processing triggers for man-db (2.10.2-1) ...
b2111933@b2111933-virtualbox: ~/cicd_tutorial$
```

Update apt index and install **Jenkins**

## - Launch Jenkins

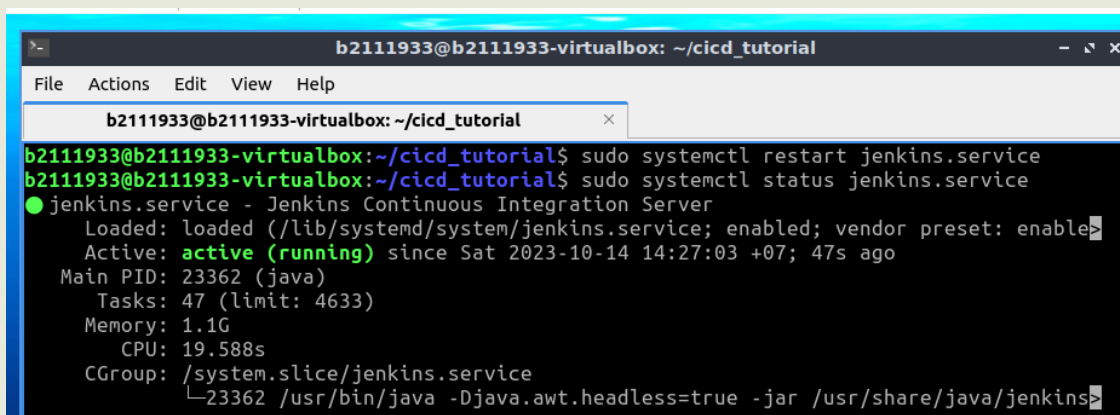
```
$sudo usermod -aG docker jenkins
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial$ sudo usermod -aG docker jenkins
b2111933@b2111933-virtualbox: ~/cicd_tutorial$
```

Add user to groups: **docker** & **jenkins** (because **jenkins** will be used to dockerize files)

```
$sudo systemctl restart jenkins.service
```

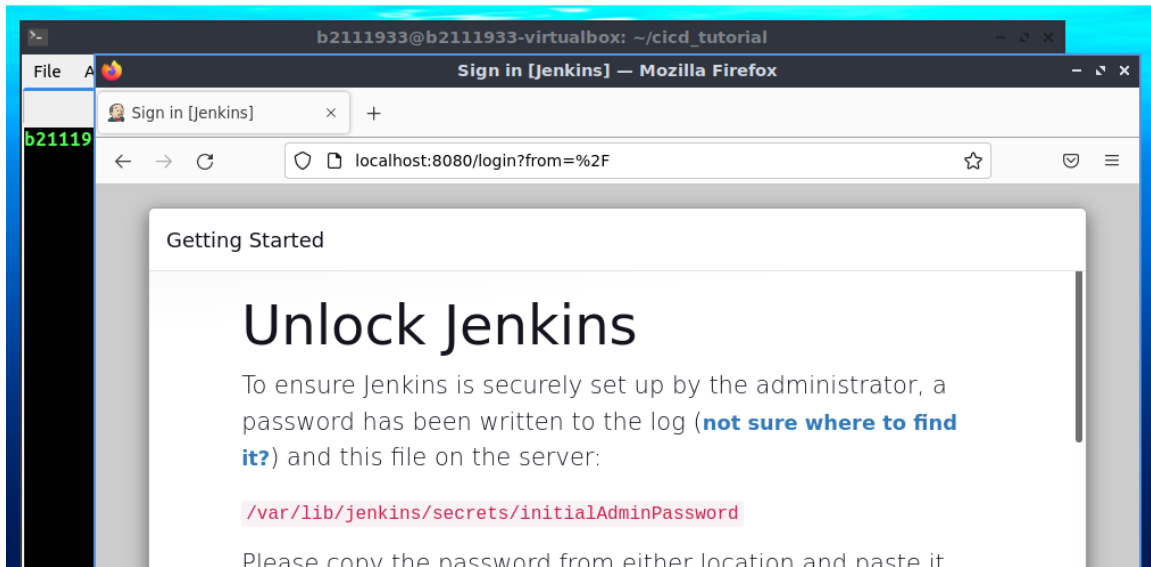


