Bab 10

CI/CD Jenkins

SEKOLAH DEVOPS CILSY

Detail Materi

Konsep Dasar Jenkins.

Indikator :
Dapat menjalankan konsep CI/CD menggunakan Jenkins pada server testing dan Server Production.

InstallasiJenkins pada Host dan juga Container

CI/CD Menggunakan Jenkins

Management Node Untuk Membagi Beban.

Deployment Aplikasi pada Node

Modul Sekolah DevOps Cilsy

Penulis : Muhammad Fakhri Abdillah
Editor: Muhammad Fakhri Abdillah
Revisi Batch 7

Penerbit : **PT. Cilsy Fiolution Indonesia**
Web Site : https://cilsyfiolution.com , https://devops.cilsy.id

Daftar Isi

Cover....1
9. CI/CD Jenkins....4
Learning Outcomes....4
Outline Materi....4
9.1. Jenkins Pipeline....5
9.1.1. Apa itu Jenkins Pipeline?....5
9.1.2. Apa itu Continuous Delivery Pipeline?....5
9.1.3. Install Build Pipeline Plugin Jenkins....6
9.1.4. Membuat Jenkins Pipeline....6
9.2. JenkinsFile....9
9.2.1. Declarative vs Scripted Pipeline Syntax....10
9.3. Konsep Dasar Jenkins Pipeline....10
9.4. Membuat JenkinsFile Sederhana....11
9.4.1. Java....11
9.4.2. Node.js / Javascript....13
9.4.3. Ruby....14
9.4.4. Python....14
9.4.5. PHP....14
9.4.6. Exercise....15

9.
CI/CD Jenkins

Learning Outcomes

Setelah selesai mempelajari bab ini, peserta mampu :

1. Memahami Konsep Dasar Jenkins pipeline.
2. Memahami cara membuat JenkinsFile

Outline Materi

1. Jenkins Pipeline
2. JenkinsFile

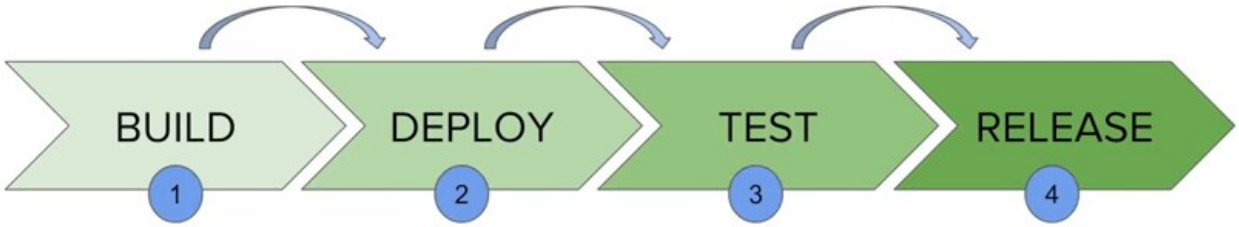
9.1. Jenkins Pipeline

9.1.1. Apa itu Jenkins Pipeline?

Jenkins pipeline adalah kombinasi dari plugin yang menduking integrasi dan implementasi dari continous delivery pipeline. Jenkins pipeline memiliki server otomatisasi yang dapat diperluas (extensible) untuk membuat pipeline pengiriman yang sederhana dan kompleks sebagai kode melalui pipeline DSL. Pipeline adalah sekelompok peristiwa yang saling terkait satu sama lain secara berurutan,

9.1.2. Apa itu Continuous Delivery Pipeline?

Pada Jenkins pipeline, setiap job atau event memiliki semacam dependency, setidaknya satu atau lebih event.



