

Varshni D S

05.02.2025

Review of Jenkins Pipeline, Docker, and GitHub Repository Setup

1. Jenkins Dashboard & Job Creation

The first section displays the Jenkins dashboard, illustrating how to create a new job. The available options include:






- **Freestyle Project:** A flexible choice allowing various build steps and configurations.
- **Maven Project:** Uses POM files for efficient builds.
- **Multi-Configuration Project:** Supports projects requiring multiple configurations, such as matrix builds.
- **Pipeline Project:** Implements CI/CD automation using scripted or declarative pipelines.

Dashboard > All > New Item

Enter an item name

Task_2

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.
-  **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.
-  **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

OK

2. Jenkins Configuration Options

The second section highlights the Jenkins pipeline job configuration. Key settings include:

- **Pipeline Speed/Durability Override:** Adjusts execution priority and resource allocation.
- **Preserve Stashes from Completed Builds:** Retains stashes for debugging purposes.

- **GitHub Triggers:** Supports Webhooks and SCM polling for automatic builds.
- **Throttle Builds:** Restricts concurrent builds to optimize resource usage.

The screenshot shows the Jenkins configuration page for a job named 'Task_2'. The breadcrumb trail is 'Dashboard > Task_2 > Configuration'. The page has a dark header with the Jenkins logo and user information. The left sidebar shows the configuration menu with 'General' selected. The main content area is divided into two sections: 'General' and 'Triggers'.

General Section:

- Description:** A text area containing the word 'done'.
- Plain text / Preview:** A toggle switch set to 'Preview'.
- Discard old builds:** A checkbox that is unchecked.
- Do not allow concurrent builds:** A checkbox that is unchecked.
- Do not allow the pipeline to resume if the controller restarts:** A checkbox that is unchecked.
- GitHub project:** A checkbox that is checked.
- Project url:** A text field containing 'Project url'.
- Buttons:** 'Save' and 'Apply' buttons.

Triggers Section:

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- Build after other projects are built:** A checkbox that is unchecked.
- Build periodically:** A checkbox that is unchecked.
- GitHub hook trigger for GITScm polling:** A checkbox that is checked.
- Poll SCM:** A checkbox that is unchecked.
- Trigger builds remotely (e.g., from scripts):** A checkbox that is unchecked.

Pipeline Section:

Define your Pipeline using Groovy directly or pull it from source control.

Definition: A dropdown menu showing 'Pipeline script from SCM'.

SCM: A dropdown menu showing 'Git'.

Buttons: 'Save' and 'Apply' buttons.

3. Pipeline Script & Source Control Management

This section displays:

- **Pipeline Script from SCM:** Jenkins fetches the pipeline script from a Git repository.
- **Repository URL:** `https://github.com/varshnids/capstone.git`.
- **Branch Specification:** Configured to main, ensuring Jenkins builds this branch.

- **Jenkinsfile Usage:** Defines pipeline stages and execution steps.

Dashboard > Task_2 > Configuration

Define your Pipeline using Groovy directly or pull it from source control.

Configure

- General
- Triggers
- Pipeline**
- Advanced

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/Varshnids/capstone.git

Credentials ?

- none -

+ Add

Save Apply

Dashboard > Task_2 > Configuration

Configure

- General
- Triggers
- Pipeline**
- Advanced

Repository URL ?

https://github.com/Varshnids/capstone.git

Credentials ?

- none -

+ Add

Advanced

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/main

Add Branch

Save Apply

4. Docker Hub Repository Overview

The fourth section provides details about the Docker Hub repository (varshni057/sample). Notable aspects include:

- **Public Repository:** Accessible to all users.

Pushing Docker Images: Uses CLI commands:

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```

- **Automated Builds:** Enables GitHub/Bitbucket integration for automatic image builds upon code updates.

The screenshot shows the 'Create repository' page on Docker Hub. The top navigation bar includes 'dockerhub', 'Explore', 'Repositories' (active), 'Organizations', and 'Usage'. A search bar and user profile icon are on the right. Below the navigation, the breadcrumb 'Repositories / Create' is shown, along with the text 'Using 0 of 1 private repositories.'.

Create repository

Namespace: Repository Name:

Short description:

A short description to identify your repository. If the repository is public, this description is used to index your content on Docker Hub and in search engines, and is visible to users in search results.

Visibility

Using 0 of 1 private repositories. [Get more](#)

☒ **Public** Appears in Docker Hub search results ☐ **Private** Only visible to you

Pushing images

You can push a new image to this repository using the CLI:

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```

Make sure to replace `tagname` with your desired image repository tag.

The screenshot shows the repository page for 'varshni057/sample'. The top navigation bar is the same as the previous screenshot. The breadcrumb is 'varshni057 / Repositories / sample / General'.

varshni057/sample 🔗

Last pushed about 6 hours ago • Repository size: 69.4 MB

[Add a description](#) 🔗 ℹ️

[Add a category](#) 🔗 ℹ️

General | Tags | Builds | Collaborators | Webhooks | Settings

Tags

This repository contains 1 tag(s).

