LAB 5

CI/CD PIPELINE USING JENKINS, GITHUB AND DOCKER

Fullname: Tran Dang Khoa

Student ID: B2014926

- 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

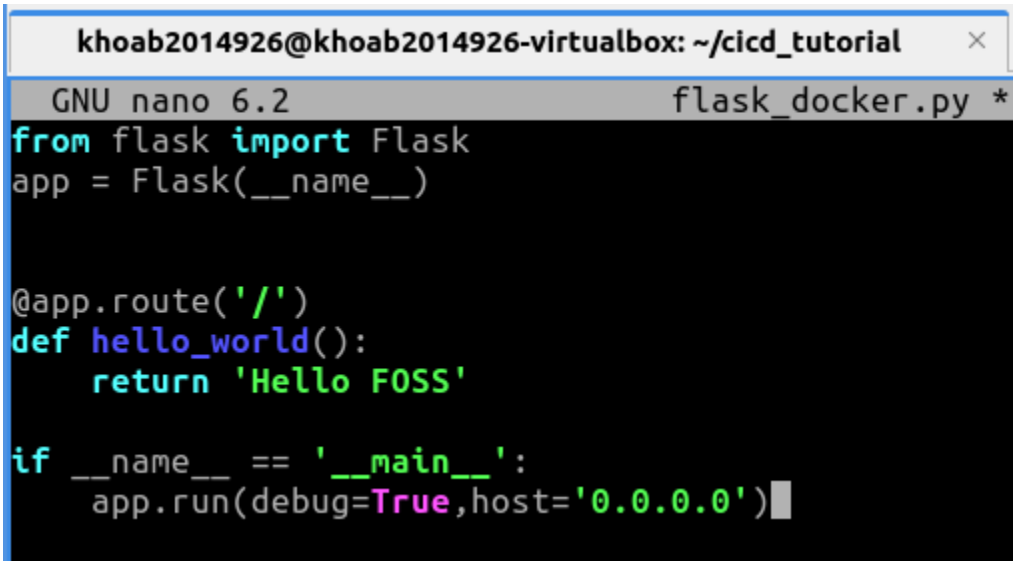
- Create a sample Flask application:

```
$mkdir cicd_tutorial ; cd cicd_tutorial
$nano flask_docker.py
```

```
khoab2014926@khoab2014926-virtualbox:~$ mkdir cicd_tutorial ; cd cicd_tutoria
l
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ nano flask_docker.py
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$
```

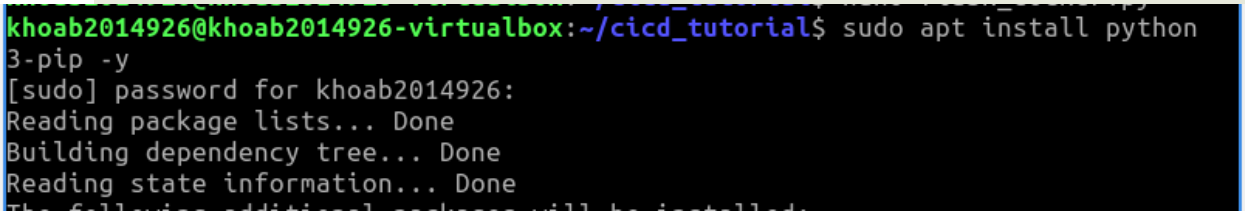
flask_docker.py

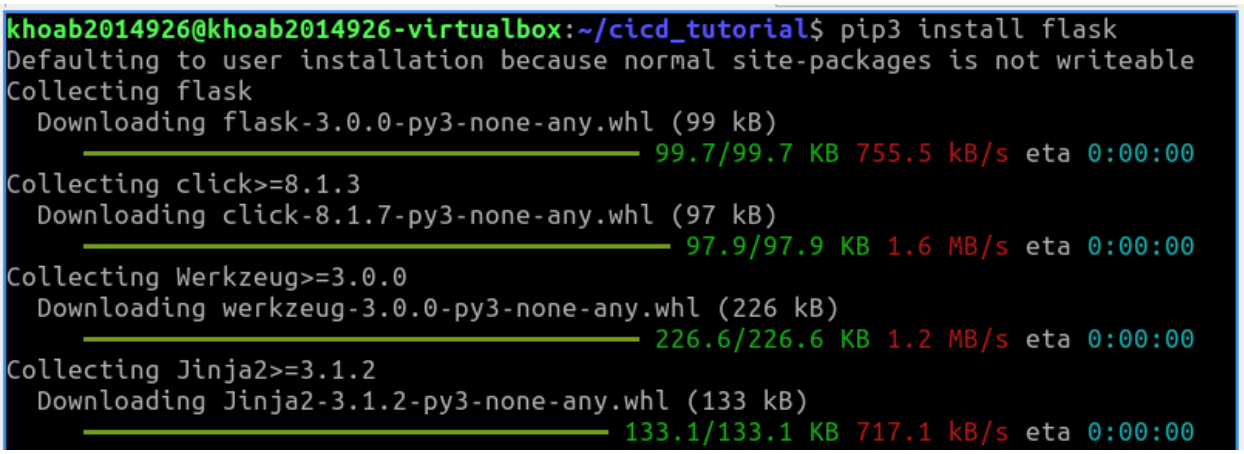
```
from flask import Flask
app = Flask(__name__)

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

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



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