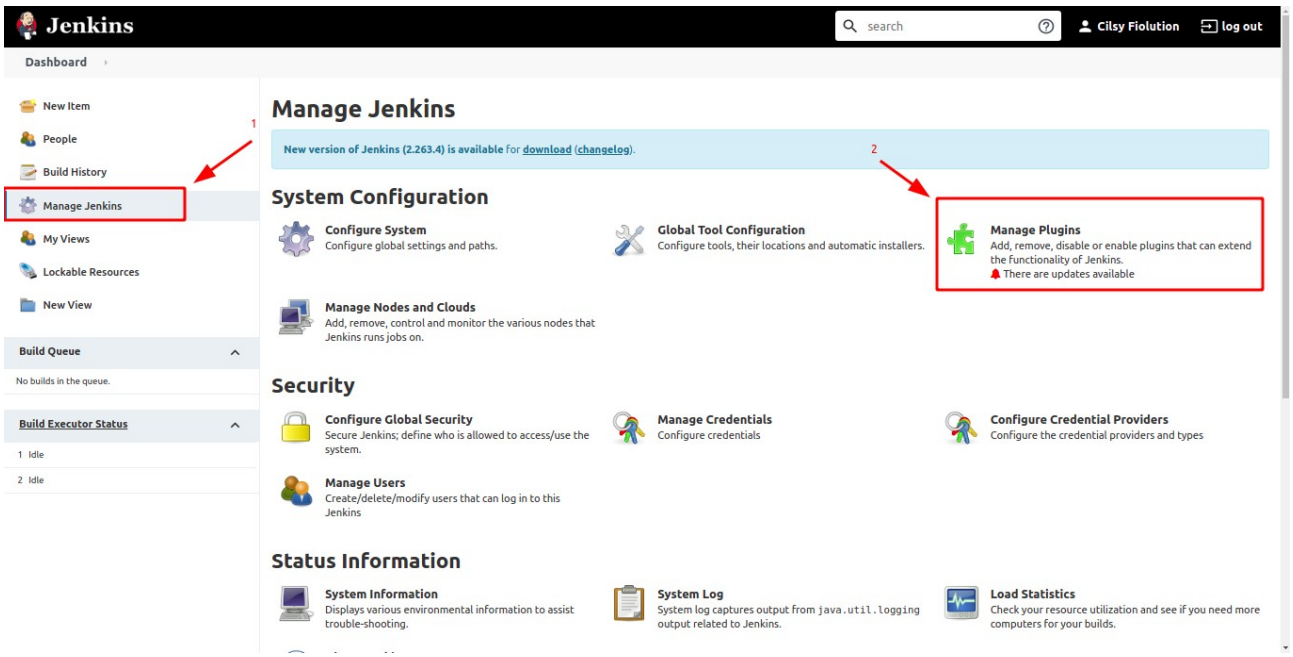
Gambar diatas merepresentasikan continuous delivery pipeline pada Jenkins. Pada proses tersebut, kita memiliki group of states yang disebut build, deploy, test, dan release. Setiap event process saling terhubung satu sama lain. Setiap state memiliki eventnya masing-masing, yang mana bekerja secara berurutan, sehingga disebut continuous delivery pipeline.

Continuous delivery pipeline adalah sebuah automated expression untuk menampilkan process kita dalam mendapatkan software untuk version control. Jadi, setiap perubahan yang dibuat dalam perangkat lunak kita melalui sejumlah proses kompleks dalam perjalanannya untuk dirilis. Ini juga melibatkan pengembangan perangkat lunak dengan cara yang andal dan dapat

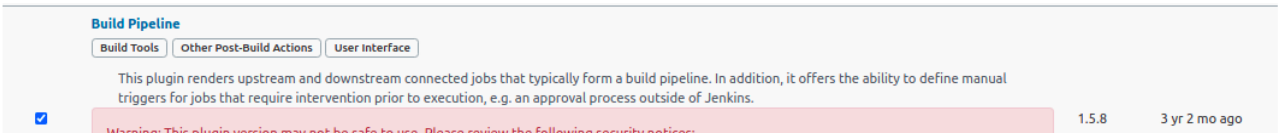
diulang, dan perkembangan perangkat lunak yang dibangun melalui berbagai tahap pengujian dan penerapan.

