

End to End DevOps Pipeline

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering Degree in Computer Science and Engineering

By

S Praveen Daniel
(Reg.No.38110427)

Balaji V (Reg.No.38110069)



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA**

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119**

MARCH – 2022



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **S Praveen Daniel (38110427) and Balaji V (38110069)** who carried out the project entitled "**End to End DevOps Pipeline**" under my supervision from November 2021 to March 2022.

Internal Guide

Mrs. Srividhya S.R.

Head of the Department

Dr. L. Lakshmanan, M.E., Ph.D.

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

We **S Praveen Daniel, Balaji V** hereby declare that the Project Report entitled **End to End DevOps Pipeline** done by us under the guidance of Mrs. **Srividhya S.R ME.** (Internal) is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering / Technology degree in Computer Science and Engineering.

DATE:

**S Praveen Daniel
Balaji V**

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing **Dr. L. Lakshmanan M.E., Ph.D.**, and **Dr.S. Vigneshwari M.E., Ph.D.** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mrs. Srividhya S.R ME.** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project

ABSTRACT

Updates and upgrades are vital for coordinated effort. Online destinations and different internet-based media should be more associated with the new change and the board. Late advances in cloud innovation have given advantages like accessibility, adaptability, and minimal expense, however purchaser life execution has been delayed because of these deficient changes. To speed up wellbeing the executives, we really want to get comfortable with the DevOps culture. DevOps is the primary course to send off a product advancement program (Dev) that coordinates data innovation (Ops), lessening programming improvement life, while continually conveying content, revisions, and updates that are firmly connected with business goals. Constant Integration (CI) and Continuous Delivery (CD) are intended to assist shoppers with working on their exhibition and furnish administration preparing determined to enable clients to constantly convey the best of both the old and the norm. The motivation behind this article is to work on the presentation of the DevOps network beginning to end, assisting us with moving quicker and fix new bugs and bugs in the startup interaction.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No
	ABSTRACT	i
	LIST OF FIGURES	iv
1	INTRODUCTION	1
	1.1 Outline of the Project	1
	1.2 Scope of The Project	1
	1.3 Existing system	2
	1.4 Proposed System	2
2	LITERATURE SURVEY	3
3	METHODOLOGY	5
	3.1 General Overview	5
	3.2 DevOps Tools Used	5
	3.3 System Design	10
	3.4 Pipeline Architecture	10
	3.5 Pipeline as Code	11
	3.6 Continuous Delivery with Pipeline	13
	3.7 DevOps Monitoring	18
4	RESULTS AND DISCUSSION	21
5	CONCLUSIONS AND FUTURE WORK	22

APPENDIX	23
SCREENSHOTS	23
PUBLICATION AND PLAGIARISM REPORT	26
REFERENCE	28

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
1	Jenkins Dashboard	7
2	Architecture diagram	10
3	Throttled stage concurrency with Pipeline	16
4	Result comparison of various CI/CD tools	21

CHAPTER 1

INTRODUCTION

1.1 OUTLINE OF THE PROJECT

Utilizing DevOps rapidly, the advancement group should organize with the improvement group in an exceptional manner to foster the product. These courses assist you with getting the limits and prerequisites of the product along with the reason for the business. DevOps (Development + Operations) is a culture that consolidates estimation and activity to conquer all snags inside the improvement group of innovation exercises. The principle reason for this culture is to join the two gatherings for the execution and quick improvement of programming. Do one-on-one gatherings on DevOps; Development is only a work underway. This incorporates programming advancement, use, and reconciliation of various programming parts. DevOps upholds Agile circulation in appropriation. Definition and design together. It is essential to share this archive, as it is vital to isolate the subtleties of the continuous turn of events and coordination of things to accelerate the entire cycle.

1.2 SCOPE OF THE PROJECT

Deliver updates and upgrades fast and efficiently by combining various DevOps tools and techniques. By implementing the system can get faster delivery cycles enabling the organization to patch bugs and upgrades quickly satisfying the customer needs.

1.3 EXISTING SYSTEM

We already have DevOps systems implemented in various organisations which speed up their development cycle drastically, but most systems are extremely expensive and complex to implement. These expensive and complex systems usually require a team of several DevOps engineers to maintain the system. Else, the system becomes unstable and will do more harm than helping. Additionally, the systems are close sourced.

1.4 PROPOSED SYSTEM

Our proposed system is built upon various DevOps tools such as Git, Jenkins, Maven, Ansible, Zabbix. When the developer commits a code update, a Jenkins job will be triggered and the automation process starts. First, the code gets copied to the building environment. Jenkins will trigger maven to start the building process. After the code built, the application is sent to the testing environment. Jenkins will trigger selenium to start the testing process. If the test failed, Jenkins will report the developers and testers by notifying the error. Else, the application is staged for further tests. If the staging is completed without any errors, the application is ready for production. The updated application is sent to the production server. The application is then continuously monitored using Zabbix monitoring server for performance monitoring. This enables us to achieve continuous development and release cycle. Our system can improve the rate of update cycle drastically.

Since reliability is a concern in the process, we have ensured many fail-safe mechanisms to make the process as reliable as possible. If any error occurred in between, the system will immediately notify the respective department to check the problem. Automating every testing process have made to achieve fast and quality testing results.

