КИИИ Лаб 4

Class Assignment / Homework

- Part 1: Setup Jenkins on your machine
  - Use the `Dockerfile` and the `docker-compose.yml` files;
  - Start the orchestration, unlock Jenkins, select plugins to install, create your admin profile, create an ngrok temporary public URL, setup your secret credentials (GitHub and Docker Hub);

PART 1

We create Dockerfile which creates our images when run

```
FROM jenkins/jenkins:lts

USER root

# Install necessary tools and dependencies
RUN apt-get update && \
    apt-get install -y sudo && \
    apt-get clean

# Add a user to the 'sudo' group and allow passwordless sudo
RUN usermod -aG sudo jenkins && \
    echo "jenkins ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers

USER jenkins
```

We also create a docker-compose.yml which defines our containers

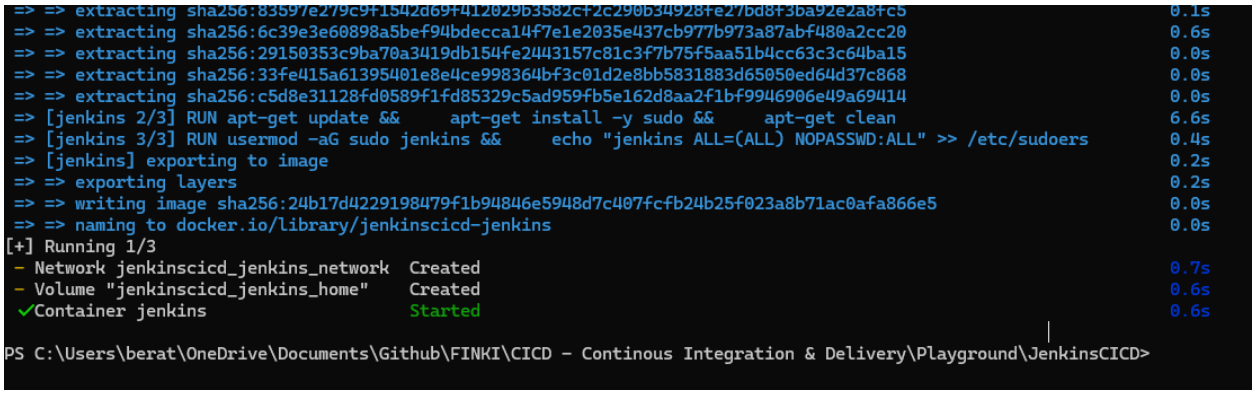
```yaml
version: '3'

services:
  jenkins:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: jenkins
    ports:
      - "8080:8080"
    volumes:
      - jenkins_home:/var/jenkins_home
    environment:
      - TZ=UTC
    networks:
      - jenkins_network

networks:
  jenkins_network:

volumes:
  jenkins_home:
```

