

Day 23 : Jenkins Freestyle Project

This is [#90DaysofDevops](#) challenge under the guidance of [Shubham Londhe](#) sir.

Day 23 TASK

check this for task:

<https://github.com/LondheShubham153/90DaysOfDevOps/blob/master/2023/day23/tasks.md>

What is CI/CD?

- CI or Continuous Integration is the practice of automating the integration of code changes from multiple developers into a single codebase. It is a software development practice where the developers commit their work frequently into the central code repository (Github or Stash). Then there are automated tools that build the newly committed code and do a code review, etc as required upon integration. The key goals of Continuous Integration are to find and address bugs quicker, make the process of integrating code across a team of developers easier, improve software quality and reduce the time it takes to release new feature updates.
- CD or Continuous Delivery is carried out after Continuous Integration to make sure that we can release new changes to our customers quickly in an error-free way. This includes running integration and regression tests in the staging area (similar to the production environment) so that the final release is not broken in production. It ensures to automate the release process so that we have a release-ready product at all times and we can deploy our application at any point in time.

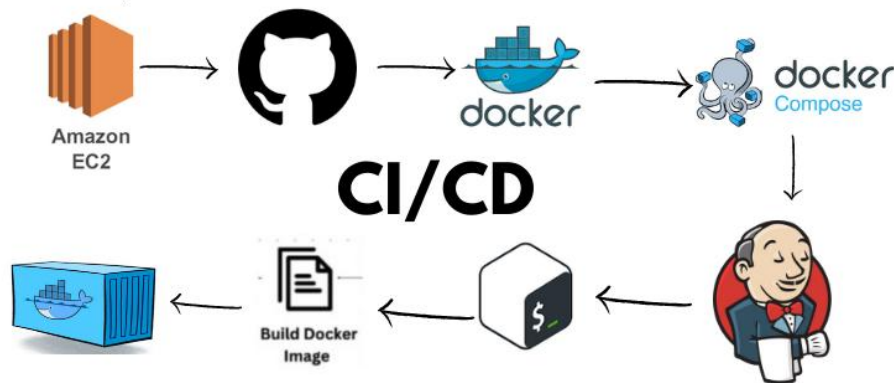
What Is a Build Job?

A Jenkins build job contains the configuration for automating a specific task or step in the application building process. These tasks include gathering dependencies, compiling, archiving, or transforming code, and testing and deploying code in different environments.

Jenkins supports several types of build jobs, such as freestyle projects, pipelines, multi-configuration projects, folders, multibranch pipelines, and organization folders.

What is Freestyle Projects ?

A freestyle project in Jenkins is a type of project that allows you to build, test, and deploy software using a variety of different options and configurations. Here are a few tasks that you could complete when working with a freestyle project in Jenkins:

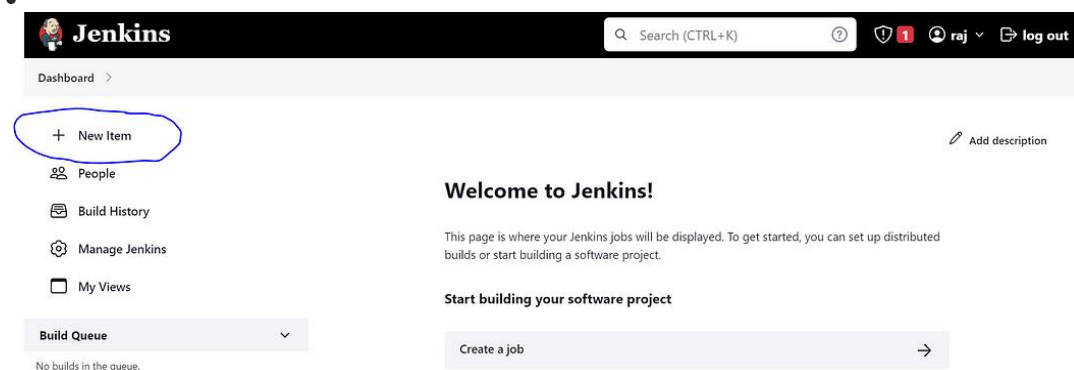


Task-01

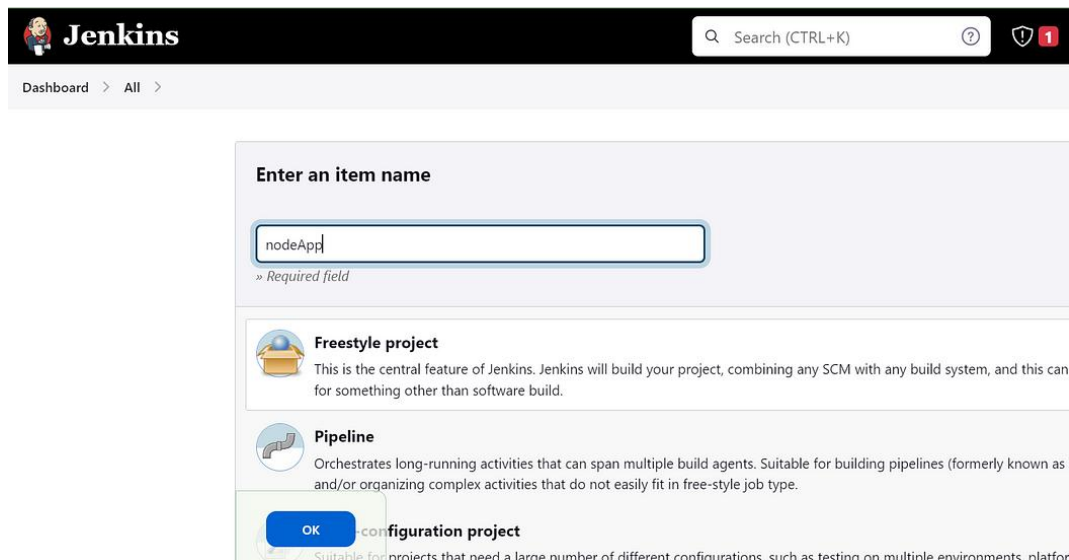
Create a agent for your app.

Create a new Jenkins freestyle project for your app.

- Log in to Jenkins and open the Dashboard
- Create a “New Item”



- Select “Freestyle project” and give name to your project.



The screenshot shows the Jenkins dashboard with a search bar and navigation links. The main content area is titled "Enter an item name" and contains a text input field with "nodeApp" entered. Below the input field is a "Required field" message. There are three options for creating a new item: "Freestyle project", "Pipeline", and "Configuration project". Each option has a brief description and an icon. The "Configuration project" option is highlighted with a blue border and a blue "OK" button.

Enter an item name

nodeApp

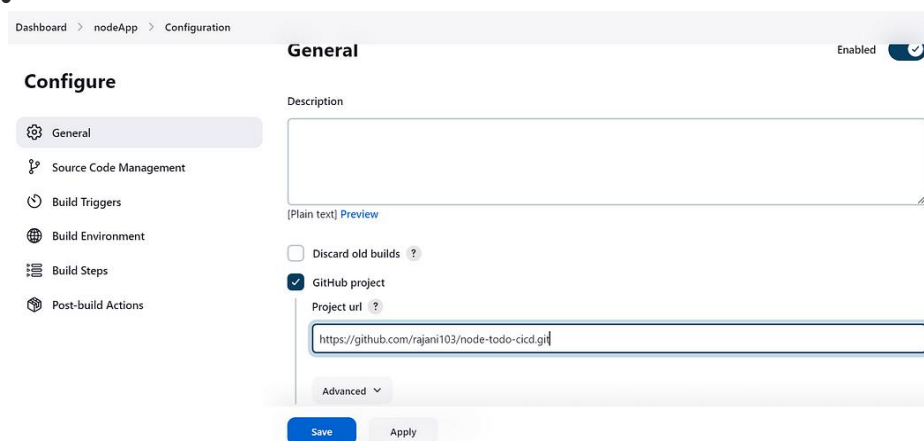
» Required field

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 used for something other than software build.

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.

Configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform

- Click on the “OK” button to create the project.
- In the project configuration, put the details of the project, such as the source code management system, build triggers, and build actions. In GitHub project put your GitHub project repository URL.



The screenshot shows the Jenkins configuration page for the "nodeApp" project. The "General" tab is selected, and the "Enabled" checkbox is checked. The "Description" field is empty. The "Discard old builds" checkbox is unchecked. The "GitHub project" checkbox is checked. The "Project url" field contains the URL "https://github.com/rajani103/node-todo-cicd.git". The "Advanced" section is collapsed. The "Save" and "Apply" buttons are at the bottom.

Dashboard > nodeApp > Configuration

General Enabled

Description

[Plain text] Preview

☐ Discard old builds ?

