

Лабораторна робота № 1

Тема: Розгортання першого сервісу за допомогою Jenkins

Мета: інсталювати, налаштувати та створити пайплайн для розгортання додатку у Jenkins за допомогою графічного інтерфейсу

Хід роботи:

У даному методичному матеріалі використовується віртуальне оточення з ОС Ubuntu 20.04

Після автентифікації до віртуальної машини виконуємо команду оновлення встановлених пакетів:

```
sudo apt-get update
```

Інсталюємо Java:

```
sudo apt-get install openjdk-11-jdk
```

Додаємо до списку репозиторіїв, репозиторій з Jenkins:

```
sudo curl -fsSL  
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key |  
sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null  
sudo echo deb  
[signed-by=/usr/share/keyrings/jenkins-keyring.asc]  
https://pkg.jenkins.io/debian-stable binary/ | sudo tee  
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

Знову запускаємо оновлення пакетів, для отримання оновлень з щойно доданого репозиторія:

```
sudo apt-get update
```

Встановлюємо Jenkins:

```
sudo apt-get install jenkins
```

Запускаємо Jenkins:

```
sudo systemctl start jenkins
```

Перевіряємо чи запустився і чи коректно працює Jenkins:

```
sudo systemctl status jenkins
```

Коректним виводом має бути:

Loaded: loaded

Active: active (running)

```

ubuntu@ip-10-0-1-63:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: ena
   Active: active (running) since Fri 2024-02-23 14:07:53 UTC; 2min 35s ago
     Main PID: 16995 (java)
        Tasks: 37 (limit: 1126)
       Memory: 318.1M
      CGroup: /system.slice/jenkins.service
              └─16995 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenk

Feb 23 14:07:15 ip-10-0-1-63 jenkins[16995]: 3f22ee319f1d4b82aeff4ed10322f9e4
Feb 23 14:07:15 ip-10-0-1-63 jenkins[16995]: This may also be found at: /var/lib/jenk
Feb 23 14:07:15 ip-10-0-1-63 jenkins[16995]: *****
Feb 23 14:07:15 ip-10-0-1-63 jenkins[16995]: *****
Feb 23 14:07:15 ip-10-0-1-63 jenkins[16995]: *****
Feb 23 14:07:53 ip-10-0-1-63 jenkins[16995]: 2024-02-23 14:07:53.258+0000 [id=30]
Feb 23 14:07:53 ip-10-0-1-63 jenkins[16995]: 2024-02-23 14:07:53.285+0000 [id=22]
Feb 23 14:07:53 ip-10-0-1-63 systemd[1]: Started Jenkins Continuous Integration Serve
Feb 23 14:07:53 ip-10-0-1-63 jenkins[16995]: 2024-02-23 14:07:53.791+0000 [id=45]
Feb 23 14:07:53 ip-10-0-1-63 jenkins[16995]: 2024-02-23 14:07:53.795+0000 [id=45]
lines 1-19/19 (END)

```

Виходимо з перегляду статусу сервісу натисканням: **Ctrl+C**

Налаштовуємо автоматичний запуск сервісу Jenkins при запуску ОС:

```
sudo systemctl enable jenkins
```

Провіряємо доступність Web інтерфейсу переходячи на адресу сервера через браузер:

`http://<your_host_ip>:8080` - де `your_host_ip` - це ір вашої віртуальної машини.

На сторінці у браузері буде повідомлення про потребу розблокувати Jenkins:

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/lib/jenkins/secrets/initialAdminPassword
```