CHAPTER 2

LITERATURE SURVEY

As indicated by Mohammed Zaid Abrahams and Joseph S. Langerman, the 2018 investigation is pointed toward building and exhibiting the capacity to coordinate DevOps, rather than making security part of the gig. This implies that the components will show a circle that involves DevOps standards in the information association strategy. The norms are intended to improve and involve the PC in these particular areas of extremely durable security by sticking to speed. The examination will survey and audit documentation connected with consistence with DevOps and Velocity proposals to comprehend research directed by the two schools and organizations. [1]

As concurred with Prashant Agrawal and Neelam Rawat in 2019, the objective of this article is to move DevOps to the Cloud and fill in as a model for program improvement and execution. To be adjusted, contemplate how to expand these DevOps cycles and mechanization as a general rule, add individual mists, and the most ideal ways to test it. In an investigation of the development of DevOps, this paper portrayed it as a coherent change in data innovation to further develop execution. The objective is to perceive the way in which DevOps and Cloud cooperate to assist associations with accomplishing their center objectives. [2]

As per Ruth W. McCarthy and Julian M. Bass, in 2020, this study gives a logical area on the execution of DevOps, featuring the connection among designers and the Ops, Outsourced Ops, DevOps and DevOps gatherings. We present an underlying arrangement on the best way to manage site and cloud-based techniques, and separate DevOps mentors. We likewise distinguished three explicit sorts of preparing in the fourth structure: giving and supporting actual levels, virtualization and systems administration, administration improvement, and joint effort, which might have carried DevOps execution to a more extensive territory. 3]

About Qianying Liao Requirements for quick, productive and successful advancement during get together, testing and conveyance. Along these lines, the ceaseless establishment and constant establishment (CI/CD) technique is clear. It centers around the plan interaction, which can present organization new companies, as well as analyses on the general effort of the CI/CD. The outline is planned for research that permits the model of a direct positioning framework to add to rationale, actuated by human-based demonstrating, as well as to try different things with different setups and conditions. [4]

As indicated by Mohammad Rizki Pratama and Dana Sulistio Kusumo in 2021, the Killing Test is essential for a formative stage test to guarantee that executed administration can screen pre-arranged exercises and separate them from execution issues. These days, with the improvement of agile, it is feasible to accomplish turning work quicker and quicker. Nonstop coordination and consistent conveyance (CI/CD) is a procedure utilized in Agile advancement to speed up and speed up the going with framework: building, testing, and approving administrations. It is wanted to do CI/CD while the analysis is running. The current review needs individuals to lead the review. Our concept of experimenting with CI/CD can be utilized on a PC to diminish the exhibition of human tests. Finishing up a CI/CD for a screening test makes the testing system steady, productive, and now and again tedious. It can likewise speed up the speed of progress and keep specialists from defining new limits for different investigations. [5]

CHAPTER 3

METHODOLOGY

3.1 GENERAL OVERVIEW

Our system is built upon various DevOps tools such as Git, Jenkins, Maven, Ansible, Zabbix. When the developer commits a code update, a Jenkins job will be triggered and the automation process starts. First, the code gets copied to the building environment. Jenkins will trigger maven to start the building process. After the code built, the application is sent to the testing environment. Jenkins will trigger selenium to start the testing process. If the test failed, Jenkins will report the developers and testers by notifying the error. Else, the application is staged for further tests. If the staging is completed without any errors, the application is ready for production. The updated application is sent to the production server. The application is then continuously monitored using Zabbix monitoring server for performance monitoring. This enables us to achieve continuous development and release cycle. Our system can improve the rate of update cycle drastically.

Since reliability is a concern in the process, we have ensured many fail-safe mechanisms to make the process as reliable as possible. If any error occurred in between, the system will immediately notify the respective department to check the problem. Automating every testing process have made to achieve fast and quality testing results.

3.2 DEVOPS TOOLS USED

In this chapter, lets discuss the DevOps Tools used in the system

3.2.1 Git

Git is one of the most broadly utilized information the executive frameworks today. It is right now thought to be an adaptation control. With this technique, experts can change code and track progress while creating programming.

Git can likewise give researchers the opportunity, since they will place it in an astounding circle. You can go to the far-off library utilizing Git. This chronicle is normally deciphered in various ways. Whenever antiques are shipped off the district, a duplicate of the file will be joined to the duplicate toward the finish of the server. Because of its similarity, Git is utilized to oversee forms of our messages.

Indeed, the most broadly involved framework for conveyance is Git. Git is an undeniable open-source project sent off in 2005 by Linus Torvalds, most popular for making Linux in view of trees. An enormous number of Git startup projects are because of Git control, like a business and open-source project. The architects working with Git accentuate on the capacity to work on the application, function admirably on the more extensive work area and IDEs (improvement climate).

The most famous Git is the DVCS picture (later known as the Celebrity Control System). Not at all like an environmental change checking framework like CVS or Subversion (SVN), which was once famous in Git, there is normally one article to show the historical backdrop of the total item, and Git likewise has a duplicate of the code for each undertaking. It might contain the entire story.

3.2.2 Jenkins

Proceeding with coordination and proceeded with conveyance or conveyance (CI/CD) is a significant piece of DevOps culture. CI/CD is really an open server that permits designers to make robots in various pieces of the organization. Jenkins, an

open-source server that keeps on associating with the server, can involve the PC for all product exercises. The way that this gadget permits clients to get to the organization is really astonishing. For this situation, the Jenkins code is utilized to enter the code storehouse, play out a comparative trial, and import the report after the test. It is utilized in customary gatherings and testing of item projects, making it simpler for specialists to change their obligations and making it more straightforward for clients to get another look. This will assist you with continuously furnishing your items with an enormous number of innovative work open doors.

With the assistance of Jenkins, associations can speed up the most common way of advancing items through PCs. Jenkins envelops the improvement of a day-to-day existence that incorporates everything from creation, capacity, trial and error, bundling, characterization, conveyance, and thorough testing.