9.1.3. Install Build Pipeline Plugin Jenkins

Dengan plugin **build pipeline**, kita dapat membuat sebuah pipeline view dari incoming dan otucoming jobs, dan membuat trigger yang dibutuhkan. Untuk menginstall plugin ini, buka dashboard Jenkins, lalu masu ke menu **Manage Jenkins > Manage Plugins.**

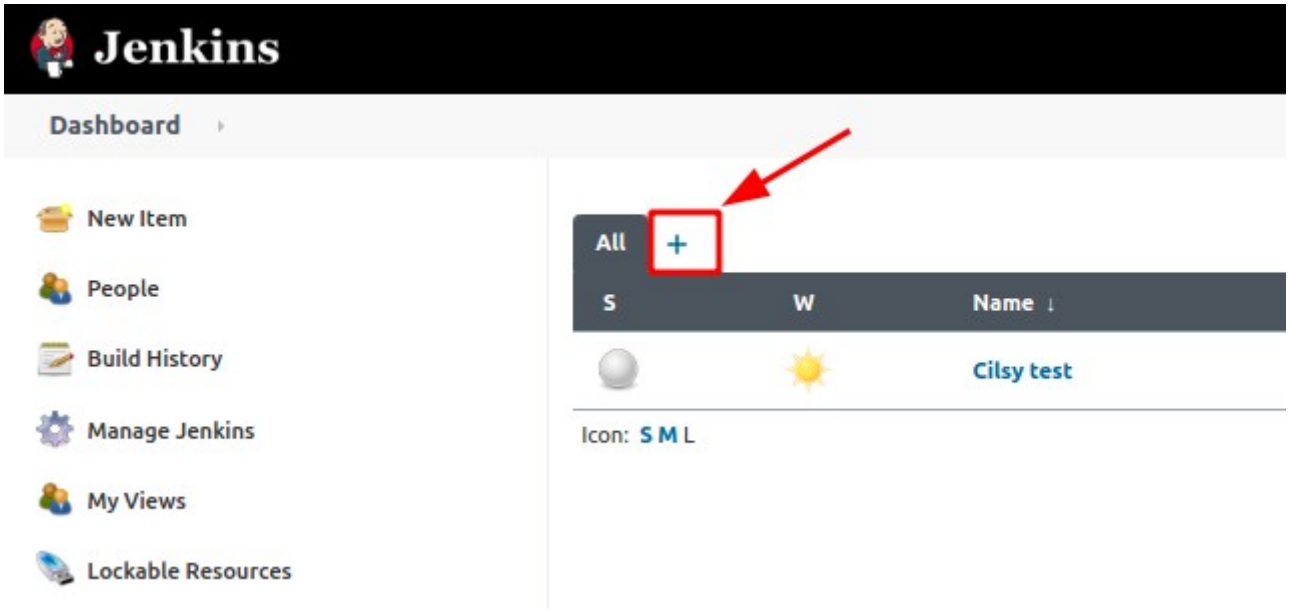


Selanjutnya, kita cari build Pipeline Plugin.