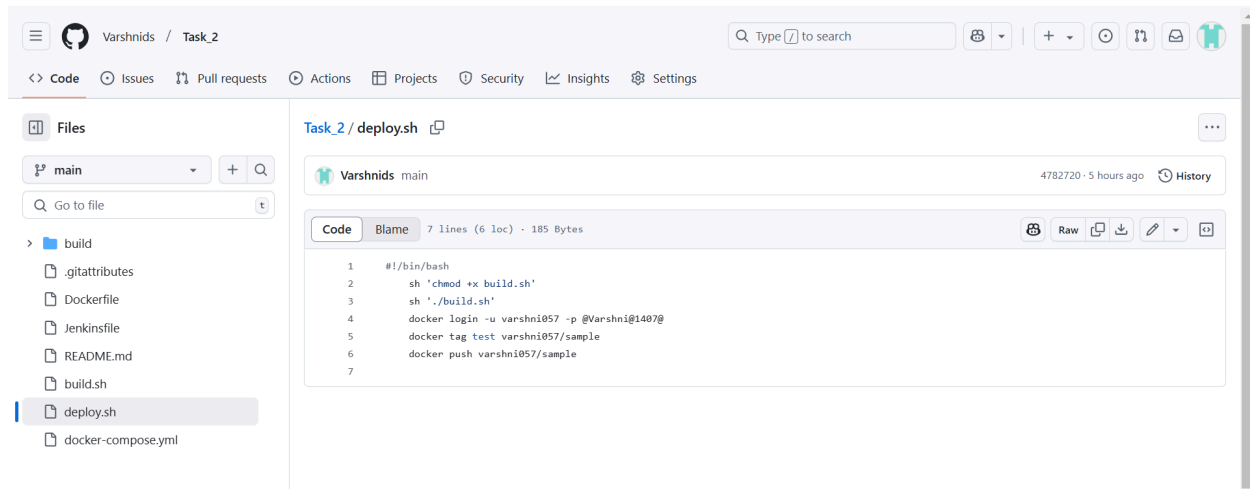
Tag	OS	Type	Pulled	Pushed
latest		Image	6 hours ago	6 hours ago

5. GitHub Repository & Deployment Workflow

This section highlights:

- **Repository Files:**
 - Dockerfile: Defines the containerized environment.

- Jenkinsfile: Contains CI/CD pipeline definitions.
- build.sh: Script for project build execution.
- deploy.sh: Script for application deployment.
- docker-compose.yml: Manages multi-service container applications.



Deployment Commands:

docker login -u username -p password
 docker push varshni057/sample:tagname

6. Jenkins Build Execution Overview

The final section captures a successfully executed Jenkins pipeline:

- **Build Number:** #1, executed on **Feb 5, 2025**.
- **Repository Revision:** 3744f0a73044a984a5951b7530715b70ccbdees.
- **Execution Duration:** 29 seconds.
- **Console Output:** Displays detailed logs of the build process.

Dashboard > Task_2 > #4

Status

</> Changes

Console Output

Edit Build Information

Delete build '#4'

Timings

Git Build Data

Pipeline Overview

Pipeline Console

Restart from Stage

Replay

Pipeline Steps

Workspaces

Previous Build

✓ #4 (05-Feb-2025, 10:51:15 am)

Add descriptionKeep this build forever

Started by user User

Started 5 hr 48 min ago
Took 20 sec

This run spent:

- 25 ms waiting;
- 20 sec build duration;
- 20 sec total from scheduled to completion.

git

Revision: e3a7577a3e12ca907dac5fd275469c9d530e2e3a
Repository: <https://github.com/Varshnids/capstone.git>

- refs/remotes/origin/main

</> No changes.

dockerhub

ExploreRepositoriesOrganizationsUsage

Search Docker Hub

ctrl+K

?

⚙️

🔔

🔄

⋮

V

varshni057 / [Repositories](#) / [sample](#) / [General](#) Using 0 of 1 private repositories.

varshni057/sample

Last pushed about 5 hours ago · Repository size: 69.4 MB

Add a descriptionAdd a category

Docker commands

To push a new tag to this repository:

docker push varshni057/sample:tagname

Public view

GeneralTagsBuildsCollaboratorsWebhooksSettings

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest	🐧	Image	6 hours ago	5 hours ago

See all

Automated builds

Manually pushing images to Docker Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. [Read more about automated builds](#)

Conclusion

The provided images illustrate a structured CI/CD pipeline integrating Jenkins, GitHub, and Docker. The workflow follows these steps:

1. **Defining a Jenkins Pipeline (Jenkinsfile).**
2. **Managing Source Code in GitHub.**
3. **Using Jenkins for Automated Builds.**
4. **Pushing Docker Images to Docker Hub.**
5. **Executing CI/CD Pipelines Efficiently.**

This setup enhances software development automation and streamlines deployment processes.