Configuration Manual

MSc Research Project
Cloud Computing

Sanket Patil
Student ID: X21229139

School of Computing
National College of Ireland

Supervisor: Sean Heeney

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Sanket Patil
Student ID:	X21229139
Programme:	Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Sean Heeney
Submission Due Date:	14/12/2023
Project Title:	Configuration Manual
Word Count:	1255
Page Count:	12

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	14th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Sanket Patil
X21229139

1 Introduction

1.1 Aim of the document

This document represents configuration manual that discuss complete step-by-step guide for deploying the methodology proposed in the research work. it also discusses tools and services required for the deployment

2 General overview

2.1 Research objective

The proposed research is focused on optimizing execution time of provisioning cloud resources. This research work utilizes Amazon Web Service as a cloud platform for deploying resources. It also uses GitHub and Jenkins for automating deployment process. Optimization of execution time was achieved by using python script. The script was developed using python 3.8 that includes concurrent.features library to facilitate Thread-Level Parallelism (TLP).

2.2 Tools and services

This section describes tools and services used during the implementation

2.2.1 GitHub

GitHub is used as a source control repository for storing the terraform scripts, python script and Jenkins file. Proposed work contains multiple GitHub repositories for storing the source code of the multiple approach proposed in the implementation.

2.2.2 Jenkins

In proposed work, Jenkins is used as an automation server for provisioning AWS resources by executing terraform scripts. In the proposed work, Jenkins server is configured with multiple pipelines for execution of the terraform scripts stored in GitHub repos.

2.2.3 AWS

AWS provides platform for provisioning the resources. In the proposed work, AWS credentials are required to deploy the resources.

3 Project configuration

3.1 Terraform scripts, Python scripts and Jenkins file

For the code artefact, please refer to the zip file submitted in ICT Solution Artefact. Modify the AWS credentials specified in Jenkins file and terraform script.

3.2 GitHub setup

The proposed research work contains three different approach for evaluating the proposed research methodology. For each approach, research work required different GitHub repository. In order to create the repository, login to "github.com" [1] or create a new account if don't have an existing account.

3.2.1 Parallel Execution Approach

Step 1: once login, click on "New" button

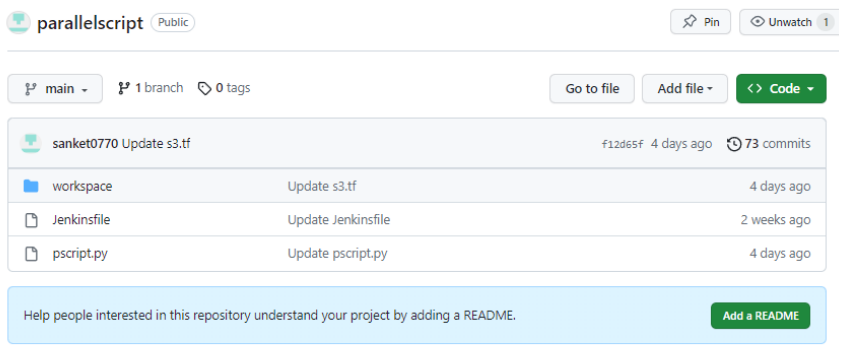
Step 2: specify the name to the repository for example "parallelscript" and select "public" access

Step 3: click on "Create Repository"

Step 4: After successful creation of repository, upload terraform script, python script and Jenkins file by using git CLI

Step 5: Once upload done, repository url will appear under "code" button.

Note: repository url is required while configuring Jenkins pipeline.



parallelscript Public

main 1 branch 0 tags

sanket0770 Update s3.tf f12d65f 4 days ago 73 commits

workspace	Update s3.tf	4 days ago
Jenkinsfile	Update Jenkinsfile	2 weeks ago
pscript.py	Update pscript.py	4 days ago

Help people interested in this repository understand your project by adding a README.