```
b2111933@b2111933-virtualbox: ~/cicd_tutorial$ sudo systemctl restart jenkins.service
b2111933@b2111933-virtualbox: ~/cicd_tutorial$ sudo systemctl status jenkins.service
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enable
   Active: active (running) since Sat 2023-10-14 14:27:03 +07; 47s ago
     Main PID: 23362 (java)
       Tasks: 47 (limit: 4633)
      Memory: 1.1G
         CPU: 19.588s
       CGroup: /system.slice/jenkins.service
              └─23362 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins
```

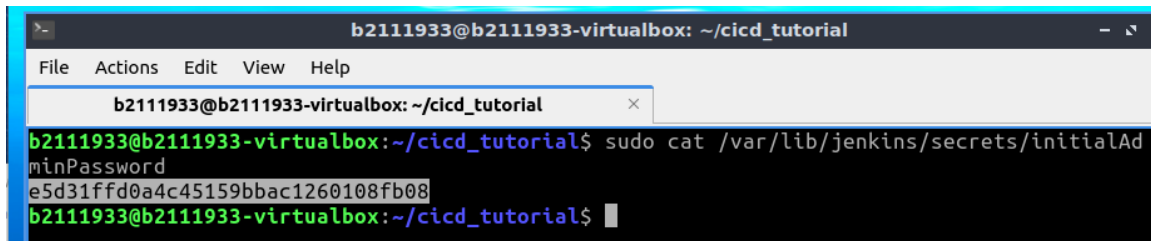
Restart **Jenkins** service and check its status



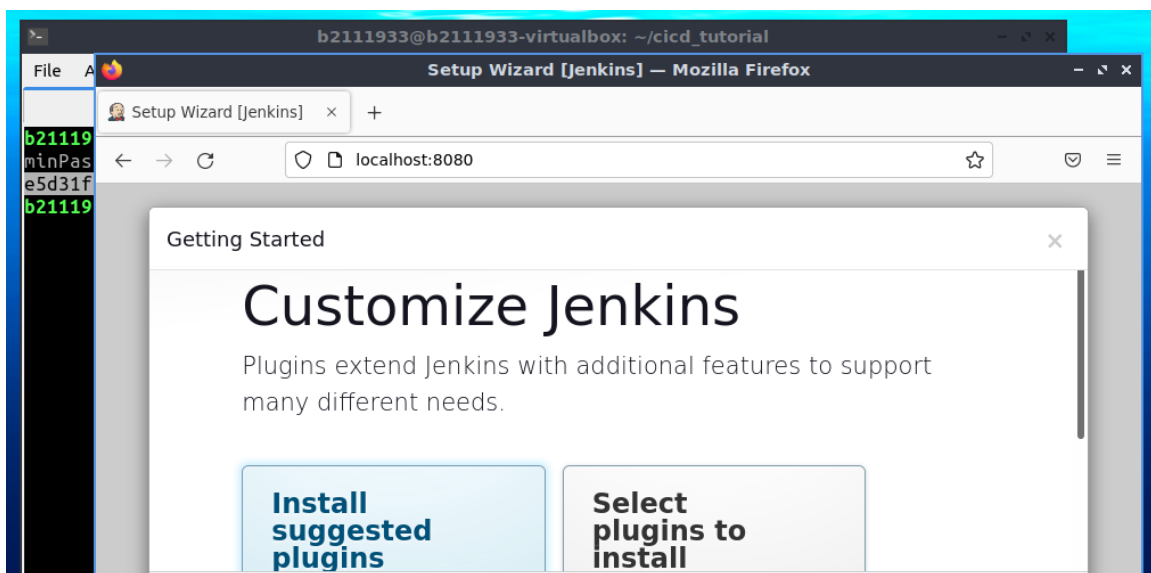
- Access Jenkins using a web browser (<http://localhost:8080>). Unlock Jenkins, install suggested plugins, create the first admin user.



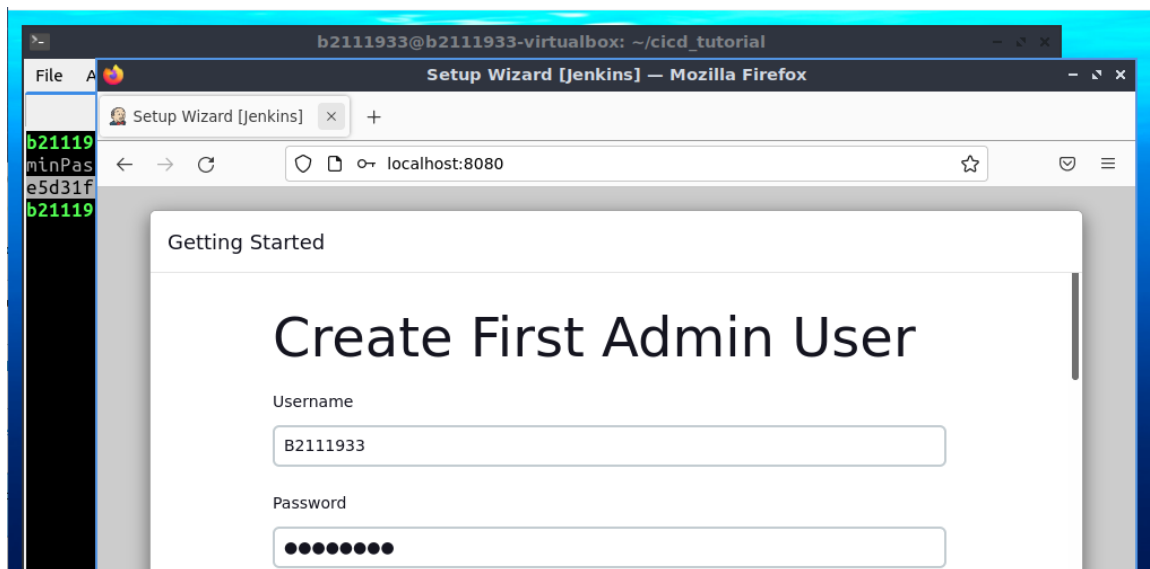
Access Jenkins via <http://localhost:8080>



Find the password that **Jenkins** request



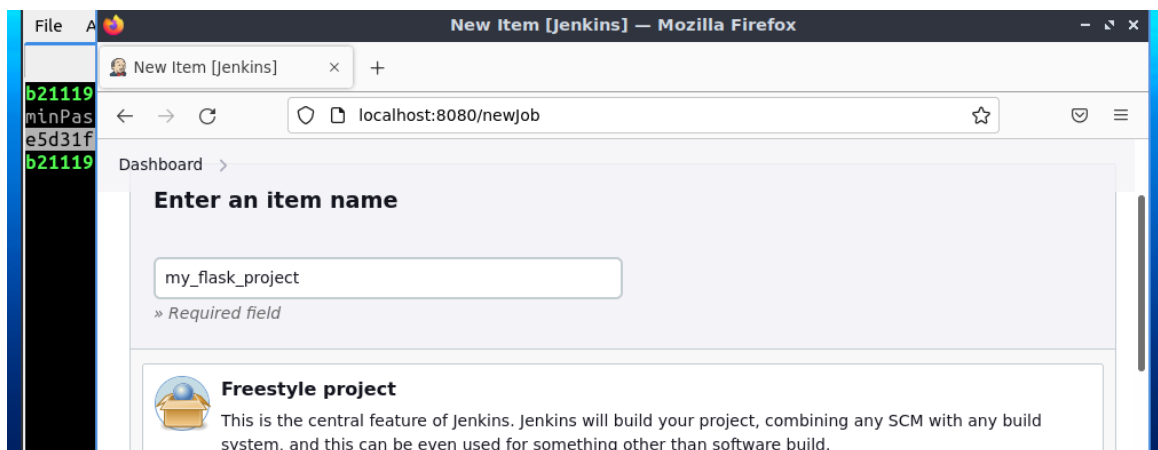
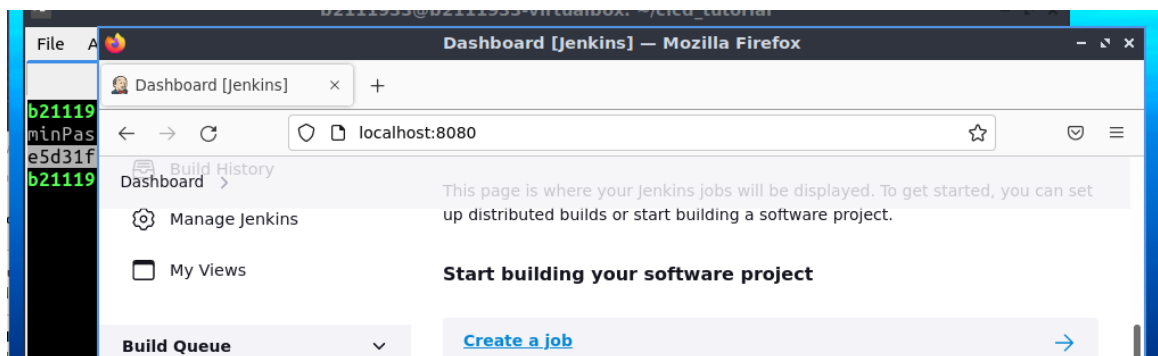
Install suggested plugins

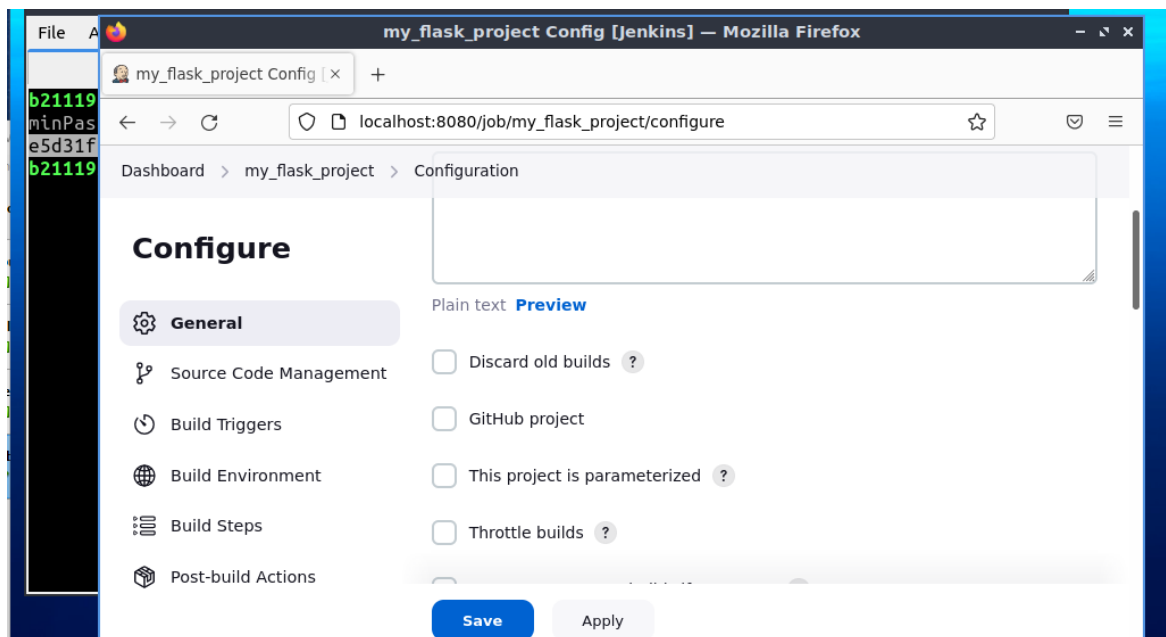