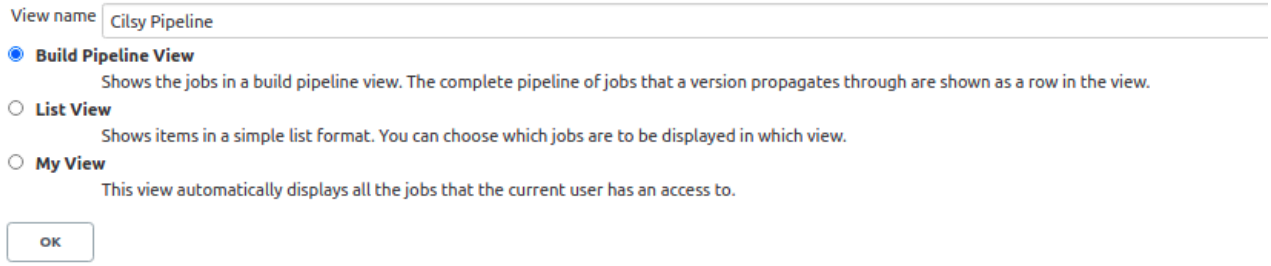


9.1.4. Membuat Jenkins Pipeline

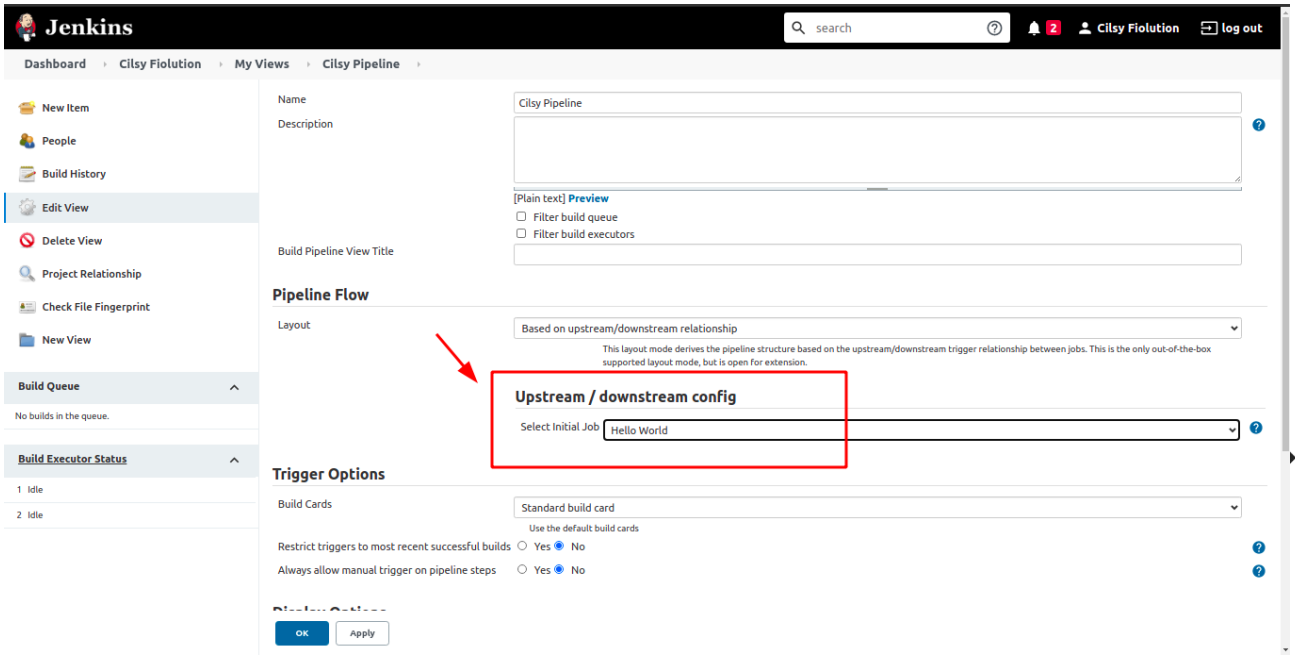
Setelah terinstall, di langkah selanjutnya kita akan mencoba membuat pipeline sederhana. Tekan tanda **+** pada Jenkins dashboard untuk membuat sebuah pipeline view.



Masukkan nama View, dan pilih Build Pipeline View.