```
$sudo apt install python3-pip -y
$pip3 install flask
```





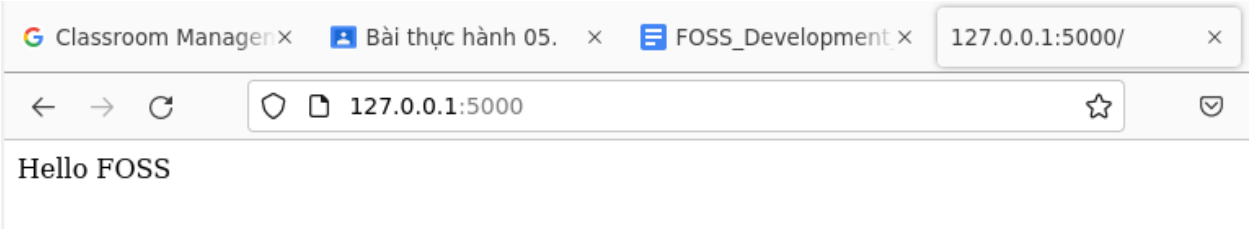
- We can test it out by running:

```
$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
```

```
khoab2014926@khoab2014926-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 deployme
nt. 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: 441-061-985
```

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



1.2. Dockerize a Flask application using Dockerfile

- Update the apt package index and install Docker

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

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ sudo apt update
Get:1 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [8
96 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/main i386 Packages [1.040 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [35
3 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/main amd64 DEP-11 Metadata [423
kB]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [1
80 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy/restricted i386 Packages [30,4
kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy/universe i386 Packages [7.474 k
```

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ sudo apt install docker
.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

- Add current user to the docker group:

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

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ sudo usermod -aG docker
 ${USER}
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ su - ${USER}
Password:
khoab2014926@khoab2014926-virtualbox:~$
```

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

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

```
khoab2014926@khoab2014926-virtualbox:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:88ec0acaa3ec199d3b7eaf73588f4518c25f9d34f58ce9a0df68429c5af48e
8d
Status: Downloaded newer image for hello-world:latest

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 i
t
```

The output will indicate that Docker is working correctly:

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

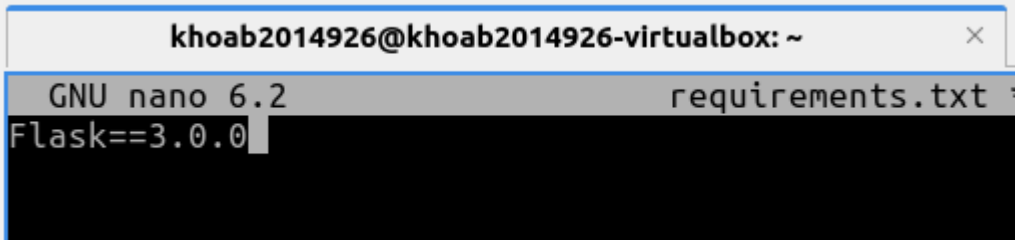
- Create a requirements.txt file

```
$nano requirements.txt
```

```
khoab2014926@khoab2014926-virtualbox:~$ nano requirements.txt
khoab2014926@khoab2014926-virtualbox:~$
```

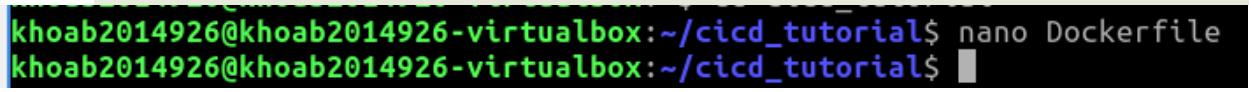
requirements.txt

```
Flask==2.2.2
```



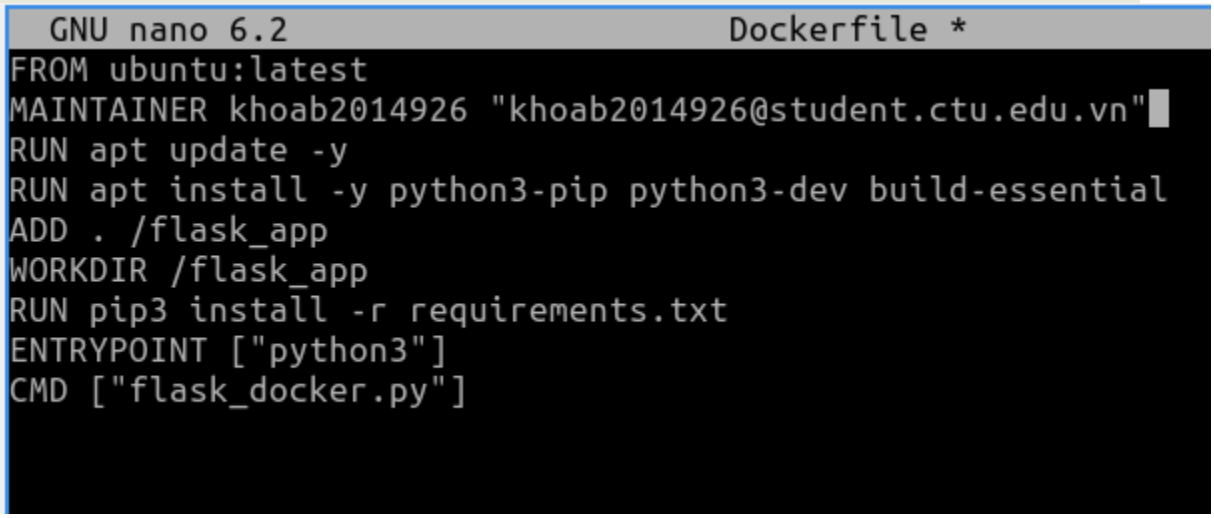
- Create a Dockerfile file