Figure 1: GitHub repository for parallel execution approach

3.2.2 Sequential Execution Approach

Step 1: once login, click on "New" button

[1]GitHub: `http://www.github.com`

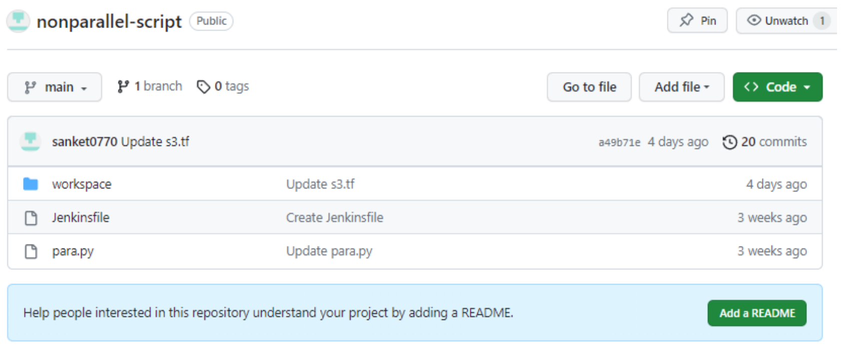
Step 2: specify the name to the repository for example "nonparallel-script" and select "public" access

Step 3: click on "Create Repository"

Step 4: After successful creation of repository, upload terraform script, python script and Jenkins file by using git CLI

Step 5: Once upload done, repository url will appear under "code" button.

Note: repository url is required while configuring Jenkins pipeline.


nonparallel-script Public

sanket0770 Update s3.tf — a49b71e 4 days ago — 20 commits

workspace	Update s3.tf	4 days ago
Jenkinsfile	Create Jenkinsfile	3 weeks ago
para.py	Update para.py	3 weeks ago

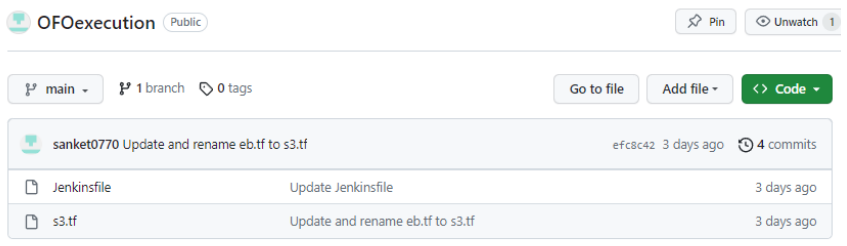
Help people interested in this repository understand your project by adding a README.

Figure 2: GitHub repository for sequential execution approach

3.2.3 Traditional Execution Approach

Traditional execution approach required three different GitHub repository to store the terraform scripts and Jenkins file.

Step 1: to create first repository for S3, follow the same steps as mentioned in parallel execution approach and sequential execution approach.
Note: Specify different repository name in step 2 of parallel or sequential execution approach, for example "OFOexecution".

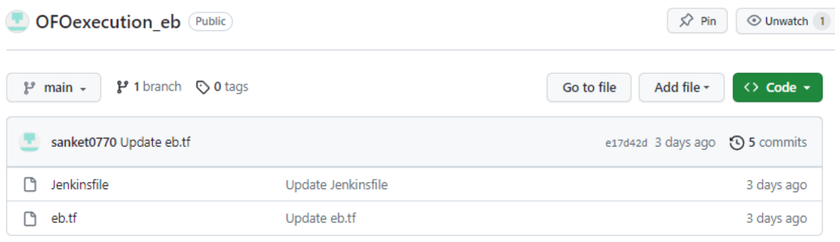

OFOexecution Public

sanket0770 Update and rename eb.tf to s3.tf — efc8c42 3 days ago — 4 commits

Jenkinsfile	Update Jenkinsfile	3 days ago
s3.tf	Update and rename eb.tf to s3.tf	3 days ago

Figure 3: GitHub repository for traditional execution approach to deploy S3 resource

Step 2: to create first repository for elastic beanstalk, follow the same steps as mentioned in parallel execution approach and sequential execution approach.


