

1. Set Up Jenkins Job

1. Open Jenkins:

- Go to your Jenkins dashboard.

2. Create a New Job:

- Click on **New Item**.

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job



Set up a distributed build

Set up an agent



Configure a cloud



Learn more about distributed builds



- Enter a name for your job (e.g., Deploy on Develop Branch).

New Item

Enter an item name

Deploy on Develop Branch

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different

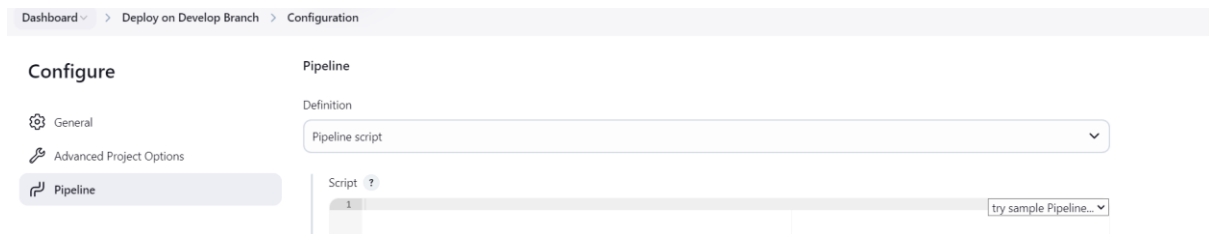
OK

- Select **Pipeline** and click **OK**.

3. Configure the Pipeline:

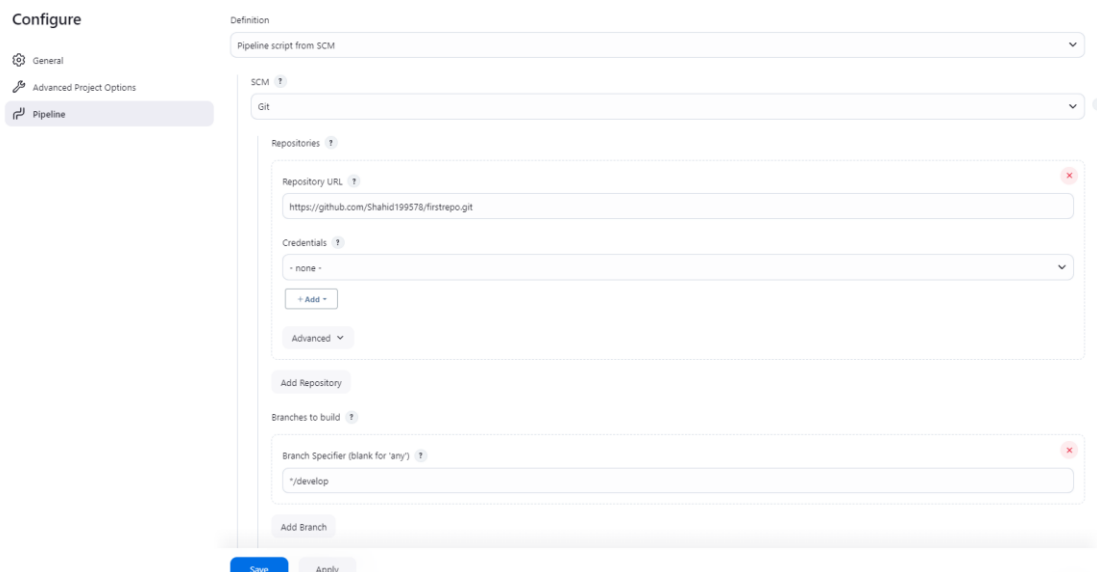
- **Pipeline Section:**

- In the **Pipeline** section, you'll configure the pipeline script.



2. Configure Pipeline Job:

- In the job configuration page:
 - Scroll to the **Pipeline** section.
 - Set **Definition** to **Pipeline script from SCM**.
 - Choose **Git** as the SCM.
 - Enter the **Repository URL** and credentials if necessary.
 - Set **Branch Specifier** to ***/develop**.
 - In **Script Path**, enter the path to your Jenkinsfile (if it's in the root of the repository, just enter Jenkinsfile).



4. Configure Build Triggers:

- Go to the **Build Triggers** section.

- Check **GitHub hook trigger for GITScm polling** if using GitHub, or **Poll SCM** if you want Jenkins to periodically check for changes.

```
jenkinsfile

pipeline {
    agent any

    environment {
        // Define the folder where the Git content will be pulled
        FOLDER_NAME = 'develop'
        REPO_URL = ' https://github.com/Shahid199578/firstrepo.git '
        BRANCH_NAME = 'develop'
    }

    stages {
        stage('Checkout') {
            steps {
                script {
                    // Clean up previous workspace
                    deleteDir()

                    // Create a directory for the Git content
                    sh "mkdir -p ${env.FOLDER_NAME}"

                    // Checkout code from Git into the specific folder
                    dir(env.FOLDER_NAME) {
                        git branch: env.BRANCH_NAME, url: env.REPO_URL
                    }
                }
            }
        }

        stage('Build') {
```

```

steps {
    // Build steps here
    echo 'Building...'
}
}

stage('Deploy') {
    steps {
        // Deploy steps here
        echo 'Deploying...'
    }
}
}
}

```

5. Save and Test

1. Save the Jenkins Job:

- Click Save to apply your changes.

2. Test the Setup:

- Make a change to your develop branch and push it to your Git repository.
- Jenkins should automatically trigger the pipeline and execute the steps defined in your Jenkinsfile.

6. Verify

- Ensure that Jenkins pulls the content to the specified folder and triggers the pipeline on each push to the develop branch.

```

● PS C:\Users\Mohd Shahid\Desktop\firstrepo> git checkout develop
Already on 'develop'
M   1.txt
Your branch is up to date with 'origin/develop'.
● PS C:\Users\Mohd Shahid\Desktop\firstrepo> git commit -m "modified"
[develop 7faa1fe] modified
1 file changed, 1 insertion(+)
● PS C:\Users\Mohd Shahid\Desktop\firstrepo> git push origin develop
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 283 bytes | 283.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Shahid199578/firstrepo.git
816ac44..7faa1fe develop -> develop
○ PS C:\Users\Mohd Shahid\Desktop\firstrepo> 

```

Pipeline