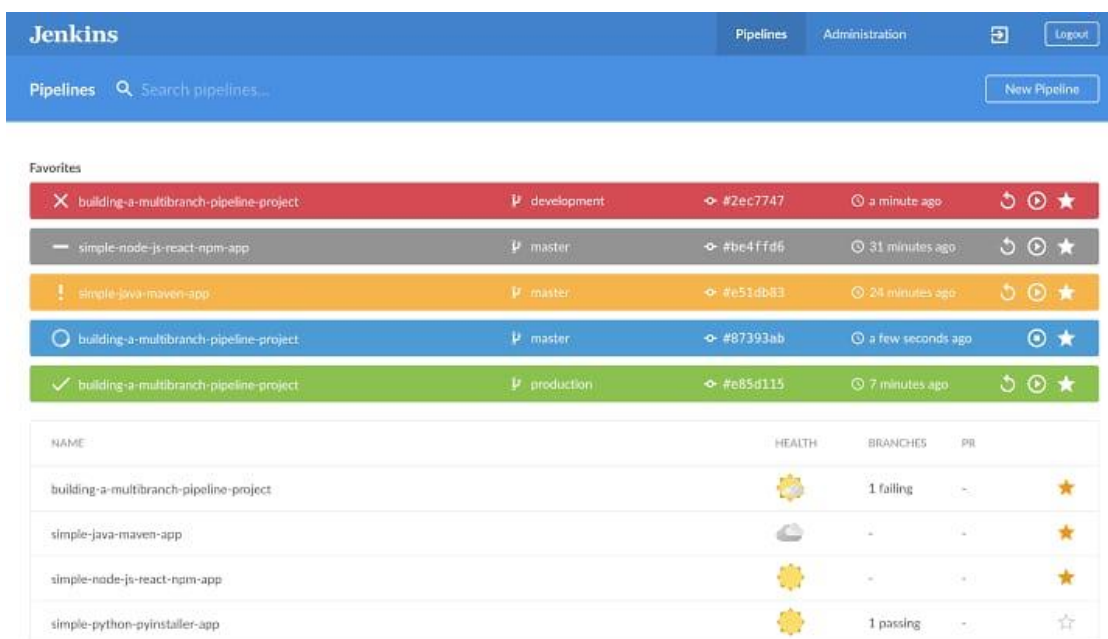


Fig 3.2 Jenkins Dashboard

3.2.3 Maven

Expert is a task the executives apparatus permits designers to make programming items. This is simple with the phone code the foundation of the item is practically general the program takes some time. Use Maven Java-based projects are to a great extent, yet tasks can be made due elegantly composed in other arranged dialects such as Scala, Ruby, C #, and so forth [7]. In our tasks, Maven is utilized to meet every one of the natural necessities.

Expert depends on the fundamental ideas of emotional wellness. This implies that the circle to fabricate and begin a specific commemoration (project) has been obvious.

For the private area, the Maven should project development process is efficient and that the POM is an assurance of accomplishing the ideal outcomes.

There are three primary kinds of life: typical, clean, and web. The standard strategy utilized is to convey your schoolwork, tidiness to clean the undertaking, and wellbeing to the site to work on the association of your family site.

3.2.4 Ansible

The arrangement is an open-source market for programming, equipment the board, and equipment-based foundation. It runs on numerous Unix-like frameworks and can introduce Unix-like frameworks and Microsoft Windows. '[6] This gadget is an uncommon specialist, and the reason for the framework is to have no clients, which makes the interaction simpler and cleaner.

In our activities, Ansible is utilized as an administration instrument.

Obviously, IT supervisors can begin involving machines in their everyday tasks. This machine liberates pioneers in exercises that assist them with rapidly assessing business and grow the way of life of advancement. At long last, Ansible gives bunches

one thing they can't get enough of: time. Permit the people who put stock in zero for good.

The response is basic machine language that can plainly tell the best way to utilize IT. It is easy to study and enlist, and you needn't bother with a four-year college education in programming to be assessed. Robotization ought not be more troublesome than required assignments

3.2.5 Zabbix

'Zabbix is an open-source programming market for different data innovation gadgets like organizations, servers, virtual machines and cloud administrations. Zabbix gives control measures, for example, other organization utilization, CPU burden, and circle use. Zabbix arrangement should be possible utilizing a XML layout that contains control components. Programming that screen [uncertain] execution on Linux, Hewlett Packard Unix (HP-UX), Mac OS X, Solaris, and working frameworks (OS); However, Windows control must be done through representatives. Zabbix can utilize MySQL, MariaDB, PostgreSQL, SQLite, Oracle or IBM DB2 to store information. '[8]

Zabbix can be shipped off charities and philanthropies. Experts in the field of data innovation are acquainted with better screen the exhibition of information assortment. At that point, examiners detailed the consolidation report to Zabbix server supervisors. This data is reviewed from the report or outside the Zabbix UI (GUI) chart. In case of any material issues, Zabbix will send a notice or notice to clients. Automated reconnaissance can prompt a perception line or a specialist reenactment instrument that incorporates resources having a place with that class.

Zabbix's web-based GUI permits clients to get to data innovation conditions utilizing intuitive whiteboards, designs, network cards, illustrations, and reports. For instance, the client can alter the report to show the degree of relationship between the SLA and the key boundaries (KPIs) on the CPU load. Arranged exercises use Zabbix to screen program execution. It helps control our framework.

3.3 SYSTEM DESIGN

The system is designed by combined a combination of previously discussed DevOps tools and Techniques. Development cycle primarily consist of four main parts i.e., Building, Testing, Staging, Production.