```
$nano Dockerfile
```



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"]
```



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

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

```
Removing intermediate container e21ed36ff52b
 ---> 39846460afd8
Step 8/9 : ENTRYPOINT ["python3"]
 ---> Running in 68fbe43e5363
Removing intermediate container 68fbe43e5363
 ---> 4ea734384139
Step 9/9 : CMD ["flask_docker.py"]
 ---> Running in a9419adf9d1f
Removing intermediate container a9419adf9d1f
 ---> b5638f321cd0
Successfully built b5638f321cd0
Successfully tagged my-flask-image:latest
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$
```

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

```
$docker images
```

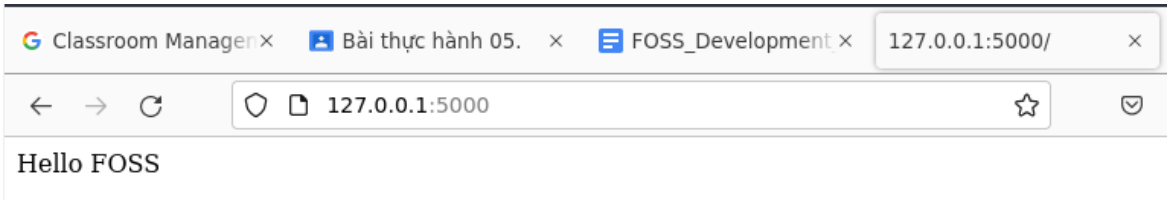
```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ docker images
REPOSITORY       TAG       IMAGE ID       CREATED              SIZE
my-flask-image   latest    b5638f321cd0   About a minute ago   476MB
ubuntu           latest    e4c58958181a   2 weeks ago          77.8MB
hello-world      latest    9c7a54a9a43c   5 months ago         13.3kB
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$
```

- Run your image (take a screenshot)

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

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ docker run -d -p 5000:5
000 my-flask-image
4b0e9be144279ad0e5585bd95a3b6b8f203cc8cf79d919fed42badb8fcb05667
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ docker ps
CONTAINER ID   IMAGE            COMMAND                  CREATED          STA
TUS          PORTS                                       NAMES
4b0e9be14427   my-flask-image   "python3 flask_docke…"   16 seconds ago   Up
14 seconds   0.0.0.0:5000->5000/tcp, :::5000->5000/tcp   competent_roentgen
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$
```

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

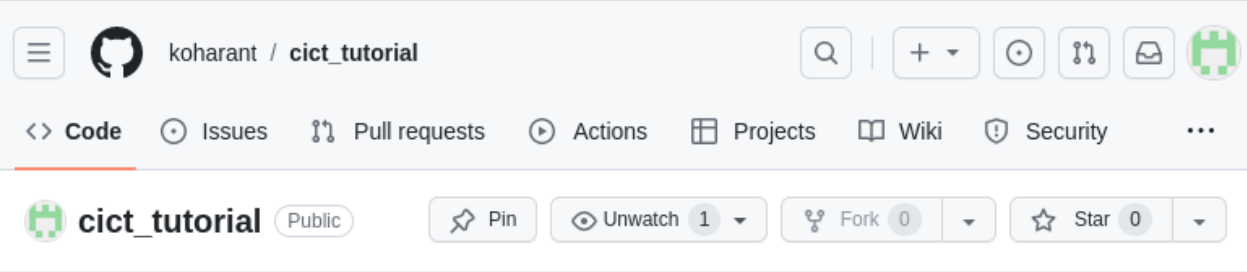


2. Automatically dockerize a Flask project using Jenkins

2.1. Push your code to a Github repository

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

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



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

```
$sudo apt update ; sudo apt install git -y
$git config --global user.name "Firstname Lastname"
$git config --global user.email "example@.ctu.edu.vn"
```

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ sudo apt update ; sudo
apt install git -y
[sudo] password for khoab2014926:
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:2 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Hit:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Fetched 229 kB in 5s (45,0 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
479 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk
  gitweb git-cvs git-mediawiki git-svn
The following packages will be upgraded:
  git
1 upgraded, 0 newly installed, 0 to remove and 478 not upgraded.
Need to get 3.166 kB of archives.
After this operation, 127 kB of additional disk space will be used.
```

