Lab Experiment 2: Creating a Jenkins Pipeline with a Jenkins file

Objective: Create a Jenkins pipeline using a Jenkinsfile that builds a simple project, runs tests, and deploys the project to a designated environment.

Prerequisites:

1. Jenkins server up and running.
2. A sample project hosted in a version control repository (e.g., Git).

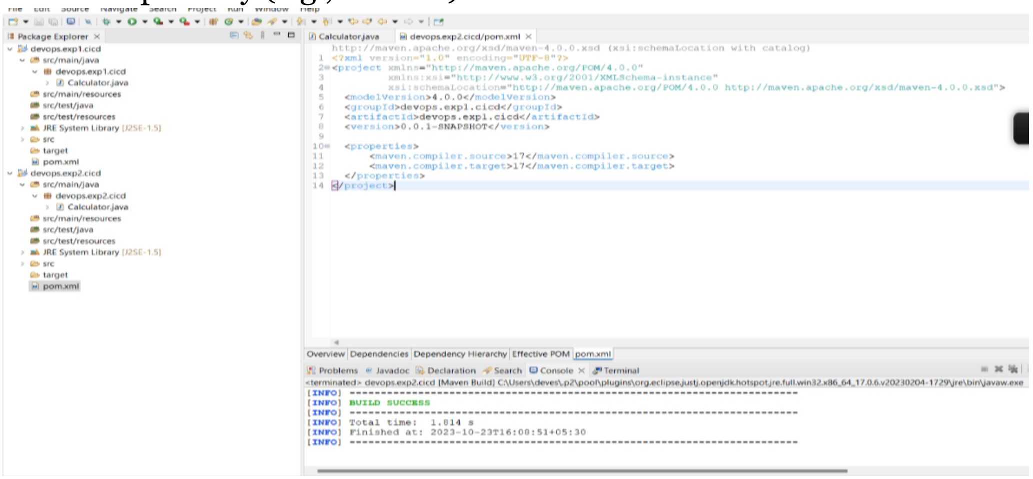
Steps:

Jenkins Configuration:

- Ensure that Jenkins is installed and accessible.
- Install necessary plugins: Pipeline and any plugins specific to your version control system (e.g., Git Plugin).

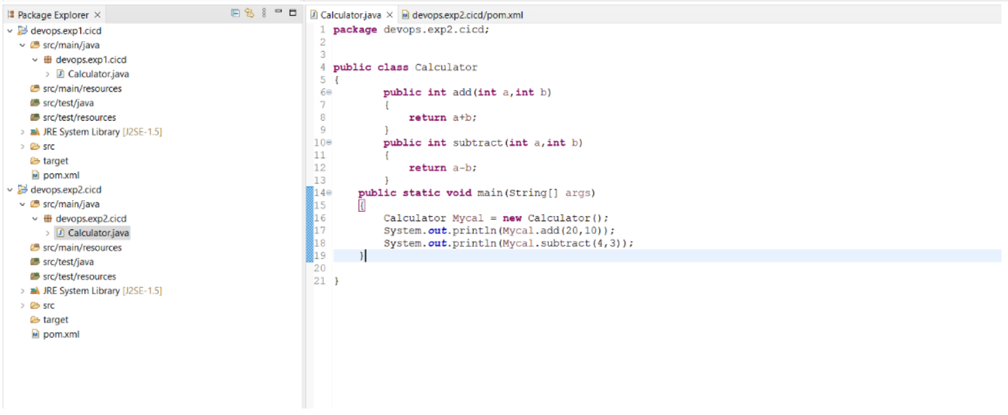
Setting Up the Project:

- Create a sample project (e.g., a simple web application) and host it on a version control repository (e.g., GitHub).



```xml
http://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation with catalog)
<?xml version="1.0" encoding="UTF-8"?>
<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/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>devops.exp1.cicd</groupId>
  <artifactId>devops.exp1.cicd</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <properties>
      <maven.compiler.source>17</maven.compiler.source>
      <maven.compiler.target>17</maven.compiler.target>
  </properties>
</project>
```