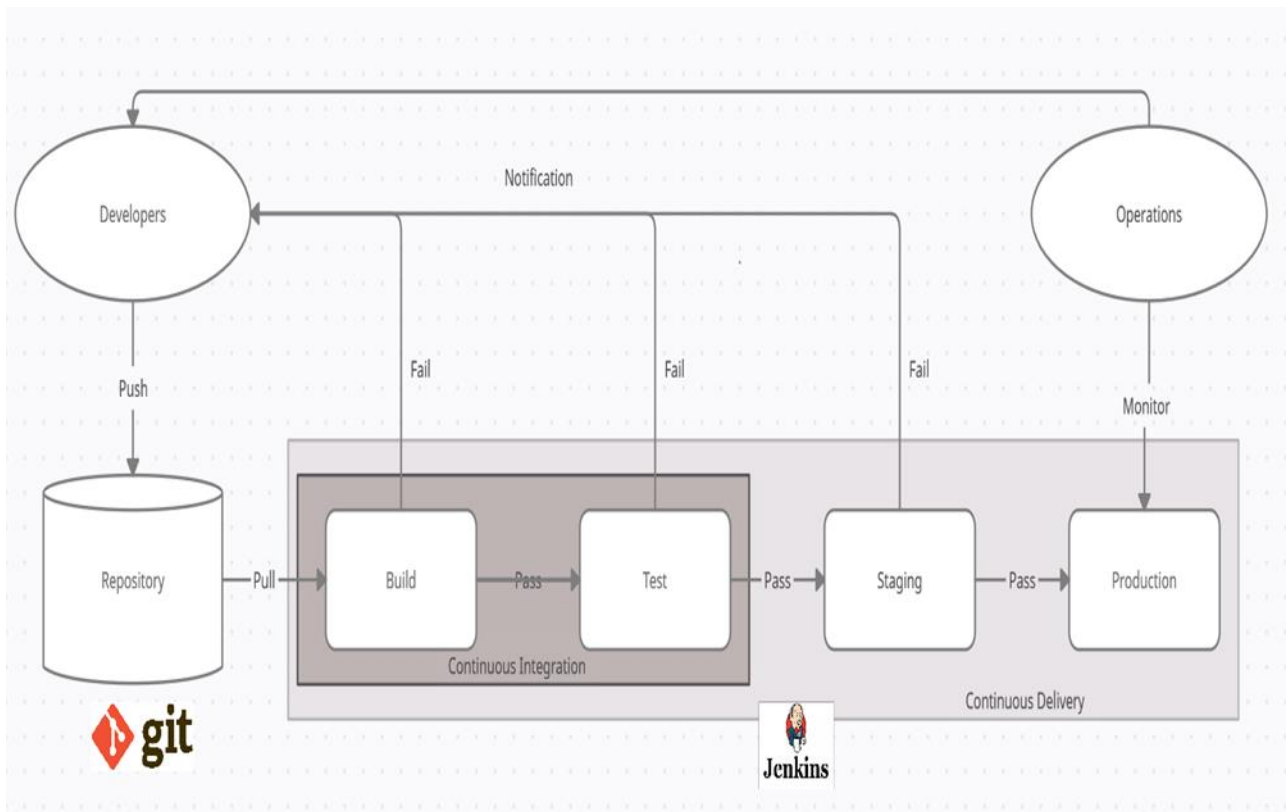


Fig 3.1 Architecture diagram

3.4 PIPELINE ARCHITECTURE

- The developers push their code changes / changes in GitHub.
- Jenkins downloads the project in the workspace.
- In this step Jenkins will perform in parallel the compilation of the Maven project (generation of * .class) and the static analysis of

the code with the CheckStyle plugin using Jenkins Docker agents.

- Running Unit Tests using JUnit and Mockito.
- Running Integration Tests using JUnit and Mockito.
- In this step we find the generation of documentation and the parallel execution of a set of code analyzes using SonarQube, PMD and FindBugs.
- Storing artifacts in the Nexus repository.
- Provisioning with Ansible. That is the preparation of the environment to ensure the installation of jdk and tomcat on our servers.
- Deploy the artifact in the staging server which is a Docker container with the Ubuntu OS.

3.5 PIPELINE AS CODE

Pipeline as Code describes a set of features that allow Jenkins users to define pipelined job processes with code, stored and versioned in a source repository. These features allow Jenkins to discover, manage, and run jobs for multiple source repositories and branches — eliminating the need for manual job creation and management.

To use *Pipeline as Code*, projects must contain a file named Jenkinsfile in the repository root, which contains a "Pipeline script."

Additionally, one of the enabling jobs needs to be configured in Jenkins:

- *Multibranch Pipeline*: build multiple branches of a *single* repository automatically
- *Organization Folders*: scan a **GitHub Organization** or **Bitbucket Team** to discover an organization's repositories, automatically creating managed *Multibranch Pipeline* jobs for them

- *Pipeline*: Regular Pipeline jobs have an option when specifying the pipeline to "Use SCM".

Fundamentally, an organization's repositories can be viewed as a hierarchy, where each repository may have child elements of branches and pull requests.

3.5.1 The Jenkinsfile

Presence of the Jenkinsfile in the root of a repository makes it eligible for Jenkins to automatically manage and execute jobs based on repository branches.

The Jenkinsfile should contain a Pipeline script, specifying the steps to execute the job. The script has all the power of Pipeline available, from something as simple as invoking a Maven builder, to a series of interdependent steps, which have coordinated parallel execution with deployment and validation phases.

A simple way to get started with Pipeline is to use the *Snippet Generator* available in the configuration screen for a Jenkins *Pipeline* job. Using the *Snippet Generator*, you can create a Pipeline script as you might through the dropdowns in other Jenkins jobs.

3.5.2 Folder Computation

Multibranch Pipeline projects and *Organization Folders* extend the existing folder functionality by introducing 'computed' folders. Computed folders automatically run a process to manage the folder contents. This computation, in the case of *Multibranch Pipeline* projects, creates child items for each eligible branch within the child. For *Organization Folders*, computation populates child items for repositories as individual *Multibranch Pipelines*.

Folder computation may happen automatically via webhook callbacks, as branches and repositories are created or removed. Computation may also be triggered by the *Build Trigger* defined in the configuration, which will automatically run a computation task after a period of inactivity (this defaults to run after one day).

3.5.3 Configuration

Both *Multibranch Pipeline* projects and *Organization Folders* have configuration options to allow precise selection of repositories. These features also allow selection of two types of credentials to use when connecting to the remote systems:

- *scan* credentials, which are used for accessing the GitHub or Bitbucket APIs
- *checkout* credentials, which are used when the repository is cloned from the remote system; it may be useful to choose an SSH key or "- *anonymous* -", which uses the default credentials configured for the OS user

3.6 CONTINUOUS DELIVERY WITH PIPELINE

Continuous delivery allows organizations to deliver software with lower risk. The path to continuous delivery starts by modelling the software delivery pipeline used within the organization and then focusing on the automation of it all. Early, directed feedback, enabled by pipeline automation enables software delivery more quickly over traditional methods of delivery.

Jenkins is the Swiss army knife in the software delivery toolchain. Developers and operations (DevOps) personnel have different mindsets and use different tools to get their respective jobs done. Since Jenkins integrates with a huge variety of toolsets, it serves as the intersection point between development and operations teams.

Many organizations have been orchestrating pipelines with existing Jenkins plugins for several years. As their automation sophistication and their own Jenkins experience increases, organizations inevitably want to move beyond simple pipelines and create complex flows specific to their delivery process.

These Jenkins users require a feature that treats complex pipelines as a first-class object, and so the Pipeline plugin was developed.

Continuous delivery is a process - rather than a tool - and requires a mindset and culture that must percolate from the top-down within an organization. Once the organization has bought into the philosophy, the next and most difficult part is mapping the flow of software as it makes its way from development to production.

