Mulesoft Deployments Guide

Deployment

Jagadishwar Reddy
MULEBYTES Bangalore

Contents

Introduction

Concept of Deployment

Deployment is a process of Bringing your Application to ACTIVE state by RUNNING it in RUNTIME. Mulesoft applications require MULE RUNTIME to bring them to working (running) state.
Deployable Archive: *File Generated by Exporting Mule Project, Mule 3 (.zip), Mule 4 (.jar)*

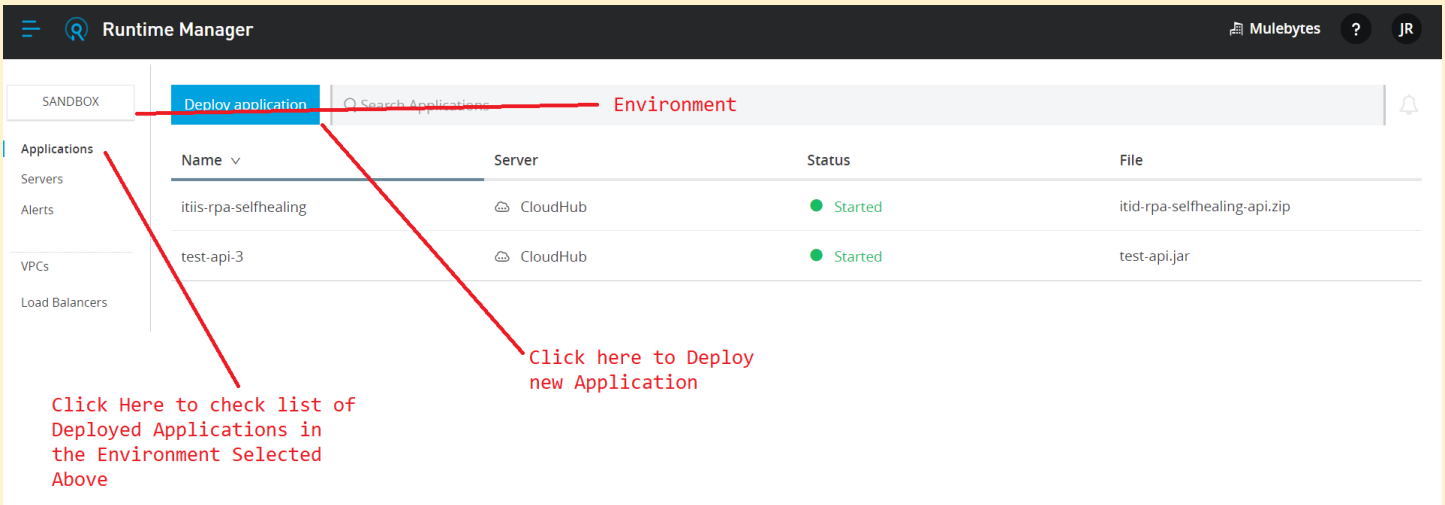
Types of Mule Runtime

Cloudhub

CloudHub is an Integration Platform as a Service (iPaaS). It enables you to deploy and run the application in the cloud via Runtime Manager. CloudHub is a scalable, multi-tenant, elastic, secure, and highly available iPaas. CloudHub is managed via the Runtime Manager console in the Anypoint platform.

Runtime Manager

Generic URL to Access Runtime Manager: https://anypoint.mulesoft.com/cloudhub



Worker

Worker is a Virtual Mule Server Instance, where Application will be deployed

Worker Size (vCore)

Worker Size or vCore is Max Memory assigned to a Worker.

Worker Type	Number of vCores for Max Use	vCore Size
MICRO	0.1 vCore	500 MB
SMALL	0.2 vCore	1 GB
MEDIUM	1 vCore	1.5 GB
LARGE	2 vCore	3.5 GB
XLARGE	4 vCore	7.5 GB
XXLARGE	8 vCore	15 GB
4XLARGE	16 vCore	32 GB

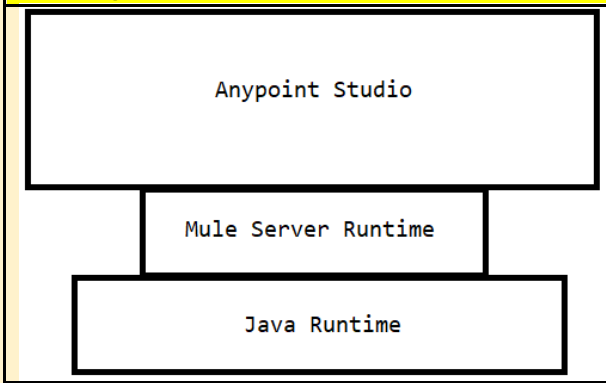
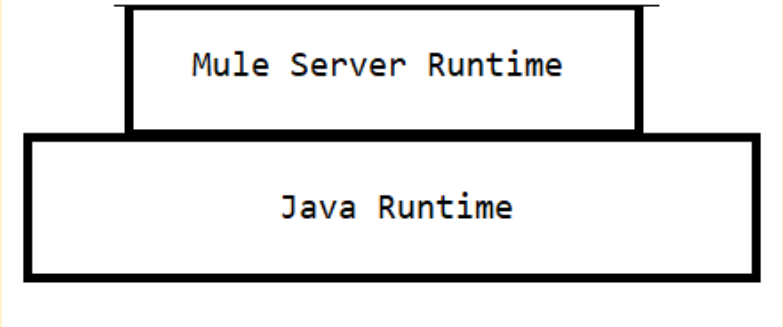
On-Premise

On-premises software (commonly misstated as on-premise, and alternatively abbreviated "on-prem") is installed and runs on computers on the premises of the person or organization using the software, rather than at a remote facility such as a server farm or cloud.

How does Mulesoft Runtime work

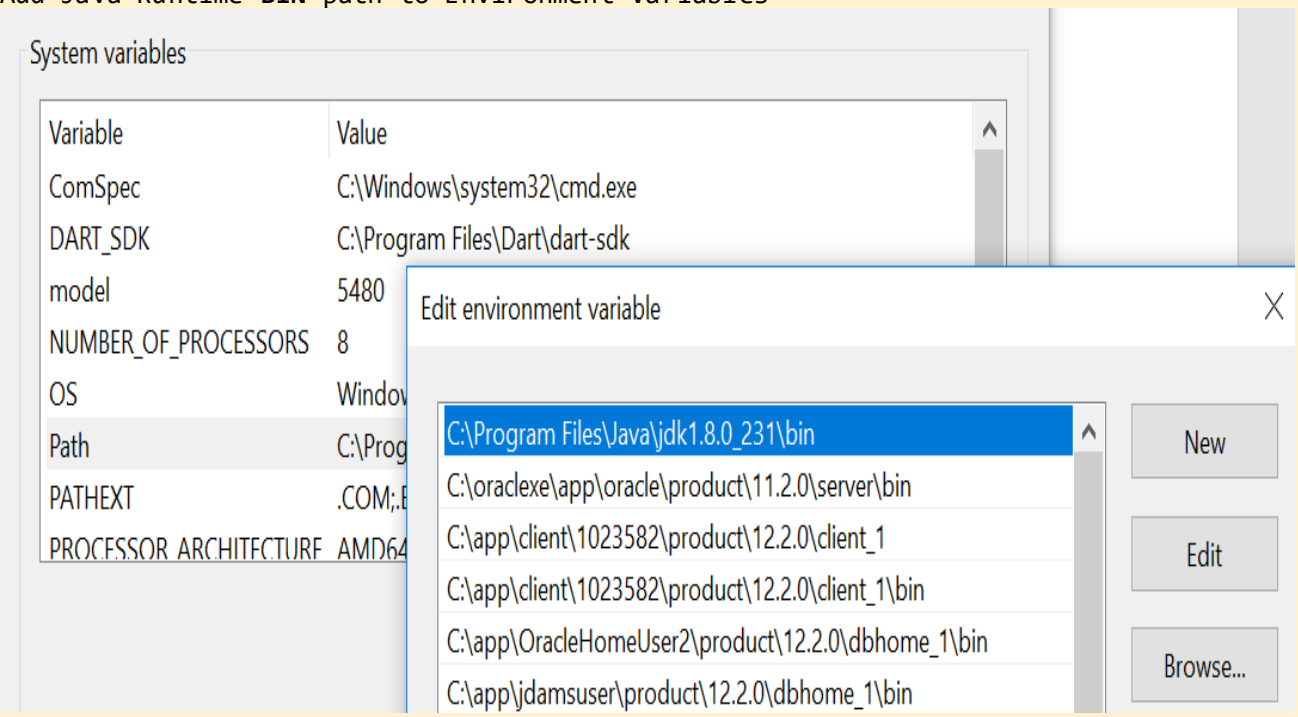
Mulesoft is a Java based integration tool, which mean it requires Java to run.
Every server which needs to be called as Mule Server should have

a. Java Runtime
b. Mule Server Runtime

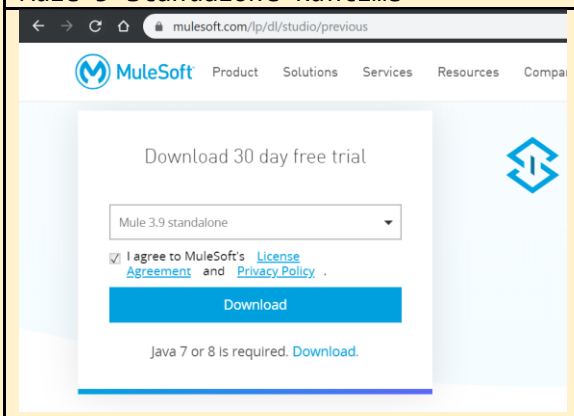
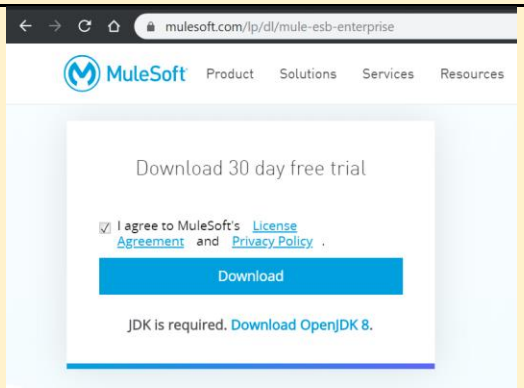
Development Environment	Deployment Environment
	

How to Setup Mulesoft On-Premise Runtime

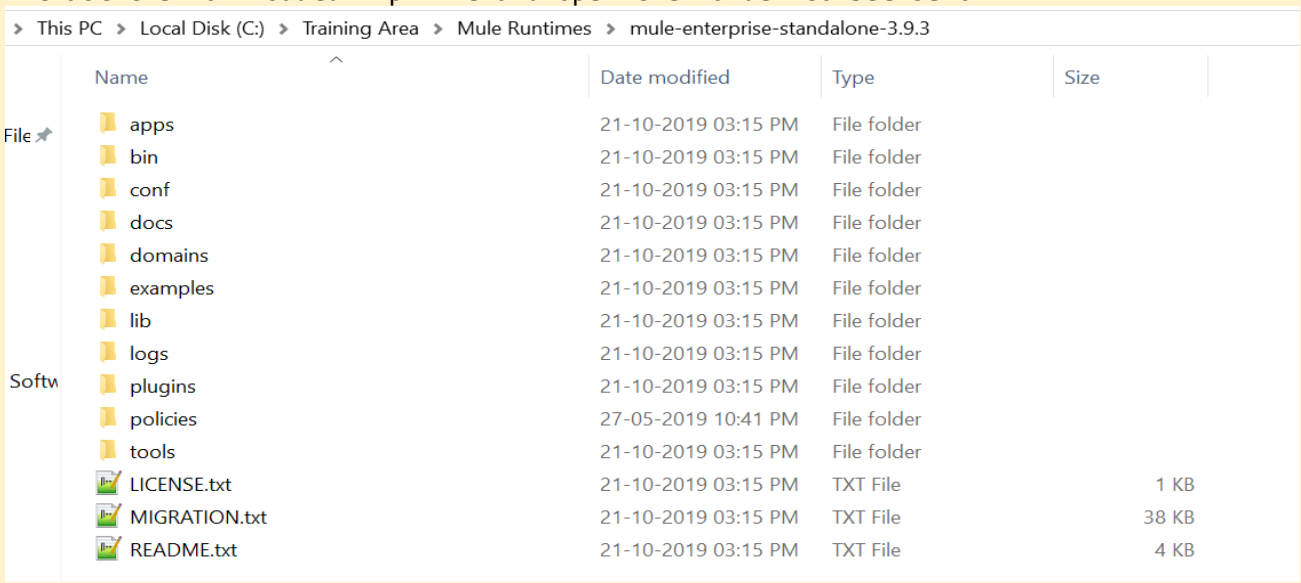
1. Setup Java Runtime
2. Add Java Runtime **BIN** path to Environment Variables

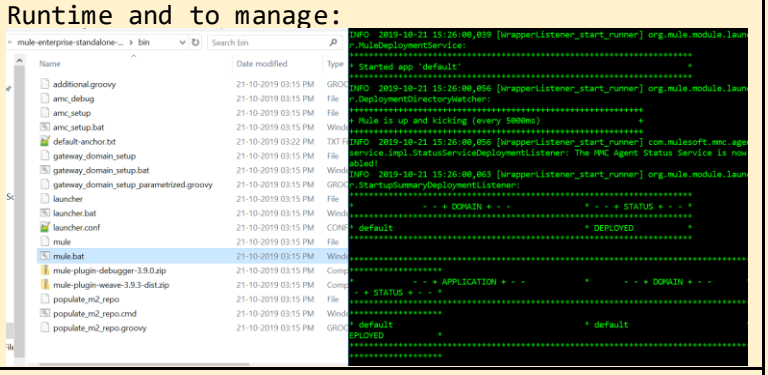


3. Download Mulesoft Standalone Runtime:

Mule 3 Standalone Runtime	Mule 4 Standalone Runtime
 https://www.mulesoft.com/lp/dl/studio/previous	 https://www.mulesoft.com/lp/dl/mule-esb-enterprise

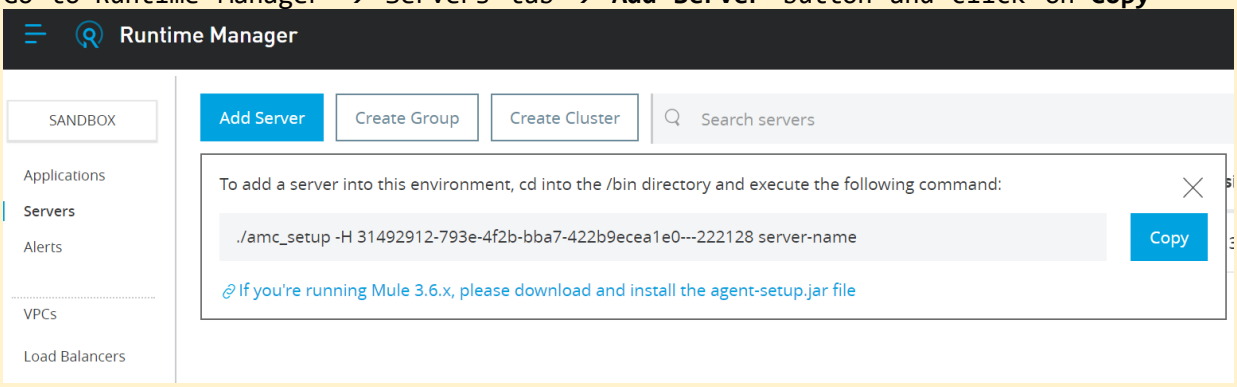
4. Extract the Downloaded Zip File and Open the Folder to see below