After installing plugins, we create first admin user

### 2.3. Using Jenkins to automatically dockerize your application

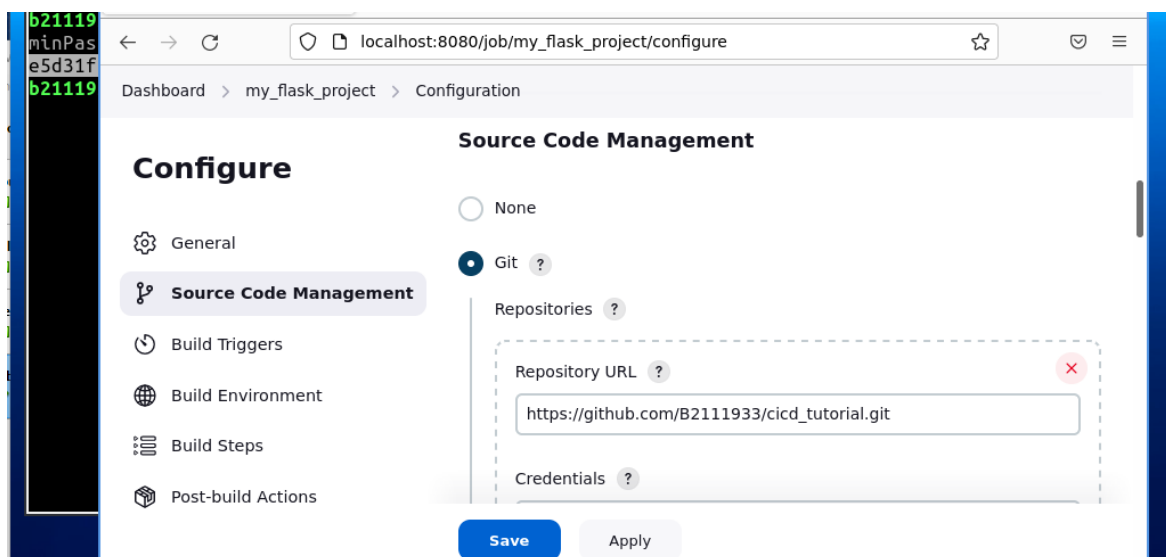
- On Jenkins dashboard, click "Create a new job", then choose "Freestyle project". Name your project as "my\_flask\_project"





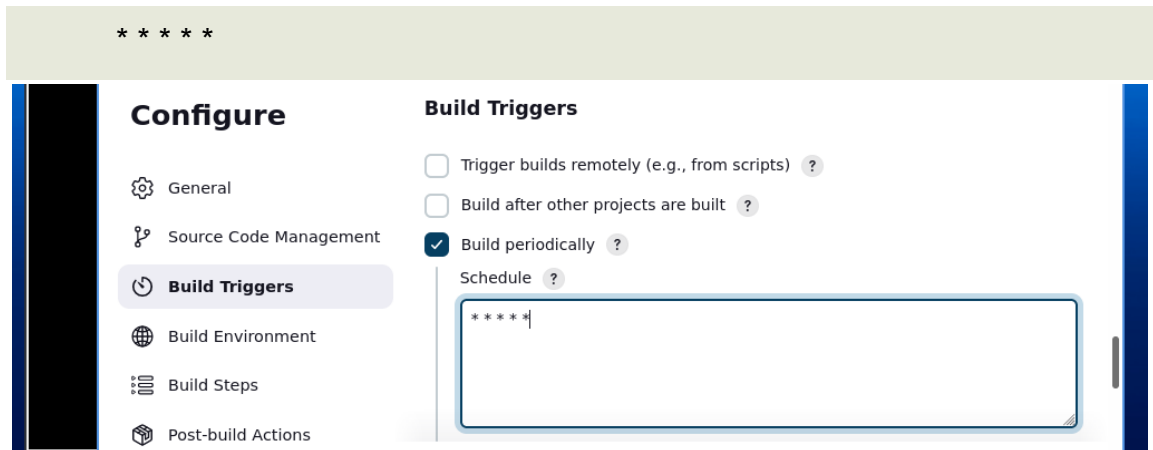
On Jenkins, create a project named "my\_flask\_project"

- Under "Source Code Management" choose "Git", fill in your GitHub repository URL



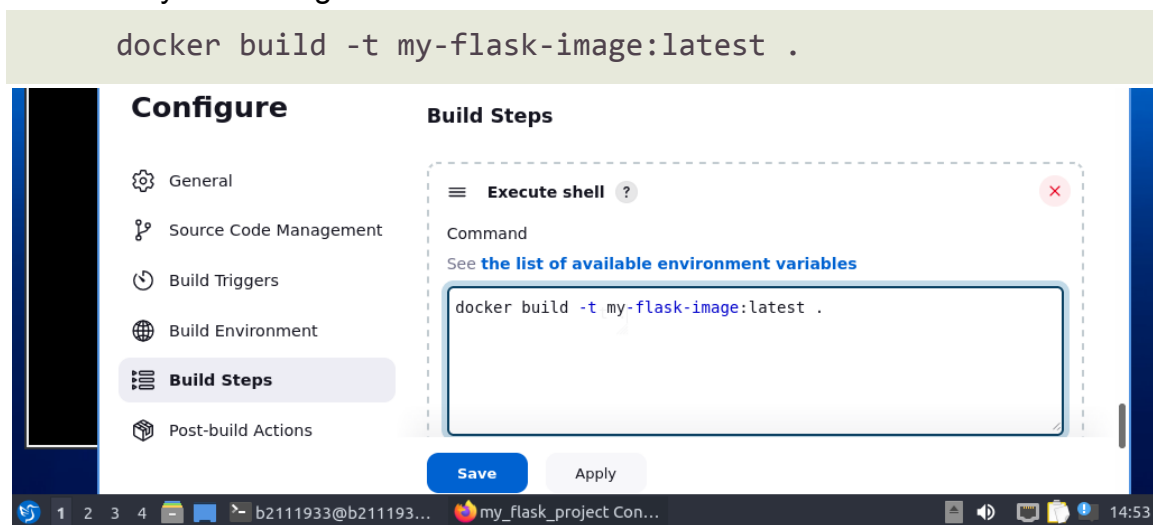
Configure the source code

