WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Continuous Integration and Continuous Delivery using Jenkins and AWS Services

[1]Nalla Siva Kumar, [2]Mondi Sai Durga, [3]Sirisha Hanumanthu and [4]Chaitanya

[1,2,3,4]Aditya Engineering College, Surampalem, Ap, India.

ABSTRACT:

Now a days in software industry many companies are using automation for developing a complex project. Continuous integration and continuous delivery using Jenkins is a concept for making the web applications simple and easy to perform. The time is a very crucial part while building a project. To save a time, automation in integration and deploying using Jenkins is used. If any modifications need to performed, then it makes the developer to feel free to update the code frequently. The process of updating, testing, deploying will be performed automatically by the developer approval. The developer needs to learn multiple DevOps toolkits like docker, Kubernetes, ansible etc. to work on DevOps project. Sometimes it is difficult to learn different technologies simultaneously. So, to overcome the problem, the menu-driven is created which will be helpful for developers to perform their operation on their required technology without any knowledge on Linux commands, Redhat operating system. The automatic installations, statuses will be checked and performed automatically without developer involvement.

Key Word–Jenkins, Docker, AWS services, Continuous Integration, Continuous Delivery.

Introduction

Overview

The main proposal of this project is to test and deploy the source code into the server for time to time. The installations of DevOps toolkits and checking of container statuses will be performed automatically.

Objective

Our main proposal of this project is to test the source code and deploy it without any human interaction. The DevOps toolkits will be automatically installed without learning of any Linux commands and we can start or stop the containers and statuses of containers can be checked.

In software companies learning of new languages is a main task for the developers. There will be multiple DevOps projects which need to be done for the clients. So, the developer had to gain knowledge on the various DevOps toolkits technologies. So, to overcome that the menu-driven program is created. A program that obtains input from a user by displaying a list of options the menu from which the user indicates his/her choice. As it will contain a Menu that can directly help multiple users at a single given time. It will be very much helpful to the coming generation. It reduces the most time for learning and installing technologies.

Proposed System

- This approach involves these steps.
- Firstly, build the source code using Guvi language in github for create the pipeline for integration, testing and deploying.
- Then create a Jenkins account and aws account and login to it.
- Now create an instance for executing the code like master node, DevOps tool kit, maven slave.
- The master node produces ip address which can be run in 8989 port. Now go to the dashboard and click on manage node and cloud.
- Now move to the instance and click on the box of master node. Then copy the public ip address and paste it in the new tab by giving 8989 ports i.e, <public-ip- address>:8989.
- Move to the Jenkins account and click on manage Jenkins and click on manage nodes and clouds. Then select maven selven and click configure. There will be a presence of host box and remove the ip address in it and paste the master node ip address.

- There will be a docker pipeline in docker hub and click the build now which runs the source code and perform testing and deploying automatically when the user approves.
- Implemented a code in shell language and place it in a folder. Then by using commands in command prompt the DevOps toolkit menu will be opened which can be only accessed by root. Then we can easily install the tools without any knowledge of commands. If the tool is already installed in a device then the containers status can be viewed and we can start or stop the containers in prompt easily.

Advantages

- User friendly.
- Bugs in source code will be identified quickly.
- We can commit changes whenever we require.
- Time consumption will be very less.