OFOexecution_eb Public

Pin Unwatch 1

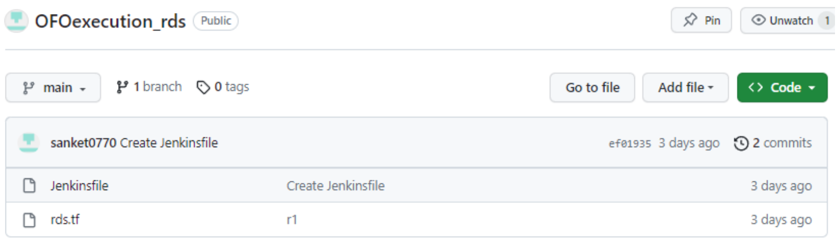
main 1 branch 0 tags Go to file Add file Code

sanket0770 Update eb.tf e17d42d 3 days ago 5 commits

Jenkinsfile	Update Jenkinsfile	3 days ago
eb.tf	Update eb.tf	3 days ago

Figure 4: GitHub repository for traditional execution approach to deploy Elastic Beanstalk resource

Note: Specify different repository name in step 2 of parallel or sequential execution approach, for example "OFOexecution_eb".
Step 3: to create first repository for RDS, follow the same steps as mentioned in parallel execution approach and sequential execution approach.
Note: Specify different repository name in step 2 of parallel or sequential execution approach, for example "OFOexecution_rds".


OFOexecution_rds Public

Pin Unwatch 1

main 1 branch 0 tags Go to file Add file Code

sanket0770 Create Jenkinsfile ef01935 3 days ago 2 commits

Jenkinsfile	Create Jenkinsfile	3 days ago
rds.tf	r1	3 days ago

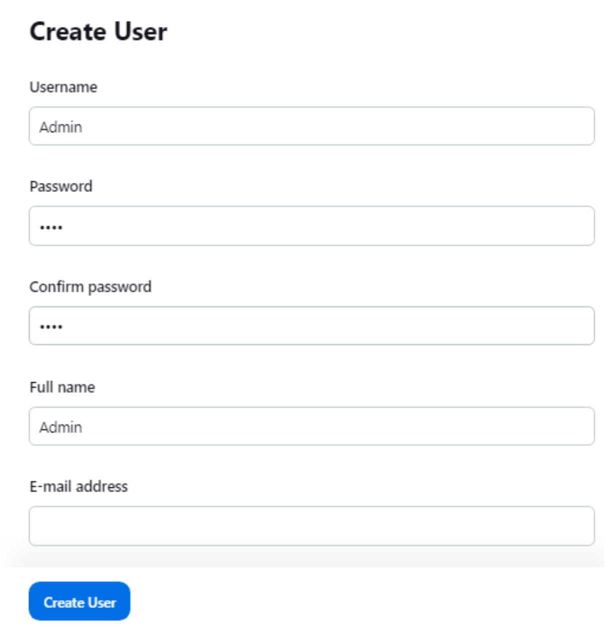
Figure 5: GitHub repository for traditional execution approach to deploy AWS RDS resource

3.3 Jenkins configuration

The methodology proposed in this research requires Jenkins automation server for execution of terraform scripts. For evaluation purpose, multiple Jenkins pipeline need to be configured in order to implement different approaches proposed in the methodology.

Step 1: Download Jenkins from "https://www.jenkins.io/download/" [2] ,and configure username and password.