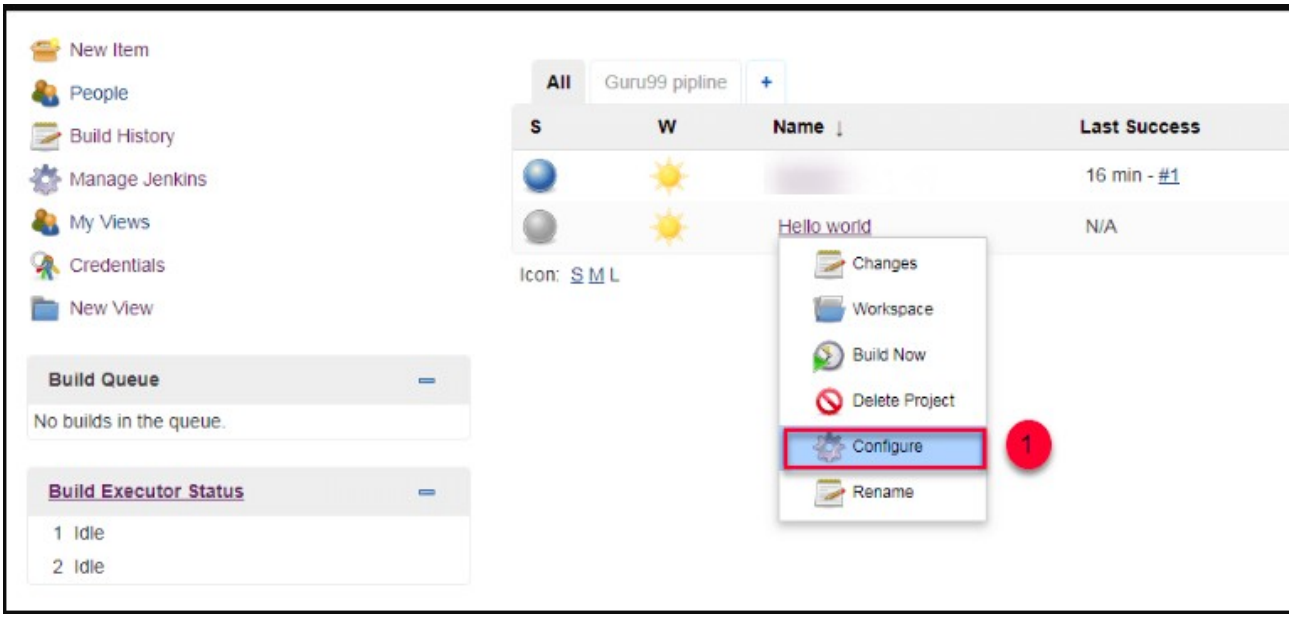


Kemudian, pilih job yang akan ditampilkan. Contohnya disini penulis sudah membuat job HelloWorld yang isinya adalah untuk menampilkan file HelloWorld dengan bahasa Java. Setelah itu, pilih Apply, lalu OK.

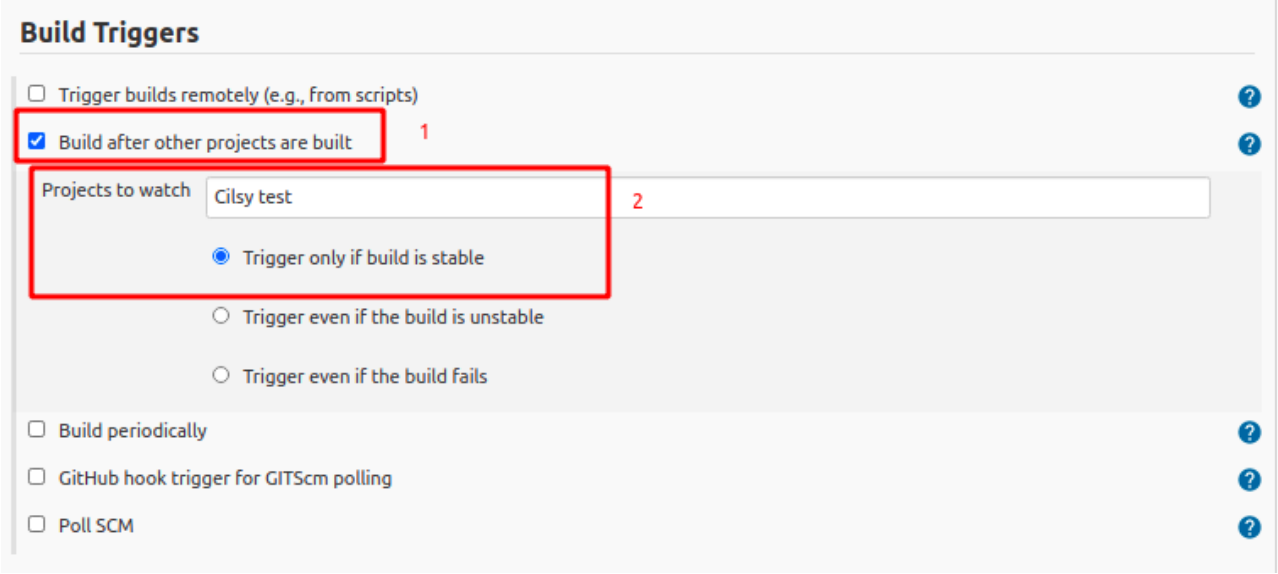


Setelah itu, akan muncul tampilan sebagai berikut

Untuk menjalankan pipeline build, kita perlu menghubungkan job terlebih dahulu. Caranya, adalah dengan cara kembali ke halaman job dan pilih configure.