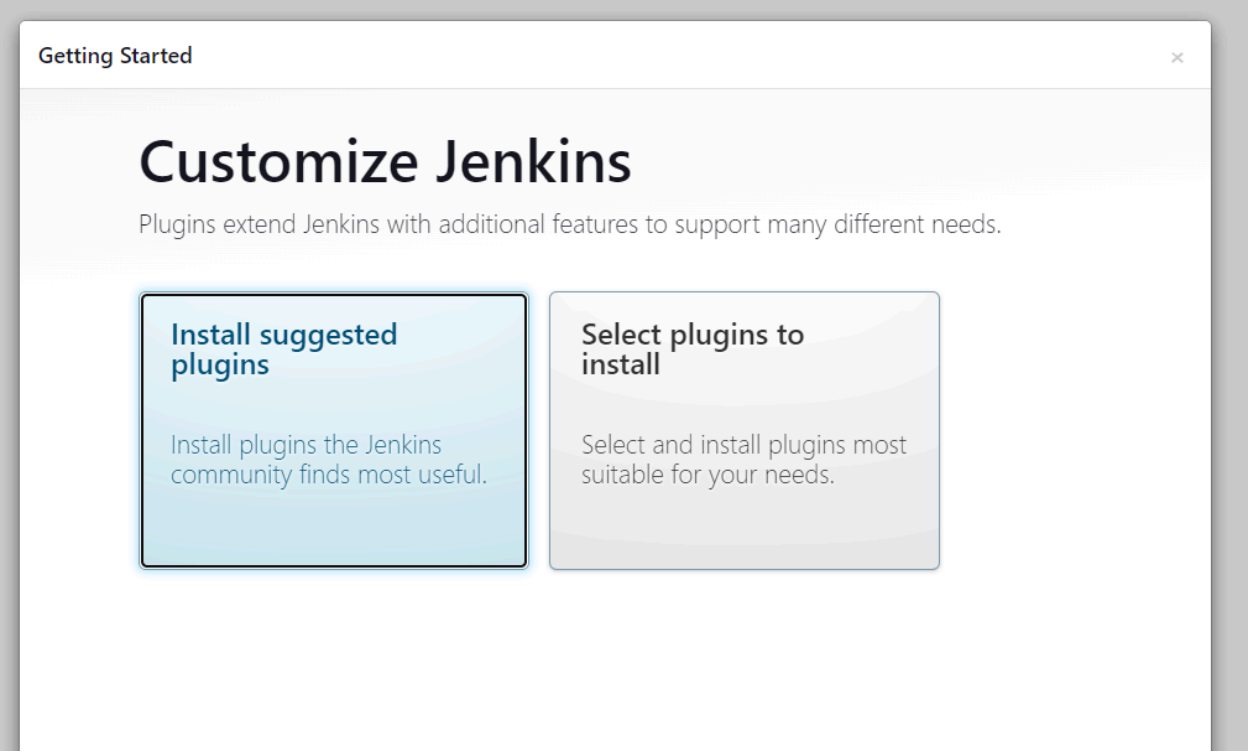
We run command
Docker-compose up -d

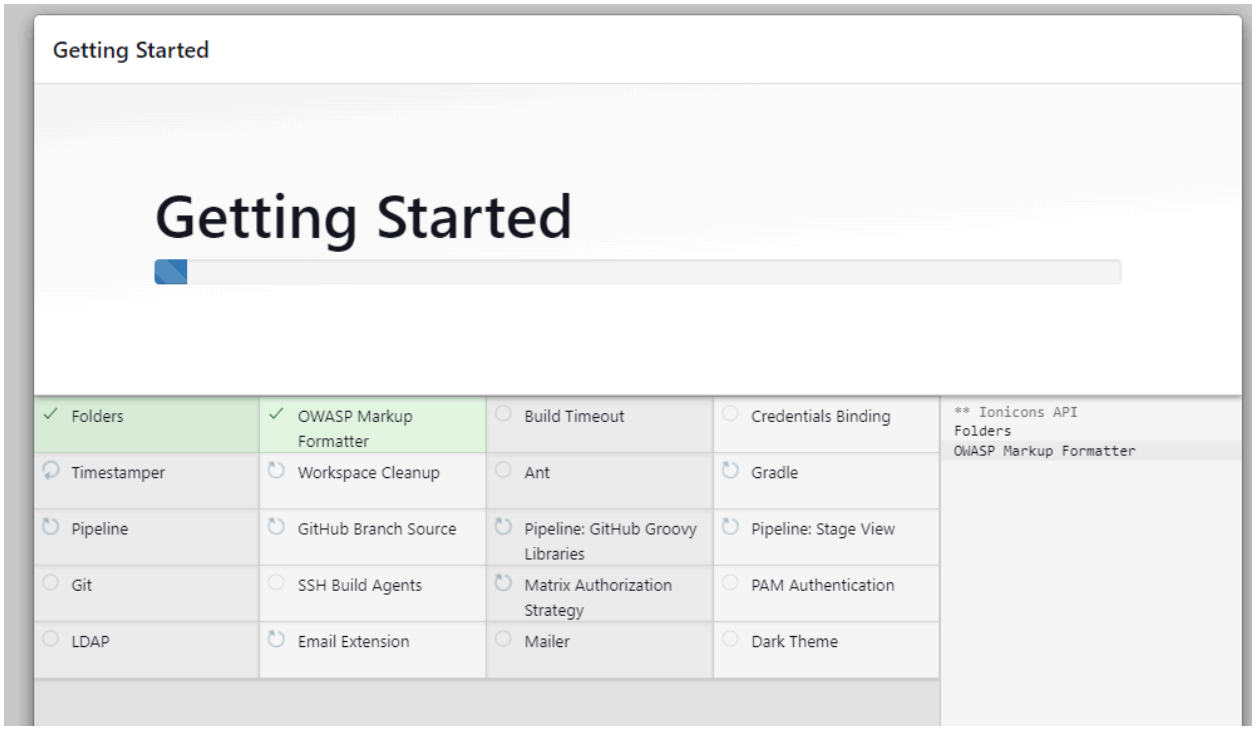


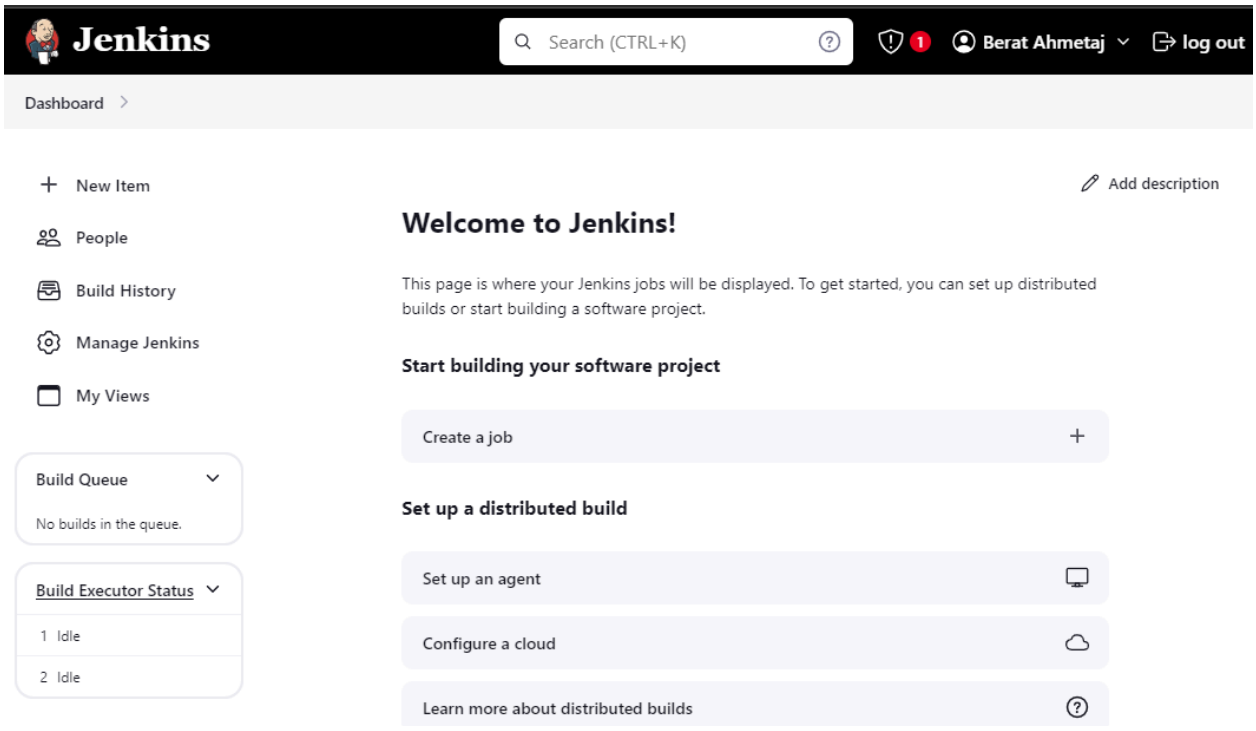
Unlock Jenkins:
After the containers are up and running, we access Jenkins on browser by visiting http://localhost:8080.