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ git config --global use
r.name "koharant"
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ git config --global use
r.email "khoab2014926@student.ctu.edu.vn"
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$
```

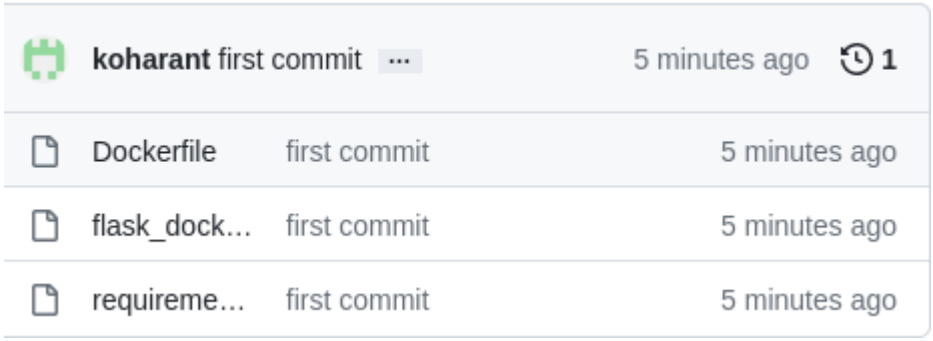
- Initialize git, commit and push your flask project files to Github

```
$mv ~/cicd_tutorial
$git init
$git add .
$git commit -m "first commit"
$git remote add origin <your repository URL>
$git push -u origin master
```

```
khoab2014926@khoab2014926-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 al
l
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/khoab2014926/cicd_tutorial/.git/
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ git add .
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ git commit -m "first co
mmit"
[master (root-commit) 09a05e5] first commit
 3 files changed, 20 insertions(+)
 create mode 100644 Dockerfile
 create mode 100644 flask_docker.py
 create mode 100644 requirements.txt
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$
```

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ git remote add origin h
ttps://github.com/koharant/cict_tutorial.git
```

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ git push -u origin mast
er
Username for 'https://github.com': koharant
Password for 'https://koharant@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), 649 bytes | 324.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/koharant/cict_tutorial.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$
```



2.2. Install and configure Jenkins

- Install Java and Jenkins

```
$sudo apt install openjdk-11-jdk -y
$wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
$sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
$sudo apt update ; sudo apt install jenkins -y
```

```
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$ sudo apt install openjdk-11-jdk -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

```
khaob2014926@khaob2014926-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-key(8)).
OK
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$ sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$
```

```
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$ java --version
openjdk 11.0.20.1 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04, mixed mode, sharing)
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$
```

```
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$ sudo apt-get update
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]
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:5 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:6 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:7 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:8 https://pkg.jenkins.io/debian-stable binary/ Packages [25,8 kB]
Fetched 28,7 kB in 2s (15,1 kB/s)
Reading package lists... Done
```

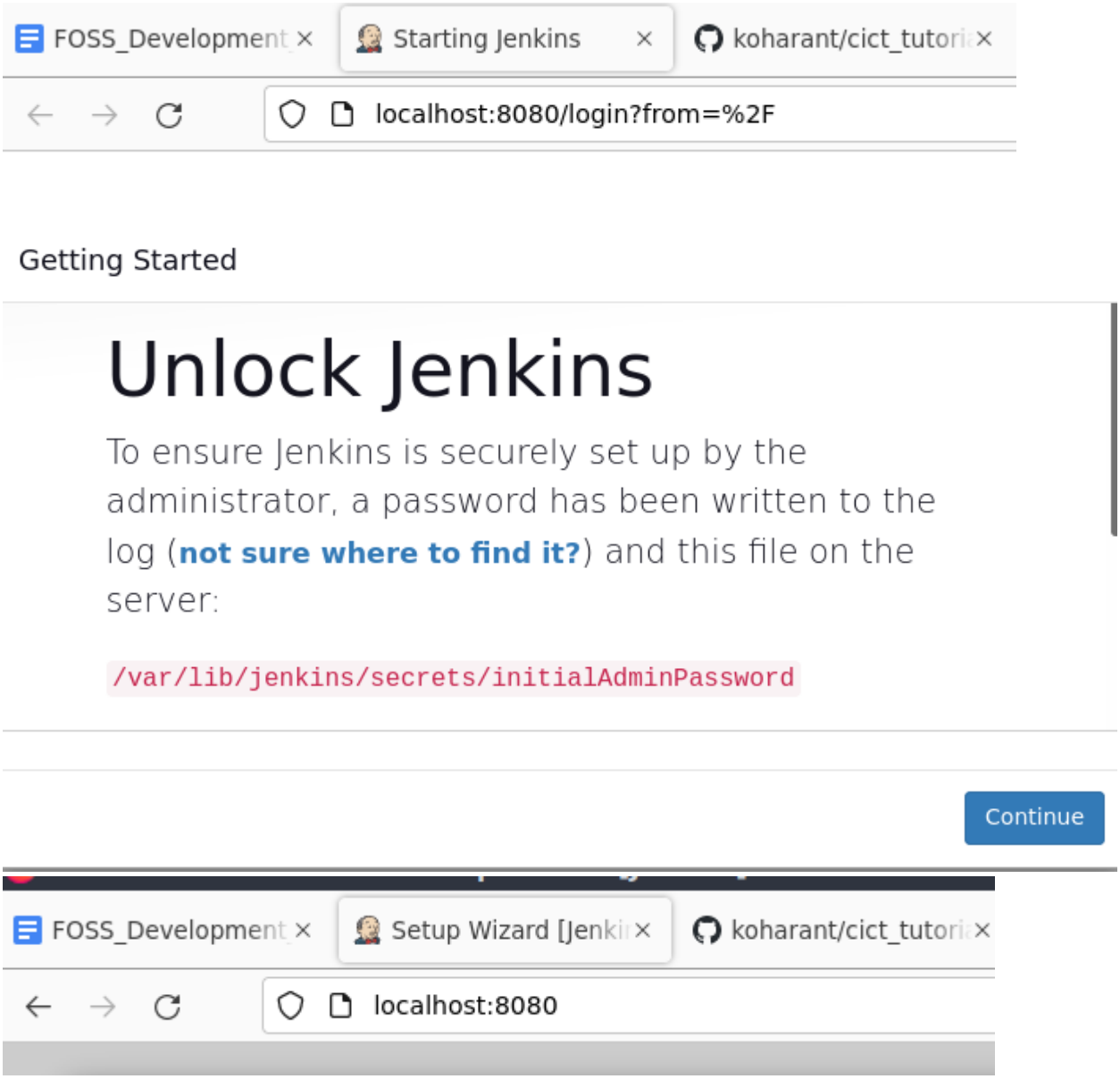
```
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$ sudo apt install jenkin
s -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

- Launch Jenkins