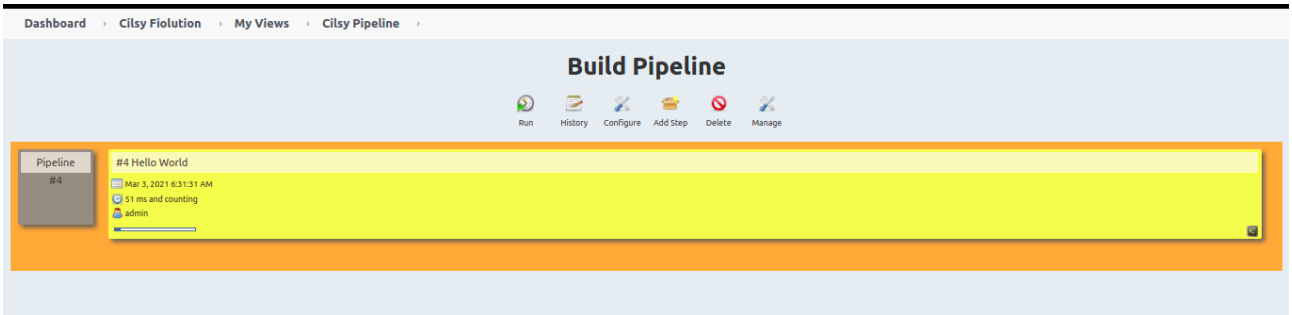


Pada bagian Build Triggers, pilih Build after other project are built. Disini penulis memilih proect pertama yang telah dibuat, yaitu Cilsy Test.



Selanjutnya, masuk ke Jenkins Dashboard dan buat sebuah view lagi dengan menekan tombol + . Pilih Build pipeline View, kemudian klik OK.

Untuk menjalankan Pipeline, nantinya akan muncul tampilan sebagai berikut.



9.2. JenkinsFile

Jenkins pipeline dapat didefinisikan menggunakan sebuah text file bernama JenkinsFile. Kita dapat mengimplementasikan pipeline as code menggunakan JenkinsFile, dan ini dapat didefinisikan dengan menggunakan domain specific language (DSL). Dengan JenkinsFile, kita dapat menulis step yang diperlukan untuk menjalankan sebuah Jenkins pipeline.

Keuntungan menggunakan JenkinsFile:

1. Kita dapat membuat pipeline secara otomatis untuk semua branch dan menjalankan pull request hanya dengan satu JenkinsFile.

   AndaKita dapat mereview kode Jenkins kita pada pipeline

2. Kita dapat mengaudit pipeline Jenkins kita
3. Ini adalah sumber tunggal untuk pipeline kita dan dapat dimodifikasi oleh banyak pengguna.

JenkinsFIle dapat didefinisikan baik menggunakan Web UI ataupun menggunakan Jenkins File.

9.2.1. Declarative vs Scripted Pipeline Syntax

Ada 2 type Jenkins pipeline syntax yang digunakan untuk mendefinisikan JenkinsFile, yaitu:

- Declarative

Declarative pipeline syntax menawarkan cara mudah untuk membuat pipeline. Ini berisi predefined hierarki untuk membuat Jenkins pipeline. Ini memberikan kita kemampuan untuk mengontrol semua aspek eksekusi pipeline dengan cara yang sederhana dan langsung.

- Scripted

Scripted Jenkins pipeline dijalankan pada master Jenkins dengan bantuan lightweight executor. Jenis ini menggunakan resource yang sangat sedikit untuk menerjemahkan pipeline menjadi atomic commands.

9.3. Konsep Dasar Jenkins Pipeline

Sebuah script JenkinsFile terdiri dari beberapa block, yaitu **Pipeline**, **Node**, **Stage**, dan **Step**

Pertama adalah block **pipeline**, block pipeline adalah bagian penting ketika ingin membuat sebuah jenkinsfile dengan tipe declarative. Pipeline merupakan

serangkaian instruksi yang dibutuhkan untuk keseluruhan build process. Dengan pipeline, kita dapat mem-build, test dan deliver sebuah aplikasi.

Kemudian, **Node** merupakan penamaan Jenkins machine. Sebuah node block biasanya digunakan untuk scripted pipeline syntax.

