NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
08258 - 281039 - 281263, Fax: 08258 - 281265

Department of Computer Science and Engineering

REPORT ON

Jenkins

Course Code: 20CS701

Academic Year – 2023-2024

Course Name: **SOFTWARE TESTING**

Semester: 7 Section : C

Submitted To,
Course Instructor:
Dr. Shabari Shedthi B.
Associate Professor

Submitted By:

1. USN: 4NM20CS122 NAME: Nischitha Shetty SIGN:
2. USN: 4NM20CS139 NAME: Prathiksha Kini SIGN:

Date of submission:
02/11/2023

Introduction

Jenkins is an open source automation tool written in Java programming language that allows continuous integration. Jenkins **builds** and **tests** our software projects which continuously makes it easier for developers to integrate changes to the project, and makes it easier for users to obtain a fresh build.

Continuous Integration (CI) is a software development practice where code changes from multiple contributors are frequently integrated into a central repository. Each integration triggers automated builds and tests, allowing teams to detect and address integration issues early in the development process.

Why Jenkins?

- **Integration and Extensibility:** Jenkins supports a vast number of plugins, which allows it to integrate with various testing frameworks, version control systems, and other tools. This extensibility makes it a versatile choice for automation testing, as you can easily adapt it to your specific needs.
- **Automation Pipelines:** Jenkins allows you to create complex automation pipelines, which can include multiple stages for building, testing, and deploying your application.
- **Scheduled and Triggered Builds:** Jenkins can be configured to run tests automatically based on triggers, such as code commits to a version control repository or on a predefined schedule. This automation ensures that tests are consistently executed without manual intervention.
- **Scalability:** You can distribute tests across multiple nodes to parallelize test execution, which can significantly reduce the time required to complete test suites.
- **Reporting and Notifications:** Jenkins provides comprehensive reporting capabilities, which allow you to analyze the results of your tests. It can also send notifications and alerts in case of test failures, making it easier to identify and address issues promptly.

- **Version Control Integration:** Jenkins seamlessly integrates with version control systems like Git, allowing you to trigger tests automatically when changes are pushed to the repository.

Advantages

- Jenkins is an open source tool with good community support.
- Jenkins has 1000+ plugins to ease your work. If a plugin does not exist for your requirement, you can code it and share with the community.
- This tool is free of cost and easy to install.
- It can execute bash scripts, shell scripts, ANT and Maven Targets.
- It also provides support for scheduled builds & automation test execution.

Disadvantages

- Not easy to maintain because it runs on a server and requires some skills as server administrator to monitor its activity.
- CI regularly breaks due to some small setting changes. CI will be paused and therefore requires some developer's team attention.

Demonstration

Once the Jenkins Installation is successful navigate to the url localhost:8080 or any other port number which you have specified during the Jenkins installation.

Let’s create a simple job/project in the Jenkins environment and then integrate it with a Version Control System like Github.

- Click on create a new item in the left pane of the Jenkins dashboard. (Refer Fig1)
- Enter the item name and select either Free Style Project or Pipeline as the Project type and then click on OK.

- Now you will be directed to the configuration page of the Jenkins instance which you have created.
- Scroll down to find source code management.
- Select Git as the source code management tool and add the repository url as Jenkins-Demo
- In the branches to build type */main as we want to pull the code and automate the build process in the main branch.
- Scroll down to find the build triggers and tick the checkbox “Build Periodically”.
- Specify the duration after which you want to automatically build your code.
- In my case I have specified it as 3 min (ie; After every 3 min it will pull my code from repo, build the code and test it).
- To do so, specify */3 * * * * in the textbox provided (Click on the que mark for more info about cron jobs).
- Scroll down and select Execute Windows Batch Command in Build Steps.
- I have written very simple lines of script for the demonstration. Write the script which you want to get automated.

```
javac Hello.java

if %errorlevel% EQU 0 (
          java Hello
) else (
     echo Build unsuccessful due to compilation errors
     exit /b %errorlevel%
)
```

- Once done click on save and then click on build now in order to build your project
- You will get either success or failure as the console output after the build is complete
- And the cycle continues for the duration you have configured for the job