☒ GitHub project

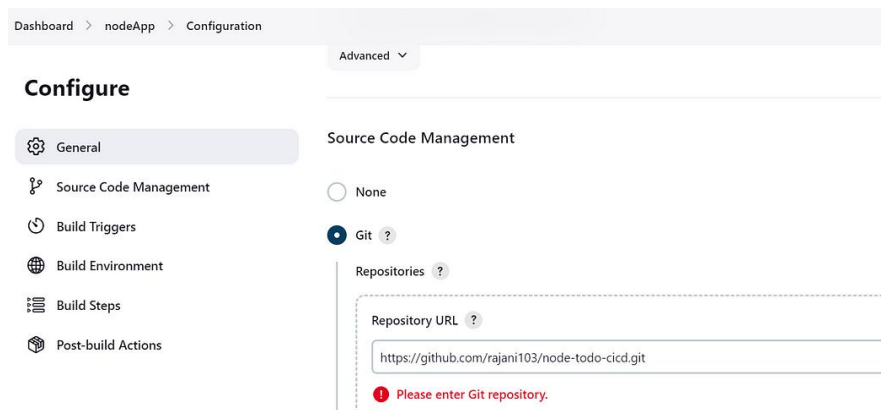
Project url ?

https://github.com/rajani103/node-todo-cicd.git

Advanced

Save Apply

In the “Source Code Management” section, add your repository Link.



The screenshot shows the Jenkins configuration page for the "nodeApp" project, specifically the "Source Code Management" section. The "Advanced" section is expanded. The "Source Code Management" section has two options: "None" and "Git". The "Git" option is selected. The "Repositories" section is expanded, showing a "Repository URL" field with the URL "https://github.com/rajani103/node-todo-cicd.git". A red error message "Please enter Git repository." is displayed below the field.

Dashboard > nodeApp > Configuration

Advanced

Configure

General

Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/rajani103/node-todo-cicd.git

Please enter Git repository.

Specify branch where the source code is present.

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/master

Add Branch

- In the “Build” section of the project, add a build step to run the “docker build” command to build the image for the container.

Dashboard > nodeApp > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Build Steps

Execute shell ?

Command

See [the list of available environment variables](#)

```
docker build -t nodeapp .
```

- Add a second step to run the “docker run” command to start a container using the image created in step 3.

Dashboard > nodeApp > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Build Steps

Execute shell ?

Command

See [the list of available environment variables](#)

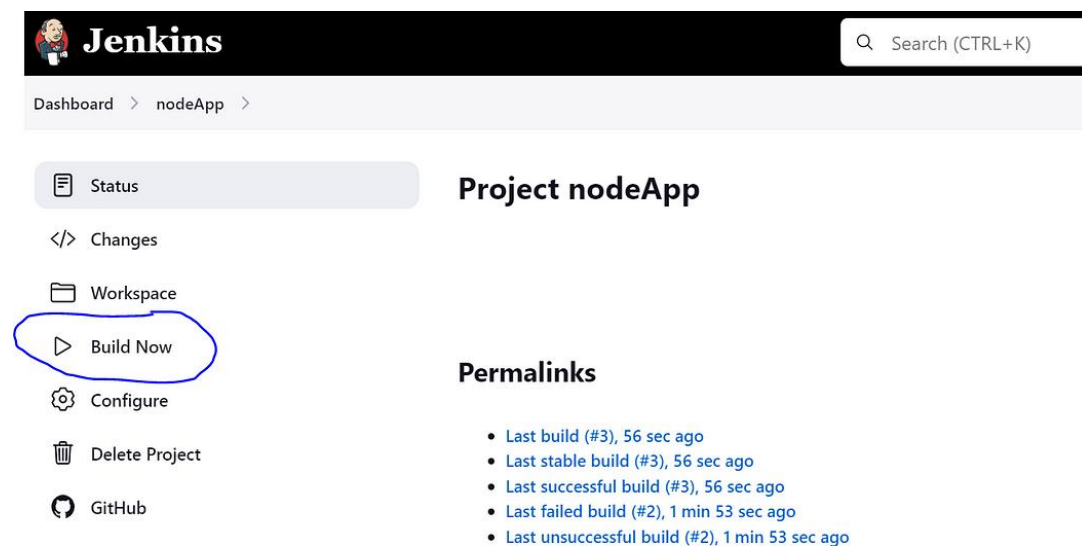
```
docker build -t nodeapp .  
docker run -d --name nodeapp -p 8000:8000 nodeapp:latest
```

```
docker build -t nodeapp .  
docker run -d --name nodeapp -p 8000:8000 nodeapp:latest
```

Save the project configurations by clicking on the save button.

