**Task on terraform 3-4**

1) Watch terraform-03 video.

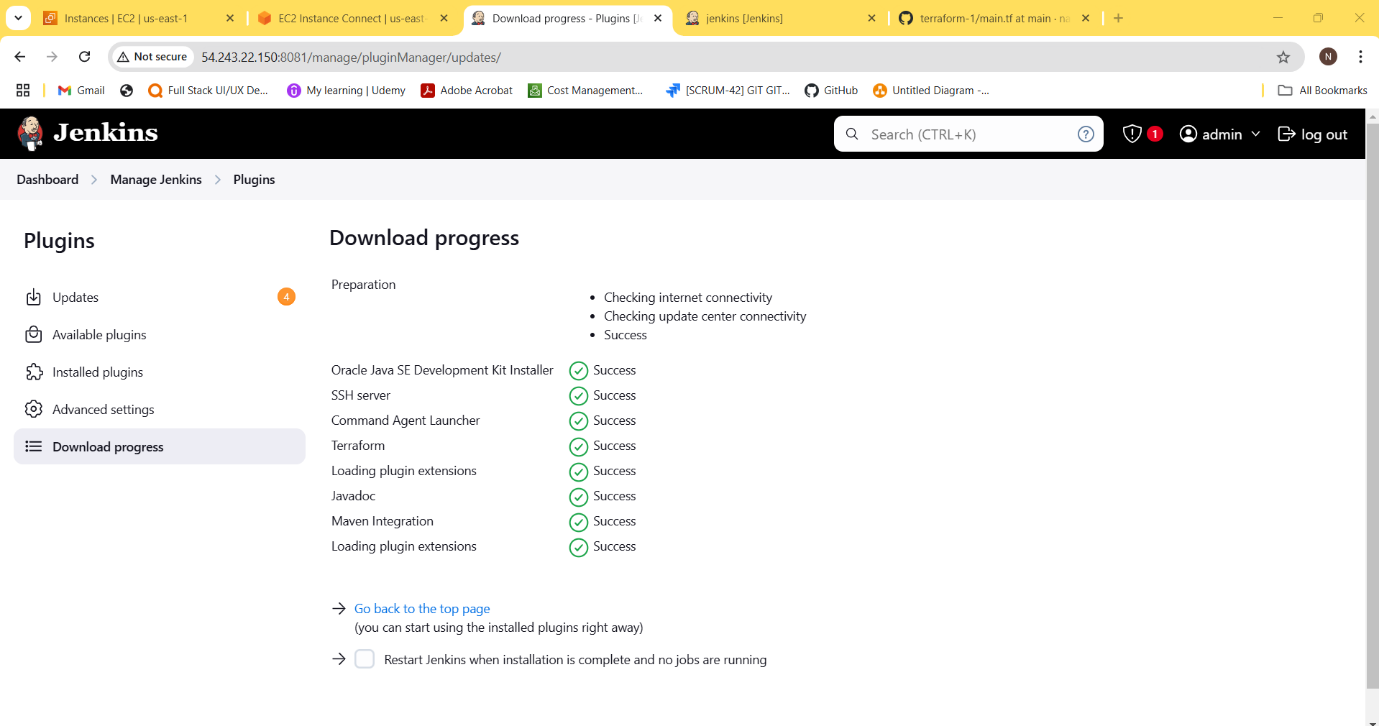
* Done

2) Execute the script shown in video.  
A screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated

3) Intergrate terrafrom in jenkins using Terraform plugin.

  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated

4) Create one jenkins job using MAVEN PROJECT for the below code with two stagesA screenshot of a computer

Description automatically generated

stage 1: Git clone  
A screenshot of a computer

Description automatically generated

stage 2: Maven Compilation  
A close-up of a white page

Description automatically generated

Code: https://github.com/betawins/java-Working-app.git

5) Use the below code and create a parameterized job in jenkins

stage 1: Git clone

stage 2: Maven Compilation

Code: <https://github.com/betawins/java-Working-app.git>  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated  
A screenshot of a chat

Description automatically generated  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

6) What are the global varaiables in jenkins?

**1.BUILD\_NUMBER**: This variable contains the current build number. It's automatically

incremented with each build.

**2.BUILD\_ID**: A unique identifier for the current build,typically in the format "YYYY-MM-DD\_hh-mm-ss."

**3.JOB\_NAME :** The name of the current Jenkins job or project.

**4.WORKSPACE**: The directory where the current build is executed. This can be used to reference

files and directories within the workspace.

**5.JENKINS\_HOME**: The path to the Jenkins installation directory.

**6.EXECUTOR\_NUMBER**: The unique number of the current build executor (e.g., 0, 1, 2).

**7.NODE\_NAME**: The name of the agent (slave) on which the current build is running. If the build is

running on the master, this will be "master."

**8.JOB\_URL**: The URL of the current Jenkins job or project.

**9.BUILD\_URL**: The URL of the current build.

**10.BUILD\_TAG**: A unique tag for the current build,typically in the format "jenkins-${JOB\_NAME}-$

{BUILD\_NUMBER}."

**11.BUILD\_DISPLAY\_NAME**: The human-readable display name for the current build.

**12.BUILD\_CAUSE**: A description of the cause that triggered the current build (e.g., SCM change, manual

start).

**13.CHANGE\_ID**: The ID of the specific change or commit that triggered the build (for SCM-triggered

builds).

