

Steps to create a Jenkins jobs

1.Go to Jenkin dashboard.

2.search the job name as per the given excel sheet (Replace . with _ while searching the job name in jenkins)

B	C	D	
Job Name / Project Name	Bitbucket Group Name	Status	Job Errors
CoreLibraries.Audio.NMF	InternalTooling	Done	
IOSP.REST	CS	Done	
SystemOverview.Authentication.Fusion	SystemOverview	Done	
SystemOverview.Compass.Channels.Plugin	SystemOverview	Done	
NiceApplications.Playback	External	Done	
CoreLibraries.SQL.MySQL	CoreLibraries	Done	
Observe.REST	CS	Done	
SystemOverview.AuditTrail.Fusion	CyberTech	Done	

Like for the above job search like NiceApplications_Playback

3.If you are not able to find the job on jenkins dashboard then create the job with the above name as explained.


4.For that in Jenkins dashboard go to new item.

5.Enter your Job name for ex: NiceApplications_Playback then choose pipeline and press ok.


Enter an item name

NiceApplications_Playback


» Required field

**Freestyle project**


This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Maven project**


Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

**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.

**Backup and Restore**

Backs up and restores Jenkins

**External Job**

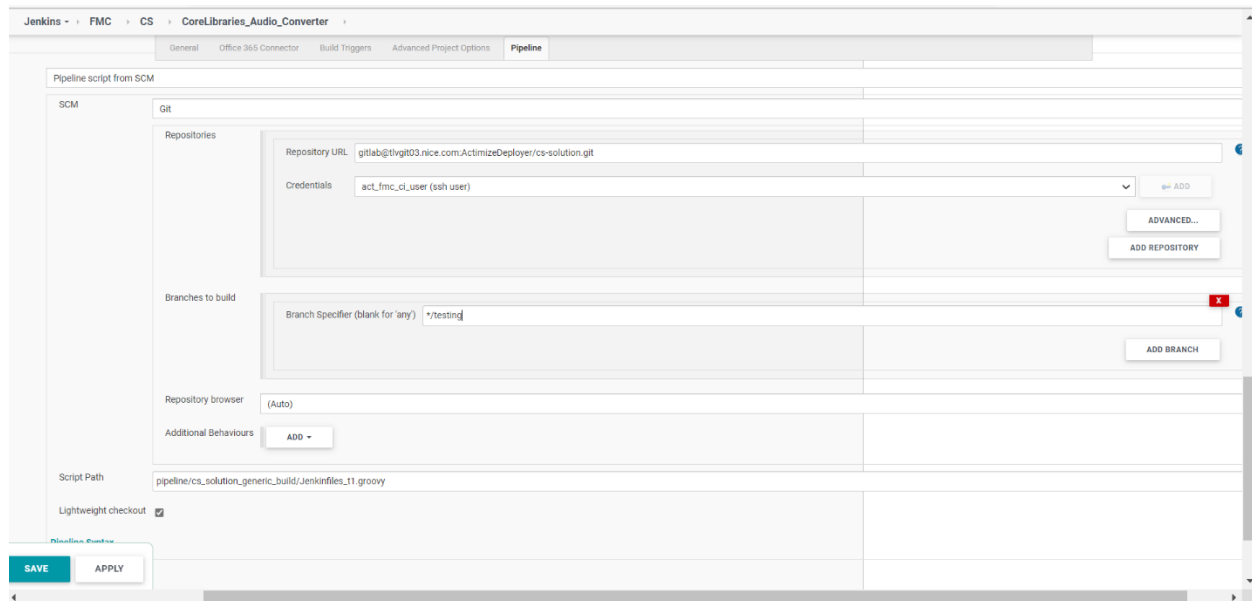
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.