[2]Jenkins: `https://www.jenkins.io/download/`

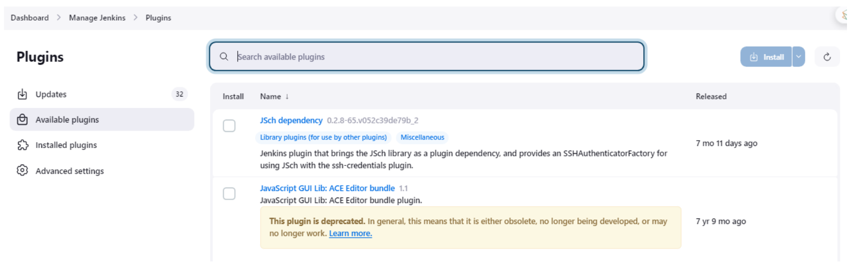


Create User

Username	Admin
Password	••••
Confirm password	••••
Full name	Admin
E-mail address	

Figure 6: Jenkins User Creation

Step 2: Install below plugins:
GitHub Plugin
Terraform Plugin
AWS credential Plugin
Go to Dashboard - manage Jenkins - Plugins - Available Plugins - select above plugin - Install



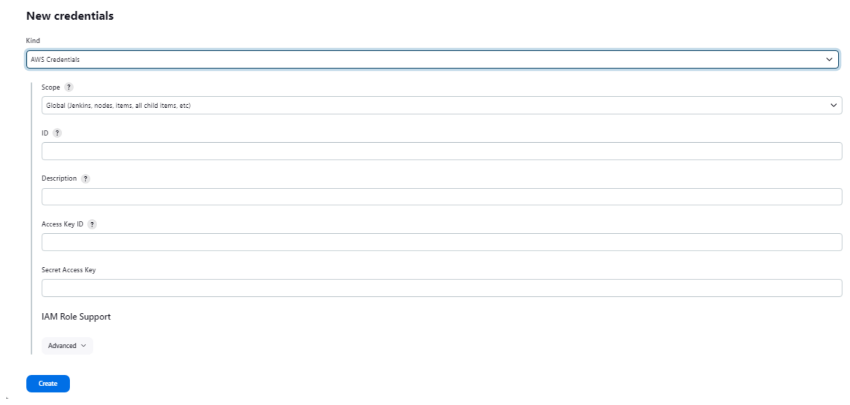
Dashboard > Manage Jenkins > Plugins

Plugins

Install	Name ↓	Released
☐	JSch dependency 0.2.8-65.v052c39de79b_2 Library plugins (for use by other plugins) Miscellaneous Jenkins plugin that brings the JSch library as a plugin dependency, and provides an SSHAuthenticatorFactory for using JSch with the ssh-credentials plugin.	7 mo 11 days ago
☐	JavaScript GUI Lib: ACE Editor bundle 1.1 JavaScript GUI Lib: ACE Editor bundle plugin. **This plugin is deprecated.** In general, this means that it is either obsolete, no longer being developed, or may no longer work. Learn more.	7 yr 9 mo ago

Figure 7: Jenkins Plugin Installation

Step 3: Configure AWS credential: Go to Dashboard - Manage Jenkins - Credentials - Add Credentials - select " AWS credentials" for kind and specify Access Key ID and Secret Access Key - Create


New credentials

Kind: AWS Credentials

Scope: Global (Jenkins, nodes, items, all child items, etc)

ID:

Description:

Access Key ID:

Secret Access Key:

IAM Role Support

Figure 8: Jenkins AWS Credentials Configuration

Step 4 : Creating Pipeline

a. **For parallel Execution**

1.From dashboard select "New item"

2.select "pipeline" and provide the name, Click ok.

3.In pipeline, under definition select "pipeline script from SCM"

