DEVOPS: IS 360-Degree view of your S/W development lifecycle.

**CI- CONTUNESOUS INTEGRATION** -VERSION CONTROL -CODE COMMIT -CODE BUILD (COMPILE, REVIEW, VALIDATE, TEST, PACKAGE .WAR OR .JAR) -TEST (UNIT TESTING, INTEGRATION TESTING)

**CONTINEOUS DELIVERY OR RELEASE**

-DEPLOY (DEPLOY THE BUILD APPLICATION ON TEST SERVERS FOR U.A.T.)

**CONTINEAOUS DEPLOY**

-DEPLOY BUILD APPLICATION ON PRODUCTION SERVER AFTER Q A CHECK

**JENKINS**

IS A OPEN SOURCE AUTOMATION TOOL, PLUGIN BASED,written in GO{JAVA} language?

-PLUGIN ALLOWS TO INTEGERATION OF VARIOUS DEVOPS STAGES

#systemctl start Jenkins #systemctl enable Jenkins #systemctl start docker

Here for maven projects -Jenkins install a maven -maven install all required package to compile the application -THEN JENKINS use docker to deploy the application.

New project -project name{project-1} ->maven -> ok -> build

Execute shell #

#!bin/bash cd java\_web\_code mvn clean install #build test started echo “test phase started” cd .. /integration testing/ mvn clean verify -p integration test

Post build action -build another project [project-2] ->apply -save

New item -> project name{project-2} ->maven -> ok -> build

Execute shell #

#!bin/bash #post build echo “integration started” cd java\_web\_code/bin/ ##copy war file on docker root directory cp target /springboot.war ../docker/ echo “provision phase started & build docker container” cd ../docker/ sudo docker build -t prashantkatare/pipelinedemo .

Post build action -> other project run [project-3 started]

project-3

New item -> project name{job2} ->maven -> ok -> build

Execute shell #

#!bin/bash sudo docker rm -rf $container # run your container sudo docker run -it -d -p 8081:80 –name devopspipelinedemo prashantkatare/pipelinedemo #completion echo “deployed here http://localhost:8081”

NOW CREATE NEW PIPELINE

-INSTALL PLUGINS - -install build pipeline plugin -I install delivery pipeline plugin

==============================================================================================

NOW CREATE NEW PIPELINE

Name=[devops\_pipeline] -pipeline flow -upstream -select initial job [project-1] -APPLY

ADD - Build pipeline view -configure the view step -run & validate

**There are two way to create pipeline**

1 . **user interface** (Browser)

2. **SCRIPT-BASED (using JENKINS CICD FILE) {Declarative Script & Scripted script}**

To create Jenkins file, we have to mention steps like in jenkins file. -Stage1-Build, -Stage-2-Deploy, -Stage-3-Test, -Stage-4-Release

*jenkinsfile (Declarative Pipeline)*

pipeline {

agent any

stages {

stage('Build') {

steps {

//

}

}

stage('Test') {

steps {

//

}

}

stage('Deploy') {

steps {

//

}

}

}

}

*Jenkinsfile (Declarative Pipeline)*

pipeline {

agent any

options {

skipStagesAfterUnstable()

}

stages {

stage('Build') {

steps {

sh 'make'

}

}

stage('Test') {

steps {

sh 'make check'

junit 'reports/\*\*/\*.xml'