We need admin password, we run this command
```
docker exec -it jenkins cat /var/jenkins_home/secrets/initialAdminPassword
```

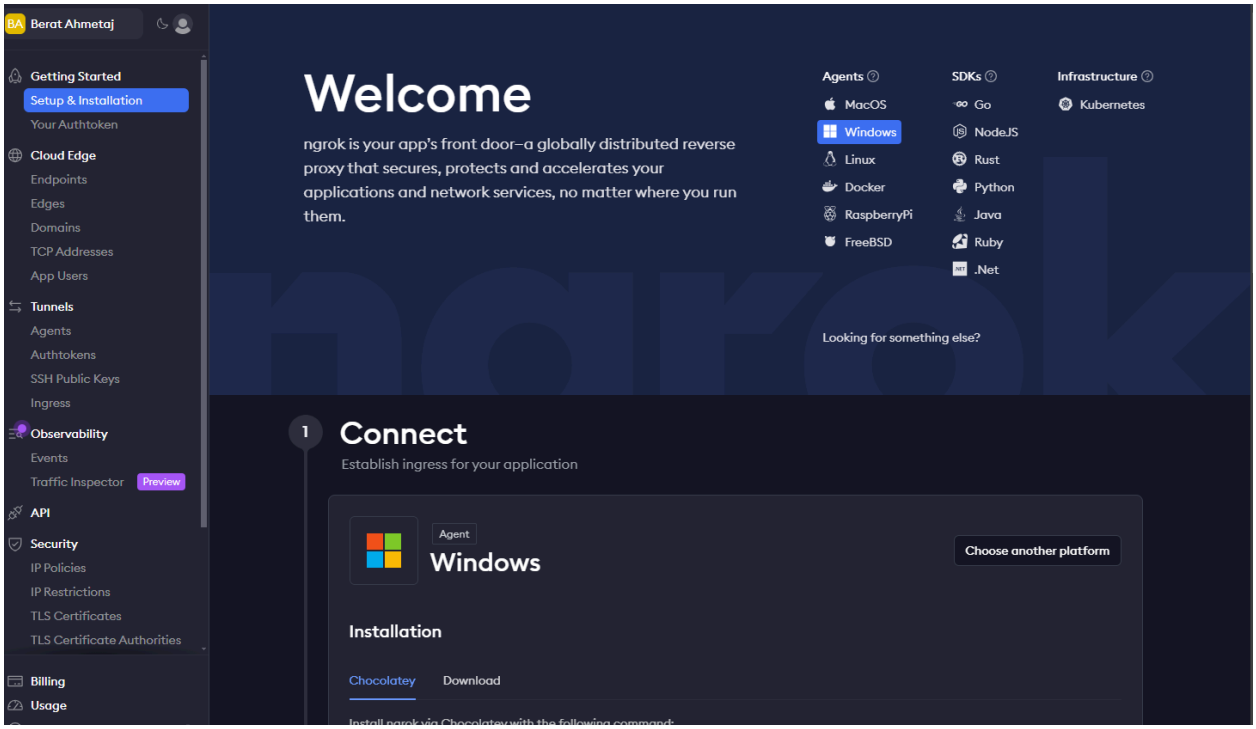
We get this
```
a15060a1b55d4cf19f5821ac618b7ee0
```