4. Under SCM select "Git"

5. Specify repository URL GitHub repository that created for parallel approach

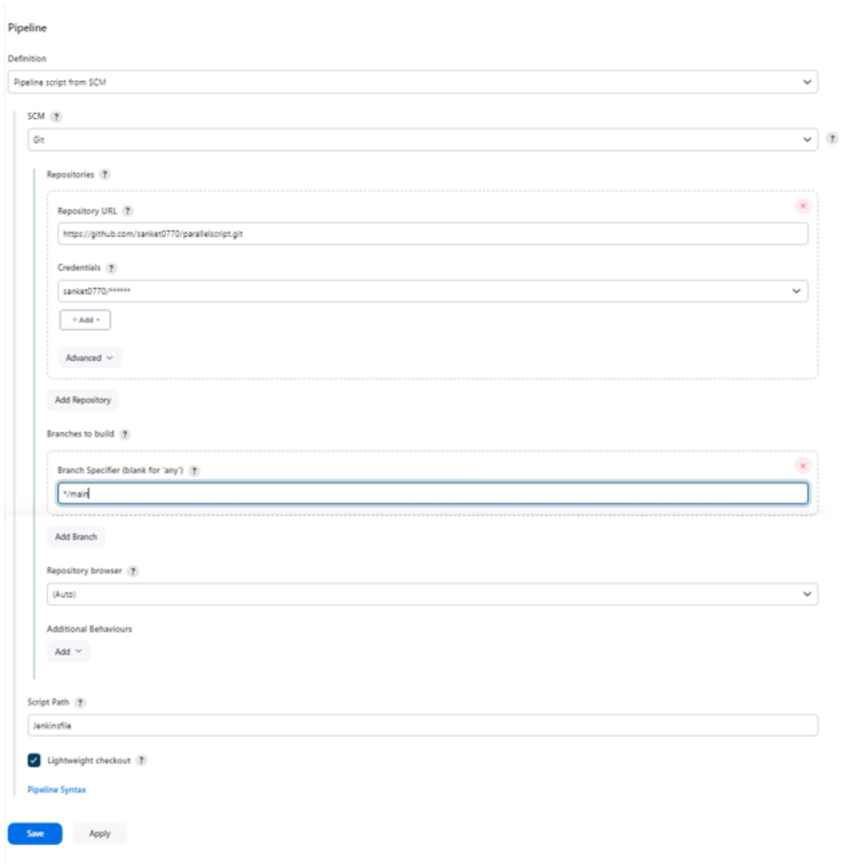
6. Specify GitHub credential, if it is not configured click on "add" and configure the GitHub credentials

7. Specify branch of GitHub repository under branch specifier

8. Specify script path as "Jenkinsfile"

b. **For sequential execution**

1.From dashboard select "New item"


Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/sanket0770/parallelscript.git

Credentials: sanket0770/******

Branches to build

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

Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

[✓] Lightweight checkout

Figure 9: Jenkins Pipeline for Parallel Approach execution

2.select “pipeline” and provide the name, Click ok.

3.In pipeline, under definition select “pipeline script from SCM”

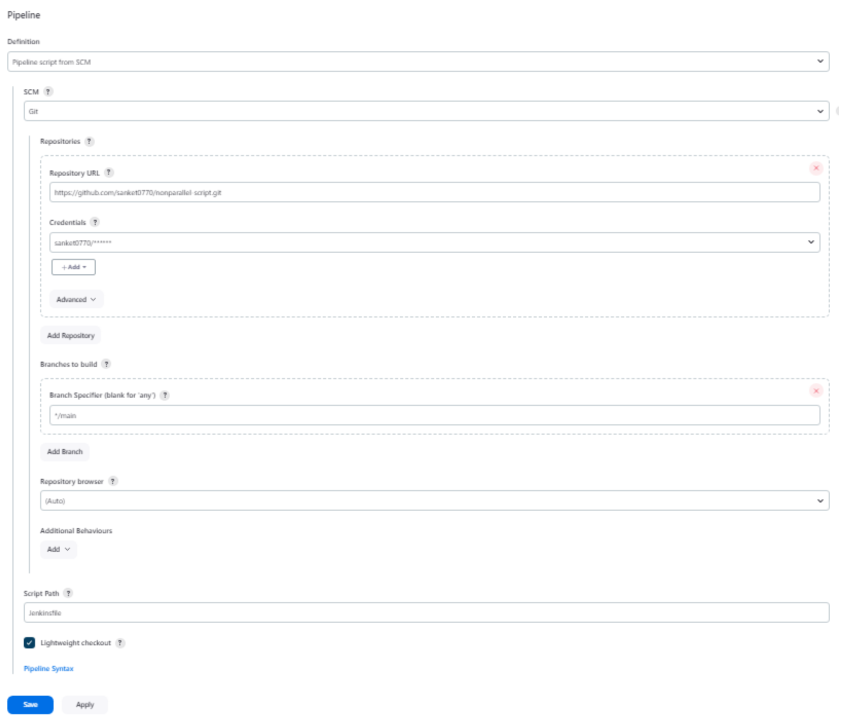
4. Under SCM select “Git”