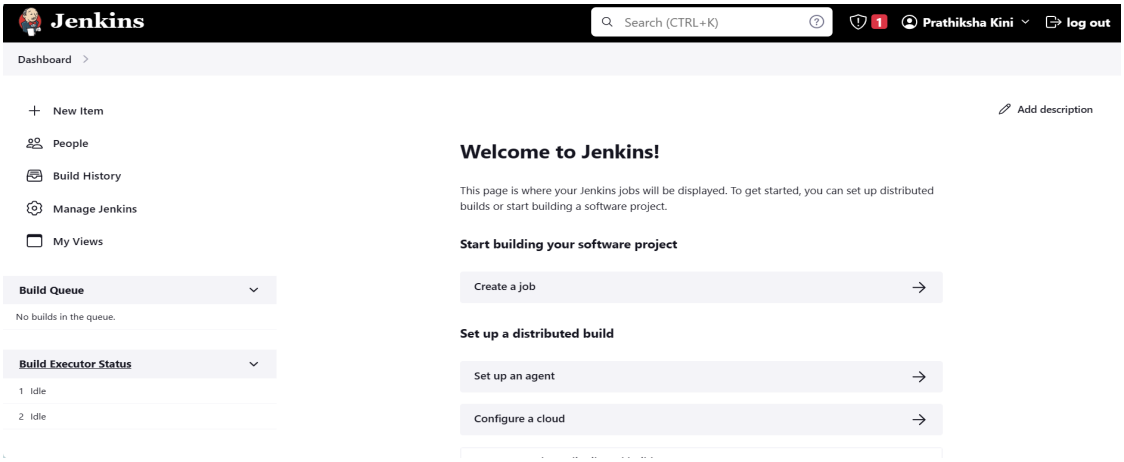
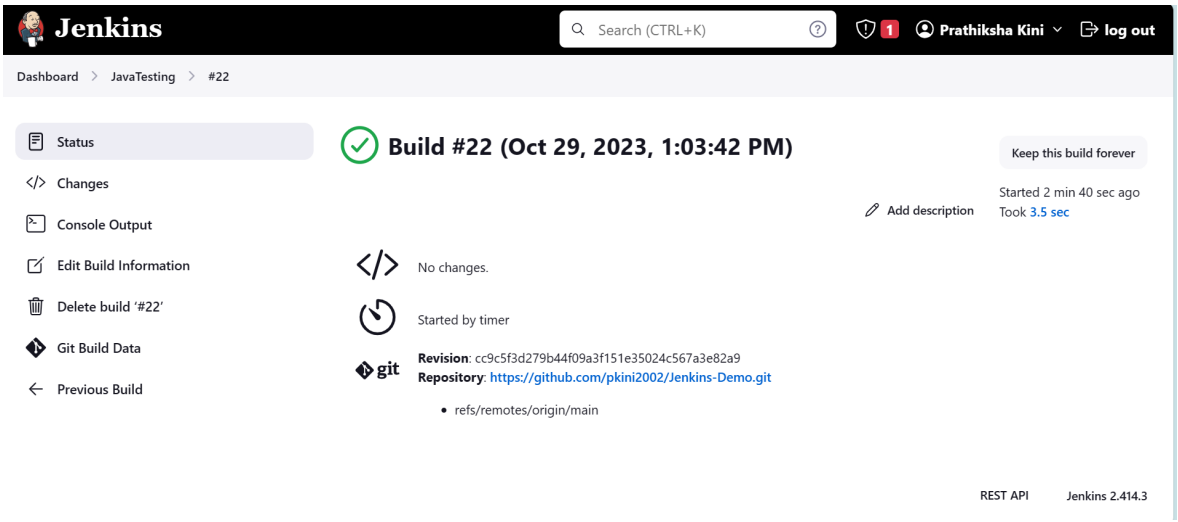
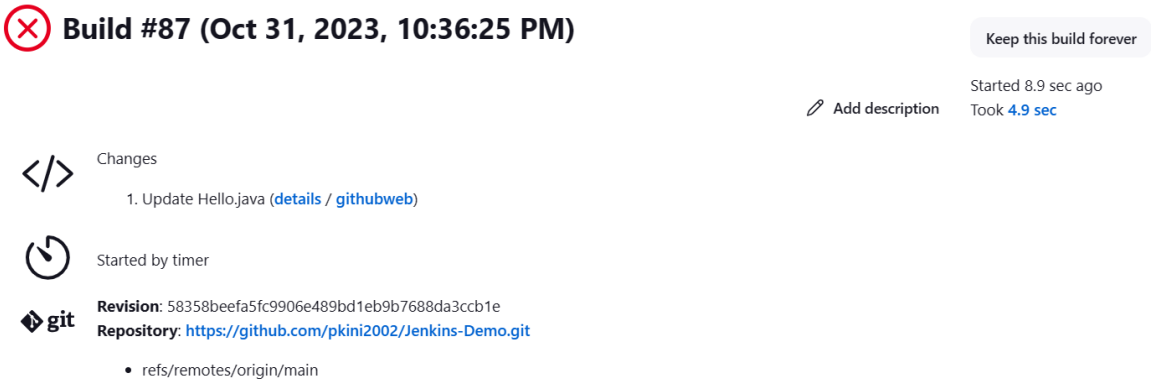


Fig1: Jenkins Dashboard



Testcase1: Correct code, returns exit status = 0



Test case 2: Incorrect code, returns exit status = 1

Conclusion

In conclusion, Jenkins stands as a powerful ally in the realm of software development, embodying the essence of efficiency, collaboration, and reliability. Through its seamless automation of build, test, and deployment processes, Jenkins not only ensures the rapid integration of code changes but also cultivates a culture of continuous improvement within development teams.

References

[1] Jenkins Tutorial- javapoint: https://www.javatpoint.com/jenkins

[2] Jenkins Full course in 5 hours [2023] | Jenkins Tutorial for Beginners | DevOpsTraining | Edureka: https://www.youtube.com/watch?v=QIhRgW7OhPY