Getting Started
Getting Started
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
SSH Build Agents
Matrix Authorization Strategy
PAM Authentication
LDAP
Email Extension
Mailer
Dark Theme
** Ionicons API
Folders
OWASP Markup Formatter


Jenkins
Search (CTRL+K)
1
Berat Ahmetaj
log out
Dashboard
New Item
People
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Add description
Welcome to Jenkins!
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.
Start building your software project
Create a job
Set up a distributed build
Set up an agent
Configure a cloud
Learn more about distributed builds

Install Ngrok (Multiplatform Tunneling Online)



We Get Authkey and install Ngrok through docker

```
docker run -it -e NGROK_AUTHTOKEN=<token> ngrok/ngrok http 80
```

We Install

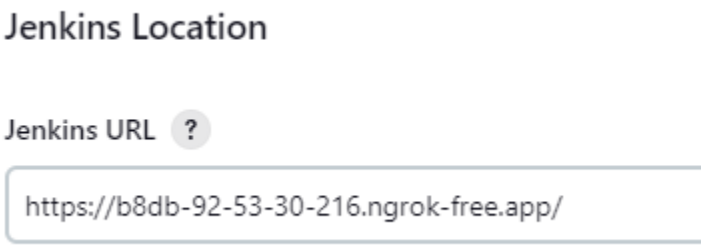
```
ngrok

Try our new Traffic Inspector Dev Preview: https://ngrok.com/r/ti

Session Status                online
Account                       Berat Ahmetaj (Plan: Free)
Update                        update available (version 3.8.0, Ctrl-U to update)