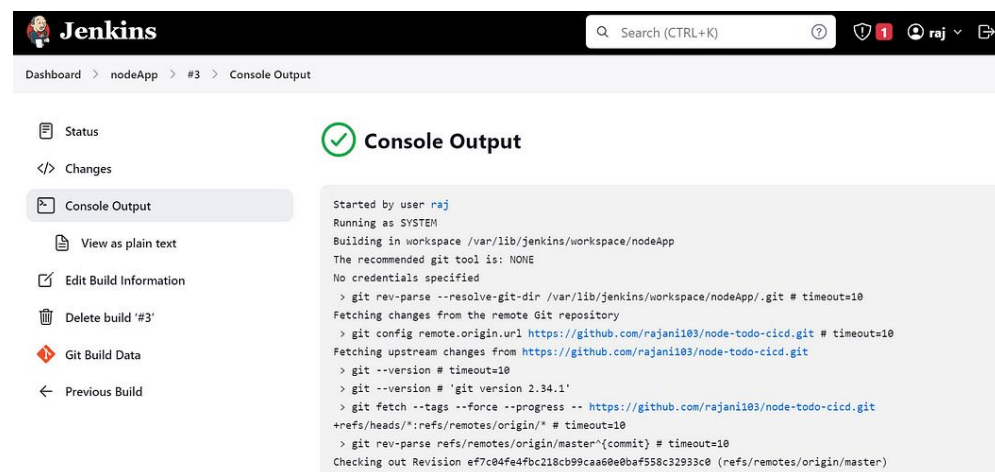
Now you can start the build process of the project by manually clicking on the “build now” tab.

This will start the build process and execute the steps specified in the project configuration (execute shell).



The screenshot shows the Jenkins web interface for a project named 'nodeApp'. The left sidebar contains a menu with options: Status, Changes, Workspace, Build Now (highlighted with a blue circle), Configure, Delete Project, and GitHub. The main area is titled 'Project nodeApp' and includes a 'Permalinks' section with a list of build links: Last build (#3), Last stable build (#3), Last successful build (#3), Last failed build (#2), and Last unsuccessful build (#2).

Once the build is successful, you can go to console output and check the output of the build.



The screenshot shows the Jenkins web interface for the 'nodeApp' project, specifically the 'Console Output' view. The left sidebar shows the 'Console Output' option selected. The main area displays the build output, which includes the following text:

```
Started by user raj  
Running as SYSTEM  
Building in workspace /var/lib/jenkins/workspace/nodeApp  
The recommended git tool is: NONE  
No credentials specified  
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/nodeApp/.git # timeout=10  
Fetching changes from the remote Git repository  
> git config remote.origin.url https://github.com/rajani103/node-todo-cicd.git # timeout=10  
Fetching upstream changes from https://github.com/rajani103/node-todo-cicd.git  
> git --version # timeout=10  
> git --version # 'git version 2.34.1'  
> git fetch --tags --force --progress -- https://github.com/rajani103/node-todo-cicd.git  
+refs/heads/*:refs/remotes/origin/* # timeout=10  
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10  
Checking out Revision ef7c84fe4fbc218cb99caa6e0ba558c32933c0 (refs/remotes/origin/master)
```

```
Dashboard > nodeApp > #3 > Console Output

#8 5.855
#8 6.087 npm WARN my-todolist@0.1.0 No repository field.
#8 6.087 npm WARN my-todolist@0.1.0 No license field.
#8 6.088
#8 6.090 added 291 packages from 653 contributors and audited 291 packages in 5.327s
#8 6.091 found 7 vulnerabilities (1 moderate, 4 high, 2 critical)
#8 6.091 run 'npm audit fix' to fix them, or 'npm audit' for details
#8 DONE 6.6s

#9 exporting to image
#9 exporting layers
#9 exporting layers 0.8s done
#9 writing image sha256:99a0a2625235d31a3bd4ee47fb8baa69a570fab9d2a9d79c0ee70c04e33c24b9 done
#9 naming to docker.io/library/nodeapp done
#9 DONE 0.8s
+ docker run -d --name nodeapp -p 8000:8000 nodeapp:latest
96437b9e9b4754dcf031c93289330f59b8a6c5c0dffe8e375dd06cfc780d19b2
Finished: SUCCESS
```

You can also go to the EC2 instance and check that the source code is been cloned on the VM and you can check the same by going to this location :