```
$sudo usermod -aG docker jenkins
$sudo systemctl restart jenkins.service
```

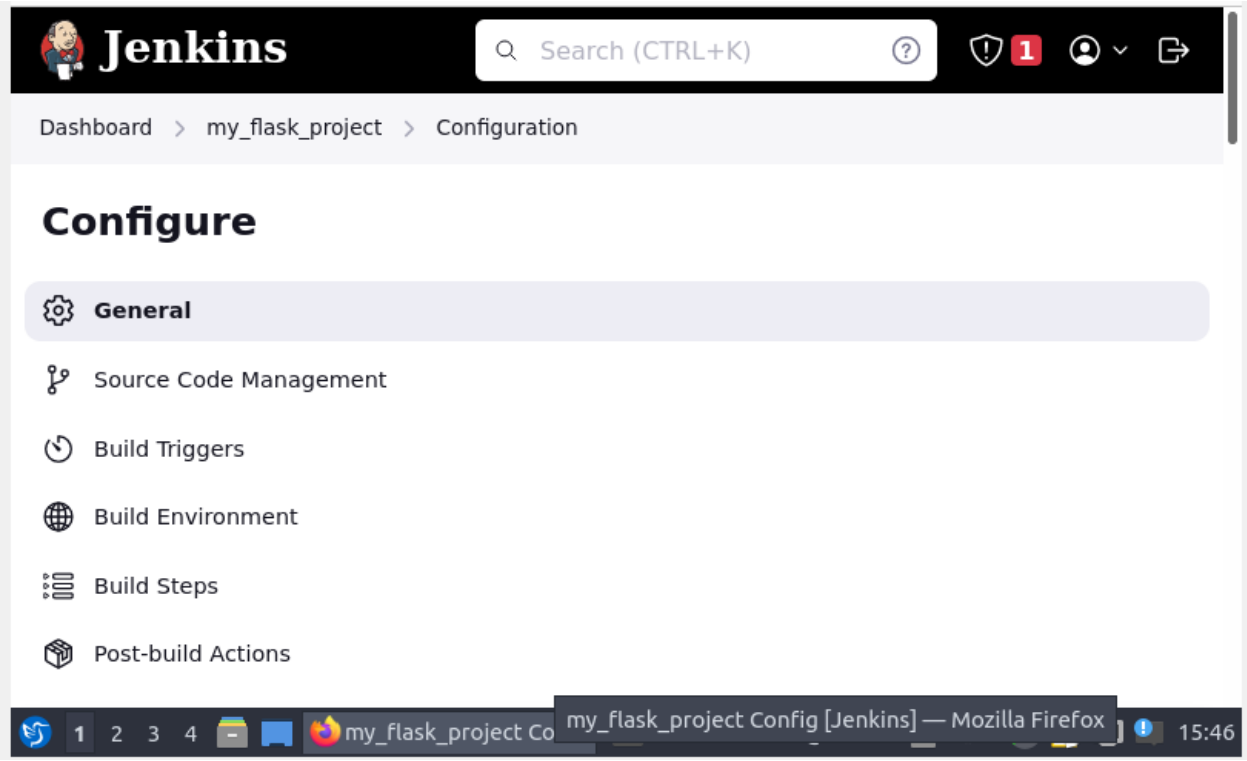
```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ sudo usermod -aG docker
 jenkins
[sudo] password for khoab2014926:
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ sudo systemctl restart
jenkins.service
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ sudo systemctl status j
enkins.service
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor pr>
     Active: active (running) since Wed 2023-10-25 19:07:29 +07; 1min 19s ago
   Main PID: 5627 (java)
      Tasks: 47 (limit: 2202)
     Memory: 647.7M
        CPU: 1min 47.063s
     CGroup: /system.slice/jenkins.service
             └─5627 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/j>