Version                       3.8.0
Region                        Europe (eu)
Latency                       50ms
Web Interface                 http://0.0.0.0:4040
Forwarding                    https://b8db-92-53-30-216.ngrok-free.app -> http://localhost:80

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

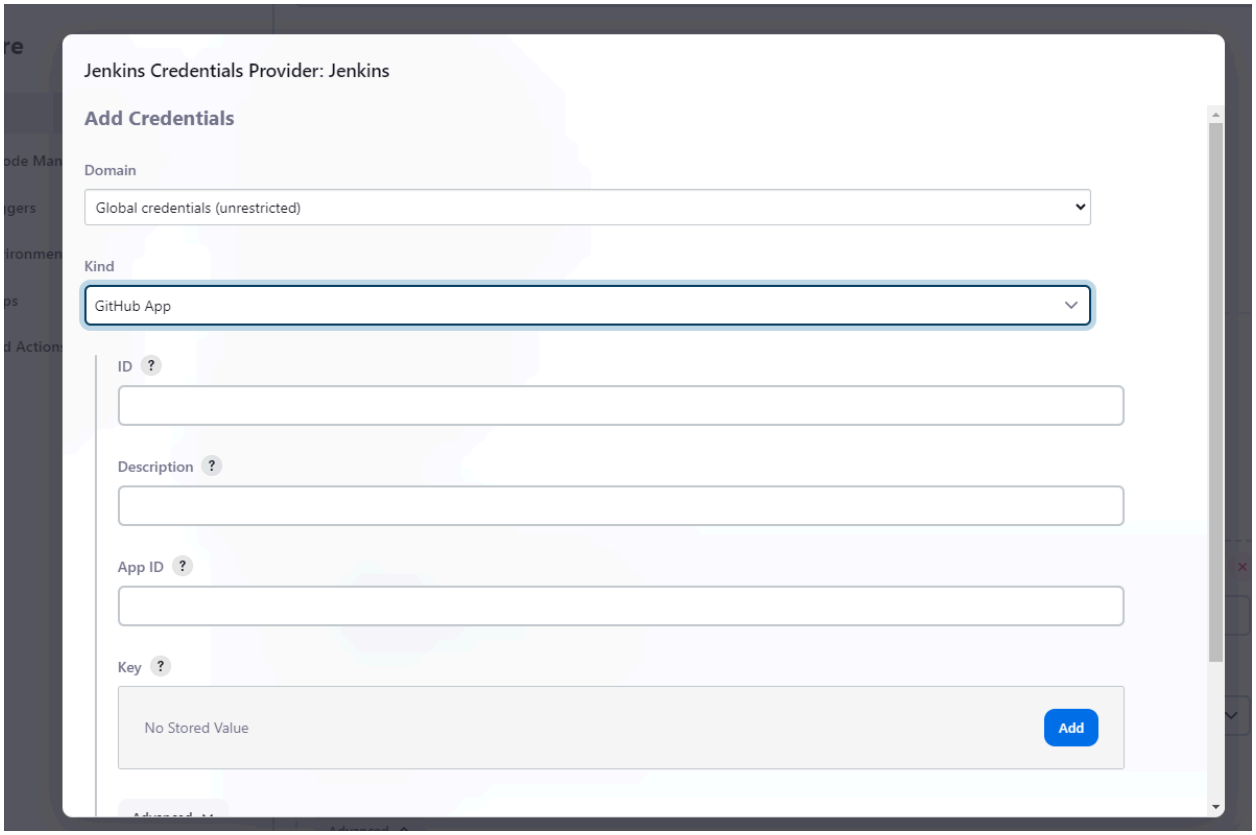
We Use The Given Tunneling Link to insert it into our Jenkins



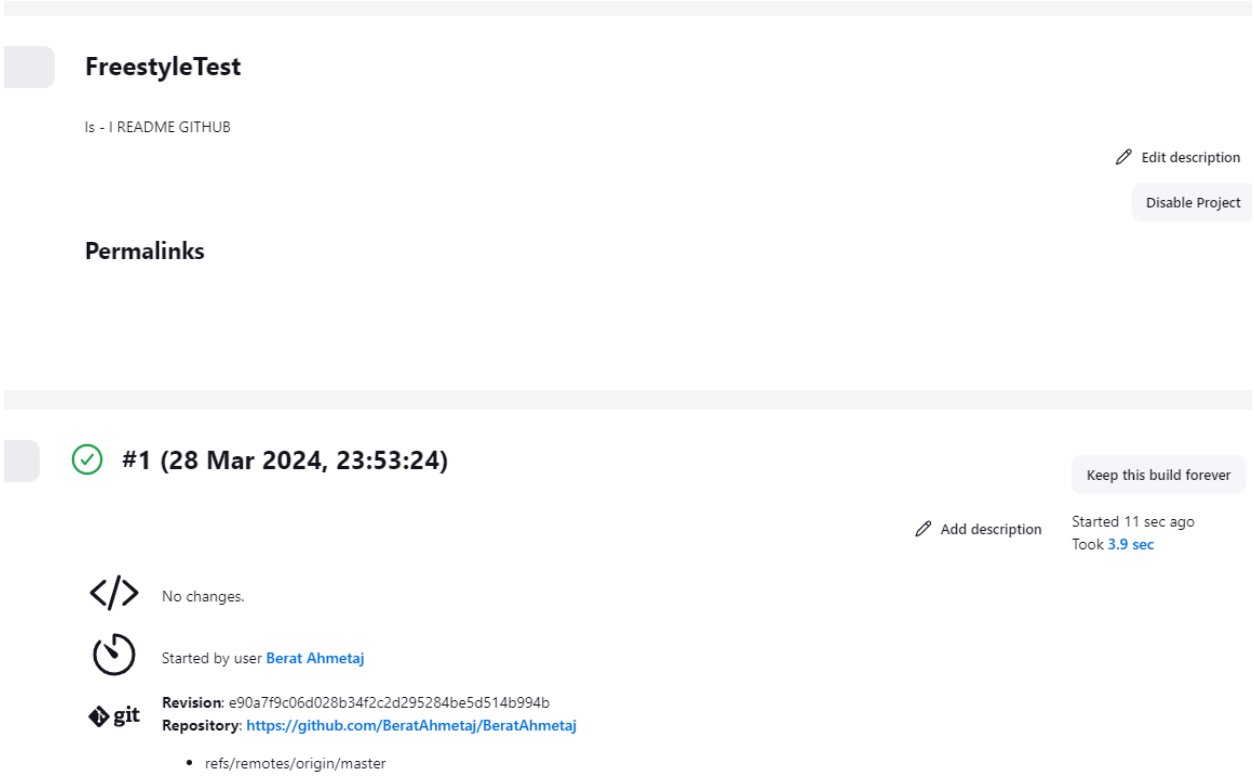
- Part 2: Create a Freestyle project
  - Connect it to a simple GitHub repository from your account;
  - Add a Build Step with a simple `ls -l` command;
  - Set up an automatic build, on each minute;

PART 2

We now create Freestyle Project connect it with github



Give it secret parameters (which arent secret)

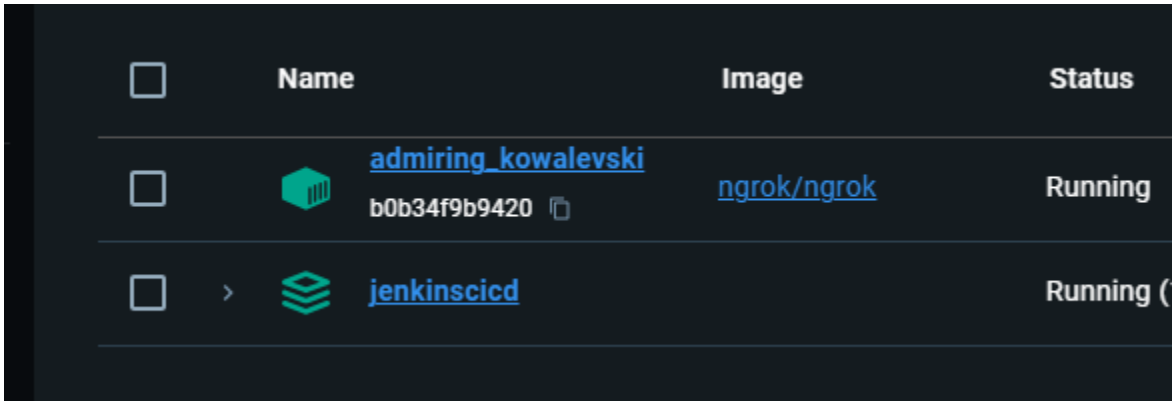


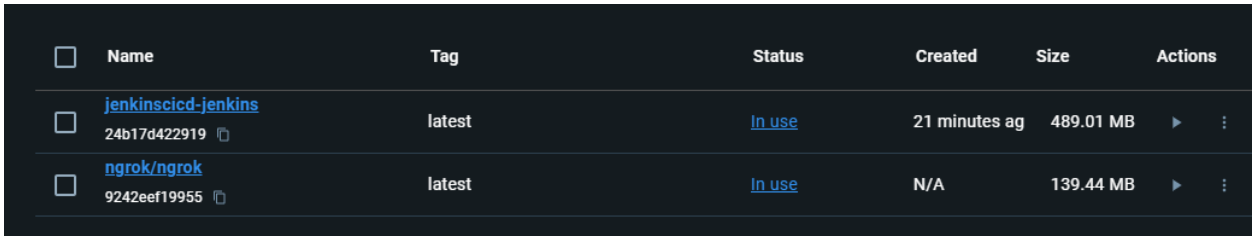
Console Output

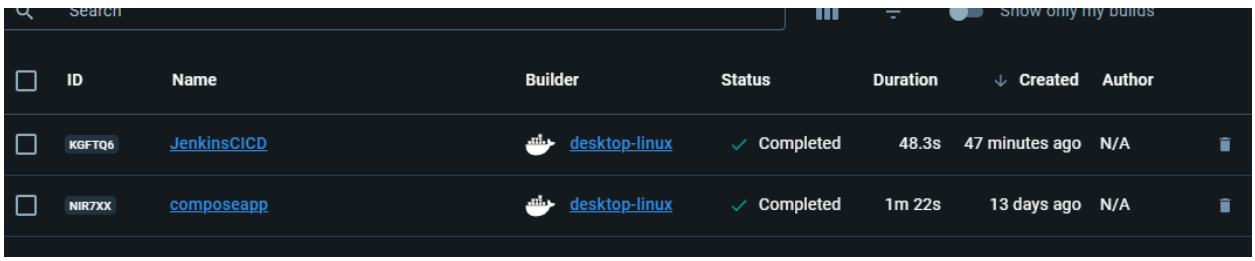
```
Started by user Berat Ahmetaj
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/FreestyleTest