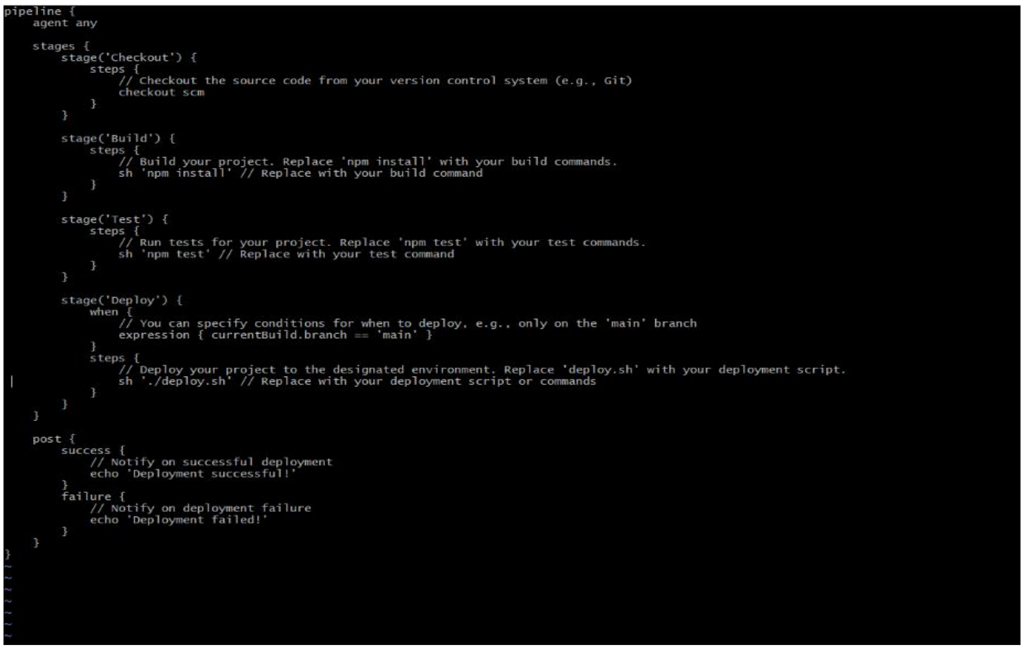
```
<terminated> devops.exp2.cicd [Maven Build] C:\Users\deves\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-1729\jre\bin\javaw.exe
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.814 s
[INFO] Finished at: 2023-10-23T16:08:51+05:30
[INFO] ------------------------------------------------------------------------
```



```java
package devops.exp2.cicd;


public class Calculator
{
        public int add(int a,int b)
        {
            return a+b;
        }
        public int subtract(int a,int b)
        {
            return a-b;
        }
    public static void main(String[] args)
    {
        Calculator Mycal = new Calculator();
        System.out.println(Mycal.add(20,10));
        System.out.println(Mycal.subtract(4,3));
    }

}
```

Creating a Jenkins file:

In the root of your project repository, create a file named Jenkinsfile.



```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                // Checkout the source code from your version control system (e.g., Git)
                checkout scm
            }
        }

        stage('Build') {
            steps {
                // Build your project. Replace 'npm install' with your build commands.
                sh 'npm install' // Replace with your build command
            }
        }

        stage('Test') {
            steps {
                // Run tests for your project. Replace 'npm test' with your test commands.
                sh 'npm test' // Replace with your test command
            }
        }

        stage('Deploy') {
            when {
                // You can specify conditions for when to deploy, e.g., only on the 'main' branch
                expression { currentBuild.branch == 'main' }
            }
            steps {
                // Deploy your project to the designated environment. Replace 'deploy.sh' with your deployment script.
                sh './deploy.sh' // Replace with your deployment script or commands
            }
        }
    }

    post {
        success {
            // Notify on successful deployment
            echo 'Deployment successful!'
        }
        failure {
            // Notify on deployment failure
            echo 'Deployment failed!'
        }
    }
}
~
~
~
~
~
~
~
```

Defining the Pipeline:

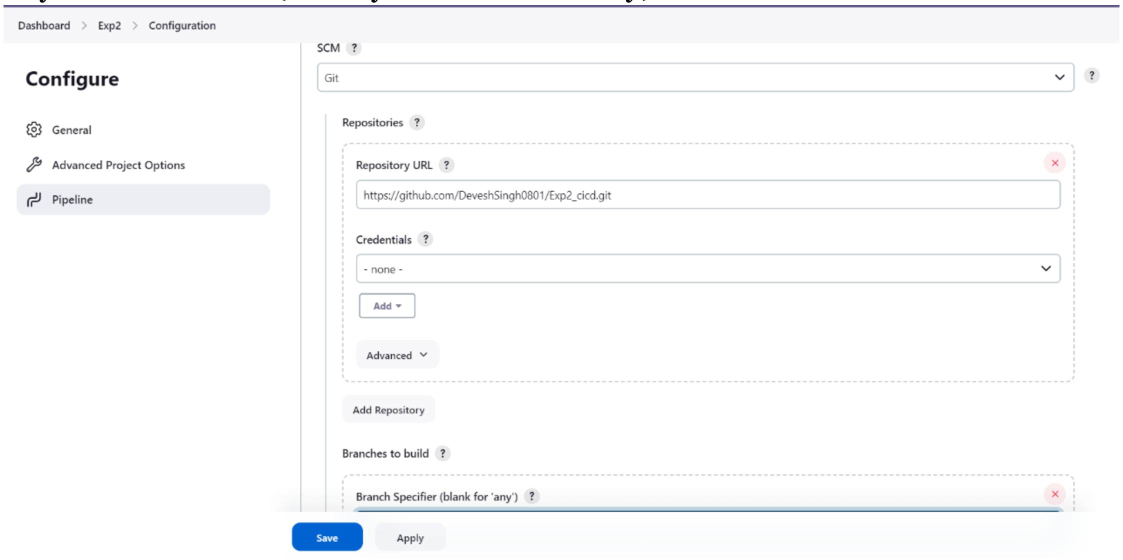
Open the Jenkins file and define the pipeline stages using the declarative pipeline syntax.