The root of such a pipeline will always be an orchestration tool like a Jenkins, but there are some key requirements that such an integral part of the pipeline must satisfy before it can be tasked with enterprise-critical processes:

- **Zero or low downtime disaster recovery:** A commit, just as a mythical hero, encounters harder and longer challenges as it makes its way down the pipeline. It is not unusual to see pipeline executions that last days. A hardware or a Jenkins failure on day six of a seven-day pipeline has serious consequences for on-time delivery of a product.
- **Audit runs and debug ability:** Build managers like to see the exact execution flow through the pipeline, so they can easily debug issues.

To ensure a tool can scale with an organization and suitably automate existing delivery pipelines without changing them, the tool should also support:

- **Complex pipelines:** Delivery pipelines are typically more complex than canonical examples (linear process: Dev→Test→Deploy, with a couple of operations at each stage). Build managers want constructs that help parallelize parts of the flow, run loops, perform retries and so forth. Stated differently, build managers want programming constructs to define pipelines.
- **Manual interventions:** Pipelines cross intra-organizational boundaries necessitating manual handoffs and interventions. Build managers seek capabilities such as being able to pause a pipeline for a human to intervene and make manual decisions.

The Pipeline plugin allows users to create such a pipeline through a new job type called Pipeline. The flow definition is captured in a Groovy script, thus adding control flow capabilities such as loops, forks and retries. Pipeline allows for stages with the

option to set concurrencies, preventing multiple builds of the same pipeline from trying to access the same resource at the same time.

3.6.1 Pipeline Job Type

There is just one job to capture the entire software delivery pipeline in an organization. Of course, you can still connect two Pipeline job types together if you want. A Pipeline job type uses a Groovy-based DSL for job definitions. The DSL affords the advantage of defining jobs programmatically:

```
node('linux'){  
  git url: 'https://github.com/jglick/simple-maven-project-with-tests.git'  
  def mvnHome = tool 'M3'  
  env.PATH = "${mvnHome}/bin:${env.PATH}"  
  sh 'mvn -B clean verify'  
}
```

3.6.2 Stages

Intra-organizational (or conceptual) boundaries are captured through a primitive called "stages." A deployment pipeline consists of various stages, where each subsequent stage builds on the previous one. The idea is to spend as few resources as possible early in the pipeline and find obvious issues, rather than spend a lot of computing resources for something that is ultimately discovered to be broken.

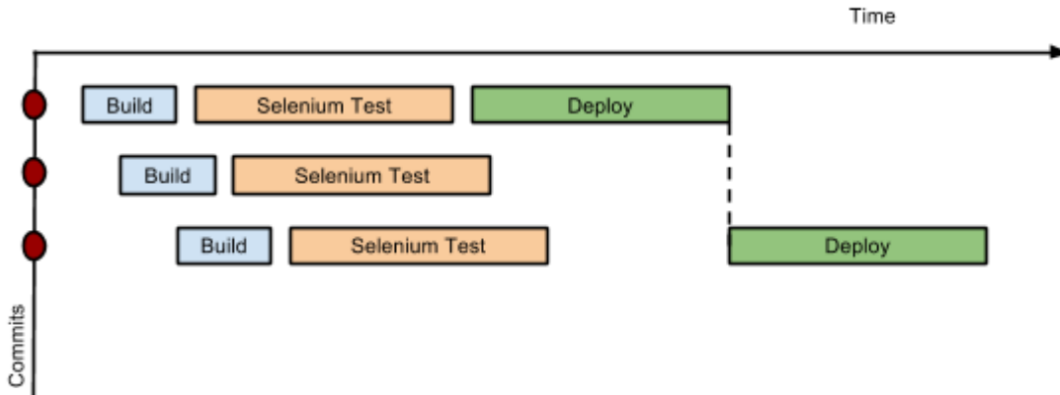


Fig 3.2 Throttled stage concurrency with Pipeline

Consider a simple pipeline with three stages. A naive implementation of this pipeline can sequentially trigger each stage on every commit. Thus, the deployment step is triggered immediately after the Selenium test steps are complete. However, this would mean that the deployment from commit two overrides the last deployment in motion from commit one. The right approach is for commits two and three to wait for the deployment from commit one to complete, consolidate all the changes that have happened since commit one and trigger the deployment. If there is an issue, developers can easily figure out if the issue was introduced in commit two or commit three.

Pipeline provides this functionality by enhancing the stage primitive. For example, a stage can have a concurrency level of one defined to indicate that at any point only one thread should be running through the stage. This achieves the desired state of running a deployment as fast as it should run.

```
stage name: 'Production', concurrency: 1
node {
    unarchive mapping: ['target/x.war' : 'x.war']
    deploy 'target/x.war', 'production'
    echo 'Deployed to http://localhost:8888/production/'
}
```

3.6.3 Gates and Approvals

Continuous delivery means having binaries in a release ready state whereas continuous deployment means pushing the binaries to production - or automated deployments. Although continuous deployment is a sexy term and a desired state, in reality organizations still want a human to give the final approval before bits are pushed to production. This is captured through the "input" primitive in Pipeline. The input step can wait indefinitely for a human to intervene.

input message: "Does http://localhost:8888/staging/ look good?"

3.6.4 Deployment of Artifacts to Staging/Production

Deployment of binaries is the last mile in a pipeline. The numerous servers employed within the organization and available in the market make it difficult to employ a uniform deployment step. Today, these are solved by third-party deployer products whose job it is to focus on deployment of a particular stack to a data center. Teams can also write their own extensions to hook into the Pipeline job type and make the deployment easier.

Meanwhile, job creators can write a plain old Groovy function to define any custom steps that can deploy (or undeploy) artifacts from production.

```
def deploy(war, id) {  
    sh "cp ${war} /tmp/webapps/${id}.war"  
}
```

3.6.5 Restartable flows

All Pipelines are resumable, so if Jenkins needs to be restarted while a flow is running, it should resume at the same point in its execution after Jenkins starts back up. Similarly, if a flow is running a lengthy sh or bat step when an agent unexpectedly disconnects, no progress should be lost when connectivity is restored.