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/BeratAhmetaj/BeratAhmetaj
 > git init /var/jenkins_home/workspace/FreestyleTest # timeout=10
Fetching upstream changes from https://github.com/BeratAhmetaj/BeratAhmetaj
 > git --version # timeout=10
 > git --version # 'git version 2.39.2'
 > git fetch --tags --force --progress -- https://github.com/BeratAhmetaj/BeratAhmetaj +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/BeratAhmetaj/BeratAhmetaj # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision e90a7f9c06d028b34f2c2d295284be5d514b994b (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f e90a7f9c06d028b34f2c2d295284be5d514b994b # timeout=10
Commit message: "Update README.md"
First time build. Skipping changelog.
[FreestyleTest] $ /bin/sh -xe /tmp/jenkins8559775056839798688.sh
+ ls -l
total 4
-rw-r--r-- 1 jenkins jenkins 558 Mar 28 23:53 README.md
Finished: SUCCESS
```

Permalinks

- Last build (#1), 7 min 41 sec ago
- Last stable build (#1), 7 min 41 sec ago
- Last successful build (#1), 7 min 41 sec ago
- Last completed build (#1), 7 min 41 sec ago

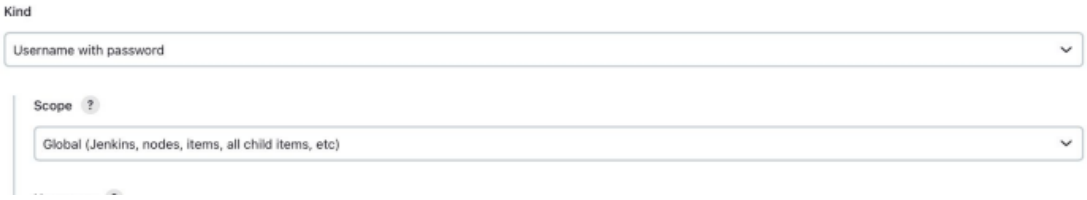


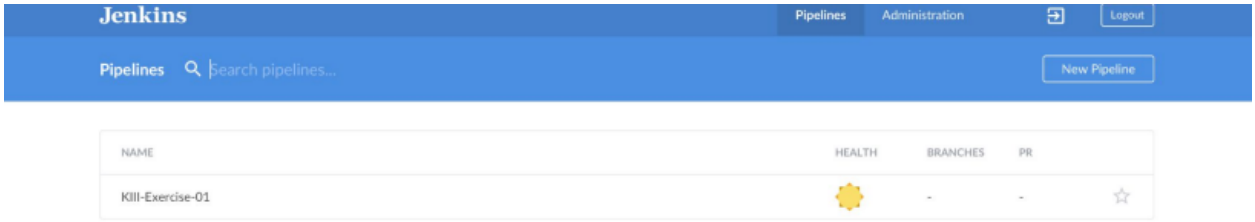




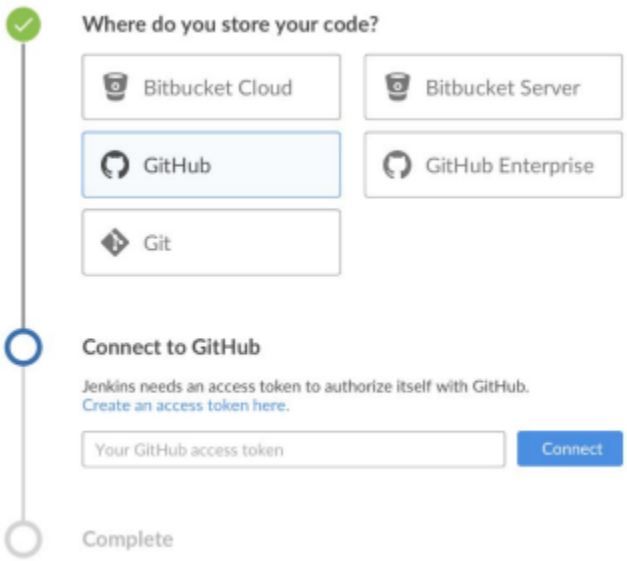
- Part 3: Create a Pipeline in Blue Ocean
  - Create a new GitHub repository with the supplied `Dockerfile` and `Jenkinsfile` files;
    - Modify the `Jenkinsfile` to match your repo and credentials;
  - Create a new Pipeline with Blue Ocean and connect it to the GitHub repo;
  - Create a Webhook in the GitHub repository;
  - Make changes in the GitHub repo, and see the automatic Pipeline run happening in Jenkins;
  - Create a new ‘dev’ branch in the repo. Make a change in it. Create a pull request from ‘dev’ to ‘main’. Accept the PR and merge the changes. Watch what happens in Jenkins.

PART 3

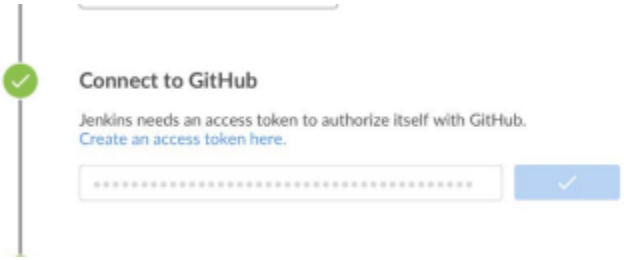


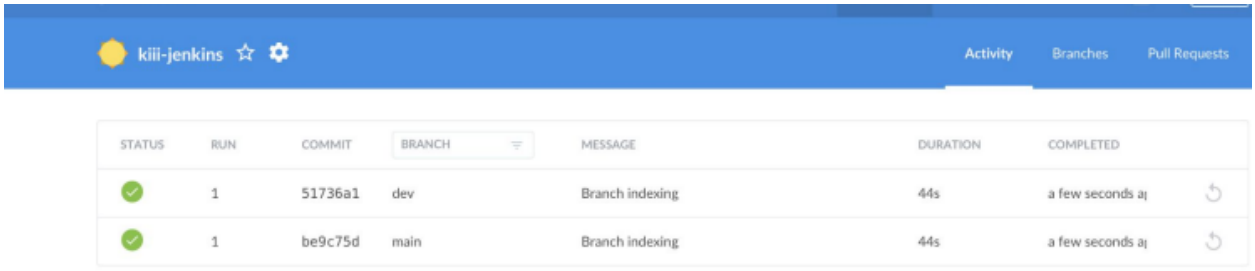


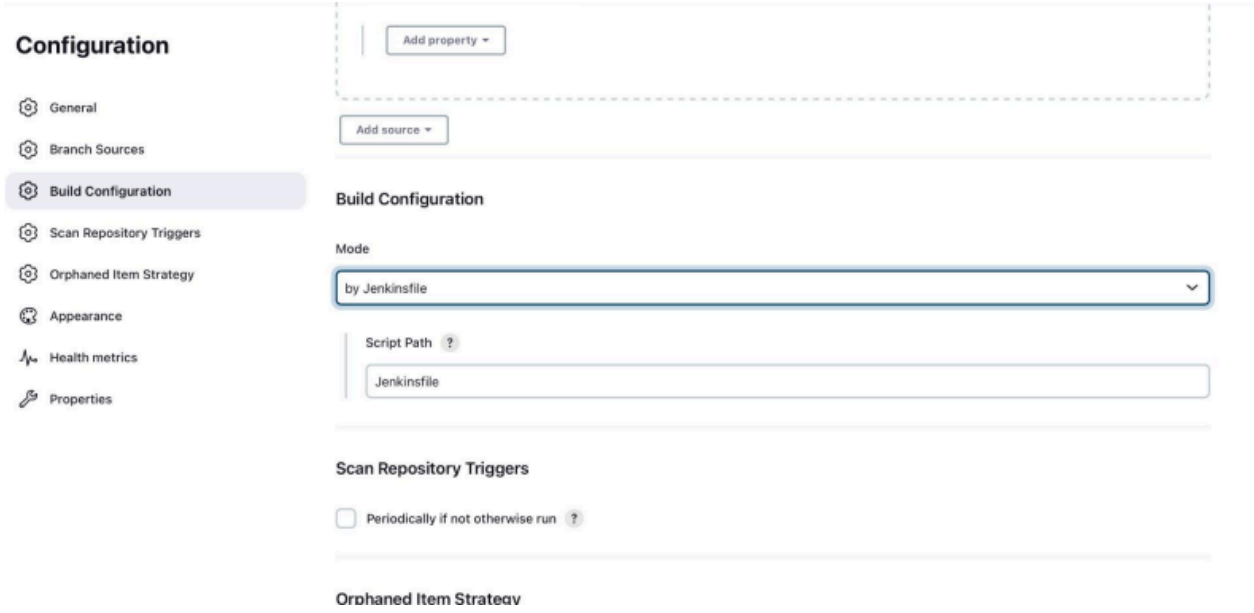
Pipeline in Blue Ocean



We connect Github



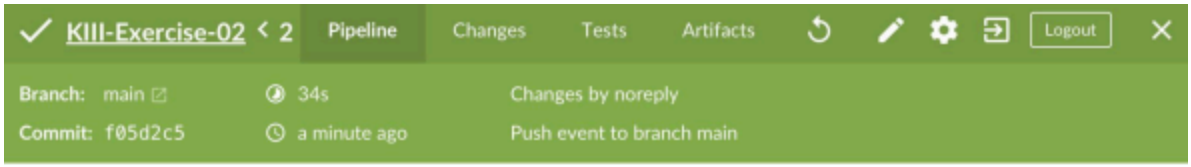


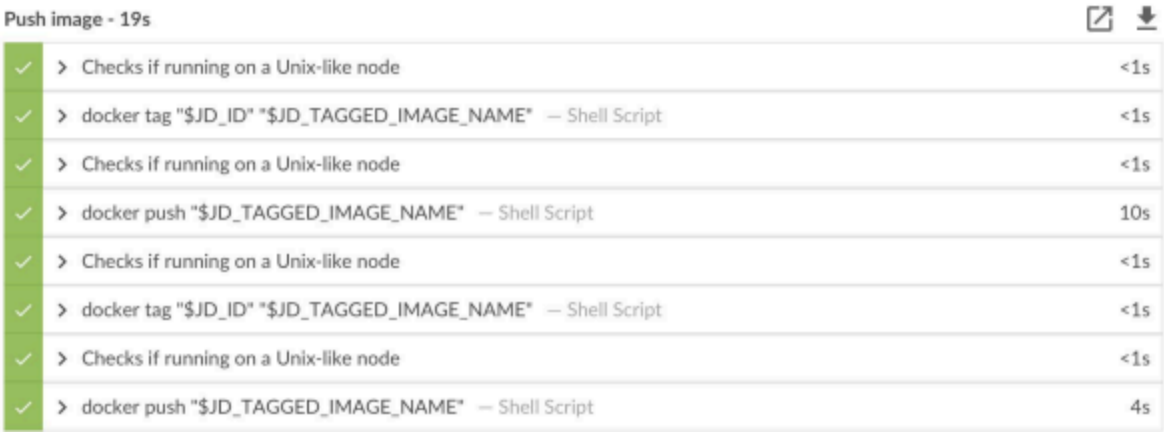


GITHUB REPO

16 lines (16 sloc) | 469 Bytes Raw Blame