Thg 10 25 19:06:20 khoab2014926-virtualbox jenkins[5627]: ****************>
Thg 10 25 19:06:20 khoab2014926-virtualbox jenkins[5627]: ****************>
Thg 10 25 19:06:20 khoab2014926-virtualbox jenkins[5627]: WARNING: An illega>
Thg 10 25 19:06:20 khoab2014926-virtualbox jenkins[5627]: WARNING: Illegal r>
Thg 10 25 19:06:20 khoab2014926-virtualbox jenkins[5627]: WARNING: Please co>
```

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

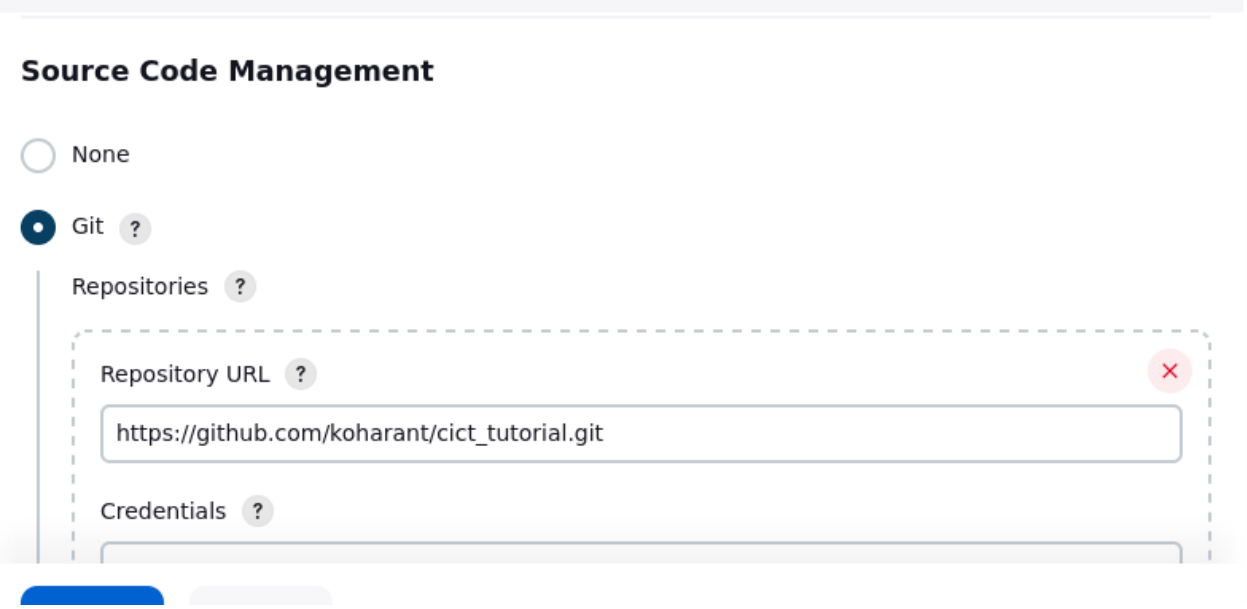


2.3. Using Jenkins to automatically dockerize your application

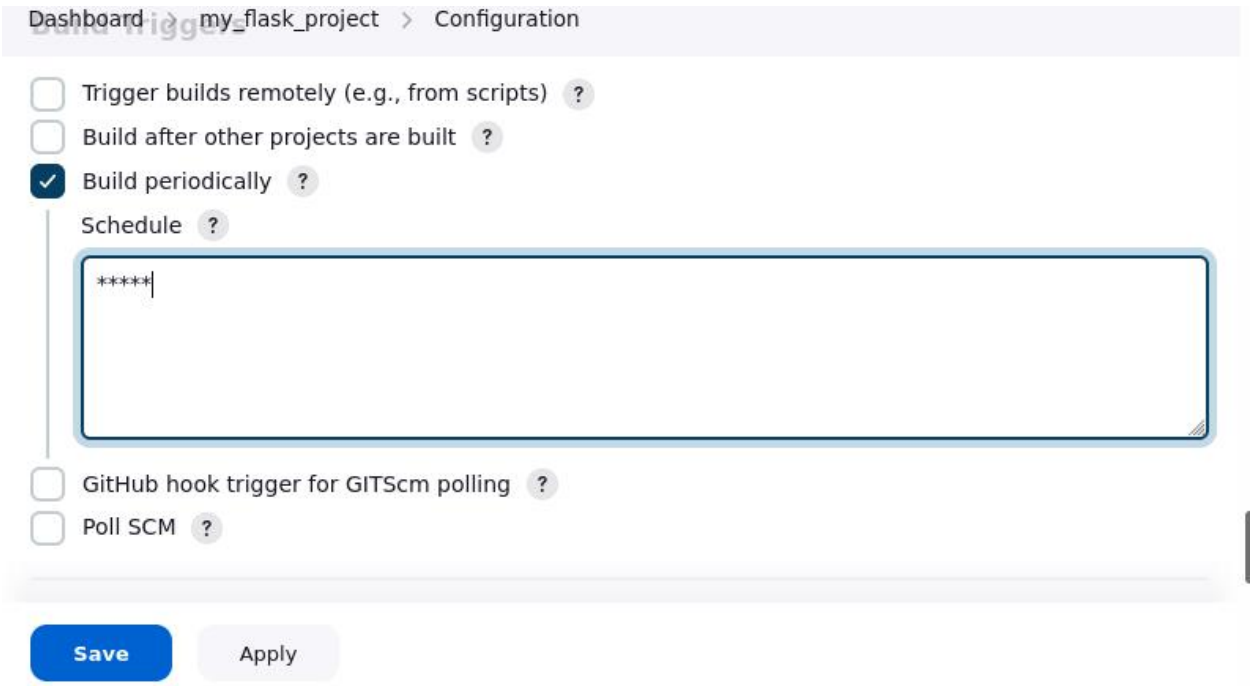
- On Jenkins dashboard, cick "Create a new job", then choose "Freestyle project". Name your project as "my_flask_project"



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



- Under "Build Triggers" select "Build periodically", fill in "* * * * *" (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 ."

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

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

Dashboard > my_flask_project > #4 > Console Output