Architecture

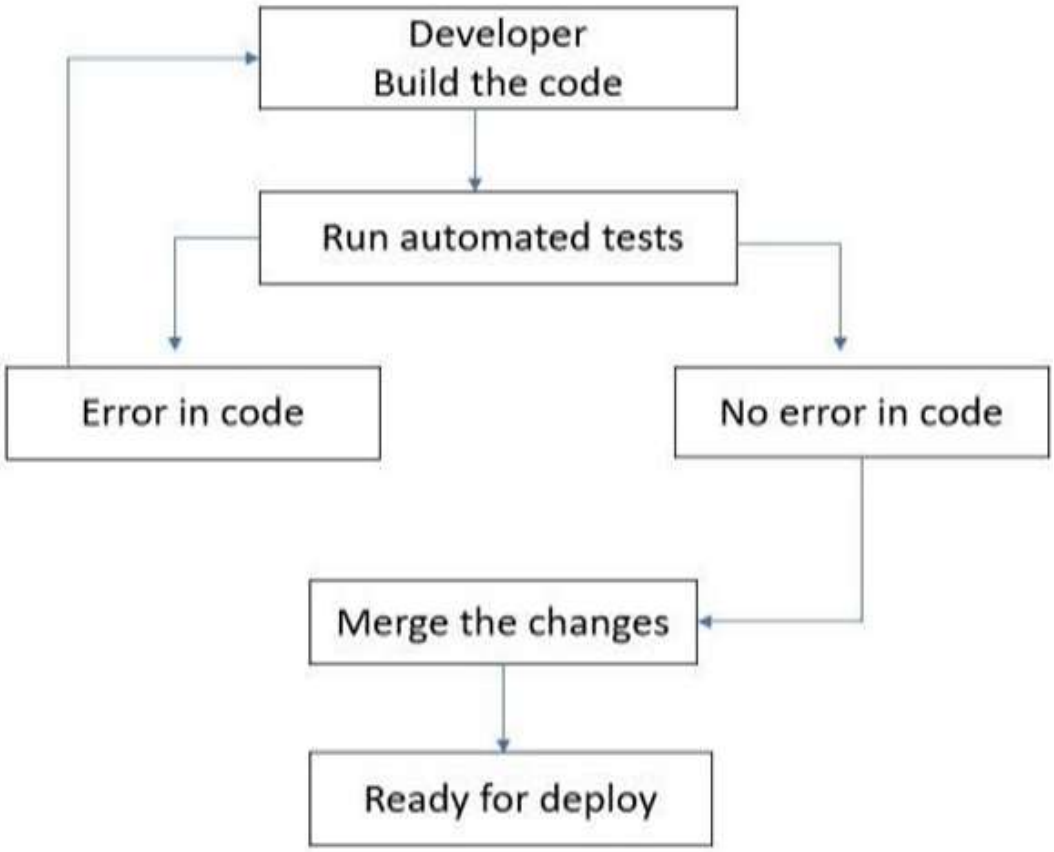

Fig.1 Architecture Diagram

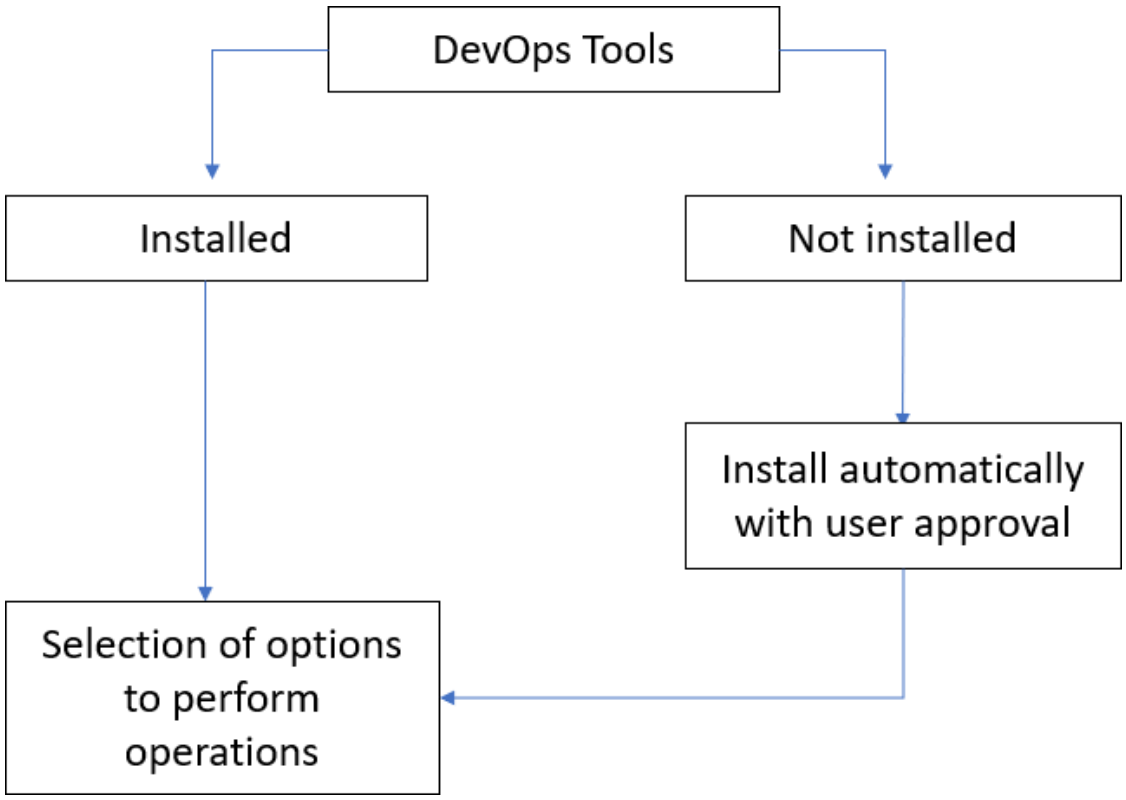

Fig.2 Architecture diagram

Software Requirements

- Jenkins tools
- Aws services
- Docker hub
- **Technology Used:** DevOps technology
- **Modules Used**

Installing Jenkins in Server

The Jenkins is installed in the server by using AWS services. This is done by the admin and after creating Linux EC2 instance, commands are given in that Linux os to set to root user. After setting to the root user the Jenkins is installed in that inux EC2 instance.

Creating Jobs in Jenkins

After the successful installation of Jenkins, signup to the Jenkins with the username and password. Create four jobs that are build, test, approved, deploy. Write respective code to perform the operations. These jobs are executed after the creation of pipeline.

Creating a pipeline

After the creation of jobs, pipeline is created for build, test and deploy. The advantage of creation of a pipeline is that the automatic execution of jobs will be done automatically.

Result

Fig.3 Automatic testing and deploying

Fig.4 DevOps Toolkit

Conclusion

Jenkins pipeline methodology is used to build and integrate automatically. Jenkins is a platform independent and flexible. It saves a lot of time. Deploying using Jenkins is very efficient and easy to setup. The DevOps concept will be acts as a container which helps to run the program. The instance is created in AWS server and the pipelines need to created in a Jenkins. It will helps to run the build, test and deploy automatically. And performing the operations of DevOps toolkit technologies will be performed automatically without getting full knowledge on any new technologies like Linux commands.

References

- [1] Ebert, C., Gallardo, G., Hernantes, J., Serrano, N., "DevOps," IEEE Software, 33(3), 94100. doi:10.1109/ms.2016.68 .
- [2] Zebula Sampedro,Aaron Holt,Thomas Hauser "Continuous Integration and Delivery for HPC: Using Singularity and Jenkins," PEARC '18 Proceedings of the Practice and Experience on Advanced Research Computing, Article No. 6, doii.10.1145/3219104.3219147.
- [3] L. Crispin and J. Gregory, Agile Testing: A Practical Guide for Testers and Agile Teams. Addison-Wesley, 2011.
- [4] Seth, N., & Khare, R. (2015). *ACI (automated Continuous Integration) using Jenkins: Key for successful embedded Software development. 2015 2nd International Conference on Recent Advances inEngineering & Computational Sciences (RAECS).* doi:10.1109/raecs.2015.745 3279