Folder Name	**Purpose**
apps	Folder where Deployed Applications will be saved with **<<app-name>>-anchor.txt** to verify they are deployed and Running.
bin	Contains, Commands to Start/Stop Mulesoft Runtime and to manage: 
logs	Application Log Files are saved here
conf	Contains Mulesoft Runtime License and Configuration File (wrapper.conf) for Mulesoft Runtime

How to add On-Premise Mule Runtime to Anypoint Platform (AMC Setup)

1. Login to Anypoint Platform https://anypoint.mulesoft.com/home/
2. Go to Runtime Manager → Servers tab → **Add Server** button and click on **Copy**



server-name: Give your own name to the server
Execute following command in **MULE_HOME\bin**

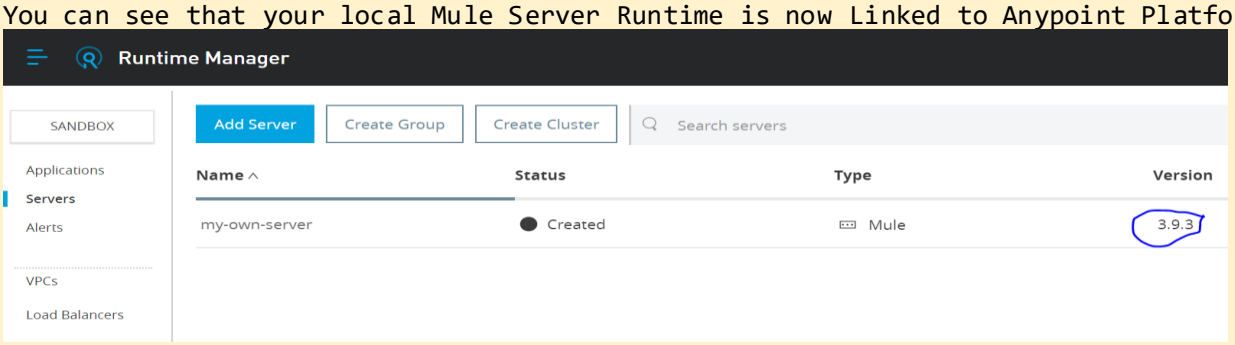
```
C:\Training Area\Mule Runtimes\mule-enterprise-standalone-3.9.3\bin>amc_setup -H 31492912-793e-4f2b-bba7-422b9ecea1e0---222128 my-own-server

Mule Agent Installer
----------------------------

For help please run the script with --help option


       INFO: MULE_HOME is set as C:\Training Area\Mule Runtimes\mule-enterprise-standalone-3.9.3
```

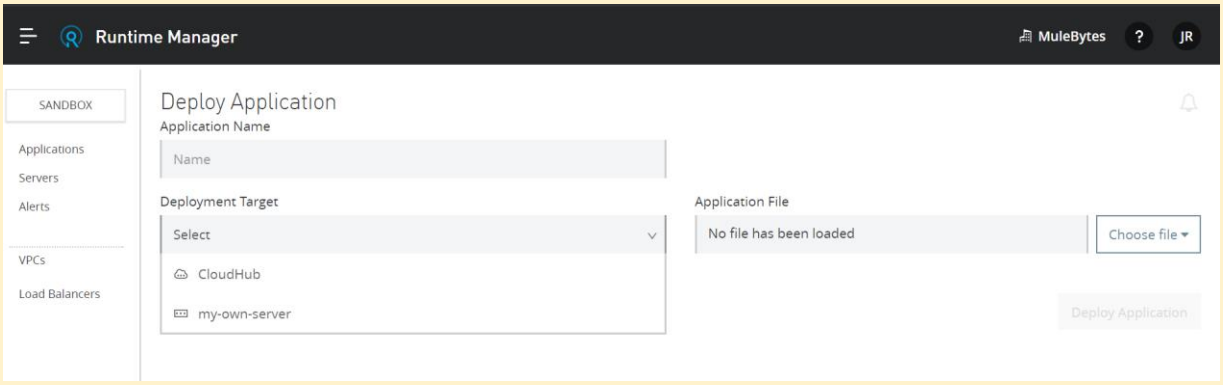
3. You can see that your local Mule Server Runtime is now Linked to Anypoint Platform:



4. Start Mule Server by going to **MULE_HOME\bin** open CMD and start Mule server by executing command **mule**

```
C:\mule-ee-distribution-standalone-3.9.3\mule-enterprise-standalone-3.9.3\bin>mule
MULE_HOME is set to C:\mule-ee-distribution-standalone-3.9.3\mule-enterprise-standalone-3.9.3
Running in console/foreground mode by default, use Ctrl-C to exit...
--> Wrapper Started as Console
Java Service Wrapper Standard Edition 64-bit 3.5.37
  Copyright (C) 1999-2018 Tanuki Software, Ltd. All Rights Reserved.
    http://wrapper.tanukisoftware.com
  Licensed to MuleSoft Inc. for Mule Runtime Enterprise Edition
```

5. Now try to Deploy an Application, you can see that Deployment Target now has your local server as well

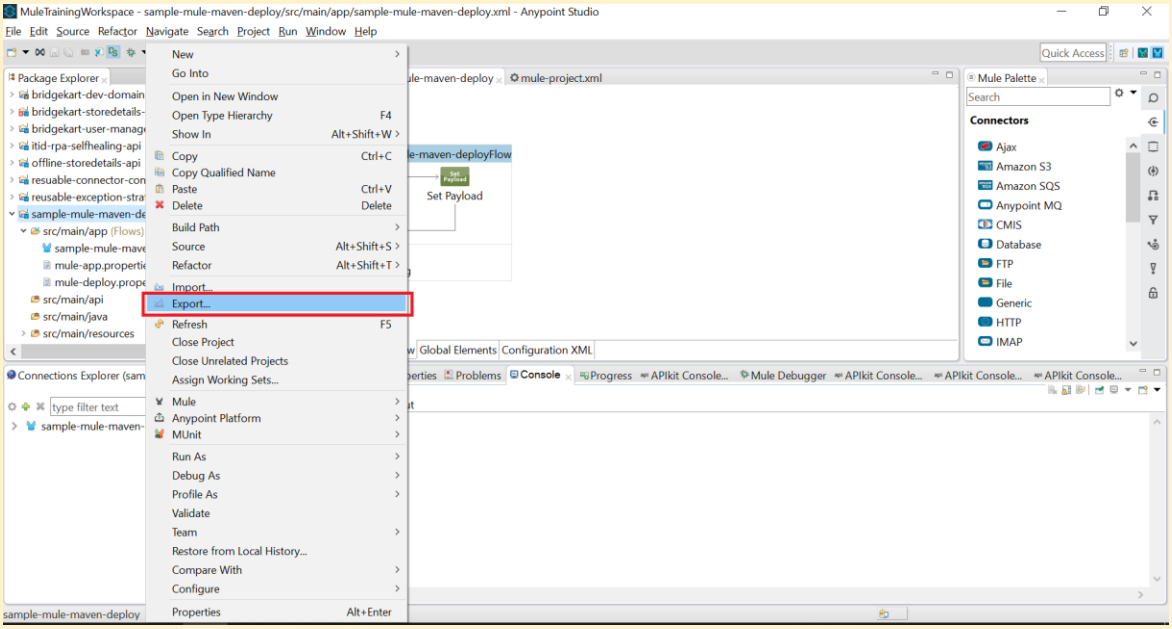


Deployment Methods

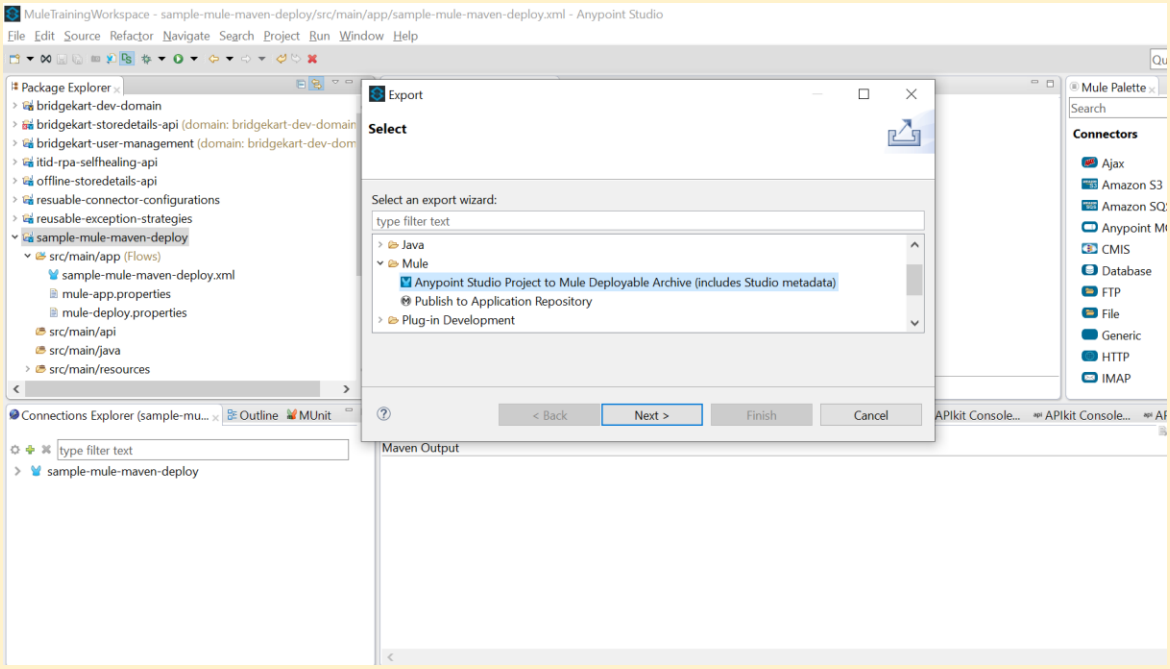
Deployment via Direct .zip or .jar Upload

CloudHub Deployment by Directly Uploading the .zip or .jar

1. Right Click on Project and Click on **Export**

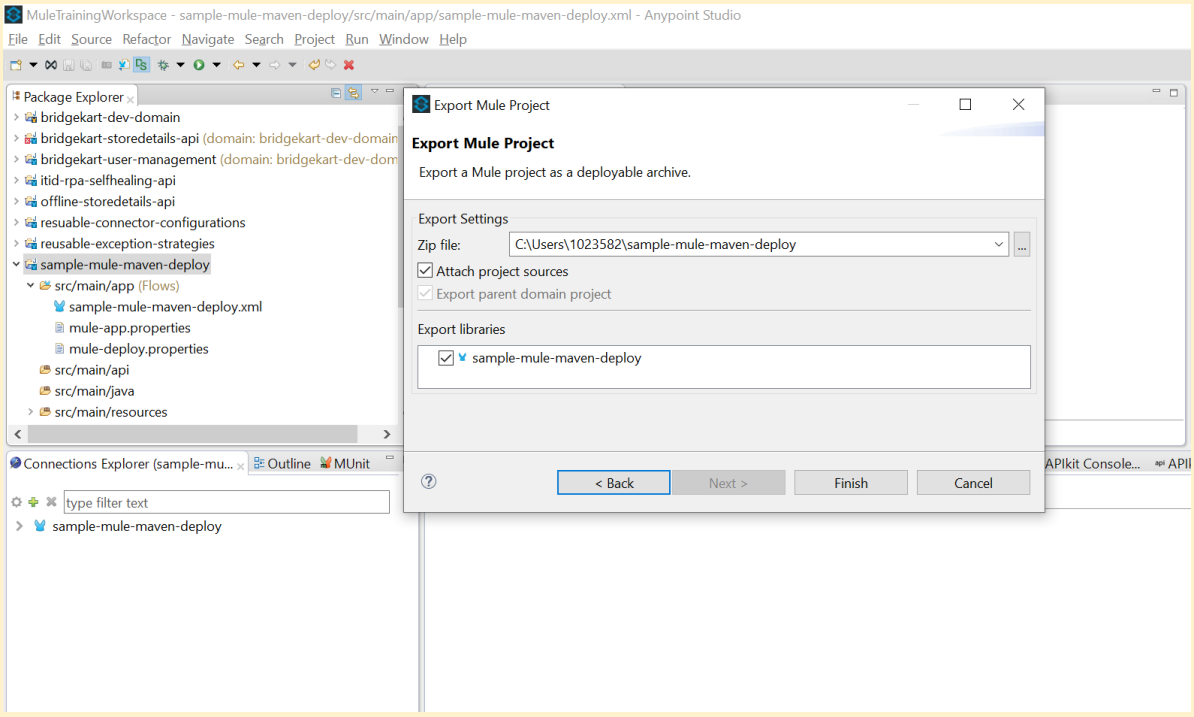


2. **Select Mule → Anypoint Studio Project to Mule Deployable Archive**

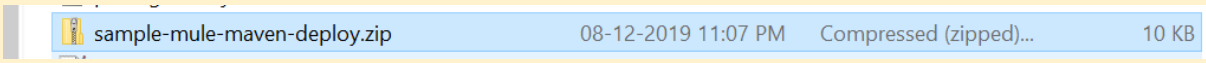


3. **Select the Location, where Project will be exported**
   Select "Attach Project Sources" an if "Export Parent domain project" is enabled, select

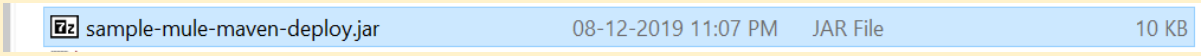
that option as well and Click on “Finish”



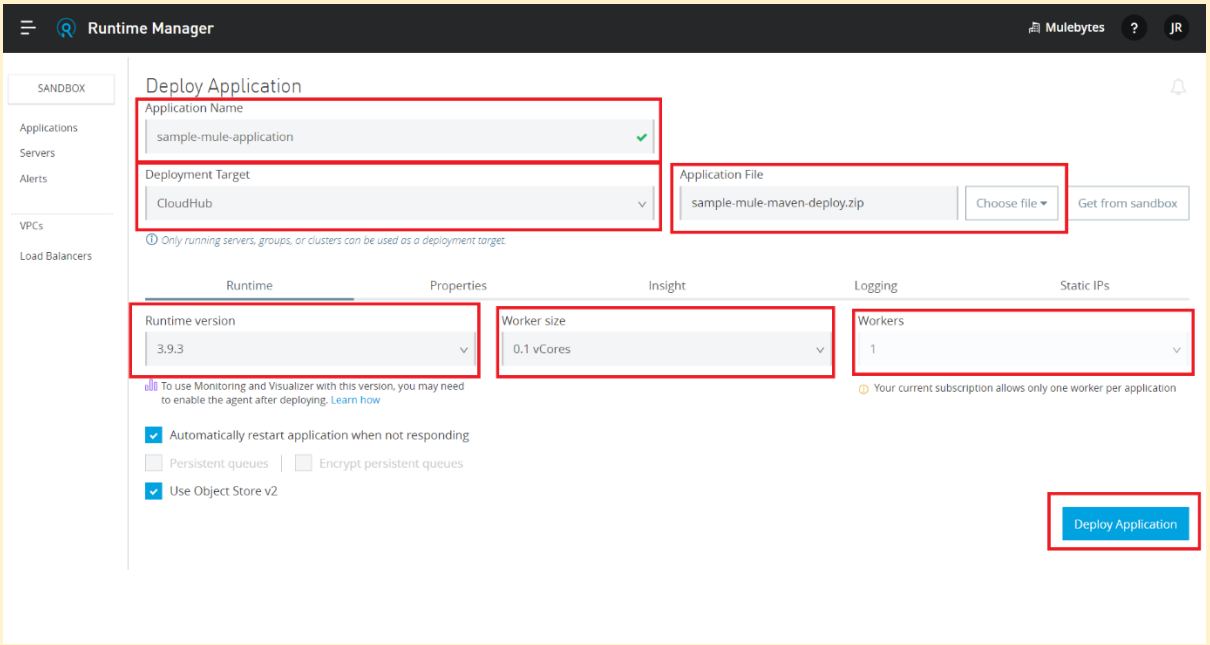
4. Project is Exported in filesystem as below:

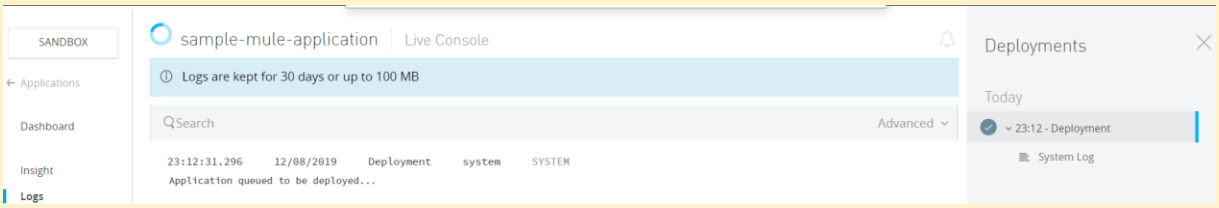


For Mule 4 .jar file will be generated:



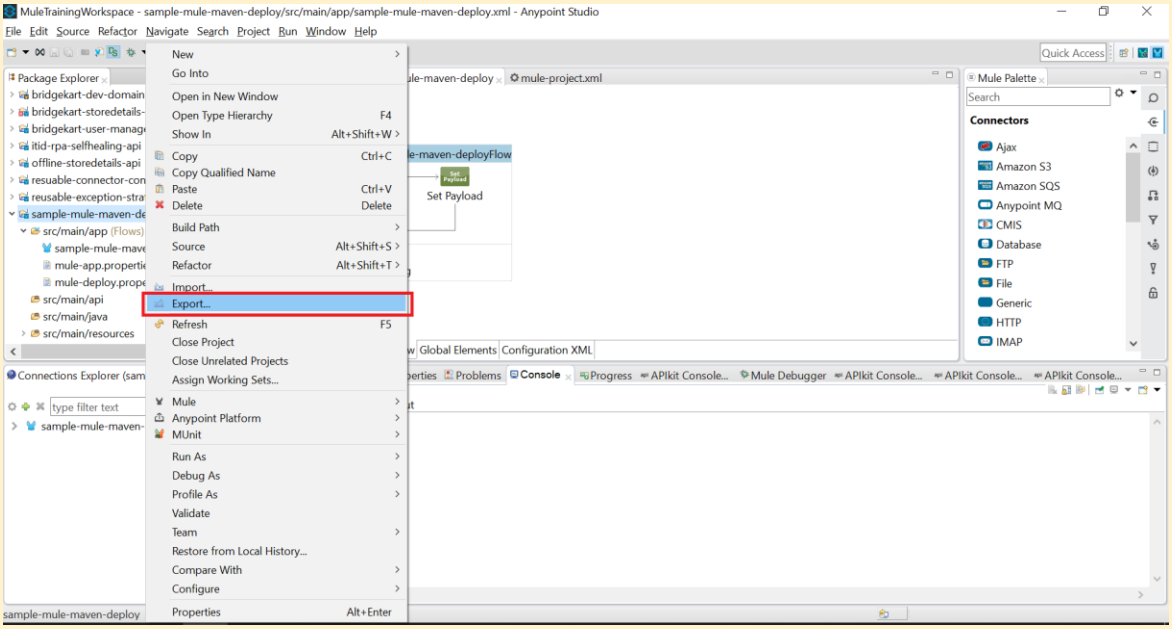
5. Goto Anypoint Platform → Runtime Manager → Deploy Application
Verify the Options highlighted below and click on “Deploy Application”



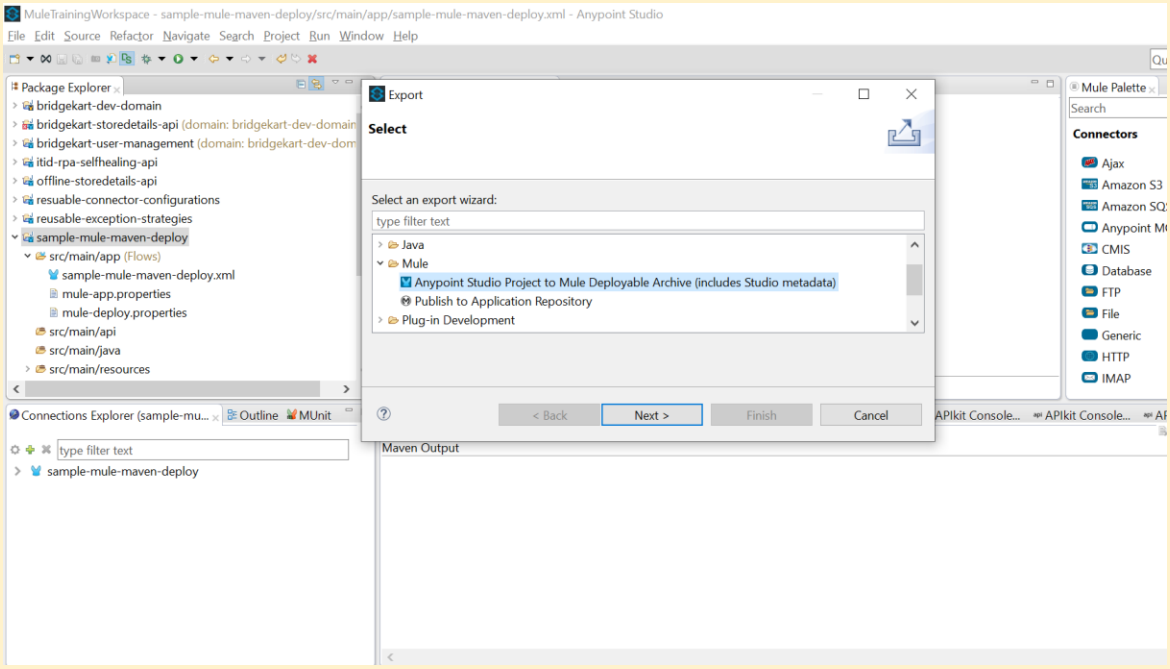


On-Premise Deployment by Directly placing the .zip or .jar

1. Right Click on Project and Click on **Export**

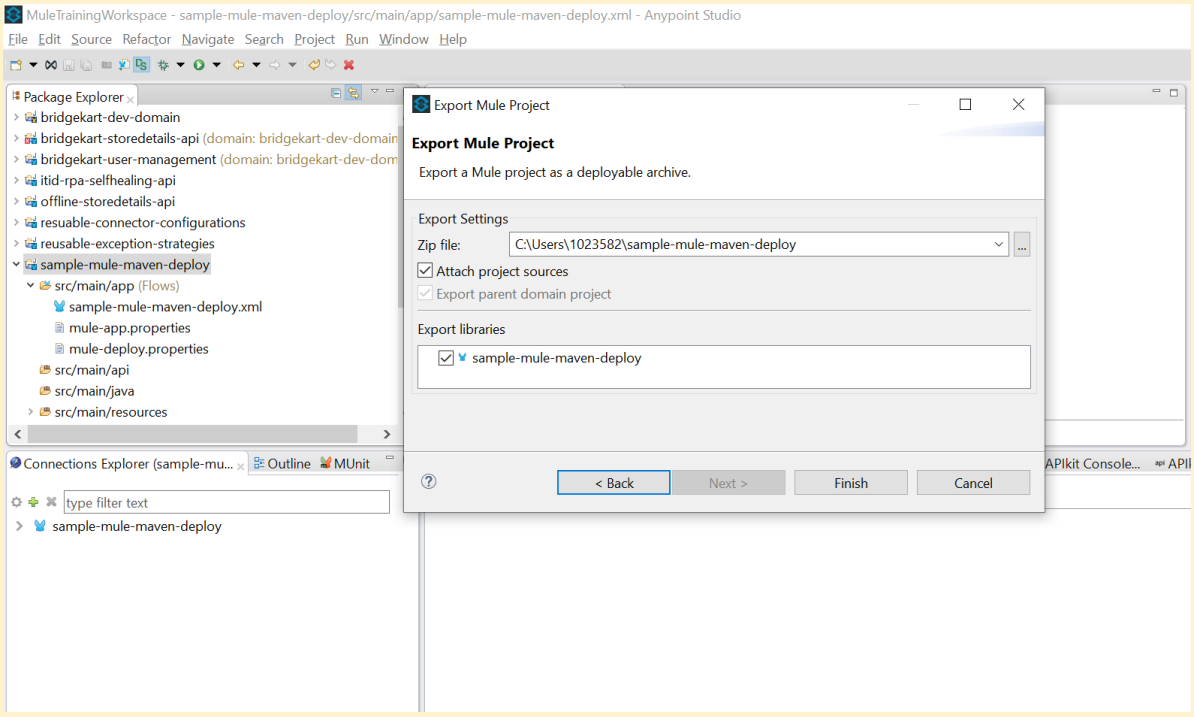


2. **Select Mule → Anypoint Studio Project to Mule Deployable Archive**

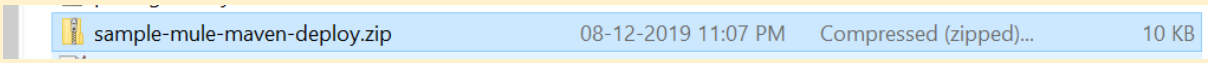


3. **Select the Location, where Project will be exported**
   Select "Attach Project Sources" an if "Export Parent domain project" is enabled, select

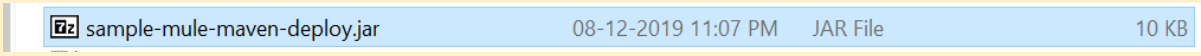
that option as well and Click on “Finish”



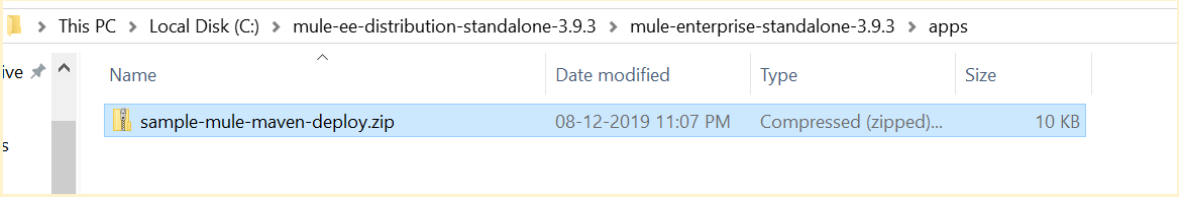
4. Project is Exported in filesystem as below:



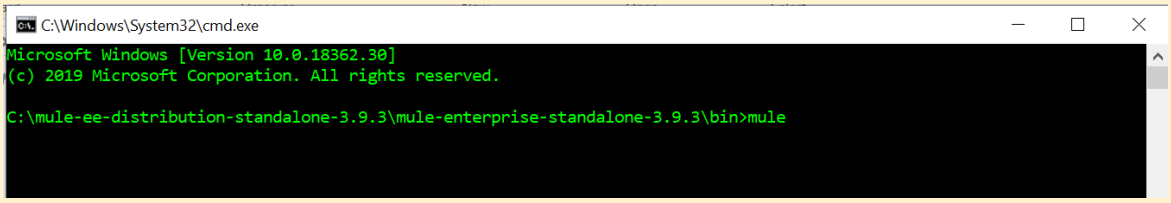
For Mule 4 .jar file will be generated:

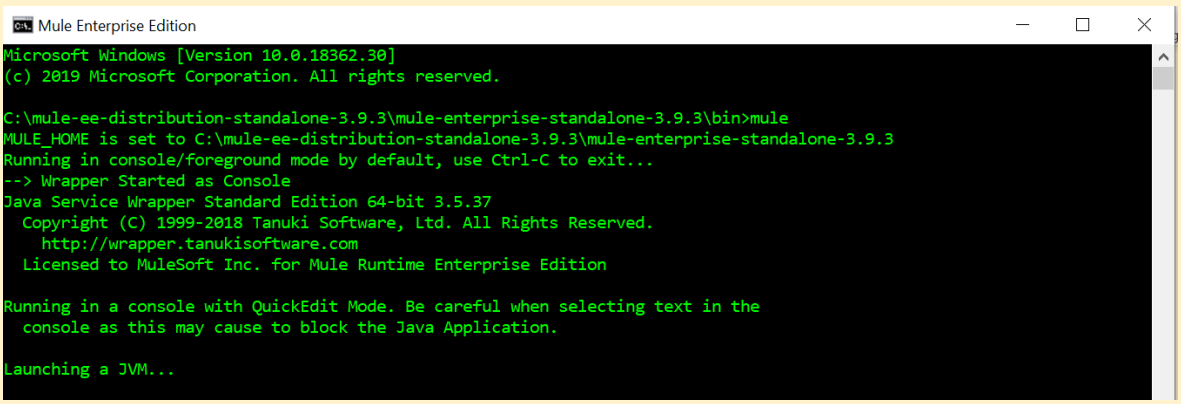


5. Goto <u>MULE_HOME\apps</u> folder, place the Generated Deployable Archive



6. Goto <u>MULE_HOME\bin</u> folder, open Command Prompt and enter `mule` command

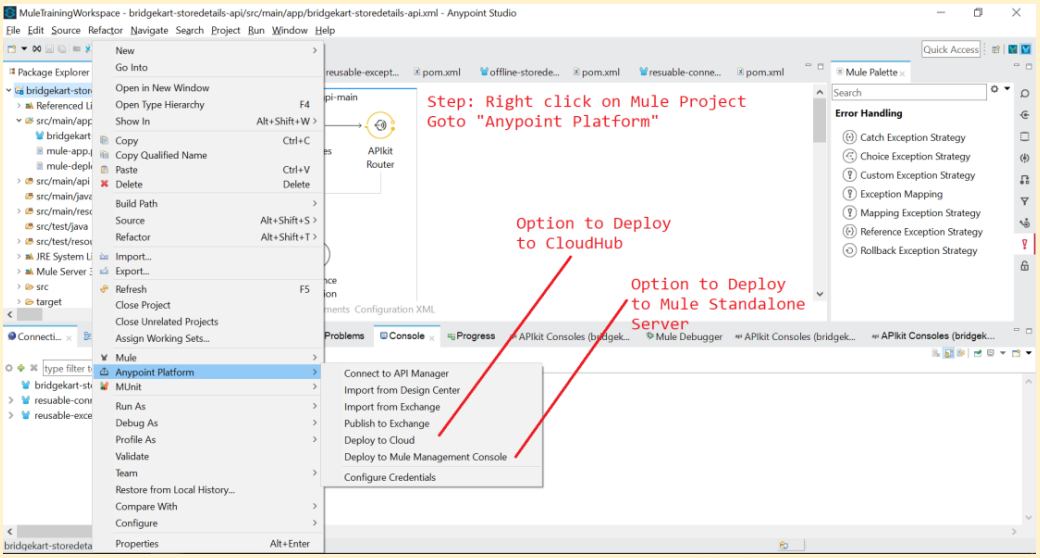




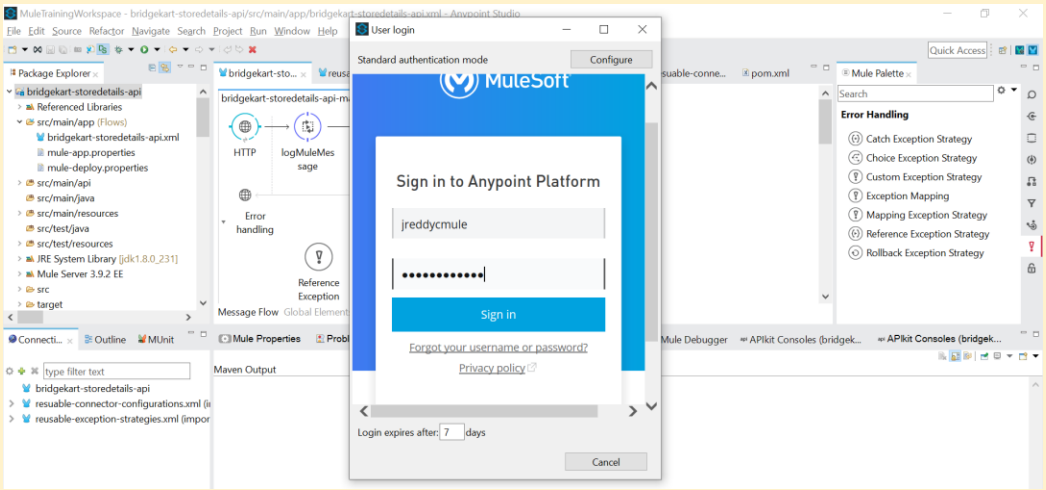
Deployment via Anypoint Studio

CloudHub Deployment from Anypoint Studio

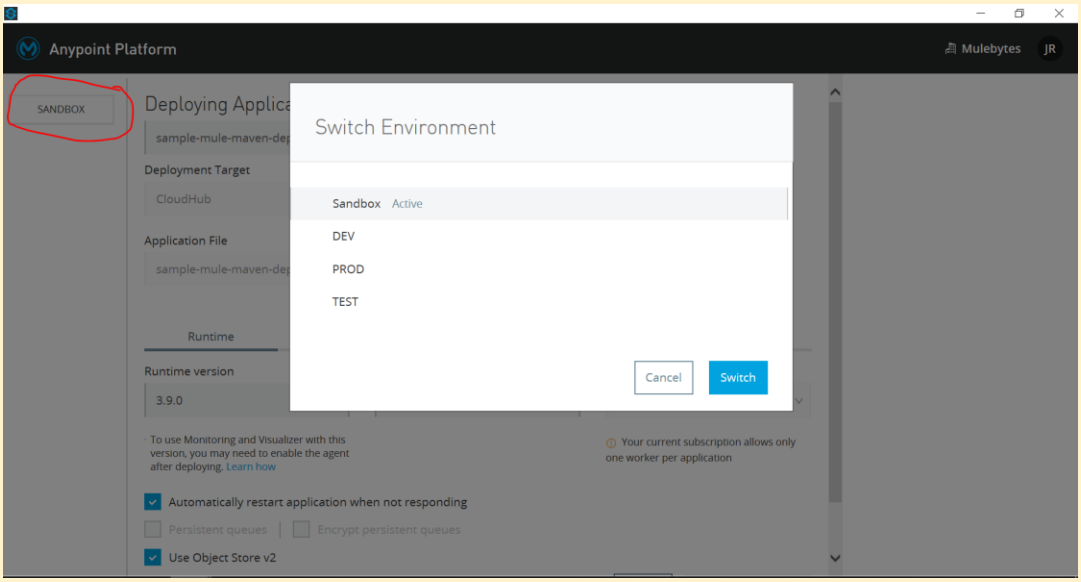
1. **Right Click on the Project → Anypoint Platform → Deploy to Cloud**



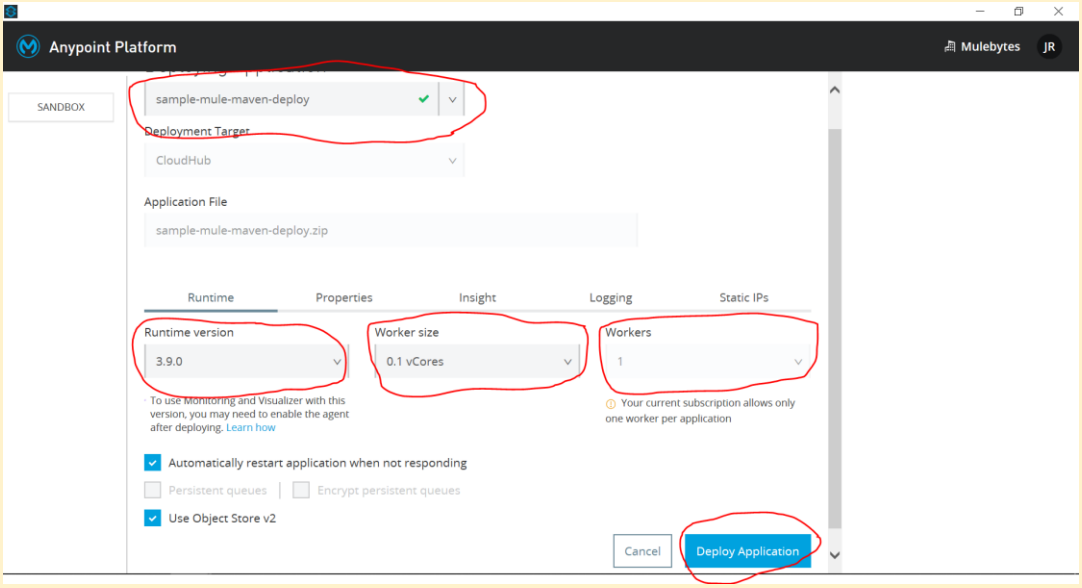
2. **Login with Anypoint Platform Credentials**



3. **Select the Environment by clicking on the Red Circle above and click on "Switch"**

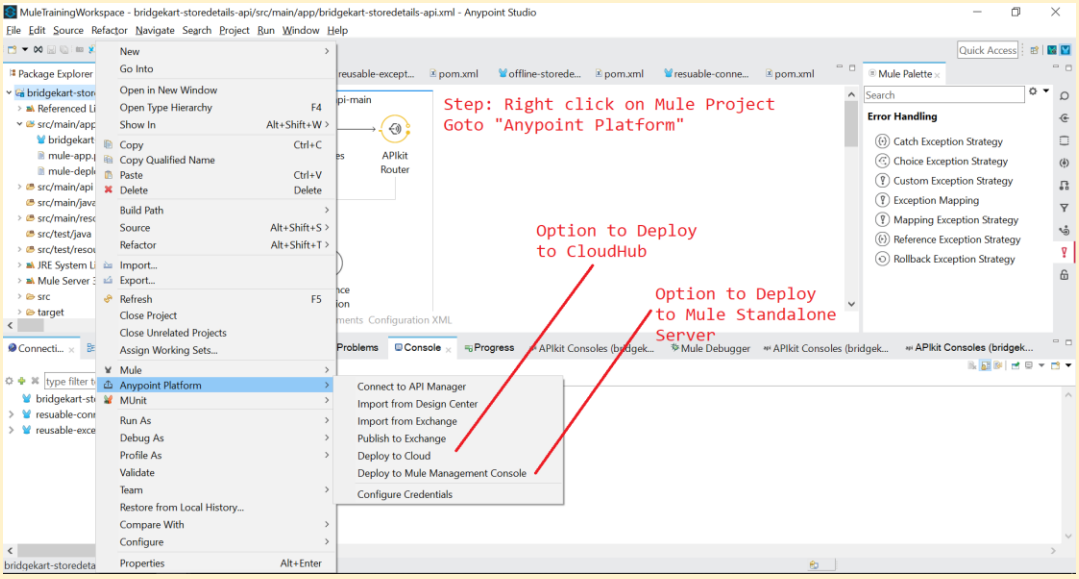


4. **Verify Highlighted configurations below and Click on “Deploy Application”**

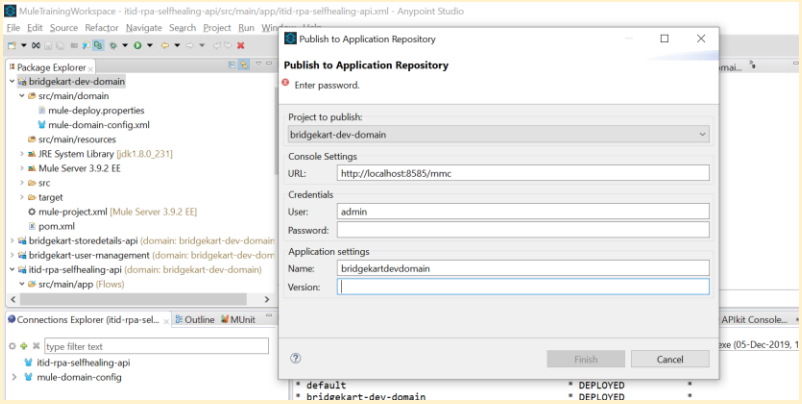


On-Premise Deployment from Anypoint Studio

1. **Right Click on the Project → Anypoint Platform → Deploy to Mule Management Console**



2. **Enter MMC URL (Provided by your project)**
   Enter Credentials (User/Password) details
   Enter Application Settings details
   Click on “Finish”



Deployment via Jenkins CI/CD

Pre-Requisites

1. Code Repository (GitHub, Bitbucket, Stash etc.)
2. Maven
3. Mule Runtime (On-Premise or Cloudhub)

CI/CD Overview

CI (Continuous Integration):

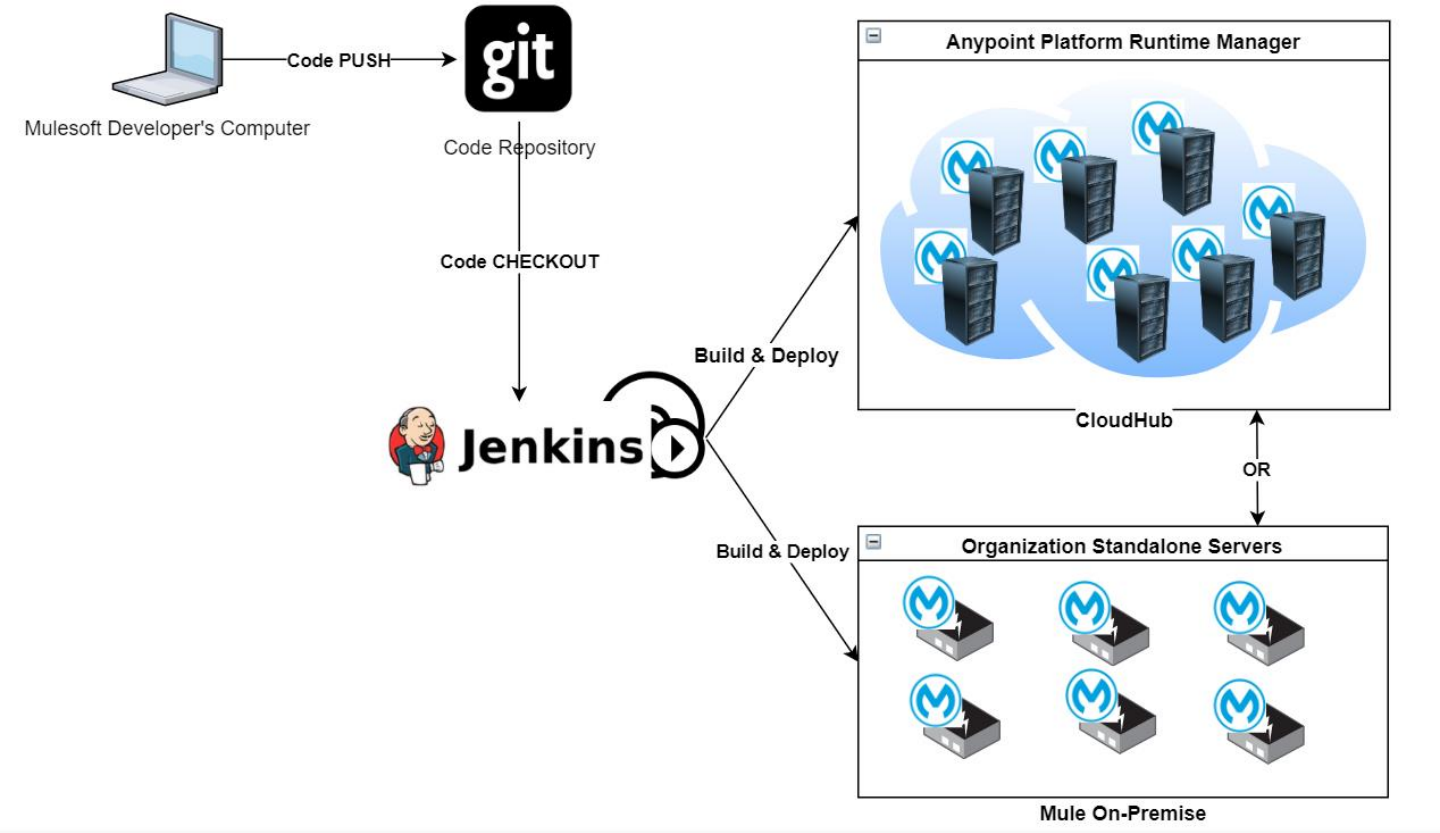
Process of building the Deployable Archive as and when Developer Check-in the code to Code repository.

CD (Continuous Deployment):

Process of Deploying the Archive Generated as part of CI process above to the desired environment as and when Developer Check-in the code to Code repository.

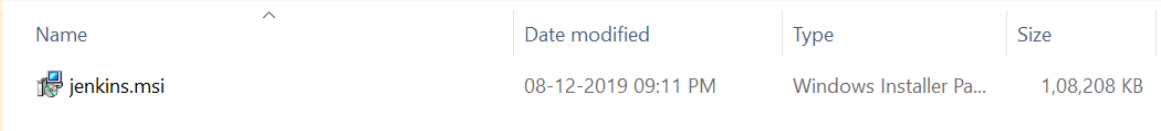
Using CI/CD Process you can automate the process of Deployment and can reduce the risk of human errors.