**Spotinst AAF**

Provision with Spotinst + Deploy + AAF Test

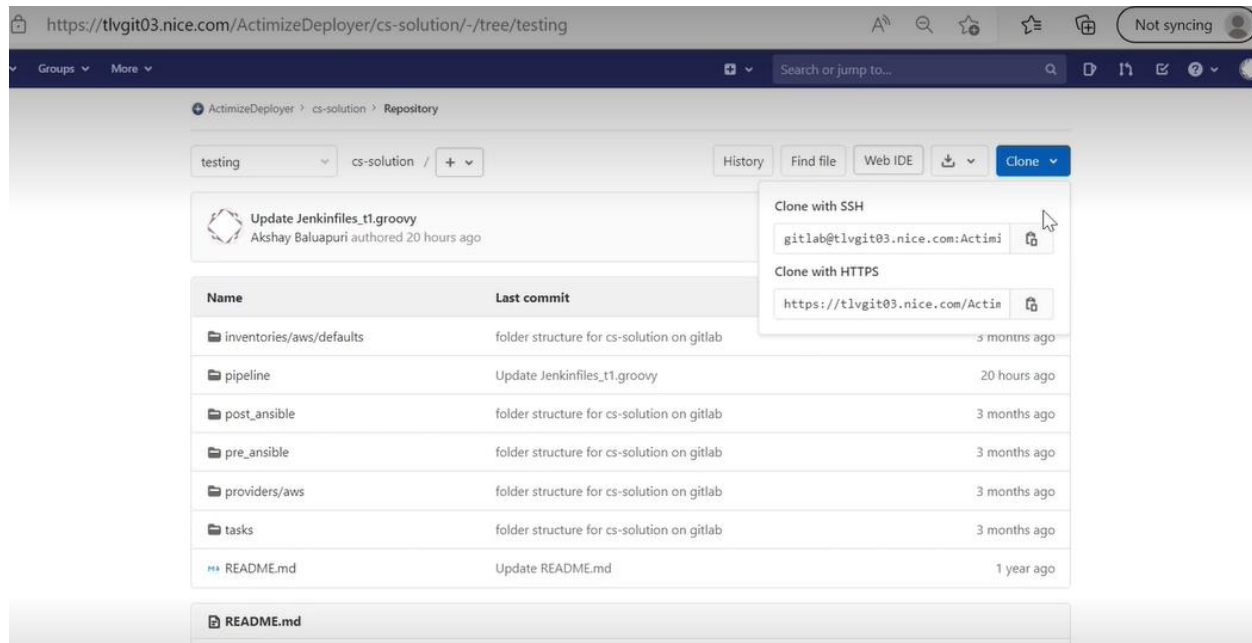
Waiting for ilactris nice.com

6.then in job go to pipeline section and choose pipeline script from SCM and in SCM Choose GIT, then it will ask GIT Repo URL and password ,Under branches to build choose testing branch */testing



The screenshot shows the Jenkins Pipeline configuration page for a job named 'CoreLibraries_Audio_Converter'. The 'Pipeline script from SCM' section is selected. Under 'SCM', 'Git' is chosen. The 'Repository URL' is 'gitlab@tlvgit03.nice.com:ActimizeDeployer/cs-solution.git'. The 'Credentials' dropdown shows 'act_fmc_ci_user (ssh user)'. The 'Branches to build' section has a 'Branch Specifier (blank for \'any\')' set to '*/testing'. The 'Repository browser' is set to '(Auto)'. The 'Script Path' is 'pipeline/cs_solution_generic_build/jenkinsfiles_t1.groovy'. The 'Lightweight checkout' checkbox is checked. At the bottom, there are 'SAVE' and 'APPLY' buttons.

For GIT URL go to git lab then CS project then choose Testing branch from there you can choose the SSH protocol URL.



The screenshot shows the GitLab repository page for 'ActimizeDeployer > cs-solution'. The 'testing' branch is selected. The 'Clone' button is clicked, showing a dropdown menu with 'Clone with SSH' and 'Clone with HTTPS' options. The SSH URL is 'gitlab@tlvgit03.nice.com:ActimizeDeployer/cs-solution.git'. The repository contains a commit 'Update Jenkinsfiles_t1.groovy' by Akshay Baluapuri 20 hours ago. Below the commit list, there is a table of files and folders.