Selanjutnya adalah block **stages**, pada block ini kita define stage apa saja yang mau kita implementasikan pada pipeline kita, di dalam satu stages dapat memiliki 1 atau lebih stage, lalu di dalam block stage terdapat block step, pada block inilah kita melakukan eksekusi untuk setiap stage yang terkait. Itu adalah beberapa block yang setidaknya ada pada sebuah jenkinsfile.

9.4. Membuat JenkinsFile Sederhana

Pada bagian ini, kita akan mengenal beberapa langkah sederhana untuk membuat JenkinsFile. Proses pembuatan JenkinsFile ini akan sangat mudah apabila kita sudah menguasai bash programming.

Berikut adalah beberapa contoh JenkinsFIle pada beberapa bahasa pemrograman.

9.4.1. Java

Untuk membuat JenkinsFile yang memuat job Java, berikut adalah contoh sintaks sederhananya.

```
pipeline {
    agent { docker { image 'maven:3.3.3' } }
    stages {
        stage('build') {
            steps {
                sh 'mvn --version'
            }
        }
    }
}
```

Berikut adalah contoh penggunaan JenkinsFile pada production

```
pipeline {
    agent any
    tools {
        maven 'maven 3.5.4'
        jdk 'JDK-1.8'
    }
    stages {
        stage ('Pull source code') {
            steps {
                 git branch: 'development', credentialsId: '3965d3f2-9d56-40b5-
b293-c9a10af6d0f6', poll: false, url:
'https://fakhri@bitbucket.org/sdcilsy/cilsy-backend.git'
            }
        }


        stage ('Build java') {
             steps {
                 sh "mvn clean package -DskipTests=true
            }
         }


        stage('Archive artifact') {
            steps {
                archiveArtifacts 'src/**, target/cilsy-0.0.1-SNAPSHOT.jar,
projectctl.sh, onlyIfSuccessful: true'
            }
        }


        stage('Publish artifact') {
            steps {
                sshPublisher(publishers: [sshPublisherDesc(configName: 'cilsy-
be', transfers: [sshTransfer(cleanRemote: false, excludes: '', execCommand:
'''cd /home/cilsy/cilsy-dev;
                ./projectctl.sh stop development;
                sleep 3;
                DATE=`date \'+%Y-%m-%d %H:%M:%S\'`
                mv *.jar jar-archive/cilsy-"$DATE".jar;
```

```
                cd /home/cilsy;
                ls -ls;
                mv target/*.jar ~/cilsy-dev;
                sleep 1;
                mv src ~/cilsy-dev;
                sleep 1;
                mv projectctl.sh ~/cilsy-dev;
                sleep 1;
                cd /home/cilsy/cilsy-dev;
                chmod +x projectctl.sh;
                ./projectctl.sh start development;
                sleep 30;
                ''', execTimeout: 120000, flatten: false, makeEmptyDirs: false,
noDefaultExcludes: false, patternSeparator: '[, ]+', remoteDirectory: '',
remoteDirectorySDF: false, removePrefix: '', sourceFiles: 'target/cilsy-0.0.1-
SNAPSHOT.jar, projectctl.sh, src/**')], usePromotionTimestamp: false,
useWorkspaceInPromotion: false, verbose: false)])
            }
        }
    }
}
```

9.4.2. Node.js / Javascript

```
pipeline {
    agent { docker { image 'node:14-alpine' } }
    stages {
        stage('build') {
            steps {
                sh 'npm --version'
            }
        }
    }
}
```

9.4.3. Ruby

```
pipeline {
    agent { docker { image 'ruby' } }
    stages {
        stage('build') {
            steps {
                sh 'ruby --version'
            }
        }
    }
}
```

9.4.4. Python

```
pipeline {
    agent { docker { image 'python:3.5.1' } }
    stages {
        stage('build') {
            steps {
                sh 'python --version'
            }
        }
    }
}
```

9.4.5. PHP

```
pipeline {
    agent { docker { image 'php' } }
    stages {
        stage('build') {
            steps {
                sh 'php --version'
            }
        }
    }
```

```
}
```

9.4.6. Exercise

Praktek :

1. Buat sebuah Jenkins pipeline untuk melakukan deployment web sederhana menggunakan PHP!
2. Buat juga JenkinsFile-nya.
3. Pastikan bahwa aplikasi sudah berjalan dengan baik.