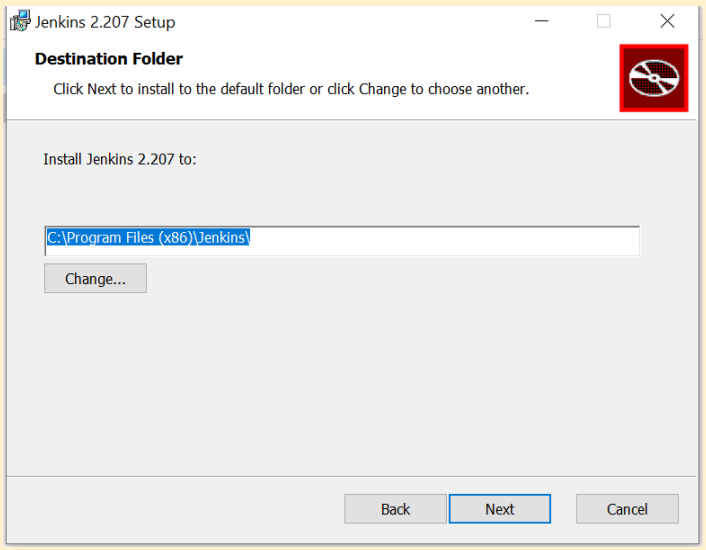
Following diagram explains, how CI/CD process works.

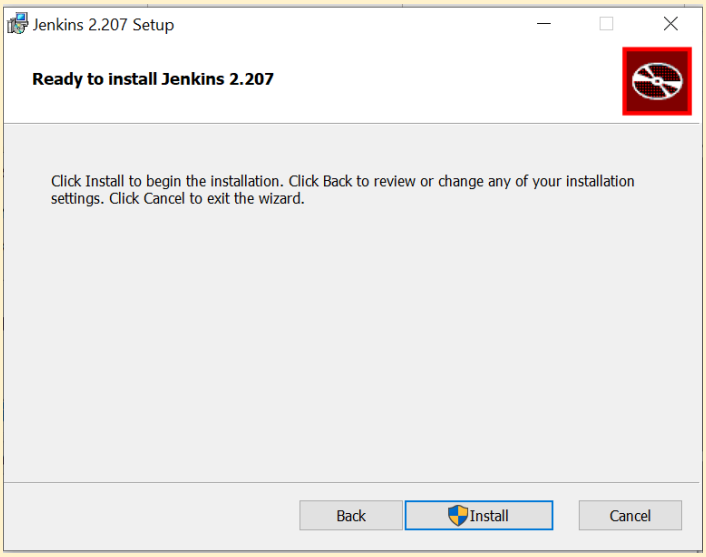


Environment Setup

1. Download Jenkins from below link and Install it:
   https://jenkins.io/download/thank-you-downloading-windows-installer/
   Extract Windows Installer from the downloaded .zip file and Double Click on below file to install it.

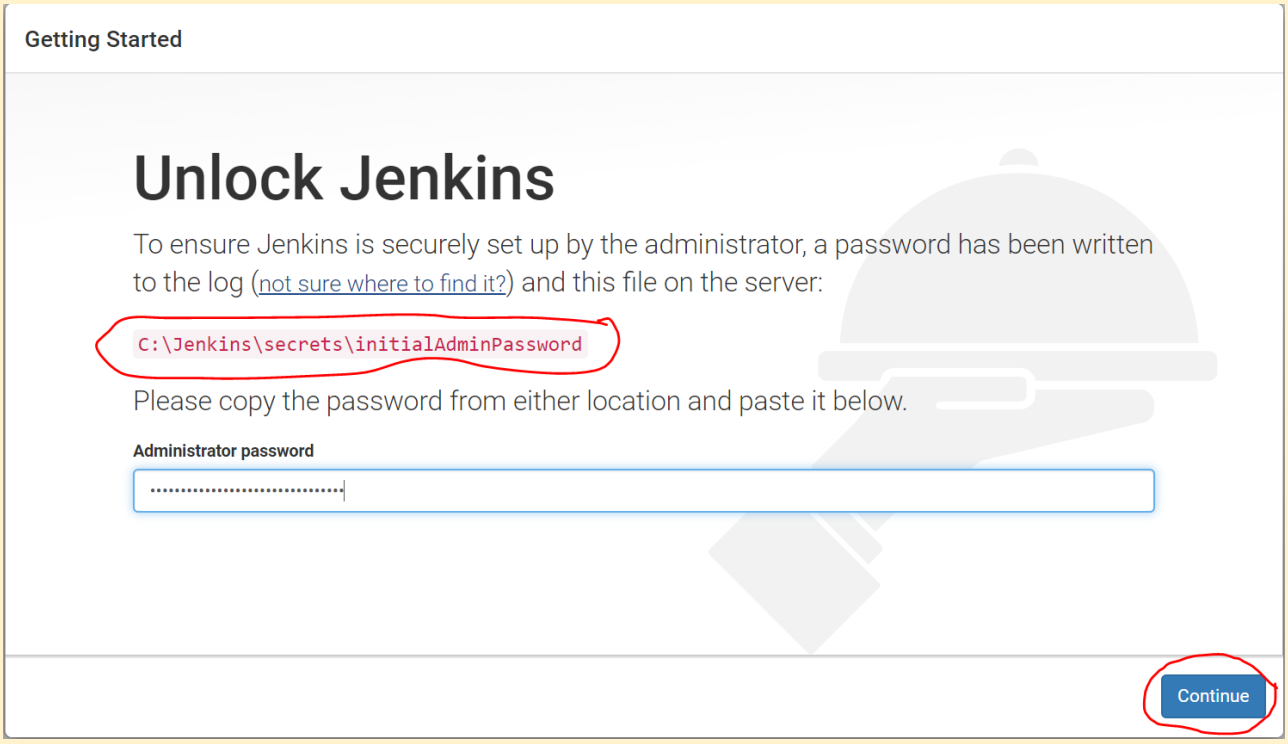




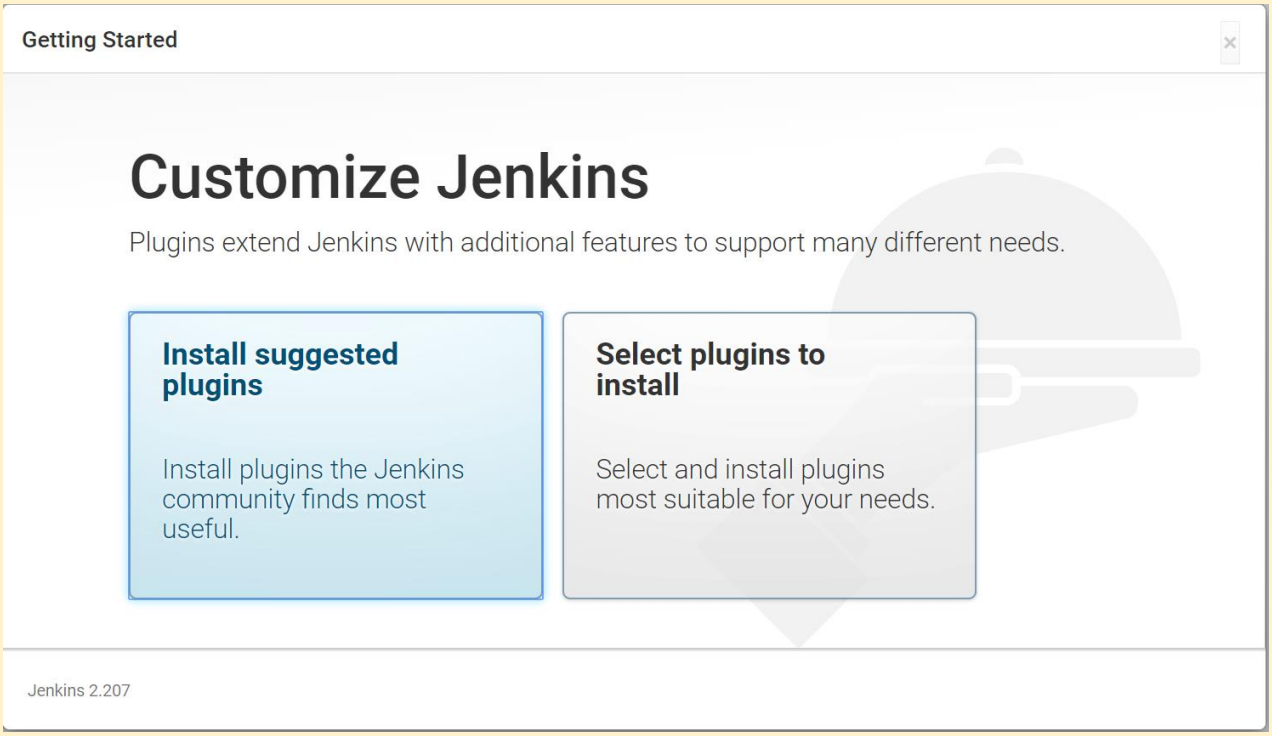


2. After Installation is complete, you can access Jenkins by using below URL: http://localhost:8080/

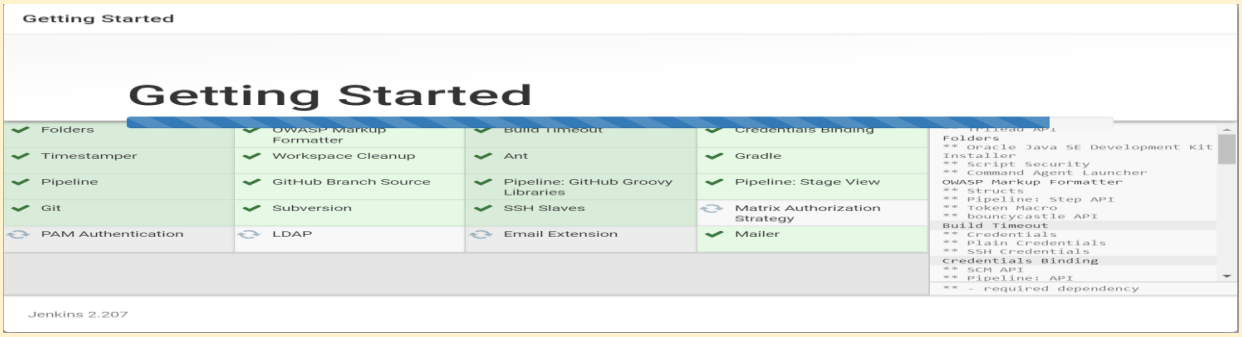
Open the file shown in below path and copy the content and paste it in below text box and click on "Continue"



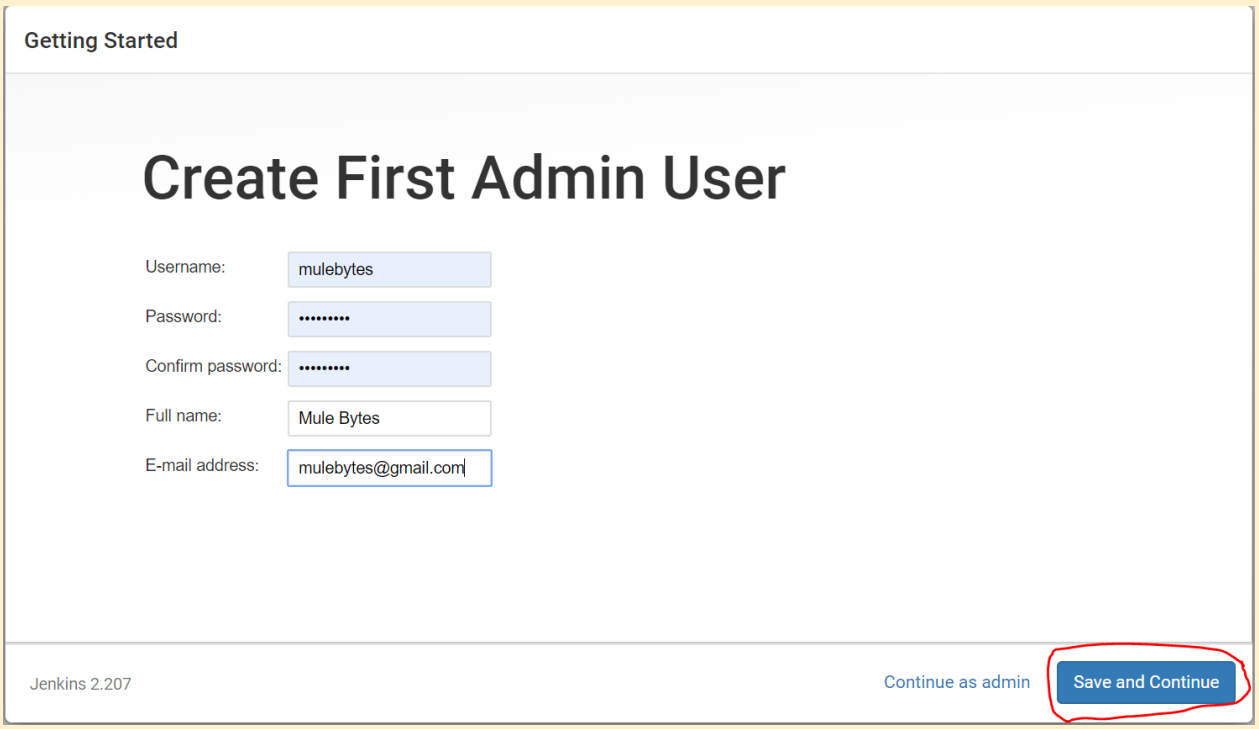
Click on "Install Suggested Plugins"