```
Building in workspace /var/lib/jenkins/workspace/my_flask_project
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/my_flask_project/.gi
timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/koharant/cict_tutorial.git #
timeout=10
Fetching upstream changes from https://github.com/koharant/cict_tutorial.git
 > git --version # timeout=10
 > git --version # 'git version 2.34.1'
 > git fetch --tags --force --progress -- https://github.com/koharant/cict_tutoria
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 09a05e5b85d8f5c871cde3096e72ac5d767fa8d9 (refs/remotes/origi
/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 09a05e5b85d8f5c871cde3096e72ac5d767fa8d9 # timeout=10
Commit message: "first commit"
 > git rev-list --no-walk 09a05e5b85d8f5c871cde3096e72ac5d767fa8d9 # timeout=10
[my_flask_pro[redacted]sh
```

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

```
$docker images
```

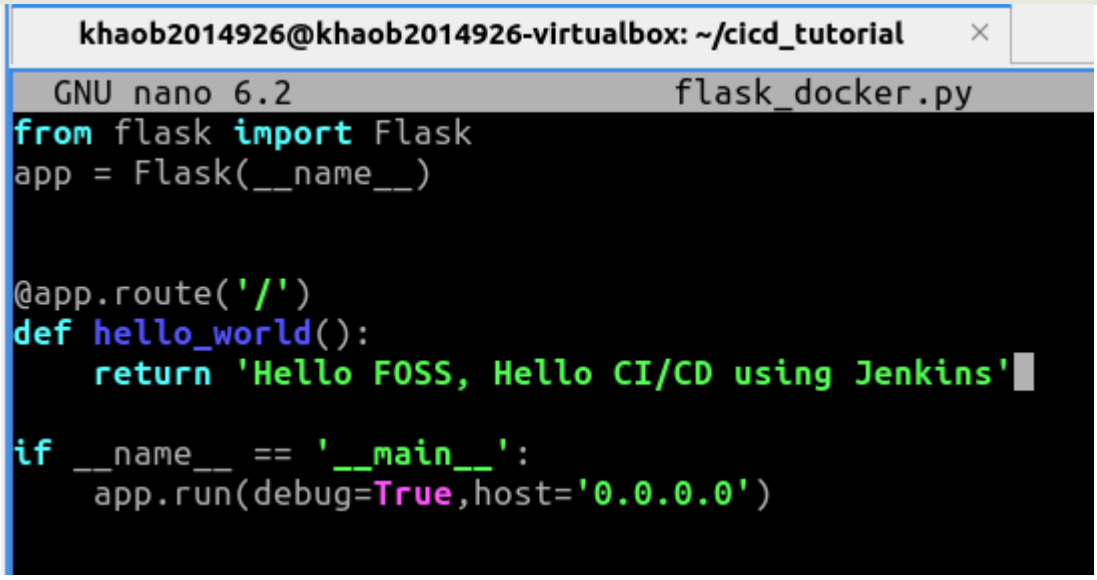
```
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$ sudo docker images
REPOSITORY       TAG       IMAGE ID       CREATED             SIZE
my-flask-image   latest    73ee5edabd61   About an hour ago   476MB
<none>           <none>    b3a001c463ae   2 days ago          476MB
ubuntu           latest    e4c58958181a   2 weeks ago         77.8MB
hello-world      latest    9c7a54a9a43c   5 months ago        13.3kB
khaob2014926@khaob2014926-virtualbox:~/cicd_tutorial$
```

- Modify your Flask application:

```
$nano flask_docker.py
```

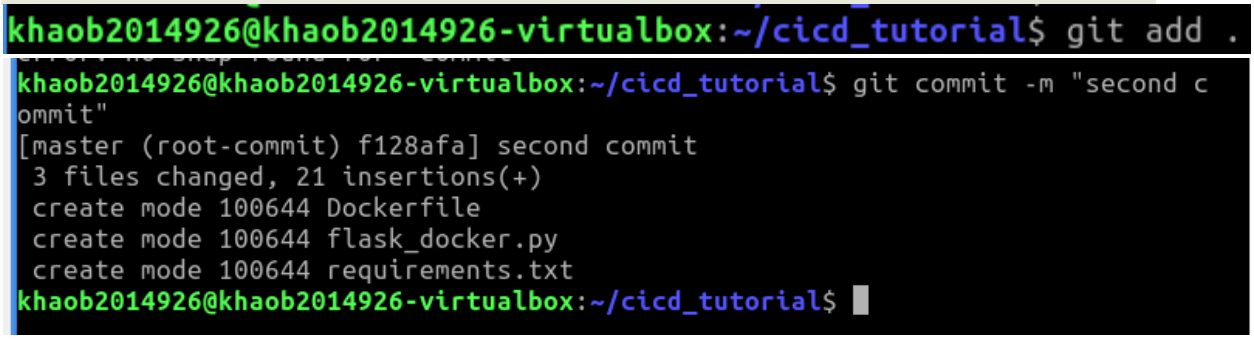
```
from flask import Flask
app = Flask(__name__)
@app.route('/')