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

Administrator password

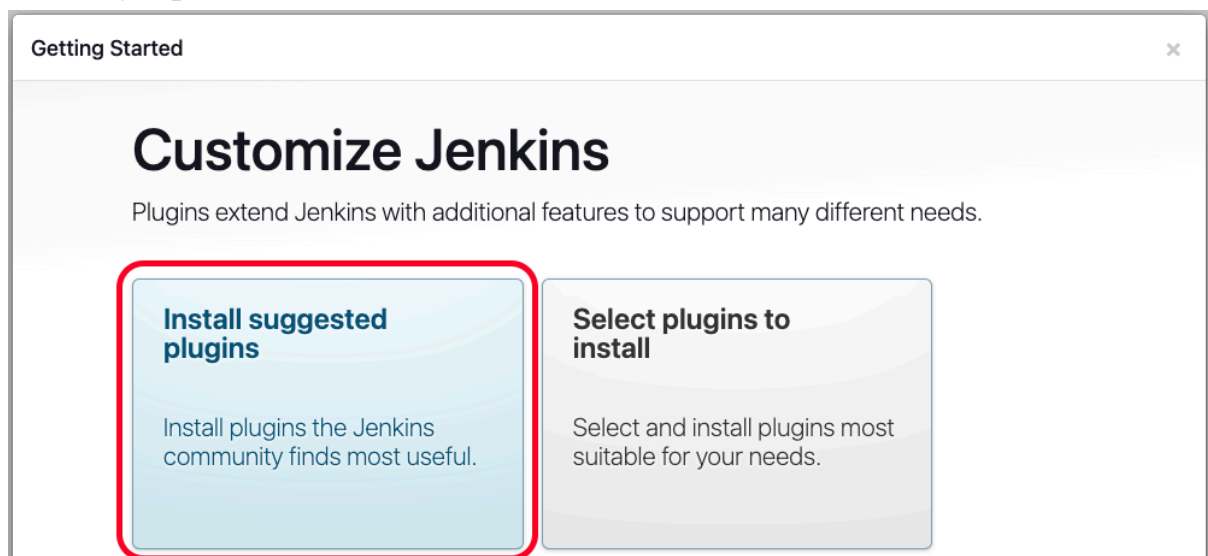
Як зазначено на сторінці, щоб почати користуватись Jenkins, потрібно ввести початковий пароль адміністратора що лежить у файлі за шляхом:
`/var/lib/jenkins/secrets/initialAdminPassword`

Тому повертаємось до терміналу віртуальної машини і читаємо вміст вказаного файлу командою:

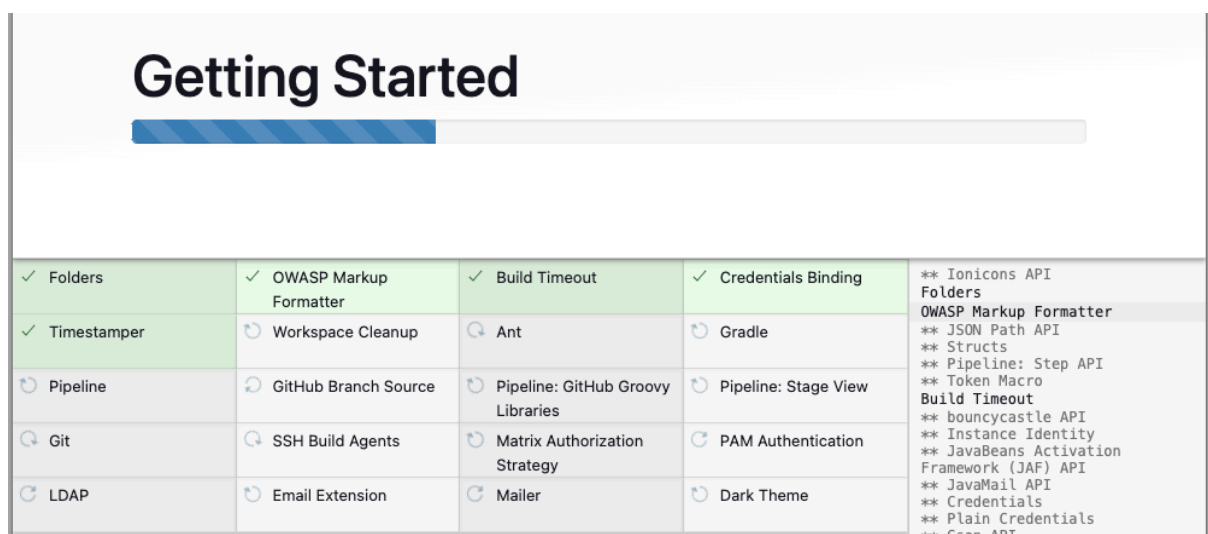
```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Отриманий пароль вводимо у поле на веб сторінці Jenkins та натискаємо **Continue**

Далі переходимо до встановлення необхідних плагінів, для початкового використання можна виконати встановлення мінімального пакету найпопулярніших з них.



Очікуємо коли встановлення плагінів завершиться



Після завершення встановлення плагінів, створюємо акаунт адміністратора:

Getting Started

Create First Admin User

Username

Password

Confirm password

Full name

E-mail address

Jenkins 2.440.1

[Skip and continue as admin](#)

Save and Continue

Зберігаємо налаштованого користувача і переглядаємо Jenkins URL, за потреби її можна змінити, але в рамках лабораторної роботи залишаємо запропоновану URL

Getting Started

Instance Configuration

Jenkins URL:

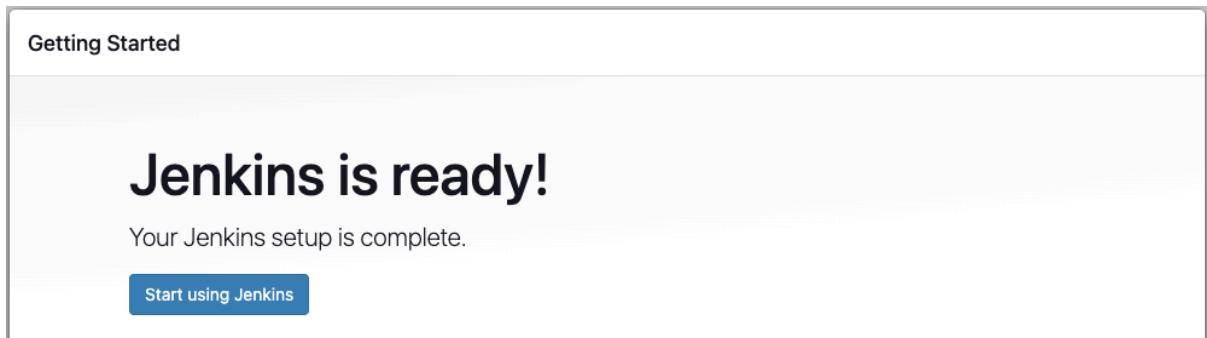
http://

:8080/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Зберігаємо налаштування та отримуємо повідомлення про готовність Jenkins до роботи:



Натискаємо **Start using Jenkins** і веб інтерфейс нас перенаправить на головну сторінку.

Після входу на головну сторінку Jenkins переходимо до налаштування Docker. Повертаємося в термінал та інсталуємо необхідні пакети для Docker:

```
sudo apt install apt-transport-https ca-certificates curl  
software-properties-common
```

Додаємо в систему репозиторій з пакетами Docker:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg |  
sudo apt-key add -  
sudo add-apt-repository "deb [arch=amd64]  
https://download.docker.com/linux/ubuntu focal stable"
```

Оновлюємо кеш :

```
apt-cache policy docker-ce
```

Інсталуємо Docker:

```
apt-cache policy docker-ce
```

Перевіряємо чи запустився і чи коректно працює Docker:

```
sudo systemctl status docker
```

Коректним виводом має бути:

Loaded: loaded

Active: active (running)

```

ubuntu@ip-10-0-1-63:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enab>
   Active: active (running) since Fri 2024-02-23 15:48:51 UTC; 2min 5s ago
   TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
    Main PID: 18897 (dockerd)
      Tasks: 7
     Memory: 57.4M
    CGroup: /system.slice/docker.service
            └─18897 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/container>

Feb 23 15:48:50 ip-10-0-1-63 systemd[1]: Starting Docker Application Container Engine>
Feb 23 15:48:50 ip-10-0-1-63 dockerd[18897]: time="2024-02-23T15:48:50.371639188Z" le>
Feb 23 15:48:50 ip-10-0-1-63 dockerd[18897]: time="2024-02-23T15:48:50.380916375Z" le>
Feb 23 15:48:50 ip-10-0-1-63 dockerd[18897]: time="2024-02-23T15:48:50.596118017Z" le>
Feb 23 15:48:50 ip-10-0-1-63 dockerd[18897]: time="2024-02-23T15:48:50.955496559Z" le>
Feb 23 15:48:51 ip-10-0-1-63 dockerd[18897]: time="2024-02-23T15:48:51.067065254Z" le>
Feb 23 15:48:51 ip-10-0-1-63 dockerd[18897]: time="2024-02-23T15:48:51.075204261Z" le>
Feb 23 15:48:51 ip-10-0-1-63 dockerd[18897]: time="2024-02-23T15:48:51.271977864Z" le>
Feb 23 15:48:51 ip-10-0-1-63 systemd[1]: Started Docker Application Container Engine.
lines 1-20/20 (END)

```

Виходимо з перегляду статусу сервісу натисканням: **Ctrl+C**

Налаштовуємо автоматичний запуск сервісу Docker при запуску ОС:

```
sudo systemctl enable docker
```