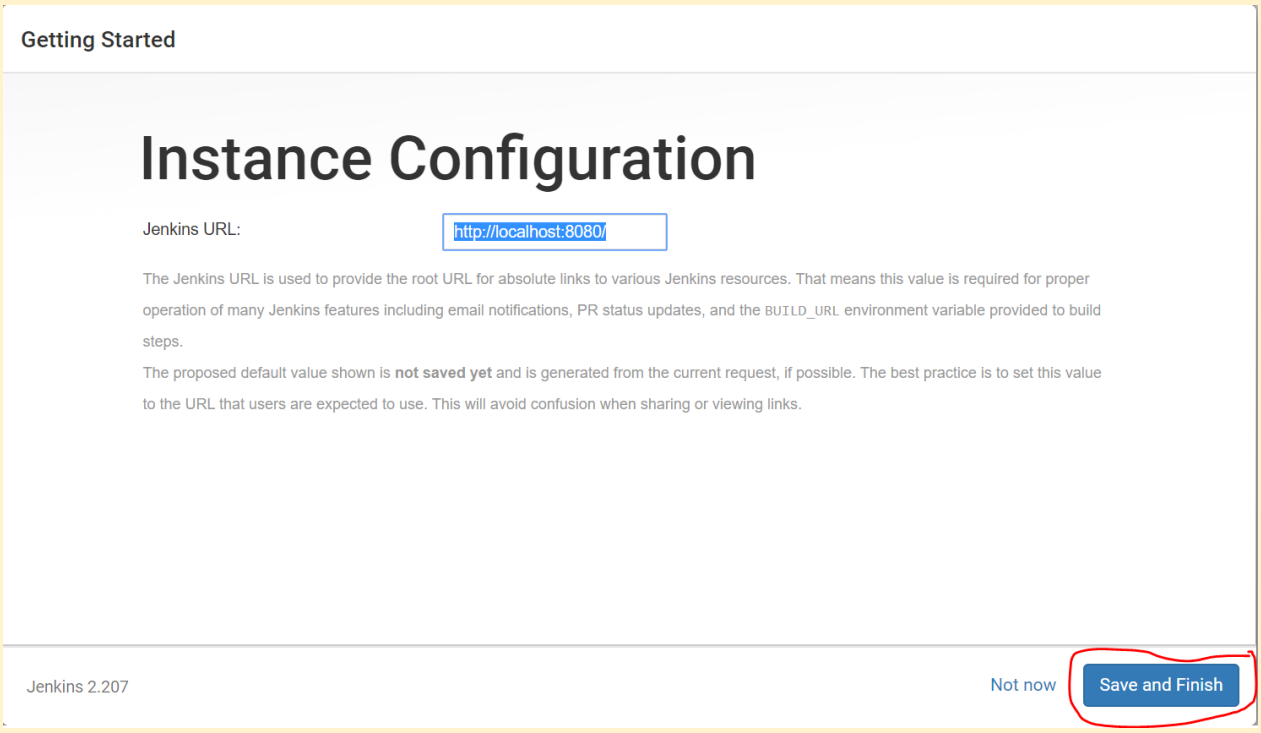


Let all the Plugins install in the next screen



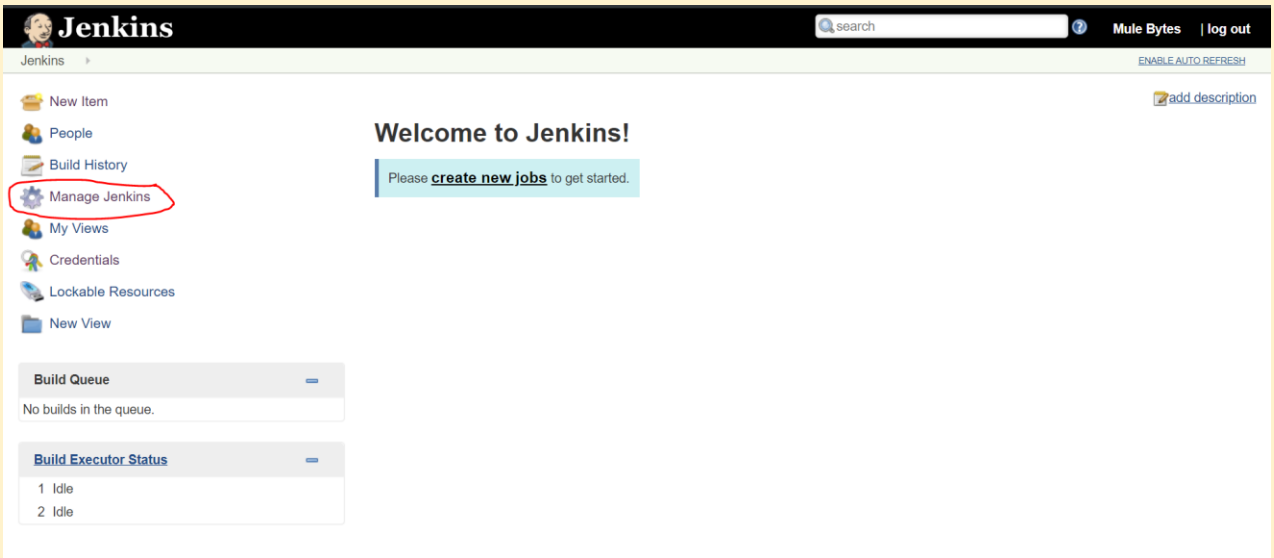
Create First Admin User and Click on “Save and Continue”



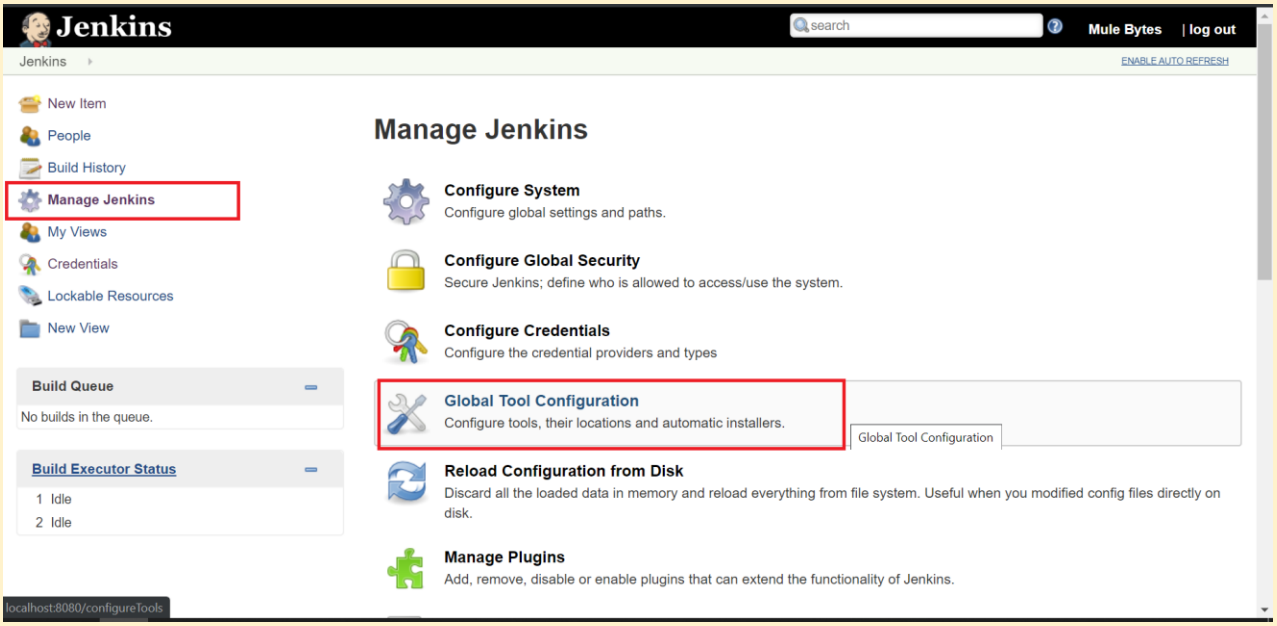


3. Configure **Maven** and **Java** installation Paths in Jenkins

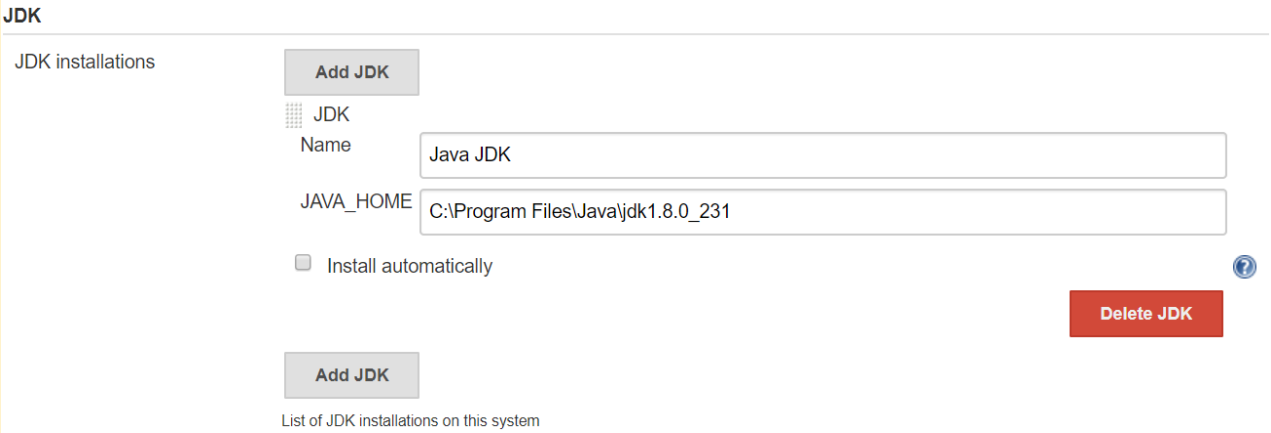
Go to "Manage Jenkins"



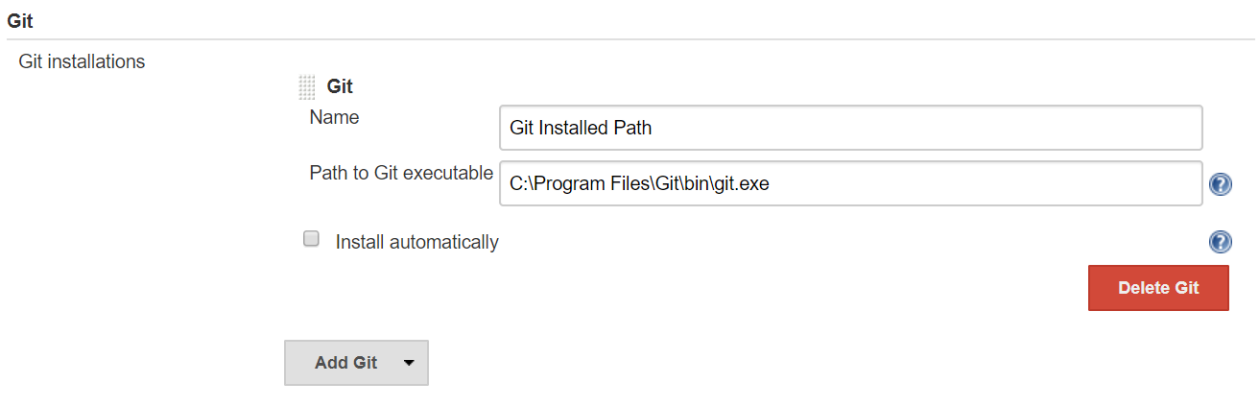
Select "Global Tool Configuration"



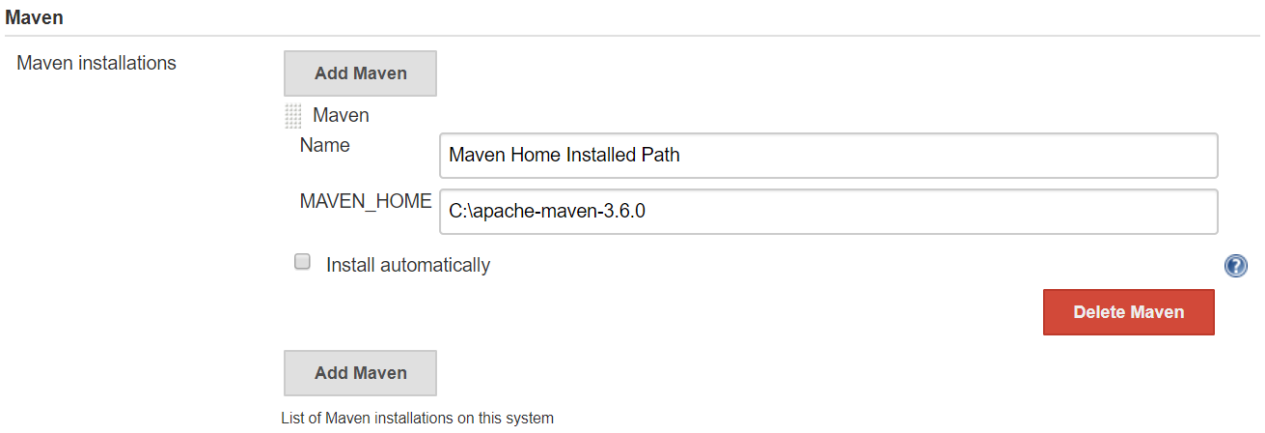
Set JDK Path



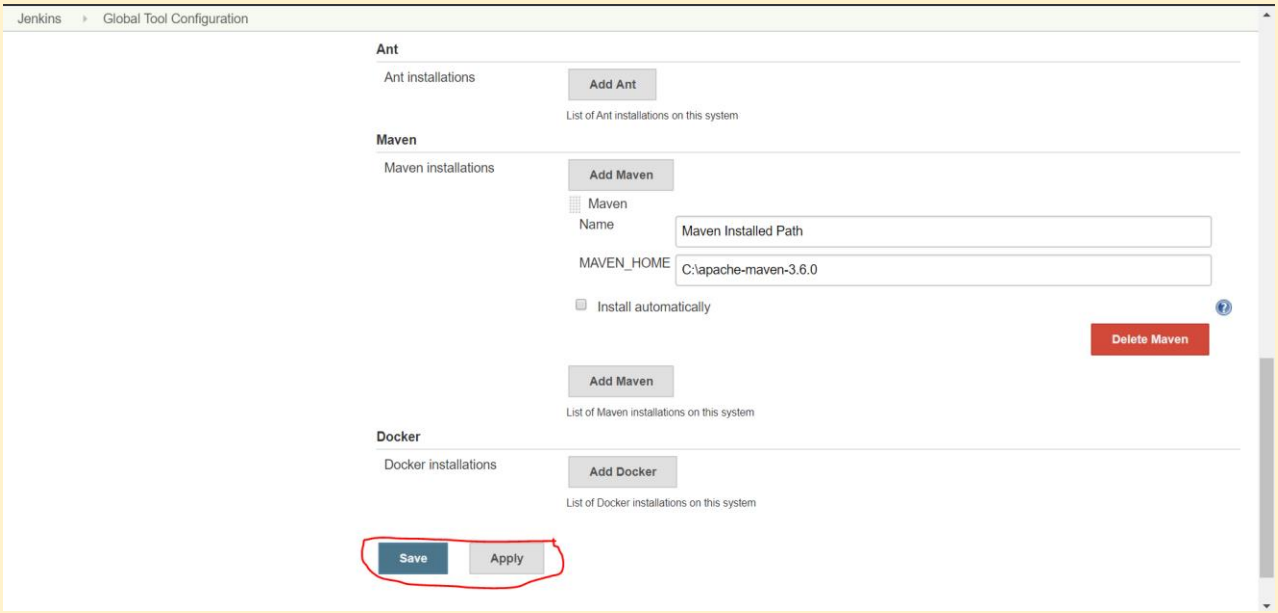
Set GIT installation Path



Set Maven Installed Path



After above configurations Click on "Save" or "Apply"