There are some cases when a flow build will have done a great deal of work and proceeded to a point where a transient error occurred: one which does not reflect the inputs to this build, such as source code changes. For example, after completing a lengthy build and test of a software component, final deployment to a server might fail because of network problems.

3.6.6 Pipeline Stage View

When you have complex builds pipelines, it is useful to see the progress of each stage and to see where build failures are occurring in the pipeline. This can enable users to debug which tests are failing at which stage or if there are other problems in their pipeline. Many organizations also want to make their pipelines user-friendly for non-developers without having to develop a homegrown UI, which can prove to be a lengthy and ongoing development effort.

The Pipeline Stage View feature offers extended visualization of Pipeline build history on the index page of a flow project. This visualization also includes helpful metrics like average run time by stage and by build, and a user-friendly interface for interacting with input steps.

The only prerequisite for this plugin is a pipeline with defined stages in the flow. There can be as many stages as you desired and they can be in a linear sequence, and the stage names will be displayed as columns in the Stage View interface.

3.7 DEVOPS MONITORING

The DevOps methodology has been widely used by businesses to improve the continuous delivery and integration process. One of the main features of DevOps is to automate the process that falls within the DevOps pipeline. Automation has improved the overall effectiveness of the business working. However, the main challenge occurs when we have to maintain and monitor these automated pipelines. You have

to keep an eye on what is going on within the business and as the network expands, this task becomes crucial. As the business grows, you should have full visibility of the system, especially when you have fewer resources to maintain the work at your disposal. Thus, there is an immense need to have a proper alerting system that will send alerts about any unexpected behaviour of the system or application.

Whether your business is small or large, you need to consider various aspects of monitoring like what part should be monitored, what type of tools you can use, and how the monitoring strategy should be implemented. You can take the help of various automated monitoring tools to identify the potential issues. With software development, the monitoring tools and their functioning has also evolved. As the DevOps methodology is gaining popularity, you need to monitor each DevOps phase from integration, provisioning, to deployment.

DevOps team can use DevOps strategies for monitoring the system's aspect. In this article, we will discuss various monitoring and alerting strategies that should be used for a seamless and effective delivery process.

To ensure the better performance of the system, you need to monitor various things. But first, you need to be sure what things you need to monitor within your system and to what level to ensure effective functioning of the system. Here, we have mentioned some of the categories that need deep monitoring:

3.7.1 Monitoring Server Health

You need to monitor the health of the server to ensure enhanced performance and uptime by optimal consumption of the available resources. The DevOps team needs to configure its resources efficiently to identify the critical endpoints within the application. Also, make sure that the applied configurations are compatible with the system to work seamlessly.

3.7.2 Monitoring the Output of an Application

One of the major aspects of DevOps that needs to be monitored is logging the output of an application which is mostly ignored. You must implement a centralized logging system that will help in checking for the potential errors within the system. Make sure that you use the right tools and strategies to throw an alert for occurring certain errors in real-time for preventing the system from vulnerabilities. These logs are used by the IT teams and system administrators to troubleshoot the expected errors and identify the root cause for a quick resolution.

3.7.3 Development Milestones

After implementing the DevOps methodology, you need to keep track of how well the DevOps strategy is working and getting better workflow insights. It allows you to monitor how well the different teams are collaborating. You should also keep a check on the sprint durations, error identification rate, resolve rate, and other features for improving the system's performance. Keeping proper checks in place will help you in achieving the tasks' deadlines.

3.7.4 Monitoring User Activity

Various types of users are using your software and any unusual behaviour of the user may hinder your software's performance. So, you need to check every user's account and make sure they are protected with proper authentication methods and have the right role-based access to different applications. Also, you need to keep an eye on what is being accessed by which user to ensure safety.

CHAPTER 4

RESULT AND DISCUSSION

The proposed system is capable of increasing development speed drastically. This allows developers and organizations to release patches and updates frequently. Persistent coordination and coherence (CI/CD) give superb opportunity to application gatherings. Just those liable for entering the code into capacity. This code is fit to be coordinated, tried, sent, retested, incorporated into the foundation and sent with astonishing exactness with security and quality control. While utilizing CI/CD, the textual style quality improves, and shells and fixes convey rapidly and obviously with next to no changes. The impact of giving something can be made and connected to work data. Thusly, CI/CD is a method for aiding assemble and foster applications that can enter the market quicker and utilize Visuals. Computerization in the advancement of programming for process and complete cycle control, which assists with building structures proficiently and economically.