- Under "Build Triggers" select "Build periodically", fill in "\* \* \* \* \*" (build your project every minute)



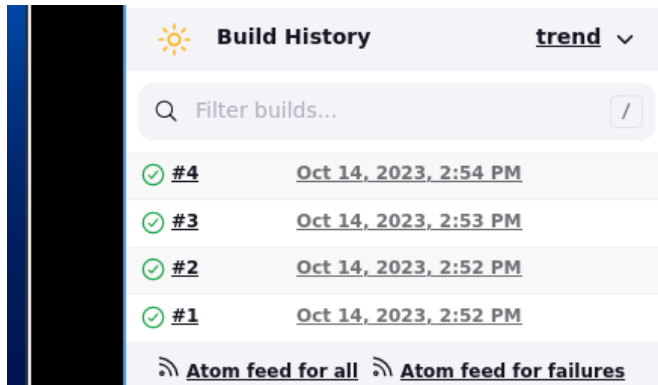
Select "**Build periodically**", fill in "\* \* \* \* \*". It will build your project every minute

- Under "Build" we will "Add build step", and select "Execute shell". Then fill in "docker build -t my-flask-image:latest ."



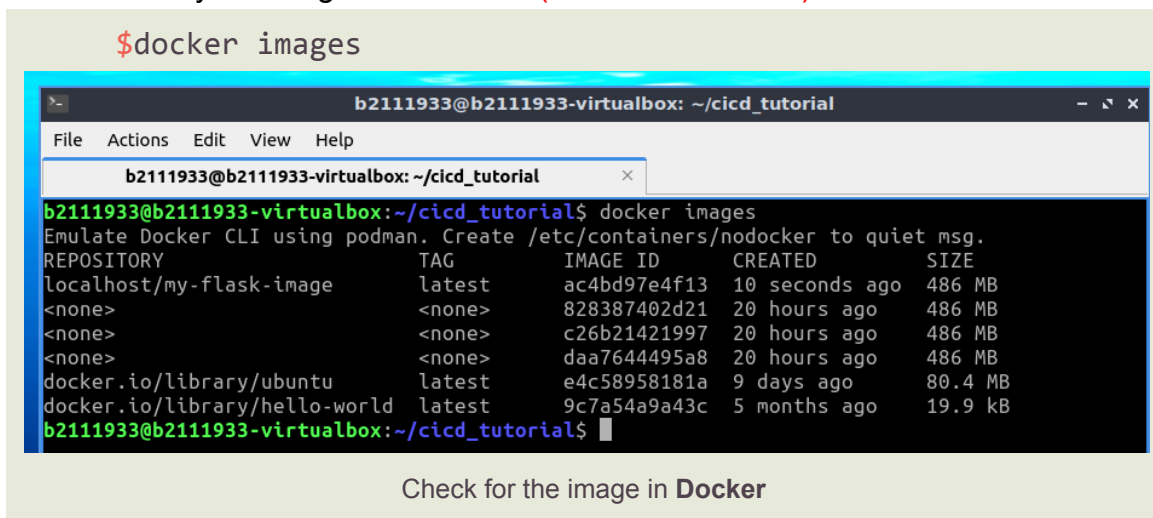
Configure **Jenkins** to build by **execute shell**

- Save your project. Then look at "Build history" to see that your project is built every minute.