/var/lib/jenkins/workspace

```
aws IAM AWS Amplify Lambda EC2 S3 VPC RDS DynamoDB
root@ip-172-31-16-214:/var/lib/jenkins/workspace# ls
nodeApp
root@ip-172-31-16-214:/var/lib/jenkins/workspace# cd nodeApp/
root@ip-172-31-16-214:/var/lib/jenkins/workspace/nodeApp# ls
Dockerfile README.md app.js package-lock.json package.json views
root@ip-172-31-16-214:/var/lib/jenkins/workspace/nodeApp# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
96437b9e9b47   nodeapp:latest "node app.js"           2 minutes ago Up 2 minutes   0.0.0.0:8000->8000/tcp, :::8000->8000/tcp   nodeapp
root@ip-172-31-16-214:/var/lib/jenkins/workspace/nodeApp# docker images
REPOSITORY    TAG         IMAGE ID      CREATED      SIZE
nodeapp       latest     99a0a2625235  2 minutes ago 104MB
root@ip-172-31-16-214:/var/lib/jenkins/workspace/nodeApp#
```

Also you can see image has been created and a container has been launched using the image that has been built, using the commands in the execute shell.

Try accessing the application using the public IP and the 8000 port on the browser.



Task-02

- Create Jenkins project to run “docker-compose up -d” command to start the multiple containers defined in the compose file.

- Create docker-compose.yml file inside your project folder.

```
root@ip-172-31-16-214:/var/lib/jenkins/workspace/nodeApp# cat docker-compose.yml
version : "3.3"
services:
  web:
    build : .
    ports:
      - "8001:8000"
  db:
    image: mysql
    ports:
      - "3306:3306"
```

- In the “Build” section of the project, add a build step “docker-compose down” command to stop and remove the containers defined in the compose file, then add “docker-compose up -d” command.
- Set up a cleanup step in the Jenkins project to run “docker-compose down” command to stop and remove the containers defined in the compose file.

Dashboard > nodeApp > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps

Execute shell ?

Command

See [the list of available environment variables](#)

```
docker-compose down
docker-compose up -d
```

Save the configurations and build the project.

Jenkins Search (CTRL+K) ? ? ?

Dashboard > nodeApp > #5 > Console Output

- Status
- Changes
- Console Output
- View as plain text
- Edit Build Information
- Git Build Data
- Previous Build

Console Output

```
Started by user raj
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/nodeApp
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/nodeApp/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/rajani103/node-todo-cicd.git # timeout=10
Fetching upstream changes from https://github.com/rajani103/node-todo-cicd.git
> git --version # timeout=10
> git --version # 'git version 2.34.1'
> git fetch --tags --force --progress -- https://github.com/rajani103/node-todo-cicd.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
```

```
Dashboard > nodeApp > #5 > Console Output

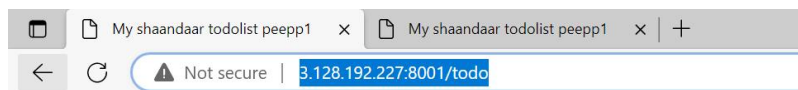
#9 exporting to image
#9 exporting layers
#9 exporting layers 0.8s done
#9 writing image sha256:0ac1b1993c93f813aad593a977078b613a55a653bd4148ae28ff00ba1c8817f3 done
#9 naming to docker.io/library/nodeapp_web done
#9 DONE 0.8s

Image for service web was built because it did not already exist. To rebuild this image you must use `docker-
compose build` or `docker-compose up --build`.
Pulling db (mysql:...)...
latest: Pulling from library/mysql
Digest: sha256:e8dc80474af0e0c48d55742815da52073f466059648738e648c9dd262ff51db
Status: Downloaded newer image for mysql:latest
Creating nodeapp_web_1 ...
Creating nodeapp_db_1 ...
Creating nodeapp_web_1 ... done
Creating nodeapp_db_1 ... done
Finished: SUCCESS
```

You can check in the instance that the containers has been created.

```
root@ip-172-31-16-214:/var/lib/jenkins/workspace/nodeApp# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
ae51826e13ec  nodeapp_web    "node app.js"           About a minute Up About a minute 0.0.0.0:8001->8000/tcp, :::8001->8000/tcp  nodeapp_web_1
96437b9e9b47  nodeapp:latest "node app.js"           19 minutes ago Up 19 minutes  0.0.0.0:8000->8000/tcp, :::8000->8000/tcp  nodeapppp
```

Now you can try accessing the application on browser.



hi swathi-123

What should I do?

Please, feel free to drop any questions in the comments below. I would be happy to answer them.

If this post was helpful, please do follow and click the clap

_Thank you for reading

_Rajani