5. Specify repository URL GitHub repository that created for sequential approach

6. Specify GitHub credential, if it is not configured click on “add” and configure the GitHub credentials

7. Specify branch of GitHub repository under branch specifier

8. Specify script path as “Jenkinsfile”



Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/sanket0770/nonparallel-script.git

Credentials: sanket0770/******

Branches to build

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

Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

[✓] Lightweight checkout

Figure 10: Jenkins Pipeline for Sequential Approach execution

c. **For traditional execution:**
Required three separate pipeline that needs to be interconnected.

For RDS:

1.From dashboard select “New item”

2.select “pipeline” and provide the name, Click ok.

3.In pipeline, under definition select “pipeline script from SCM”

4. Under SCM select “Git”

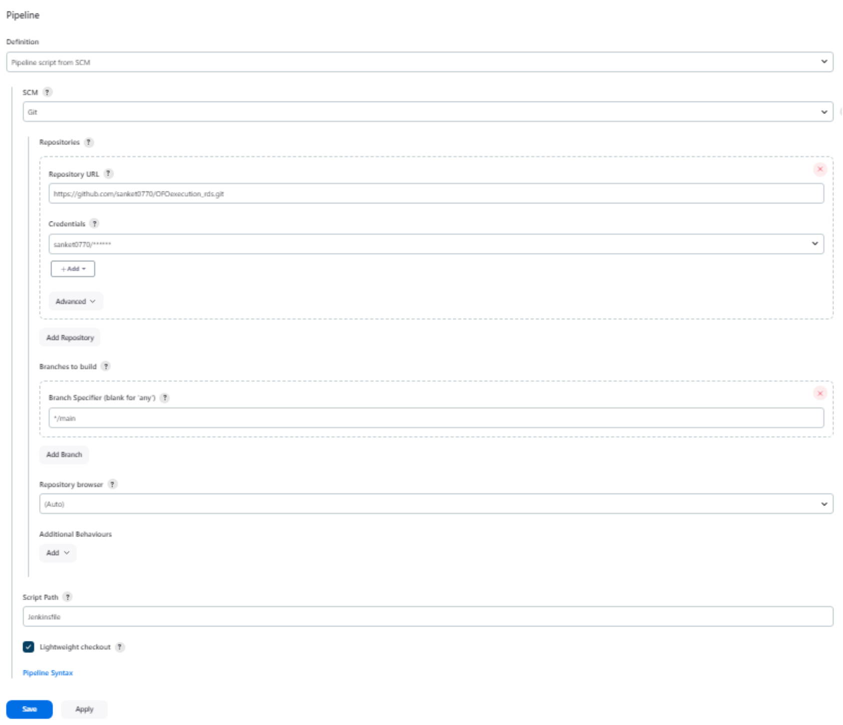
5. Specify repository URL of GitHub RDS repository

6. Specify GitHub credential, if it is not configured click on “add” and configure the GitHub credentials

7. Specify branch of GitHub repository under branch specifier

8. Specify script path as “Jenkinsfile”

9. Click “Save”


Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/sanket0770/OFOexecution_rds.git