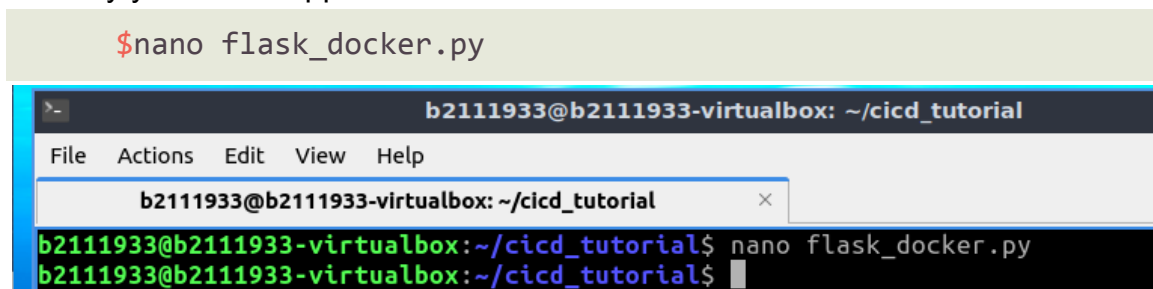


We can see that our project is built **every minute**

- Then see if your image is in Docker (take a screenshot)



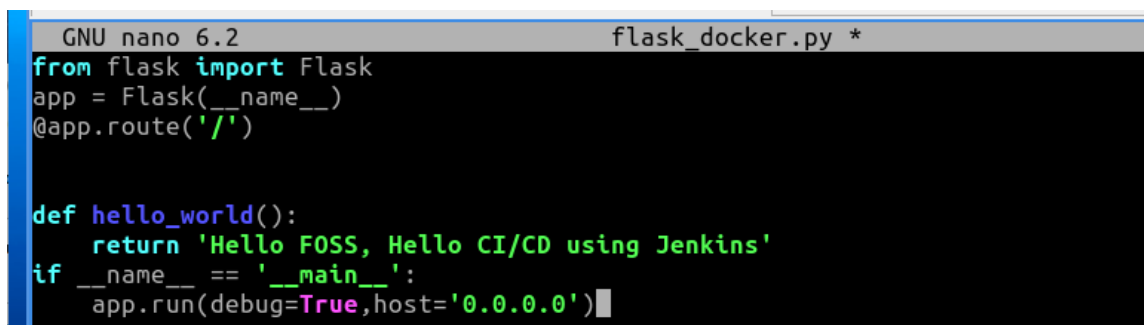
- Modify your Flask application:



```
from flask import Flask
app = Flask(__name__)
@app.route('/')

def hello_world():
    return 'Hello FOSS, Hello CI/CD using Jenkins'
if __name__ == '__main__':
    app.run(debug=True,host='0.0.0.0')
```

The contents of file:

A screenshot of a terminal window titled 'GNU nano 6.2 flask\_docker.py \*'. The terminal shows the same Python code as the previous block, with syntax highlighting: 'from flask import Flask' in blue, 'app = Flask(\_\_name\_\_)' in green, '@app.route('/')' in red, 'def hello\_world():' in yellow, 'return 'Hello FOSS, Hello CI/CD using Jenkins'' in green, 'if \_\_name\_\_ == '\_\_main\_\_':' in blue, and 'app.run(debug=True,host='0.0.0.0')' in green. The cursor is at the end of the last line.

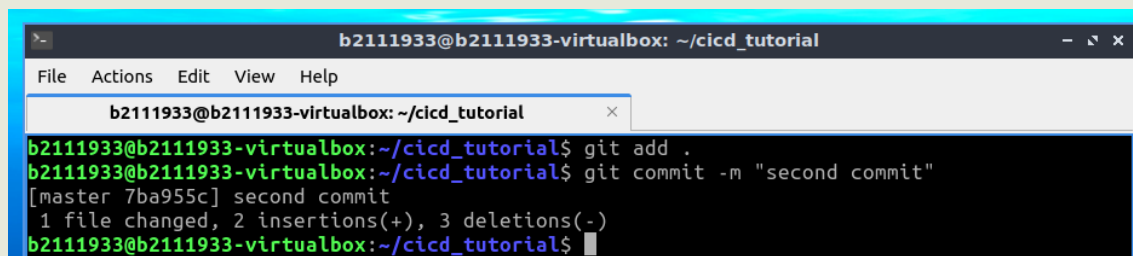
```
GNU nano 6.2 flask_docker.py *
from flask import Flask
app = Flask(__name__)
@app.route('/')

def hello_world():
    return 'Hello FOSS, Hello CI/CD using Jenkins'
if __name__ == '__main__':
    app.run(debug=True,host='0.0.0.0')
```

Modify the contents of file so the website will display: 'Hello FOSS, Hello CI/CD using Jenkins'

- Commit and push your project files to GitHub

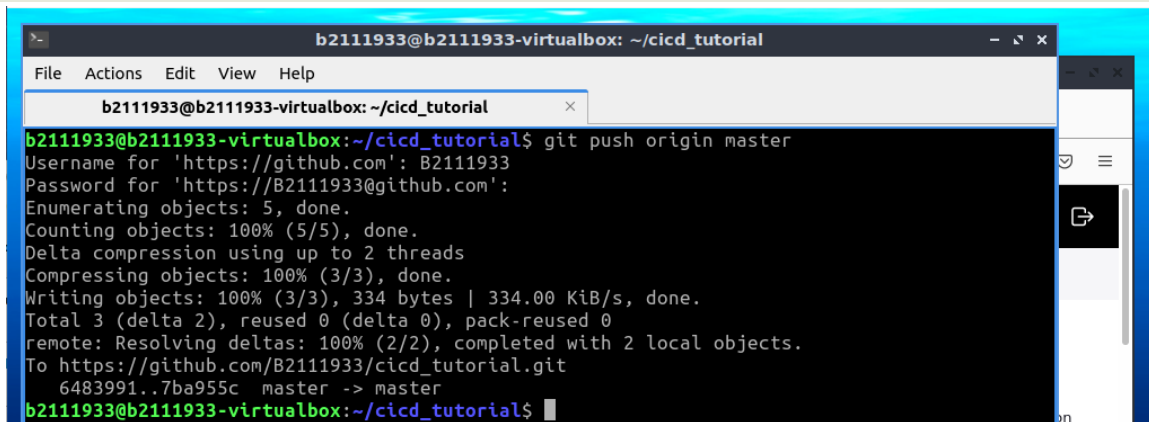
```
$git add .