def hello_world():
```

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

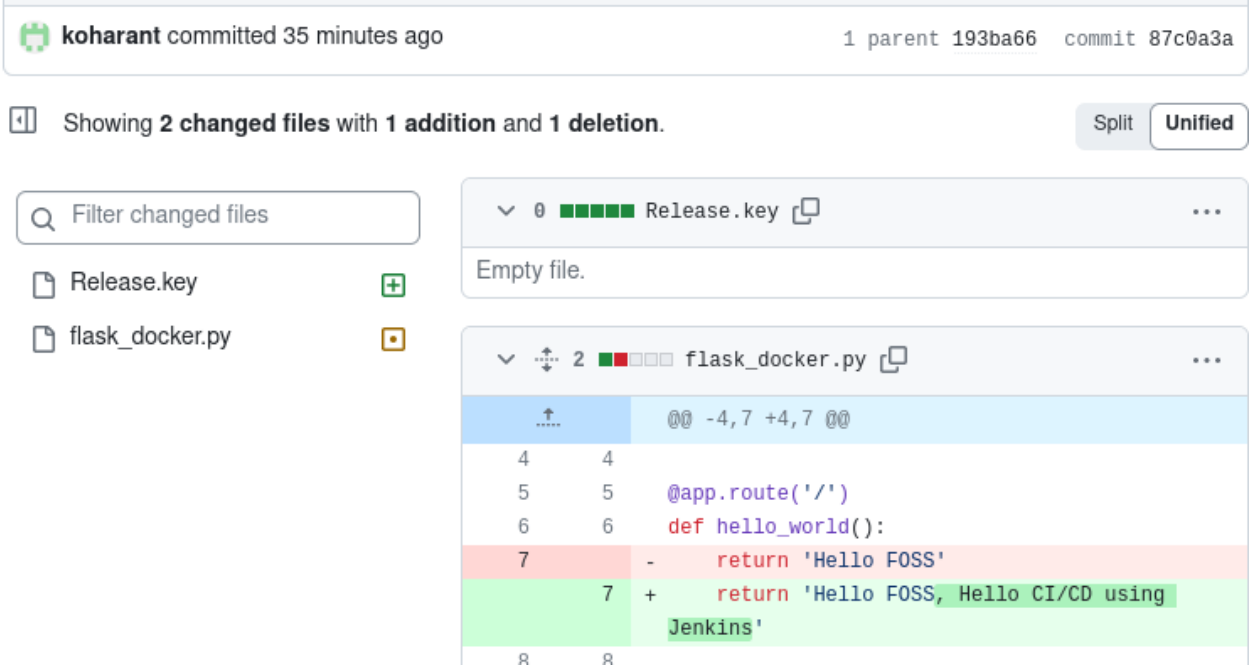


- Commit and push your project files to GitHub

```
$git add .
$git commit -m "second commit"
$git push origin master
```



```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ git push -f origin mast
er
Username for 'https://github.com': koharant
Password for 'https://koharant@github.com':
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 2 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 1.00 KiB | 342.00 KiB/s, done.
Total 9 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/koharant/cict_tutorial.git
 + 09a05e5...87c0a3a master -> master (forced update)
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$
```

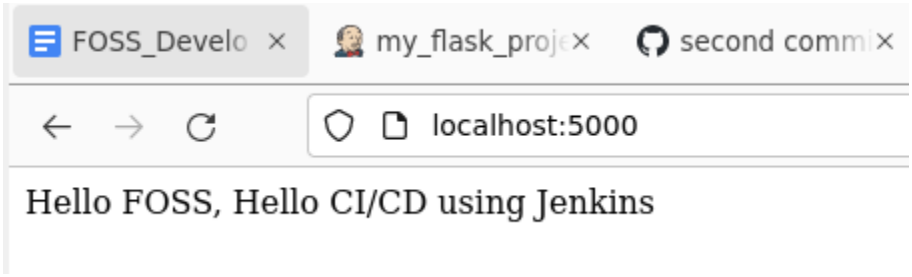


- Wait 1 minute, then run your image

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

```
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ docker run -d -p 5000:5
000 my-flask-image
bcb3bb80cd9ae0db321eced80b796045d8e1b05d4fd87ca208241e807fe83878
khoab2014926@khoab2014926-virtualbox:~/cicd_tutorial$ docker ps
CONTAINER ID   IMAGE           COMMAND                  CREATED          STA
TUS          PORTS                                       NAMES
bcb3bb80cd9a   my-flask-image   "python3 flask_docke…"   45 seconds ago   Up
43 seconds   0.0.0.0:5000->5000/tcp, :::5000->5000/tcp   charming_pike
```

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



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

---END---