Credentials: sanket0770/******

Branches to build

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

Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

[✓] Lightweight checkout

Figure 11: Jenkins Pipeline for Traditional Execution of RDS

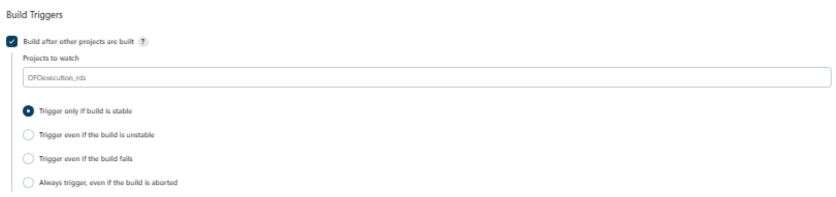
For Elastic Beanstalk:

1.From dashboard select “New item”

2.select “pipeline” and provide the name, Click ok.

3. under “Build Trigger” select “build after other projects are build” - specify “RDS pipeline” name under Projects to watch - select “Trigger only if build is stable” option

4.In pipeline, under definition select “pipeline script from SCM”



Build Triggers

✓ Build after other projects are built

Projects to watch: OFOexecution_rds

[✓ Trigger only if build is stable] Trigger even if the build is unstable Trigger even if the build fails Always trigger, even if the build is aborted

Figure 12: Trigger Configuration For Running Elastic Beanstalk Pipeline

5. Under SCM select “Git”

6. Specify repository URL of GitHub Elastic Beanstalk repository

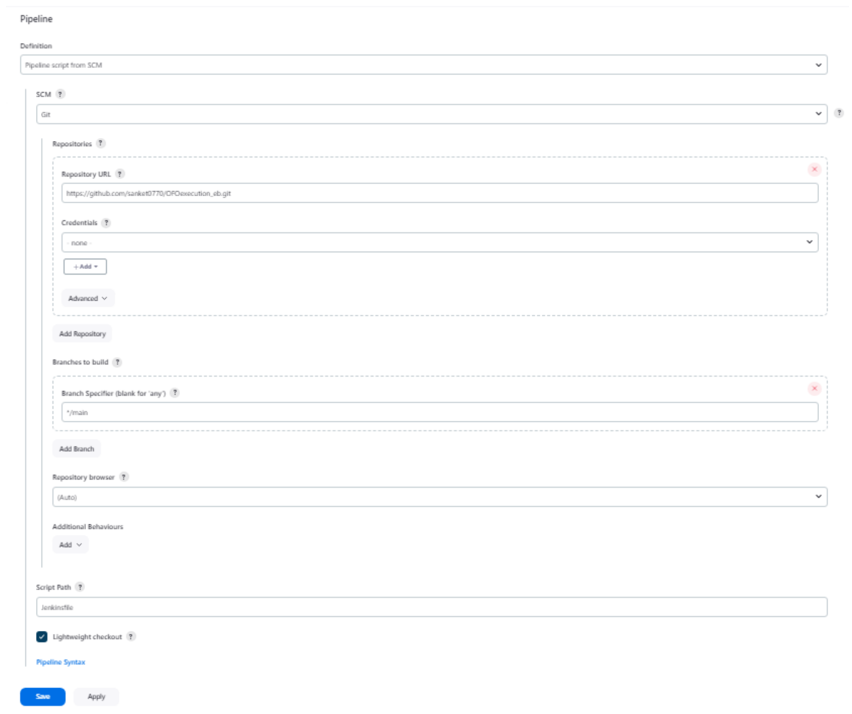
7. Specify GitHub credential, if it is not configured click on “add” and configure the GitHub credentials

8. Specify branch of GitHub repository under branch specifier

9. Specify script path as “Jenkinsfile”

10. Click “Save”

For S3:



Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/sanket0770/OFOexecution_eb.git

Credentials: - none -

Branches to build

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

Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

✓ Lightweight checkout

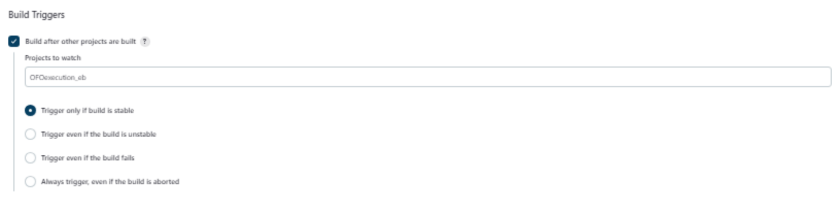
Figure 13: Jenkins Pipeline for Traditional Execution of Elastic Beanstalk

1.From dashboard select "New item"

2.select "pipeline" and provide the name, Click ok.