}

}

stage('Deploy') {

steps {

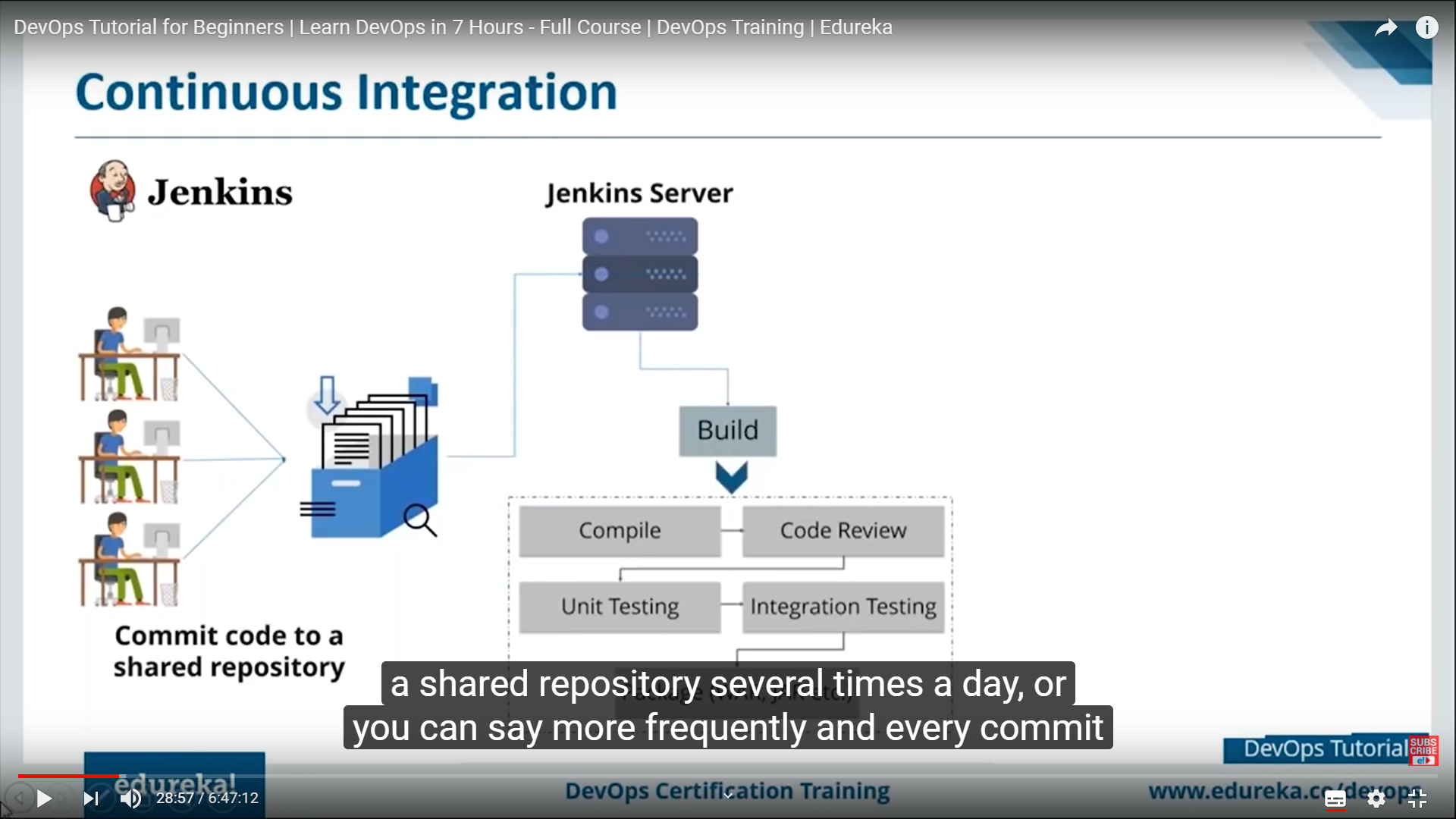
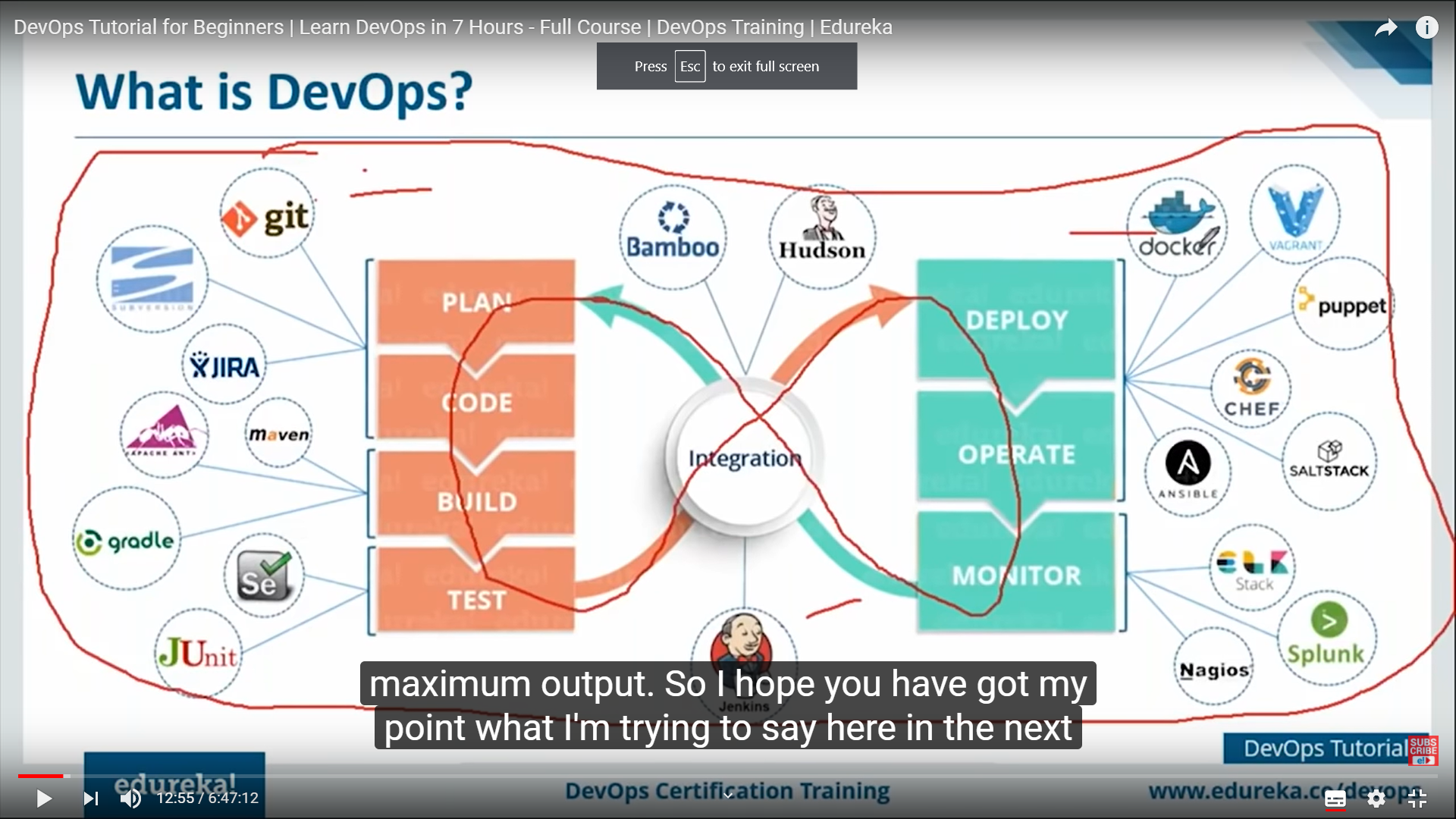
sh 'make publish'

}

}

}

}



Compile: convert all 15 files in to one .JAR or .WAR file (for JAVA).

Build=> Compile-> code review->Unit Test-> integration test-> Package in to JAR or WAR.

TEST TOOL =>Selenium, Sonar que, Junit

Continuous Delivery=>Build (JAR or WAR) deploy on Test Server (UAT Server) for User Accepting Testing.

Continuous Deploy => Then Deployment on production Server.

Snapshot= {incremental backup} => reduced the size of version file or versions become light weight.