$git commit -m "second commit"
```

A screenshot of a terminal window titled 'b2111933@b2111933-virtualbox: ~/cicd\_tutorial'. The terminal shows the execution of 'git add .' and 'git commit -m "second commit"', followed by the commit output: '[master 7ba955c] second commit' and '1 file changed, 2 insertions(+), 3 deletions(-)'. The prompt is at the end of the last line.

```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git add .
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git commit -m "second commit"
[master 7ba955c] second commit
1 file changed, 2 insertions(+), 3 deletions(-)
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

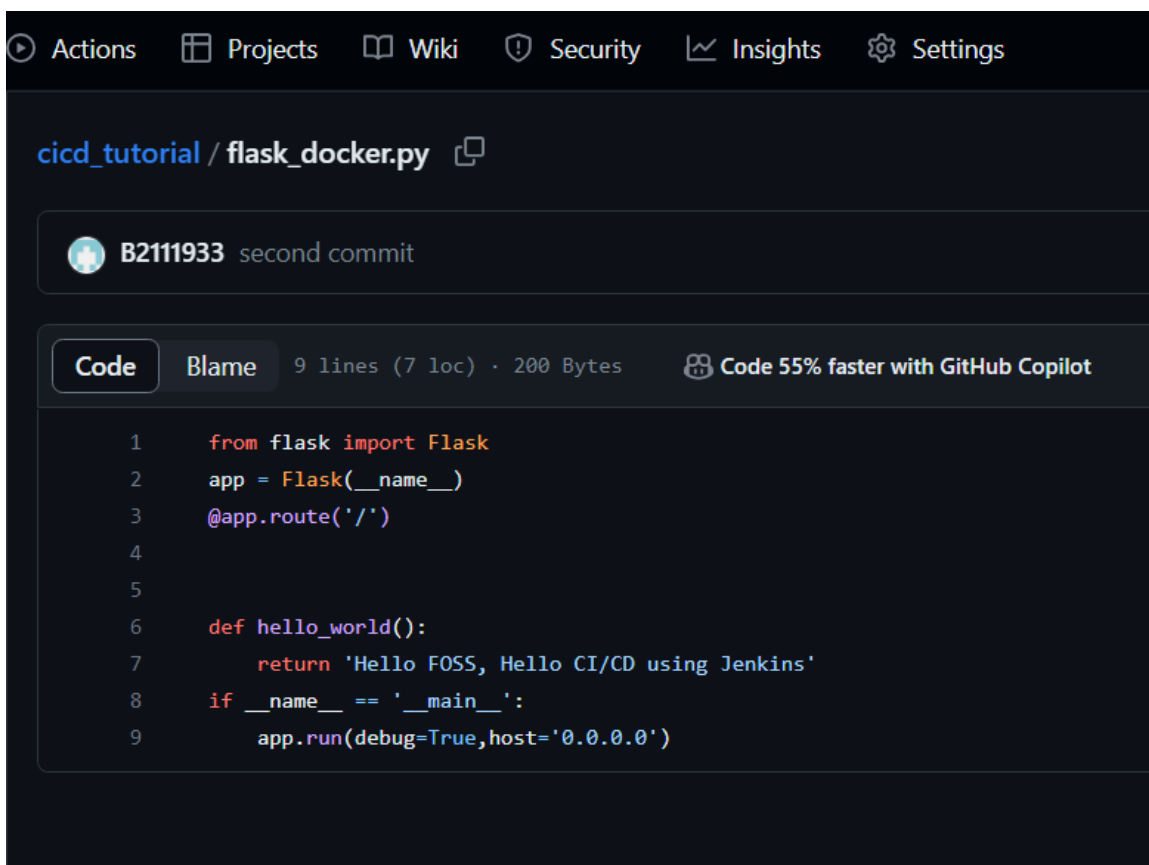
Add and commit all files from the current working repository

```
$git push origin master
```



```
b2111933@b2111933-virtualbox: ~/cicd_tutorial
b2111933@b2111933-virtualbox:~/cicd_tutorial$ git push origin master
Username for 'https://github.com': B2111933
Password for 'https://B2111933@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 334 bytes | 334.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/B2111933/cicd_tutorial.git
  6483991..7ba955c master -> master
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

Push the image to **Github**



Actions Projects Wiki Security Insights Settings

cicd\_tutorial / flask\_docker.py

B2111933 second commit

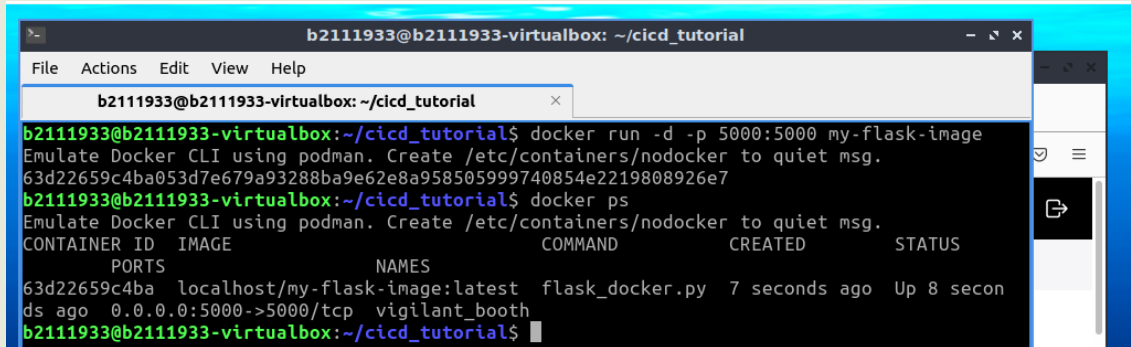
Code Blame 9 lines (7 loc) · 200 Bytes Code 55% faster with GitHub Copilot

```
1  from flask import Flask
2  app = Flask(__name__)
3  @app.route('/')
4
5
6  def hello_world():