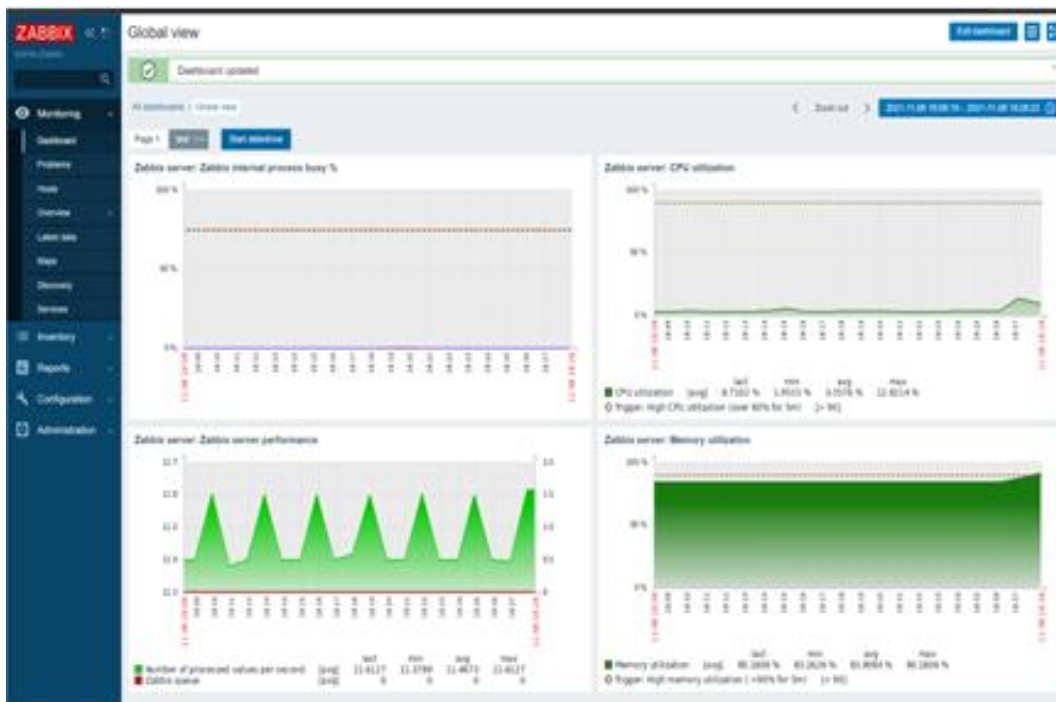


Fig 4 shows the result comparison of various CI/CD tools

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

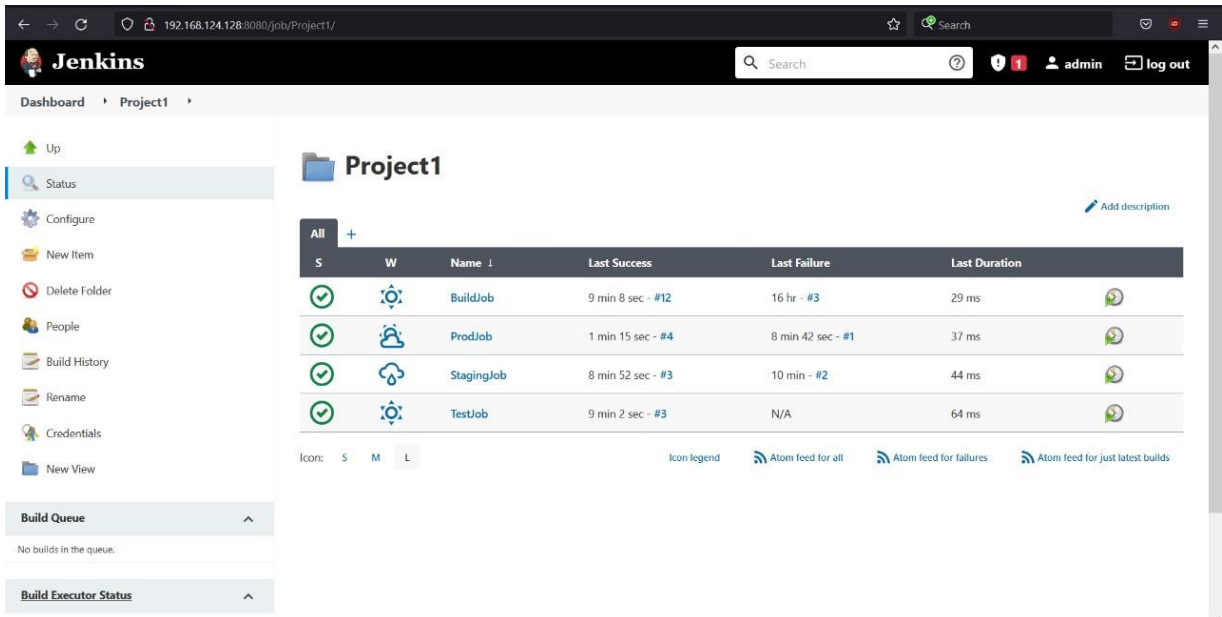
Continuous Integration and Continuous Delivery (CI/CD) gives an ideal situation to your association's application groups. Developers just push code to a repository. This code will be integrated, tested, deployed, tested again, merged with infrastructure, go through security and quality audits, and be prepared to send with amazingly high certainty. At the point when CI/CD is utilized, code quality is improved, patches and updates are conveyed rapidly and with high certainty that there will be no breaking changes. The effect of any delivery can be associated with information from creation and tasks. Thus, CI/CD procedure assists in building and improvising on the applications with quicker time to market and better User Interface. Automation in software development streamlines the processes and the complete cycle which helps in building better and more steady applications.

5.2 FUTURE WORK

The main motive is to automate the development cycle to make it fast and reliable as possible. We plan to further optimize the system to make it stable and reliable by implementing automatic updating the tools and several dependencies checks. We can improve the system by introducing new tools and further optimizing to run in a stable state.

APPENDIX

SCREEN SHOTS OF THE SYSTEM



← → ↻ 192.168.124.128:8080/job/Project1/job/BuildJob/14/console

Jenkins

Search

admin log out

Dashboard ▸ Project1 ▸ BuildJob ▸ #14

- Back to Project
- Status
- Changes
- Console Output
- View as plain text
- Edit Build Information
- Delete build '#14'
- Previous Build

Console Output

Started by user [admin](#)
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Project1/BuildJob
[BuildJob] \$ /bin/sh -xe /tmp/jenkins16286508331780542736.sh
+ rm /home/ws/test/*
rm: cannot remove '/home/ws/test/*': No such file or directory
+ true git clone https://github.com/dannyboy1/MainProject-BETA- /home/ws/test/ mv * /var/www/html
Triggering a new build of Project1 » TestJob
Finished: SUCCESS

REST API Jenkins 2.319.3

← → ↻ 192.168.124.128:8080/job/Project1/job/TestJob/4/console

Jenkins

Search

admin log out

Dashboard ▸ Project1 ▸ TestJob ▸ #4

- Back to Project
- Status
- Changes
- Console Output
- View as plain text
- Edit Build Information
- Delete build '#4'
- Previous Build

Console Output

Started by upstream project "Project1/BuildJob" build number 13
originally caused by:
Started by user [admin](#)
Started by upstream project "Project1/BuildJob" build number 14
originally caused by:
Started by user [admin](#)
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Project1/TestJob
[TestJob] \$ /bin/sh -xe /tmp/jenkins12085466535432728105.sh
+ echo TESTING....
TESTING....
+ echo TEST SUCCESSFUL
TEST SUCCESSFUL
Triggering a new build of Project1 » StagingJob
Finished: SUCCESS