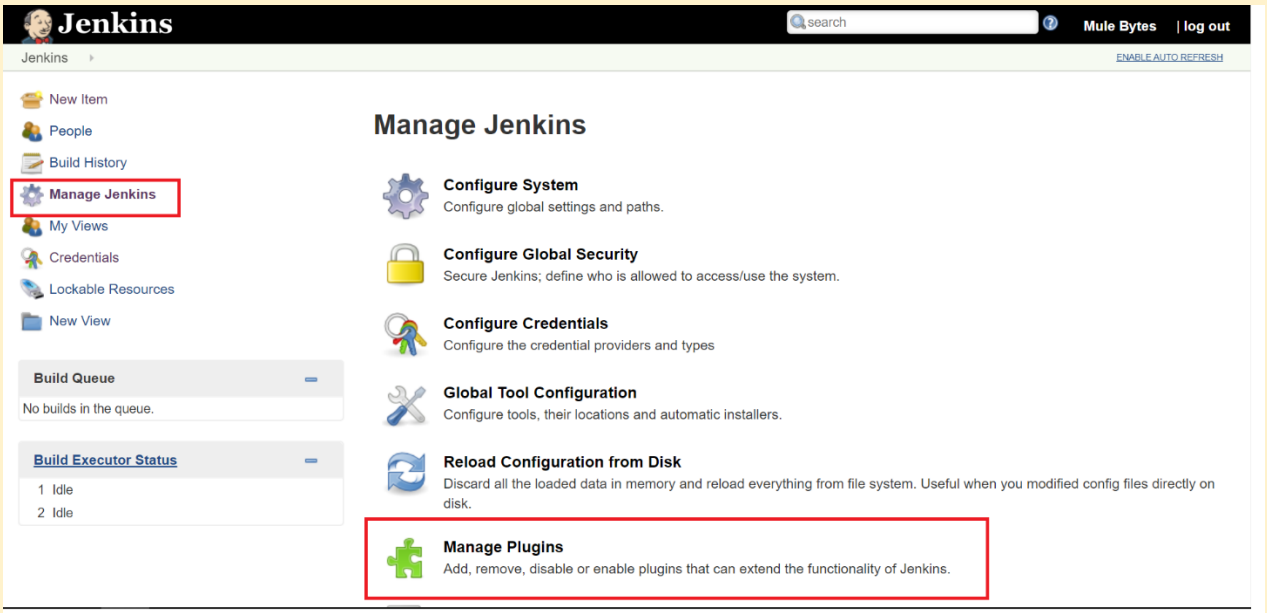


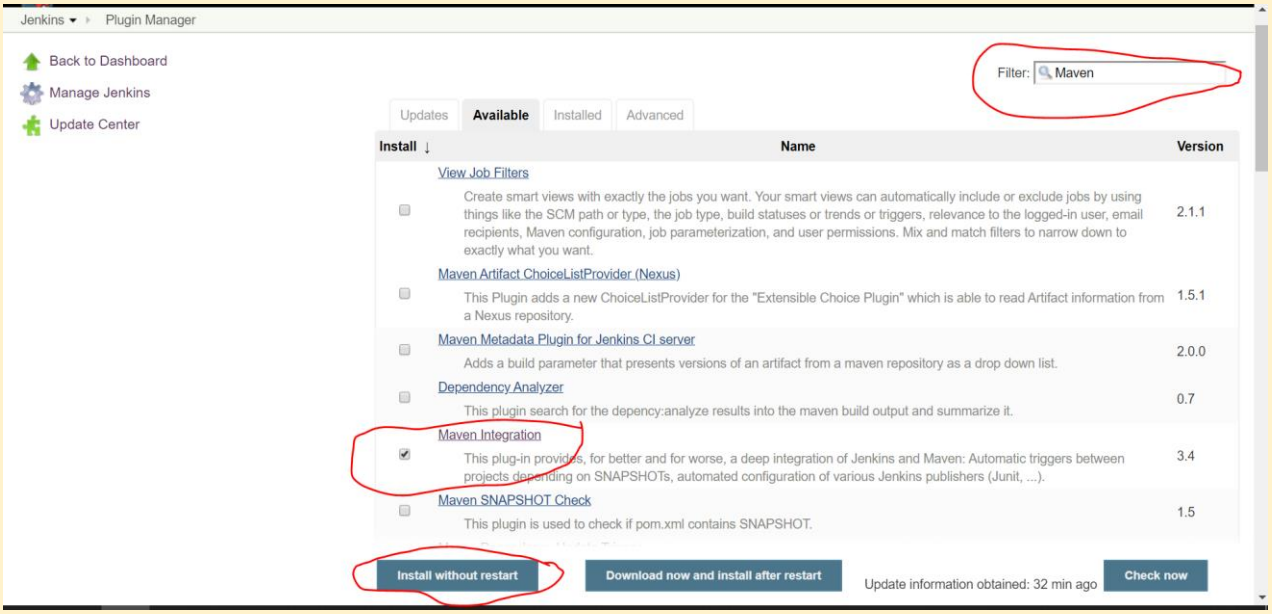
After above Setup is complete, Jenkins is now capable of Downloading the projects from Git Repository, Building the Mule Projects to generate "Mule Deployable Archive" and the same can be Deployed to CloudHub or On-Premise.

Jenkins is now Running in your Localhost and you can access it using http://localhost:8080/ URL.

4. **Install Maven Plugin**

Go to Manage Jenkins → Manage Plugins





Available Tab in Manage Plugins page.

Search for "Maven"

Select "Maven Integration"

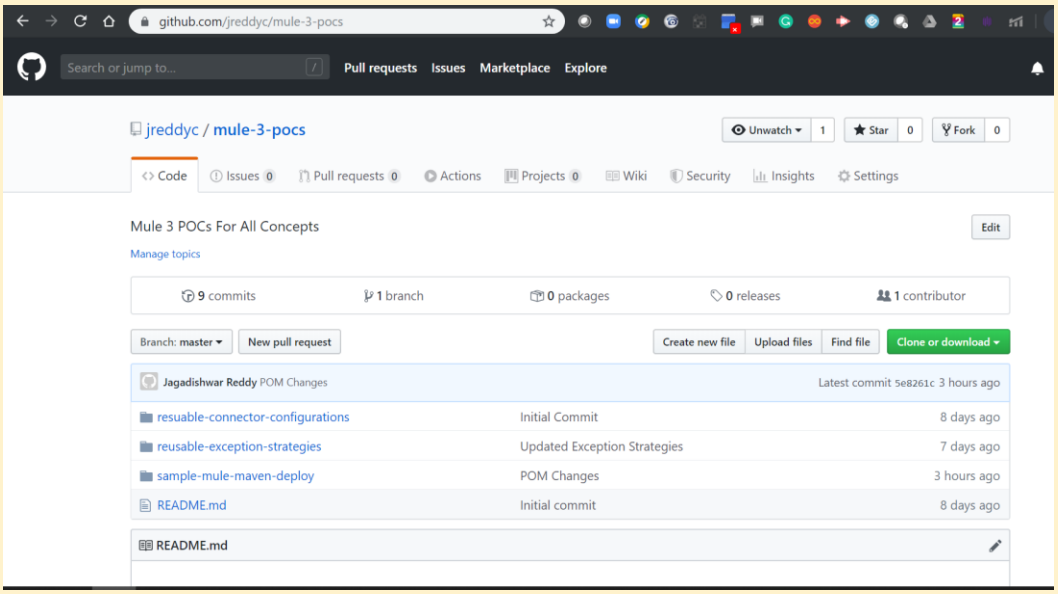
Click on "Install without restart"

After this Maven Plugin will be available in Jenkins

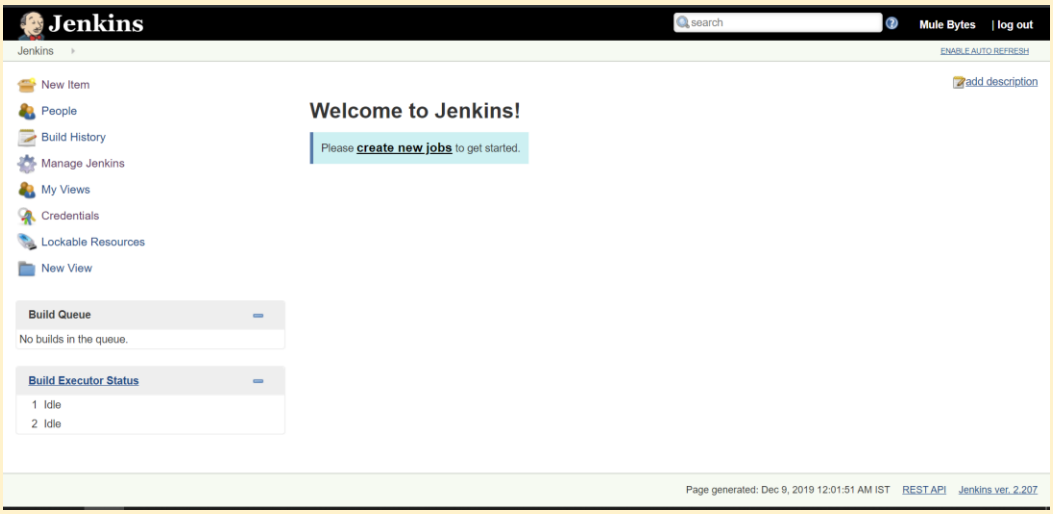
Post this Step Proceed to Create new Build Job

Create a Jenkins Build Job

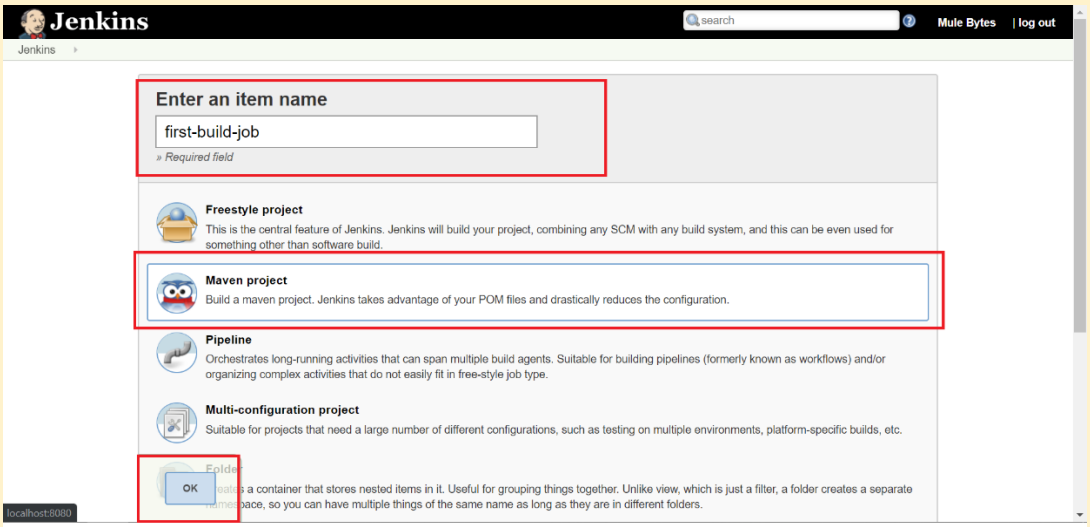
1. Make Sure Your Mulesoft Project is available in Git Repository



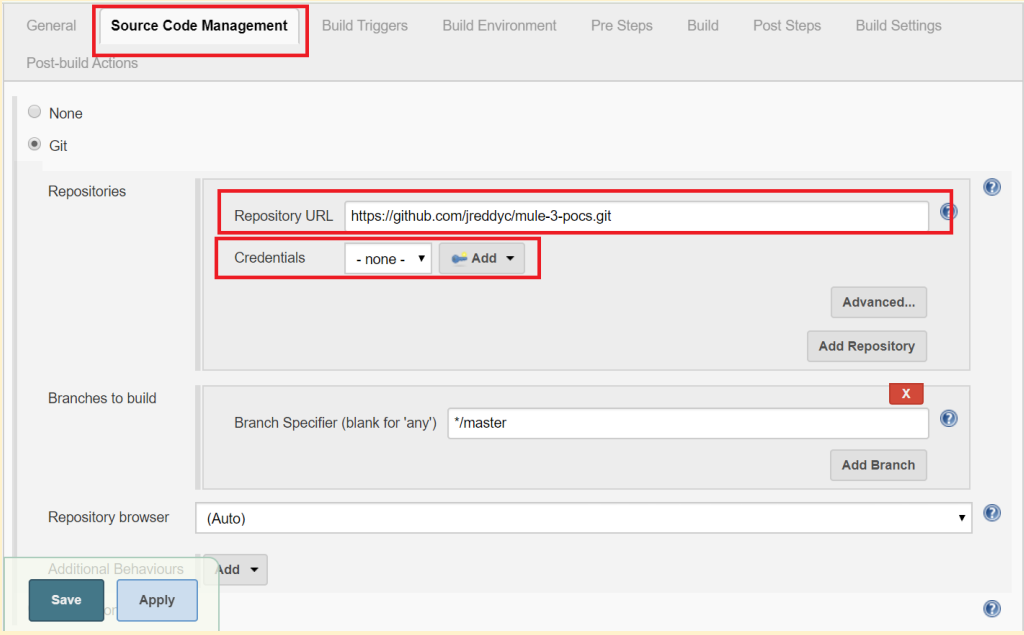
2. Click on Clone/Download Button and Copy the **.git** URL
   https://github.com/jreddyc/mule-3-pocs.git
3. Go to http://localhost:8080/ Click on **"Create New Jobs"**



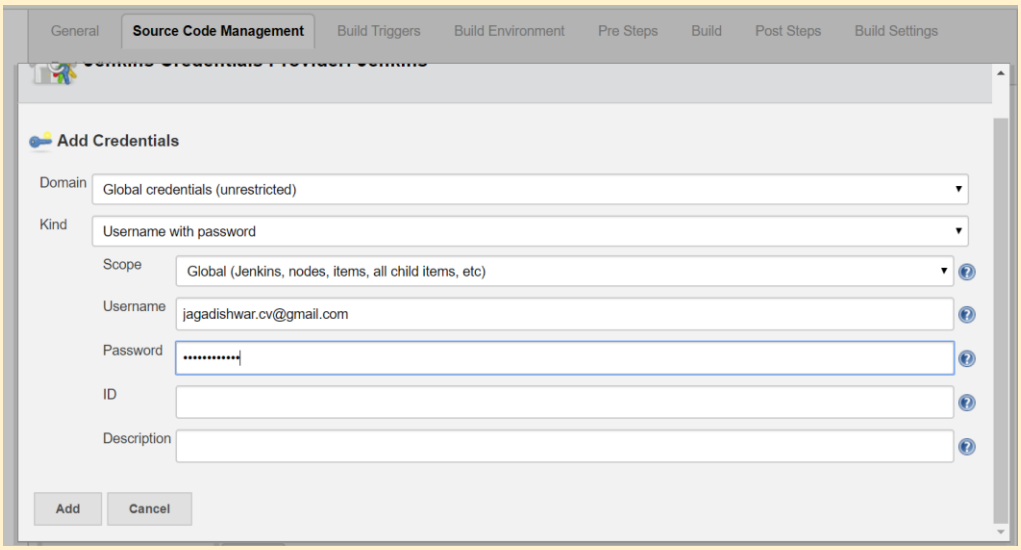
4. Enter/Select below highlighted options and click on "OK"



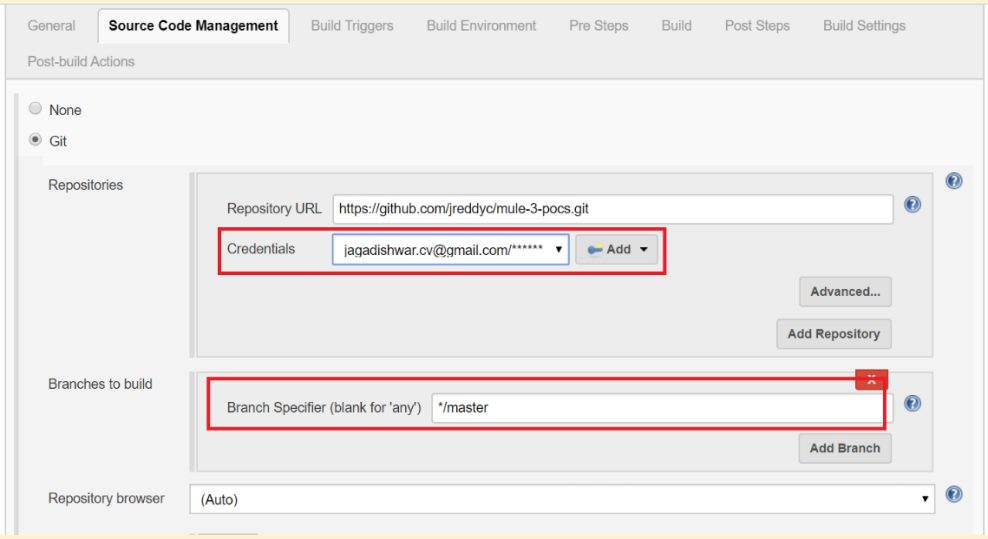
5. Configure Repository URL in “Source Code Management”
   Click on “Add” to configure Github Credentials