3. under "Build Trigger" select "build after other projects are build" - specify "Elastic Beanstalk pipeline" name under Projects to watch - select "Trigger only if build is stable" option

4.In pipeline, under definition select "pipeline script from SCM"



Build Triggers

✓ Build after other projects are built

Projects to watch: OFOexecution_eb

● Trigger only if build is stable

○ Trigger even if the build is unstable

○ Trigger even if the build fails

○ Always trigger, even if the build is aborted

Figure 14: Trigger Configuration For Running Elastic Beanstalk Pipeline

5. Under SCM select "Git"

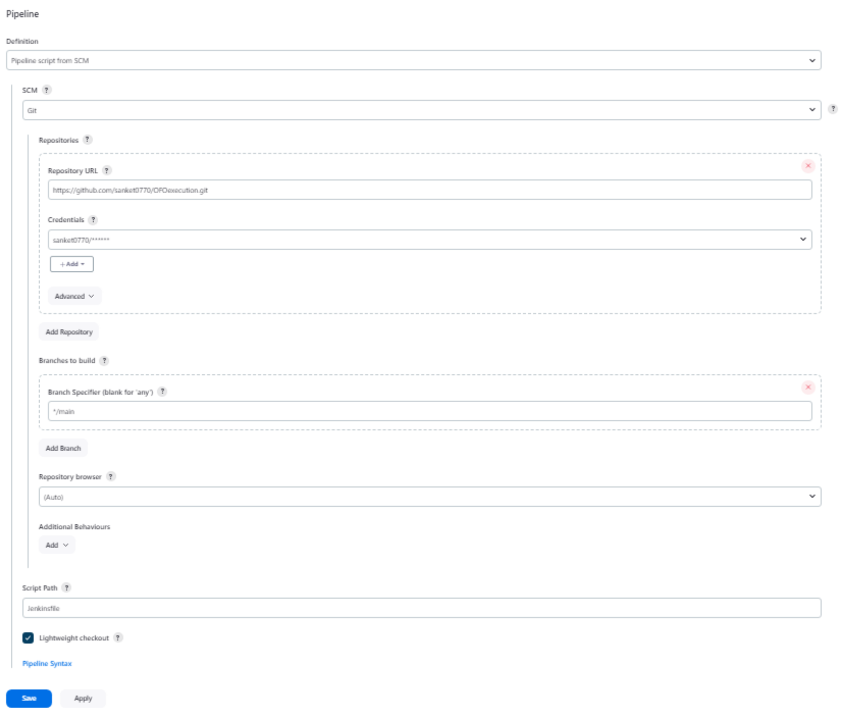
6. Specify repository URL of GitHub S3 repository

7. Specify GitHub credential, if it is not configured click on "add" and configure the GitHub credentials

8. Specify branch of GitHub repository under branch specifier

9. Specify script path as “Jenkinsfile”

10. Click “Save”


Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/sanket0770/OFOexecution.git

Credentials: sanket0770/******

Branches to build

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

Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

✓ Lightweight checkout

Figure 15: Jenkins Pipeline for Traditional Execution of S3

4 Validation

4.1 Run the pipeline

In case of sequential and parallel pipeline
Go to Jenkins dashboard and open the pipeline which you want to run and click on “Build Now” option

In case of traditional execution
Go to Jenkins dashboard and open the pipeline created for the RDS and click on build now option.