Here's an example Jenkins file with basic stages:

```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                checkout scm
            }
        }

        stage('Build') {
            steps {
                sh 'your-build-command-here'
            }
        }

        stage('Test') {
            steps {

                sh 'your-test-command-here'
            }
        }

        stage('Deploy') {
            steps {

                sh 'your-deployment-command-here'
            }
        }
    }

    post {
        success {
            echo 'Pipeline succeeded! Project built and deployed.'
        }
        failure {
            echo 'Pipeline failed! Check logs for details.'
        }
    }
}
```

Configuring the Pipeline in Jenkins:

- In Jenkins, create a new pipeline job.
- Link the job to your version control repository (e.g., provide the repository URL).
- Choose the option to use a Jenkinsfile from the repository and specify the path to your Jenkinsfile (usually the root directory).



SCM: Git

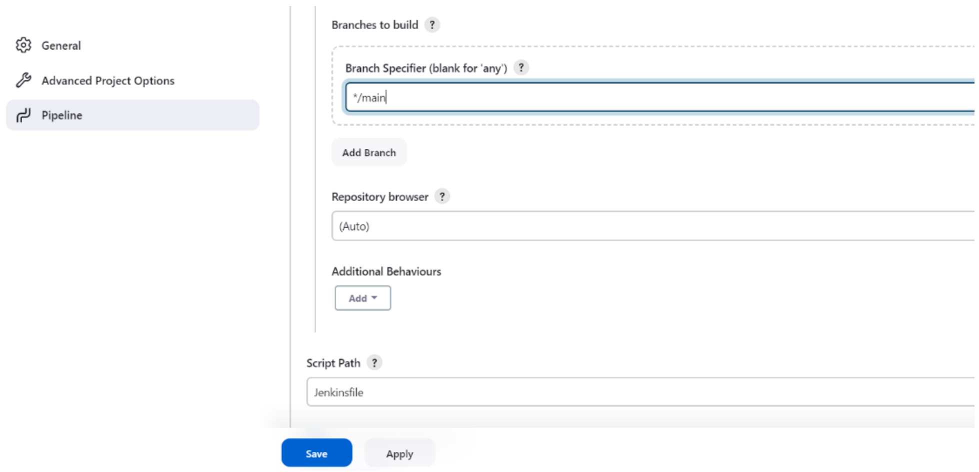
Repositories

Repository URL: https://github.com/DeveshSingh0801/Exp2_cicd.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any')



Branches to build

Branch Specifier (blank for 'any'): */main

Add Branch

Repository browser: (Auto)

Additional Behaviours

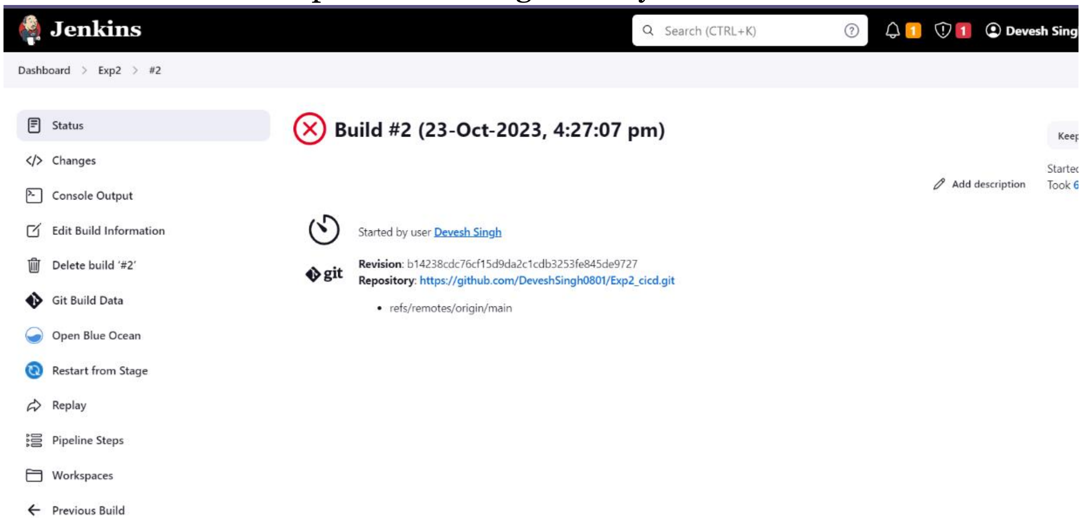
Script Path: Jenkinsfile

Running the Pipeline:

- Trigger the pipeline manually or set up a webhook to trigger it automatically on repository changes.

Observing the Results:

- Observe the pipeline execution on the Jenkins dashboard.
- Check the console output of each stage for any errors or issues.



Build #2 (23-Oct-2023, 4:27:07 pm)

Started by user Devesh Singh

Revision: b14238cdc76cf15d9da2c1cdb3253fe845de9727
Repository: https://github.com/DeveshSingh0801/Exp2_cicd.git

- refs/remotes/origin/main