Name	Last commit	Time ago
inventories/aws/defaults	folder structure for cs-solution on gitlab	3 months ago
pipeline	Update Jenkinsfiles_t1.groovy	20 hours ago
post_ansible	folder structure for cs-solution on gitlab	3 months ago
pre_ansible	folder structure for cs-solution on gitlab	3 months ago
providers/aws	folder structure for cs-solution on gitlab	3 months ago
tasks	folder structure for cs-solution on gitlab	3 months ago
README.md	Update README.md	1 year ago

GIT LAB URL: <https://tlvgit03.nice.com/ActimizeDeployer/cs-solution>

Pw: [act_fmc_ci_user \(ssh user\)](#)

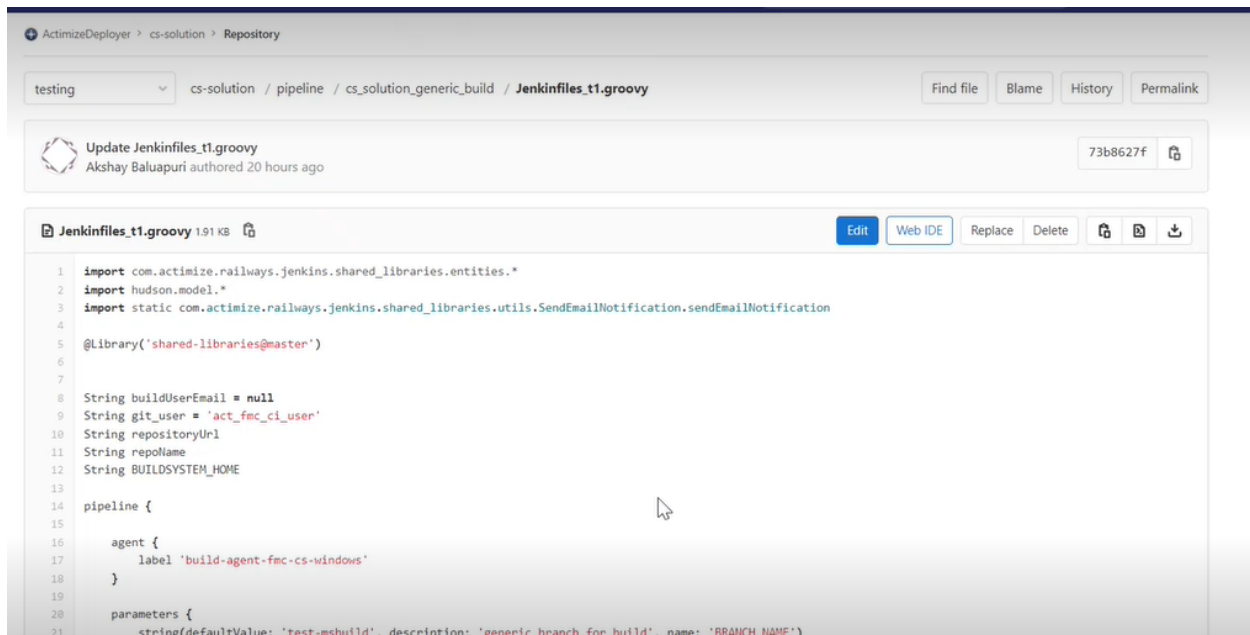
Then in jenkins >>

Then in script path : has to give the path of jenkins generic groovy file which we have already created in gitlab

Path is as below >> **cs-solution/pipeline/cs_solution_generic_build/Jenkinsfiles_t1.groovy**

Branch- testing

URL for JenkinsGenericfile: **https://tlvgit03.nice.com/ActimizeDeployer/cs-solution/-/blob/testing/pipeline/cs_solution_generic_build/Jenkinsfiles_t1.groovy**



Then save and create job. Now Jenkins will run the groovy script which we have on GitLab . As We have already provided the path of Groovy script file in our Jenkins job.