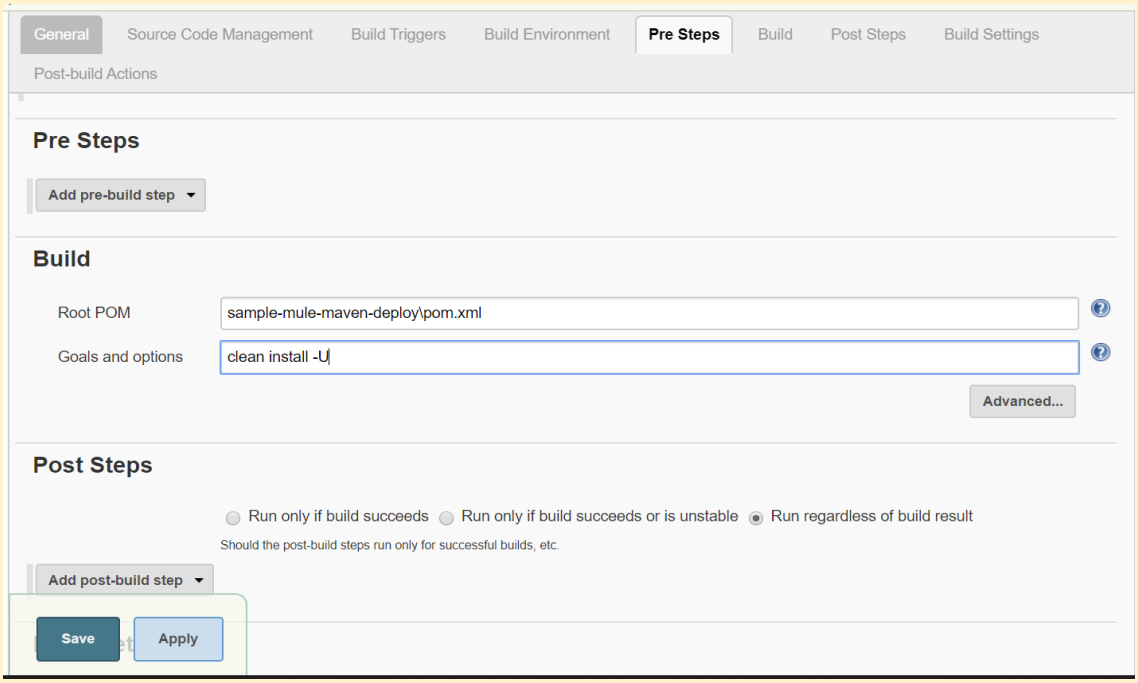
Configure Credentials as below and Click on “Add”



Select the Configured Credentials in Above Step

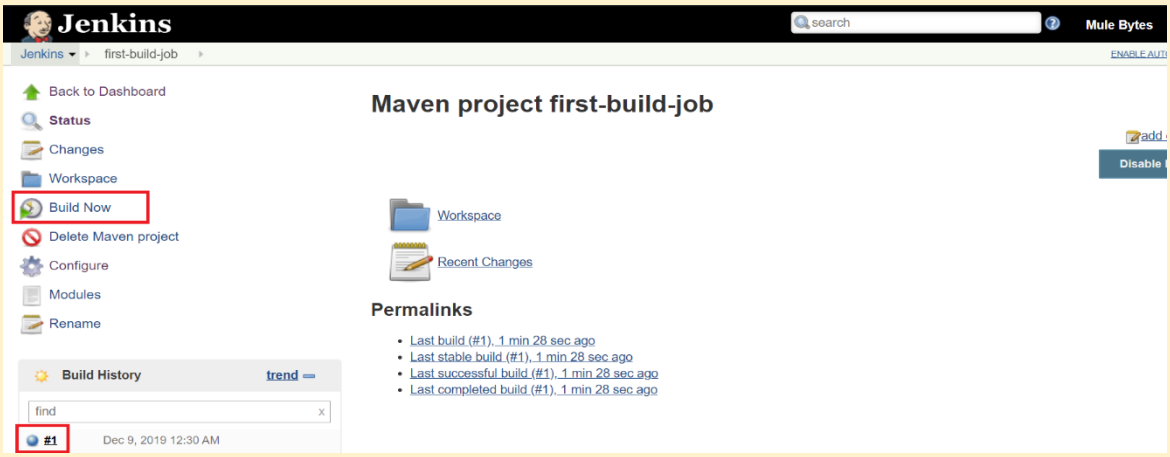


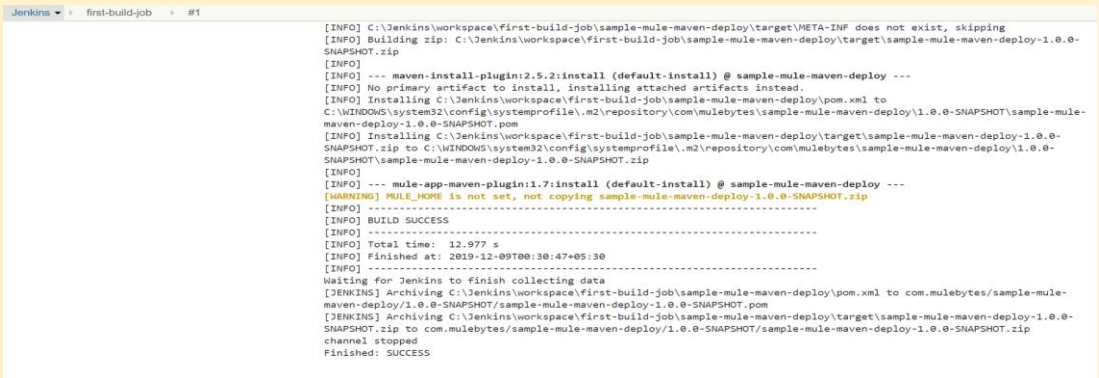
6. Select the POM file location project → Configure “Goals and options” with Maven Command and Click on “Save”



7. Go to Jenkins Home Page and Select the New Job Created and Configured

8. Click on “Build Now” post which it will display the Build Number





Create a Jenkins Deploy Job

1. Add Below Plugin in Project **POM.xml**
   under section → **build/plugins/plugin**

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-deploy-plugin</artifactId>
    <version>2.8.2</version>
    <configuration>
        <skip>true</skip>
    </configuration>
</plugin>
```

2. Create Deployment Profiles in **POM.xml** (Please find below sample file content)
   Please refer **highlighted** info in below sample POM file.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-
v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mulebytes</groupId>
  <artifactId>sample-mule-maven-deploy</artifactId>
  <version>1.0.0</version>
  <packaging>mule</packaging>
  <name>Mule sample-mule-maven-deploy Application</name>
  <properties>
    <mule.tools.version>1.7</mule.tools.version>
    <!-- <mule.home>${MULE_HOME}</mule.home> -->
    <mule.version>3.9.0</mule.version>
    <anypoint.username>mulebytesme</anypoint.username>
    <anypoint.password>Jaga#mule92#</anypoint.password>
    <cloundhub.env>Sandbox</cloundhub.env>
    <anypoint.uri>https://anypoint.mulesoft.com</anypoint.uri>
    <anypoint.businessGroup>MuleBytes</anypoint.businessGroup>
    <cloudhub.workerType>MICRO</cloudhub.workerType>
    <cloudhub.workers>1</cloudhub.workers>
    <maven.build.timestamp.format>yyMMddHHmmss</maven.build.timestamp.format>
  </properties>
  <build>
```

```
<plugins>
  <plugin>
    <groupId>org.mule.tools.maven</groupId>
    <artifactId>mule-app-maven-plugin</artifactId>
    <version>${mule.tools.version}</version>
    <extensions>true</extensions>
    <configuration>
      <copyToAppsDirectory>false</copyToAppsDirectory>
    </configuration>
  </plugin>
  <plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-deploy-plugin</artifactId>
    <version>2.8.2</version>
    <configuration>
      <skip>true</skip>
    </configuration>
  </plugin>
  <plugin>
    <groupId>org.codehaus.mojo</groupId>
    <artifactId>build-helper-maven-plugin</artifactId>
    <version>1.7</version>
    <executions>
      <execution>
        <id>add-resource</id>
        <phase>generate-resources</phase>
        <goals>
          <goal>add-resource</goal>
        </goals>
        <configuration>
          <resources>
            <resource>
              <directory>src/main/app/</directory>
            </resource>
```

```
                <resource>
                  <directory>src/main/api/</directory>
                </resource>
                <resource>
                  <directory>mappings/</directory>
                </resource>
              </resources>
            </configuration>
          </execution>
        </executions>
      </plugin>
    </plugins>
  </build>
  <profiles>
    <profile>
      <id>standalone</id>
      <build>
        <plugins>
          <plugin>
            <groupId>org.mule.tools.maven</groupId>
            <artifactId>mule-maven-plugin</artifactId>
            <version>2.3.2</version>
            <configuration>
              <standaloneDeployment>
                <muleVersion>${mule.version}</muleVersion>
                <muleHome>${mule.home}</muleHome>
                <applicationName>${artifactId}</applicationName>
              </standaloneDeployment>
            </configuration>
            <executions>
              <execution>
                <id>deploy</id>
                <phase>deploy</phase>
                <goals>
```

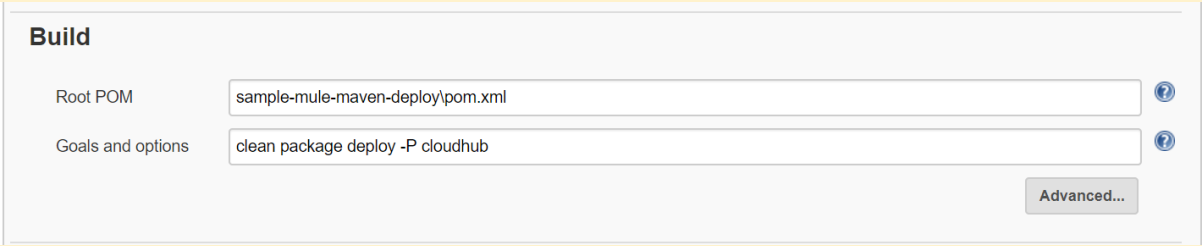
```
                <goal>deploy</goal>
              </goals>
            </execution>
          </executions>
        </plugin>
      </plugins>
    </build>
  </profile>
  <profile>
    <id>cloudhub</id>
    <build>
      <plugins>
        <plugin>
          <groupId>org.mule.tools.maven</groupId>
          <artifactId>mule-maven-plugin</artifactId>
          <version>2.3.2</version>
          <configuration>
            <cloudHubDeployment>
              <uri>${anypoint.uri}</uri>
              <muleVersion>${mule.version}</muleVersion>
              <username>${anypoint.username}</username>
              <password>${anypoint.password}</password>
              <applicationName>${artifactId}</applicationName>
              <environment>${cloundhub.env}</environment>
            </cloudHubDeployment>
          </configuration>
          <executions>
            <execution>
              <id>deploy</id>
              <phase>deploy</phase>
              <goals>
                <goal>deploy</goal>
              </goals>
            </execution>
```

```
            </executions>
          </plugin>
        </plugins>
      </build>
    </profile>
  </profiles>
  <!-- Mule Dependencies -->
  <dependencies>
    <!-- Xml configuration -->
    <dependency>
      <groupId>org.mule.modules</groupId>
      <artifactId>mule-module-spring-config</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
    <!-- Mule Transports -->
    <dependency>
      <groupId>org.mule.transports</groupId>
      <artifactId>mule-transport-file</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>org.mule.transports</groupId>
      <artifactId>mule-transport-http</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>org.mule.transports</groupId>
      <artifactId>mule-transport-jdbc</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
```

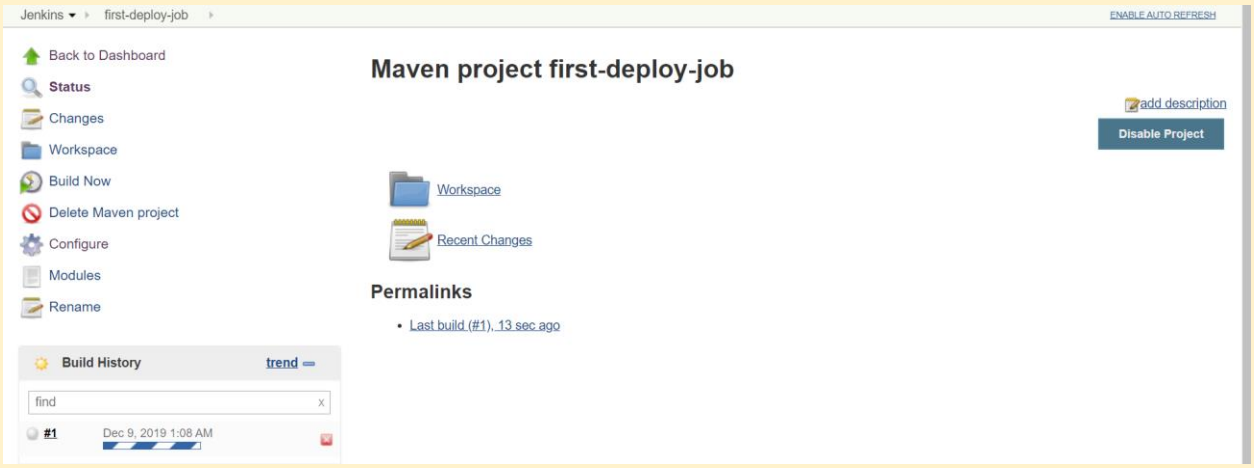
```
    <dependency>
      <groupId>org.mule.transports</groupId>
      <artifactId>mule-transport-jms</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>org.mule.transports</groupId>
      <artifactId>mule-transport-vm</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
    <!-- Mule Modules -->
    <dependency>
      <groupId>org.mule.modules</groupId>
      <artifactId>mule-module-scripting</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>org.mule.modules</groupId>
      <artifactId>mule-module-xml</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
    <!-- for testing -->
    <dependency>
      <groupId>org.mule.tests</groupId>
      <artifactId>mule-tests-functional</artifactId>
      <version>${mule.version}</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.mule.modules</groupId>
```

```
      <artifactId>mule-module-apikit</artifactId>
      <version>${mule.version}</version>
      <scope>provided</scope>
    </dependency>
  </dependencies>
  <repositories>
    <repository>
      <id>Central</id>
      <name>Central</name>
      <url>http://repo1.maven.org/maven2/</url>
      <layout>default</layout>
    </repository>
    <repository>
      <id>mulesoft-releases</id>
      <name>MuleSoft Releases Repository</name>
      <url>http://repository.mulesoft.org/releases/</url>
      <layout>default</layout>
    </repository>
  </repositories>
  <pluginRepositories>
    <pluginRepository>
      <id>mulesoft-release</id>
      <name>mulesoft release repository</name>
      <layout>default</layout>
      <url>http://repository.mulesoft.org/releases/</url>
      <snapshots>
        <enabled>false</enabled>
      </snapshots>
    </pluginRepository>
  </pluginRepositories>
</project>
```

3. Repeat the Steps mentioned in [Build Job]() but during Pre-Steps Build change like below:



4. Click on “Build Now”



5. Verify if Mule Application is deployed to CloudHub → Runtime Manager