**14.CHANGE\_AUTHOR**: The author of the change or commit that triggered the build (for SCM-triggered

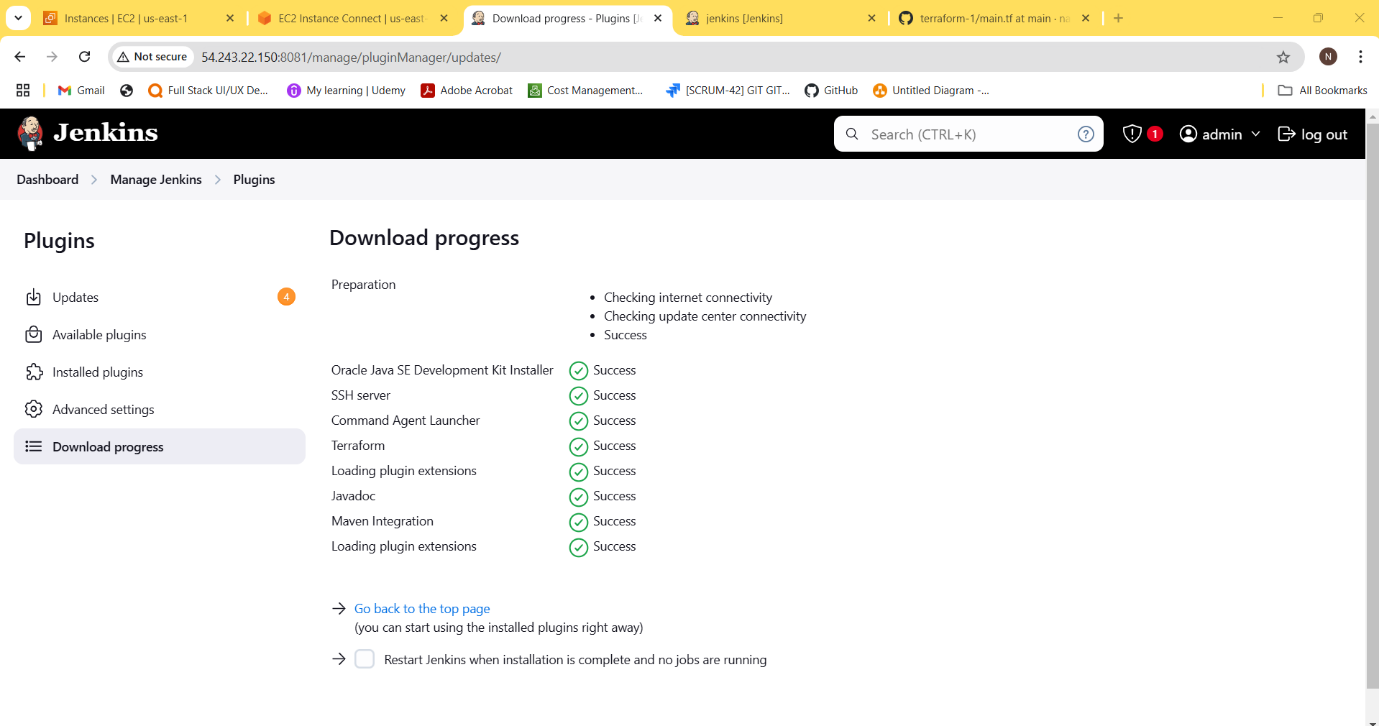
builds).

**15.BUILD\_USER**: The username of the user who triggered the build (for builds triggered by users)

7) Watch terraform-04 video

* Done

8) Execute the script shown in video

9) Integrate terrafrom in jenkins using Terraform plugin  
  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated

10) Create CICD pipeline for Nodejs Application.

https://github.com/betawins/Trading-UI.git

11) Explain 10 Maven commands.

**1.mvn clean**

Purpose: Deletes the target directory in the project, which contains all the compiled files, JARs, and temporary files created during the build process.

Usage: Cleans the project before starting a fresh build.

**2. mvn compile**

Purpose: Compiles the source code of the project.

Usage: Ensures that all .java files are converted to .class files under the target directory.

**3. mvn test**

Purpose: Runs the test cases in the project using a testing framework like JUnit or TestNG.

Usage: Ensures code changes have not broken existing functionality

**4. mvn package**

Purpose: Compiles the code, runs the tests, and packages the compiled code into a deployable format (e.g., JAR, WAR).

Usage: Used to generate the output artifact of the project

**5. mvn install**

Purpose: Installs the packaged artifact into the local Maven repository (~/.m2/repository).

Usage: Makes the artifact available for other projects on the same machine.

**6. mvn deploy**

Purpose: Uploads the packaged artifact to a remote repository.

Usage: Used for sharing artifacts across multiple teams or environments.

**7. mvn dependency:tree**

Purpose: Displays the dependency hierarchy (tree) of the project.

Usage: Useful for analyzing and troubleshooting dependency conflicts.

**8. mvn validate**

Purpose: Validates the project’s pom.xml file and checks for errors or missing required configuration.

Usage: Ensures that the project's setup is correct before proceeding with further steps.

**9. mvn site**

Purpose: Generates a website with information about the project, such as reports, documentation, and dependencies.

Usage: Used for project documentation and reporting

**10. mvn exec:java**

Purpose: Runs a Java class within the project using Maven.

Usage: Useful for testing or running small applications directly.