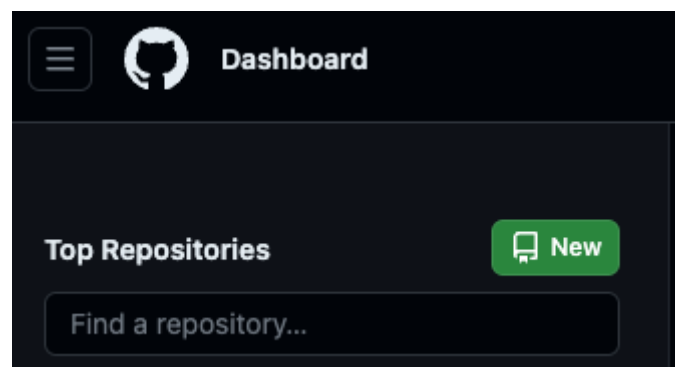
Для того щоб мати змогу запускати команди Docker не вводячи sudo:

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

Перелогінуємось в сесію терміналу або виконуємо команду:

```
su - ${USER}
```

Створюємо репозиторій в якому буде створено конфігураційні файли для Jenkins та Docker : **NEW**



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *

 VFedorch ▾

Repository name *

/ PRIKM

✔ PRIKM is available.

Great repository names are short and memorable. Need inspiration? How about **bug-free-sniffle** ?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:



Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore


.gitignore template: None ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None ▾

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a public repository in your personal account.

Create repository

Наживаємо репозиторій за бажанням, або аббревіатурою назви курсу лабораторних робіт.

Повертаємось до терміналу і ініціалізуємо репозиторій у робочій директорії:

```
git init
```

Далі створюємо **Jenkinsfile**

Приклад:

```

pipeline {
    agent any

    stages {
        stage('Start') {
            steps {
                echo 'Lab_1: nginx/custom'
            }
        }

        stage('Build nginx/custom') {
            steps {
                sh 'docker build -t nginx/custom:latest .'
            }
        }

        stage('Test nginx/custom') {
            steps {
                echo 'Pass'
            }
        }

        stage('Deploy nginx/custom'){
            steps{
                sh "docker run -d -p 80:80 nginx/custom:latest"
            }
        }
    }
}

```

Ссворюємо **Dockerfile**.

Приклад:

```

FROM nginx:latest
COPY ./index.html /usr/share/nginx/html/index.html

```


Створюємо вебсторінку **index.html**

Приклад:

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Lab_1</title>
  </head>
  <body>
    <h2>Hello from Docker, launched by Jenkins</h2>
  </body>
</html>
```

Зберігаємо всі файли, та робимо коміт у нову гілку в репозиторії