Explanation of stages in our groovy file.

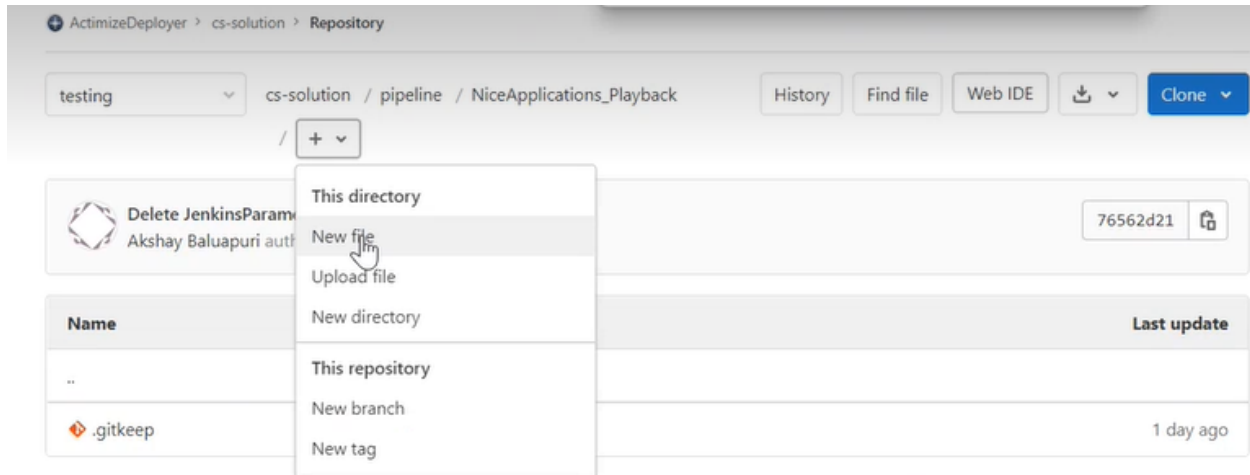
```
stages {
    stage('Preparations', {
        steps {
            script {
                repositoryUrl = load("pipeline/${env.JOB_BASE_NAME}/JenkinsParameters.groovy").repositoryUrl()
            }
        }
    })
}
```

This means when we trigger the job it will load the repository url means it will go inside pipeline folder then it will search for job name then inside job it will find the JenkinsParameters.groovy file.

So for this we have to create JenkinsParameters.groovy file inside job

Path : **Cs-solution/pipeline/ NiceApplications_Playback/**

Then create new file as **JenkinsParameters.groovy**



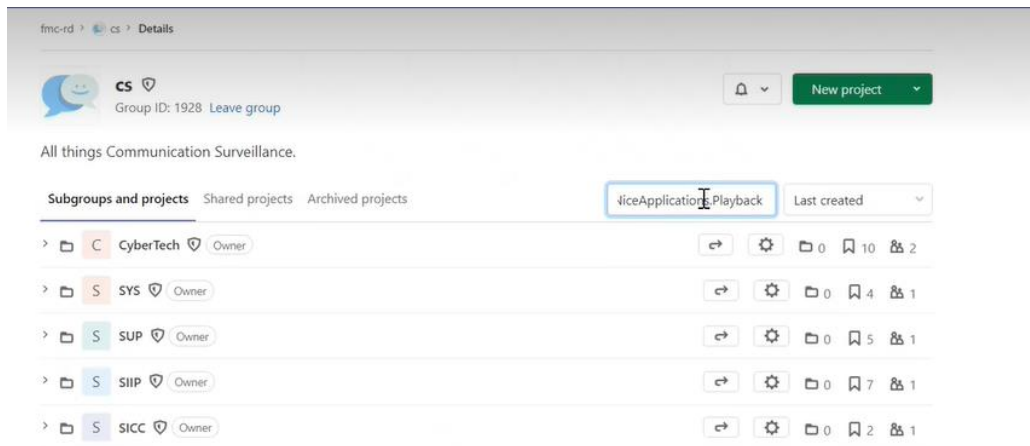
Now go to cs-solution/pipeline/any job which has successfully completed check the **JenkinsParameters.groovy** there and open the file and copy to your newly created file in your working job and modify the things as per the job like below.