7      return 'Hello FOSS, Hello CI/CD using Jenkins'
8  if __name__ == '__main__':
9      app.run(debug=True,host='0.0.0.0')
```

Check the result

- Wait 1 minute, then run your image

```
$docker run -d -p 5000:5000 my-flask-image  
$docker ps
```

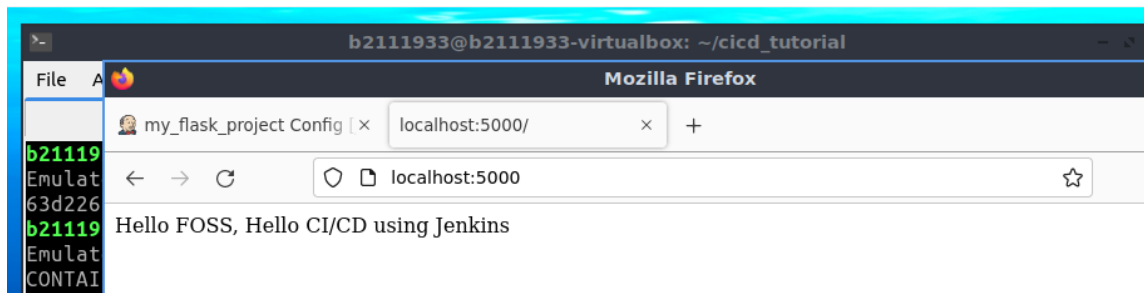


A terminal window titled 'b2111933@b2111933-virtualbox: ~/cicd\_tutorial'. It shows the execution of 'docker run -d -p 5000:5000 my-flask-image' and 'docker ps'. The output of 'docker ps' shows a container named 'vigilant\_booth' with ID '63d22659c4ba' running 'flask\_docker.py' on port 5000.

```
b2111933@b2111933-virtualbox: ~/cicd_tutorial  
b2111933@b2111933-virtualbox:~/cicd_tutorial$ docker run -d -p 5000:5000 my-flask-image  
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.  
63d22659c4ba053d7e679a93288ba9e62e8a95850599740854e2219808926e7  
b2111933@b2111933-virtualbox:~/cicd_tutorial$ docker ps  
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.  
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS  
63d22659c4ba   localhost/my-flask-image:latest     flask_docker.py         7 seconds ago Up 8 seconds  
0.0.0.0:5000->5000/tcp vigilant_booth  
b2111933@b2111933-virtualbox:~/cicd_tutorial$
```

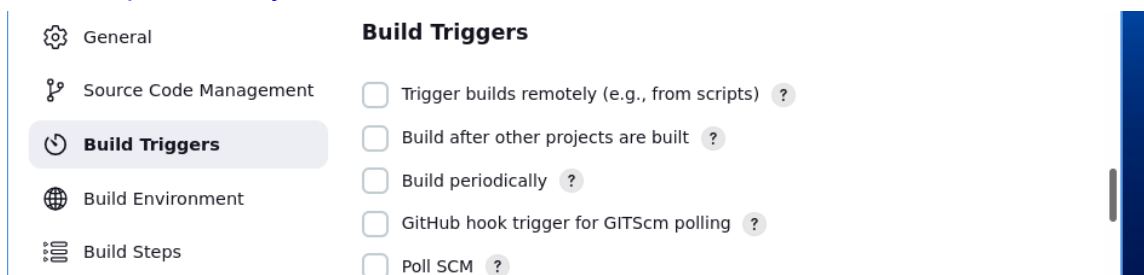
Run the image and check its status

- Access the application from a browser (<http://localhost:5000>) (take a screenshot)



Access the application via **<http://localhost:5000>**

- On your Jenkins project configure, under "Build Triggers", do not forget to **deselect** "Build periodically"



Uncheck the "**Build periodically**" to avoid wasting resources

---END---