REST API Jenkins 2.319.3

192.168.124.128:8080/job/Project1/job/StagingJob/4/console

Jenkins

Search

admin log out

Dashboard > Project1 > StagingJob > #4

- Back to Project
- Status
- Changes
- Console Output
- View as plain text
- Edit Build Information
- Delete build '#4'
- Previous Build

Console Output

Started by upstream project "Project1/TestJob" build number 4
originally caused by:
Started by upstream project "Project1/BuildJob" build number 13
originally caused by:
Started by user admin
Started by upstream project "Project1/BuildJob" build number 14
originally caused by:
Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Project1/StagingJob
[StagingJob] \$ /bin/sh -xe /tmp/jenkins680606764192803186.sh
+ mv * /home/ws/stage
mv: cannot stat '*': No such file or directory
+ true
+ echo APPLICATION STAGED
APPLICATION STAGED
Triggering a new build of [Project1 - ProdJob](#)
Finished: SUCCESS

192.168.124.128:8080/job/Project1/job/ProdJob/5

REST API Jenkins 2.319.3

192.168.124.128:8080/job/Project1/job/ProdJob/5/console

Jenkins

Search

admin log out

Dashboard > Project1 > ProdJob > #5

- Back to Project
- Status
- Changes
- Console Output
- View as plain text
- Edit Build Information
- Delete build '#5'
- Previous Build

Console Output

Started by upstream project "Project1/StagingJob" build number 4
originally caused by:
Started by upstream project "Project1/TestJob" build number 4
originally caused by:
Started by upstream project "Project1/BuildJob" build number 13
originally caused by:
Started by user admin
Started by upstream project "Project1/BuildJob" build number 14
originally caused by:
Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Project1/ProdJob
[ProdJob] \$ /bin/sh -xe /tmp/jenkins7821309890461133796.sh
+ sh /home/ws/prod/prod.sh
UPDATING
APPLICATION UPDATED
Finished: SUCCESS

REST API Jenkins 2.319.3

PUBLICATION AND PLAGIARISM REPORT

← → ↻ blob:https://web.whatsapp.com/f03e6d17-8682-4881-aa95-87ed95de402 ☆ development speed upto →

1 of 10 Automatic Zoom

End to End DevOps Pipeline.docx

by End To End Devops Pipeline.docx End To End Devops Pipeline.docx

Submission date: 05-Mar-2022 10:22AM (UTC-0500)
Submission ID: 1777134549
File name: End_to_End_DevOps_Pipeline.docx (724.6K)
Word count: 3128
Character count: 17556

End to End DevOps Pipeline

V. Balaji,
Department of Computer
Science and Engineering
Sathyabama Institute of
Science and Technology
Chennai, India
Balaji110201@gmail.com

S Praveen Daniel
Department of Computer
Science and Engineering
Sathyabama Institute of
Science and Technology
Chennai, India
Indiapraveendaniel2001@gmail.com

Mrs. Srividhya S.R.M.E
Department of Computer
Science and Engineering
Sathyabama Institute of
Science and Technology
Chennai, India
srividhyastse@sathyabama.ac.in

← → ↻ <blob:https://web.whatsapp.com/703e6d17-8682-4881-aa95-87ed95de402> ☆ development speed upto →

10 of 10 — + 110% ▾

End to End DevOps Pipeline.docx

ORIGINALITY REPORT

3%	2%	0%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Melbourne Institute of Technology Student Paper	2%
2	dokumen.pub Internet Source	<1%
3	anonradio.net Internet Source	<1%
4	Ruth W. Macarthy, Julian M. Bass. "An Empirical Taxonomy of DevOps in Practice", 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2020 Publication	<1%
5	www.91it.org Internet Source	<1%

Exclude quotes On Exclude matches Off
Exclude bibliography On

REFERENCES

- [1] M. Z. Abrahams and J. J. Langerman, "Compliance at Velocity within a DevOps Environment," 2018 Thirteenth International Conference on Digital Information Management (ICDIM), 2018, pp. 94-101, doi: 10.1109/ICDIM.2018.8847007.
- [2] P. Agrawal and N. Rawat, "Devops, A New Approach To Cloud Development & Testing," 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), 2019, pp. 1-4, doi: 10.1109/ICICT46931.2019.8977662.
- [3] R. W. Macarthy and J. M. Bass, "An Empirical Taxonomy of DevOps in Practice," 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2020, pp. 221-228, doi: 10.1109/SEAA51224.2020.00046.
- [4] Q. Liao, "Modelling CI/CD Pipeline Through Agent-Based Simulation," 2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), 2020, pp. 155-156, doi: 10.1109/ISSREW51248.2020.00059.
- [5] M. R. Pratama and D. Sulistiyo Kusumo, "Implementation of Continuous Integration and Continuous Delivery (CI/CD) on Automatic Performance Testing," 2021 9th International Conference on Information and Communication Technology (ICoICT), 2021, pp. 230-235, doi: 10.1109/ICoICT52021.2021.9527496.
- [6] Sriniketan Mysari, Vaibhav Bejgam, "Continuous Integration and Continuous Deployment Pipeline Automation Using Jenkins Ansible", Emerging Trends in Information Technology and Engineering (ic-ETITE) 2020 International Conference on, pp. 1-4, 2020.
- [7] Putu Adi Guna Permana, Evi Triandini, Ni Luh Gede Pivin Suwirmayanti, "Implementation Jenkins Automation Deployment with Scheduler and Notification",

Cybernetics and Intelligent System (ICORIS) 2021 3rd International Conference on, pp. 1-5, 2021

[8] Thorsten Rangnau, Remco v. Buijtenen, Frank Fransen, Fatih Turkmen, "Continuous Security Testing: A Case Study on Integrating Dynamic Security Testing Tools in CI/CD Pipelines", Enterprise Distributed Object Computing Conference (EDOC) 2020 IEEE 24th International, pp. 145-154, 2020.

[9] [https://en.wikipedia.org/wiki/Ansible_\(software\)](https://en.wikipedia.org/wiki/Ansible_(software))

[10] <https://mvnrepository.com/>

[11] <https://en.wikipedia.org/wiki/Zabbix>

[12] <https://docs.ansible.com/>

[13] <https://git-scm.com/doc>

[14] <https://maven.apache.org/guides/>

[15] <https://www.jenkins.io/doc/>