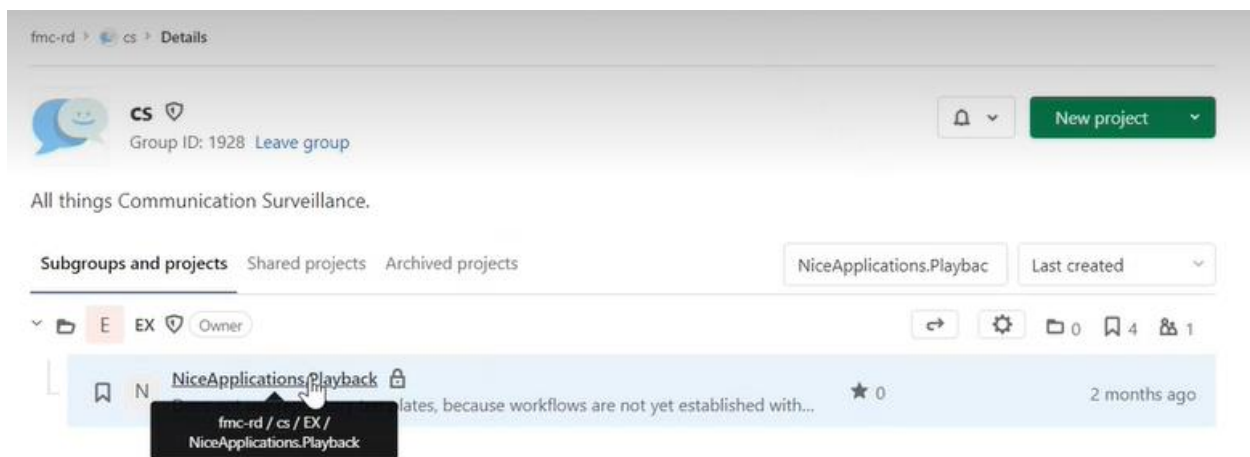
Now modify the git url according to the job, for that we have to go to our GITLAB

GIT URL: <https://tlvgit03.nice.com/fmc-rd/cs/>

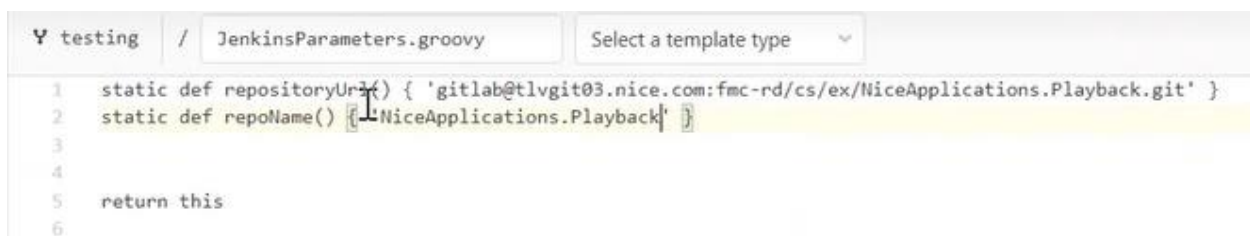
Then fmc-rd/cs/search your job name like below



Then go inside the job like below



Then copy the repository path by doing cloning and paste it inside the **JenkinsParameters.groovy** file and change the repo name as well. Like below ss



Note: Don't replace . with _