```
git add .
git commit -m "first commit"
git branch -M Lab_1
git remote add origin
https://github.com/назва\_вашого\_акаунту\_git/назва\_вашого\_репозитору.git
git push -u origin Lab_1
```

Переходимо до веб інтерфейсу Jenkins і створюємо першу Job:



+ New Item

👤 People

📁 Build History

⚙️ Manage Jenkins


📌 My Views


Обираємо тип - Pipeline, та даємо назву для Job:


Enter an item name


Lab_1


» Required field


**Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**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.

**Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

OK

Обираємо **Pipeline script from SCM**:

Configure

⚙️ General

🔧 **Advanced Project Options**

📄 Pipeline

Advanced Project Options

Advanced ▾

Pipeline

Definition

✓ Pipeline script

Pipeline script from SCM

Script ?

1

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save

Apply

SCM - Git:

Pipeline

Definition

Pipeline script from SCM

SCM ?

✓ None

Git

Script Path ?

Jenkinsfile

Вводимо назву репозиторію і назву робочої гілки у відповідні поля:

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/назва_вашого_акаунту_git/назва_вашого_репозиторію.git

Please enter Git repository.

Credentials ?

- none -

+ Add

Advanced

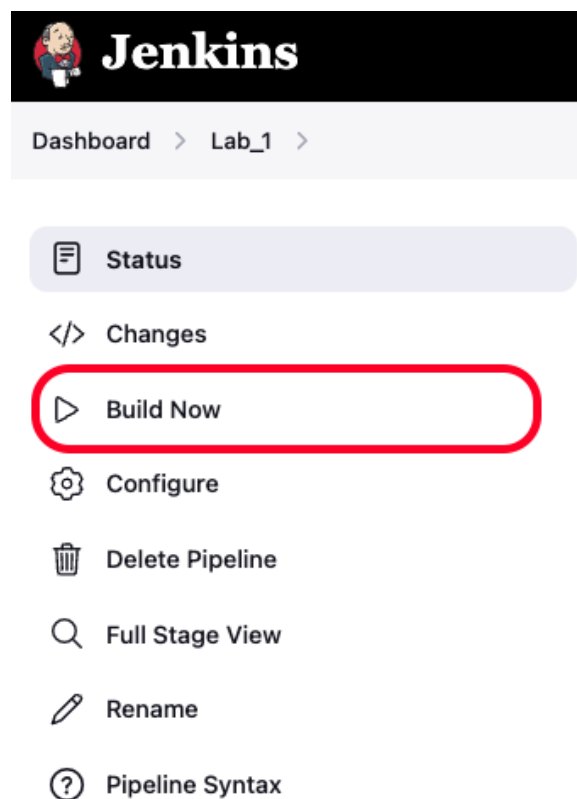
Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

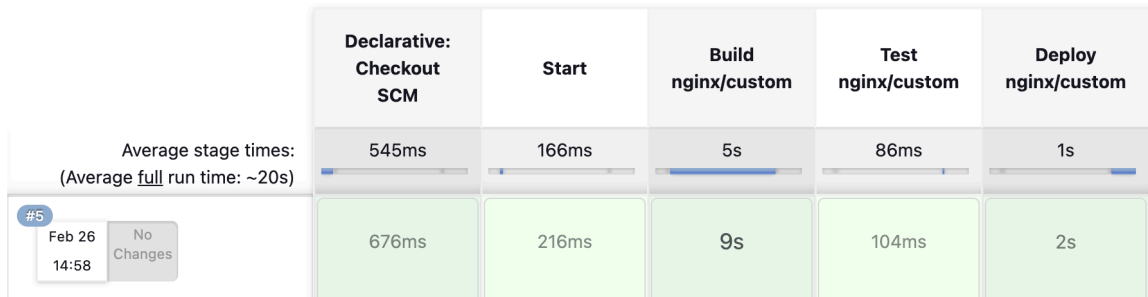
*/Lab_1

Зберігаємо натисканням кнопки **Save**
Запускаємо Job кнопкою **Build Now**:

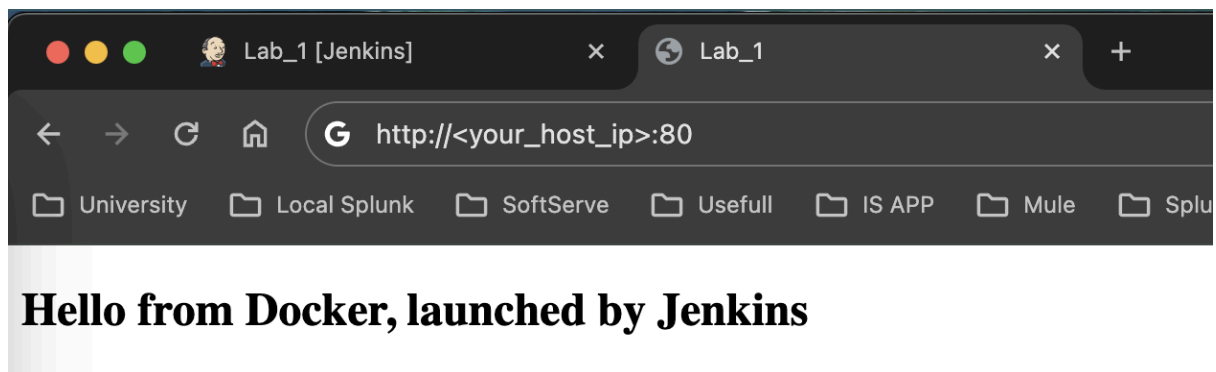


Відбувається запуск Pipeline і у випадку успішного запуску контейнера з вебсторінкою, Jenkins виведе наступне повідомлення:

Stage View



Отже можна перейти на адресу сторінки яку ми розгортали в контейнері, а саме: `http://<your_host_ip>:80`



Для повного виконання лабораторної роботи потрібно внести зміни в `index.html` та вивести своє повідомлення/зображення/тощо на сторінці а також змінити `Jenkinsfile`, додавши/змінивши стейджі в `pipeline`