Contents

[Code repository 2](#_Toc132616695)

[Environment setup 4](#_Toc132616696)

# Code repository

Instead of a single main branch, this workflow uses two branches to record the history of the project. The main branch stores the official release history, and the develop branch serves as an integration branch for features. It's also convenient to tag all commits in the main branch with a version number.

Create develop branch and push it to remote repo

*git branch develop*

*git push -u origin develop*

Create feature branch

*git checkout develop*

*git checkout -b feature\_branch*

Add test file for feature branch, commit, push it and merge

*git checkout feature\_branch*

*git add .*

*git commit -m “*[*Feature branch test*](https://github.com/anastaviki/ci-cd/commit/0d48fff08a402956b78e924a1a8602f0ad4dab24)*”*

*git push -u origin feature\_branch*

*git checkout develop*

*git merge feature\_branch*

*git push -u origin develop*

Feature branch can be deleted

*git push origin --delete feature\_branch*



*git branch -d feature\_branch*

Create release branch and push it

*git checkout develop*

*git checkout -b release/0.1.0*

*git push -u origin release/0.1.0*

Add some bug fixing in release branch commit changes and push to remote repo

*git add .*

*git commit -m “*[*Bug fix in release test*](https://github.com/anastaviki/ci-cd/commit/0d48fff08a402956b78e924a1a8602f0ad4dab24)*”*

*git push -u origin release/0.1.0*

Once the release is ready to ship, it will get merged it into main and develop, then the release branch will be deleted.

*git checkout main*

*git merge release/0.1.0*

*git push -u origin main*

*git checkout develop*

*git merge release/0.1.0*

*git push -u origin develop*

Create hot fix branch

*git checkout main*

*git checkout -b hotfix\_branch*

Add changes, commit and push to remote repo

*git add .*

*git commit -m “*[*Hot fix test*](https://github.com/anastaviki/ci-cd/commit/0d48fff08a402956b78e924a1a8602f0ad4dab24)*”*

*git push -u origin hotfix\_branch*

Similar to finishing a release branch, a hotfix branch gets merged into both main and develop.

*git checkout main*

*git merge hotfix\_branch*

*git push -u origin main*

*git checkout develop*

*git merge hotfix\_branch*

*git push -u origin develop*

Hot fix branch can be deleted

*git push origin --delete hotfix\_branch*



*git branch -D hotfix\_branch*

# Environment setup

Install Docker

Docker pull Jenkins/Jenkins

Text

Description automatically generated

docker ps – show containers



Docker images – show images

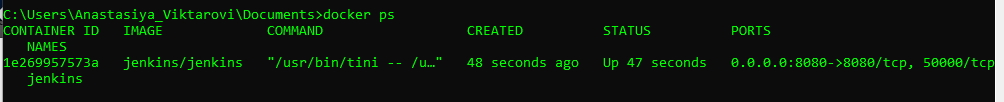
Text

Description automatically generated

docker run -p 8080:8080 --name=jenkins -d jenkins/Jenkins – run docker container



docker ps – show containers



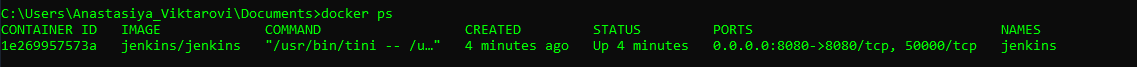
Check Jenkins

Graphical user interface, text, application, email

Description automatically generated

Find container ID

docker ps



Find admin password

docker exec -it 1e269957573a cat /var/jenkins\_home/secrets/initialAdminPassword



And log it to jenkins

Install plugins for git

Create Admin user, as I already have git on my ps, use username from git

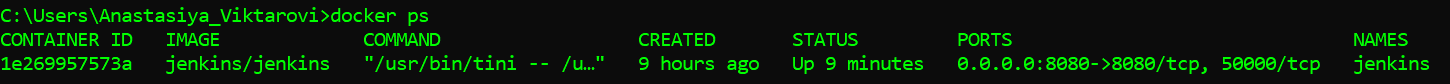
Graphical user interface

Description automatically generated with medium confidence



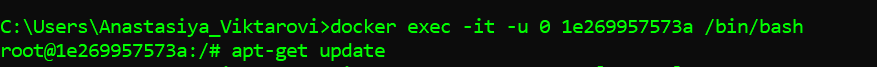
Install python on container.

docker ps



Docker exec -it -u 0 1e269957573a /bin/bash

apt-get update



apt-get install python3

python3 –version

A screenshot of a computer

Description automatically generated with medium confidence