```
node {
    def app
    stage('Clone repository') {
        checkout scm
    }
    stage('Build image') {
       app = docker.build("mjovanovik/kiii-jenkins")
    }
    stage('Push image') {
        docker.withRegistry('https://registry.hub.docker.com', 'dockerhub') {
            app.push("${env.BRANCH_NAME}-${env.BUILD_NUMBER}")
            app.push("${env.BRANCH_NAME}-latest")
            // signal the orchestrator that there is a new version
        }
    }
```






```
3     556c7114696f: Preparing
4     a1bd4a5c5a79: Preparing
5     597a12cbab02: Preparing
6     8820623d95b7: Preparing
7     338a545766ba: Preparing
8     e65242c66bbe: Preparing
9     3af14c9a24c9: Preparing
10    3af14c9a24c9: Waiting
11    e65242c66bbe: Waiting
12    556c7114696f: Layer already exists
13    a1bd4a5c5a79: Layer already exists
14    338a545766ba: Layer already exists
15    8820623d95b7: Layer already exists
16    597a12cbab02: Layer already exists
17    3af14c9a24c9: Layer already exists
18    e65242c66bbe: Layer already exists
19    main-latest: digest: sha256:917c1115ba445697525ab70fa782a399b43c6b2b5a2e81e051b0b536c909c6ab
      size: 1777
```