**CI/CD (Continuous Integration and Continuous Deployment/Delivery):**

It is a backbone of DevOps

**Scenario:** A company wants to ensure that any code changes made by developers are automatically tested and deployed.

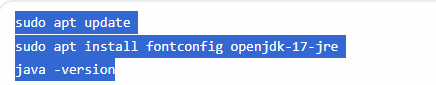
**CI: -** Once Developer push their code that code should be tested then security Check then application build and it should be ready to deploy called Continuous Integration.

**CD (Continuous Deployment): -** Once CI is completed, automatically triggered called continuous Deployment.

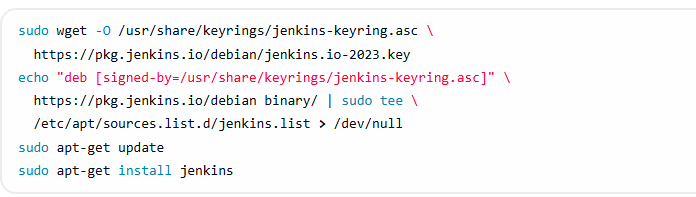
**CD (Continuous Delivery): -** Once CI is completed, we need manually triggered called continuous Deployment.

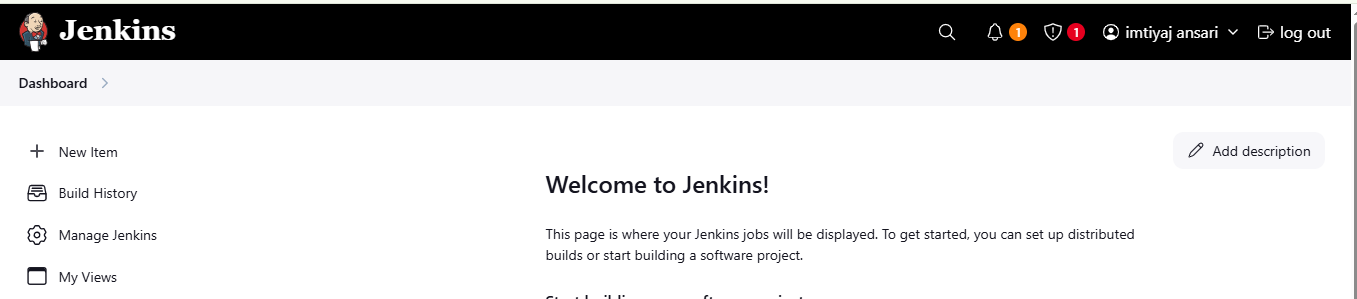
**Installation of Jenkins:-**

**First Install Java…….**



**Now Install Jenkins……**





**Discard old builds: -** Automatically removes older build records to save storage space.

**Do not allow concurrent builds: -** Prevent multiple builds from running simultaneously to avoid conflicts and ensure consistency.

**Do not allow the pipeline to resume if the controller restarts: -** Ensures a pipeline won't continue after a Jenkins controller reboot, forcing a fresh start.

**GitHub project: -** Specifies the GitHub repository from which the pipeline retrieves its source code.

**Pipeline speed/durability override: -** Adjust pipeline execution settings to prioritize either faster runs or higher durability.

**Preserve stashes from completed builds: -** Keeps stashed data (temporary files) even after a build finishes.

**This project is parameterized: -** Enables the pipeline to accept user-defined input values during execution.**Throttle builds: -** Limits the rate at which builds are executed to manage system load.

**Triggers:-**

**Build after other projects are built: -** Triggers a build when another specified Jenkins project completes successfully.

**Build periodically: -** Schedules builds to run at regular intervals, defined by a cron expression.

**GitHub hook trigger for GITScm polling: -** Automatically triggers a build when GitHub detects a push to the repository.

**Poll SCM: -** Regularly checks the source code management (SCM) system for changes and triggers a build if any are found.

**Trigger builds remotely (e.g., from scripts): -** Allows builds to be initiated via external scripts or tools using a specific URL and authentication.

**Create Demo pipeline…**

