CREATING A CONTINUOUS INTEGRATION, AND CONTINUOUS DEPLOYMENT (CI/CD) PIPELINE FOR DOCKER PROJECT

[1]Shruthi, [2]Vikash Kumar,
[1]Student, [2]Asst.Professor,
[1] Department of MCA (JAIN University) Bangalore, India
[2] Department of MCA (JAIN University) Bangalore, India

Abstract*:* The rate of innovation within the container ecosystem and throughout the community has been amazing to watch. This impressive evolution is the result of the deep engagement from the open source community. This assignation has been instrumental in fostering ongoing innovation.

Keywords: Innovation, Ecosystem, Impressive, Agile, Waterfall Model.

1.INTRODUCTION

The implementation of CI/CD has changed how developers and testers ship software. This is a first in a series of blog posts about this transition and the blog posts will provide understanding into different tools and process changes which could help developers be more successful with CI/CD

First it was waterfall next it was, Agile and now it's Devops building With the rise of DevOps has come the new methods of Continuous Integration, Continuous Delivery, (CI/CD) and Continuous Deployment. Conventional software development and delivery methods are becoming outdated , in the agile age, most companies used to deploy software in monthly, quarterly, bi-annual, or even annual releases Now, in the DevOps era, weekly, daily, and even multiple times a day is the norm. This is especially true as SaaS is taking over the world and you can easily update applications on the fly without obliging customers to download new components. Often times, they won't even realize things are changing. Development teams have adapted to the shortened delivery cycles by implementing automation throughout their software delivery pipeline. Most teams have automated processes to check in code and deploy to new environments. This has been united with a focus on automating the testing process along the way as well, but that we'll cover in a future article

2. LITERATURE REVIEW

In this project, we will learn how to set up a continuous integration and continuous delivery (CI/CD) pipeline on AWS.
A pipeline helps us to automate steps in your software delivery process, such as introducing automatic builds and then deploying to Amazon EC2 instances. we will use Jenkins, a service that builds, tests, and deploys the code every time there is a code change, based on the release process models define. As part of our setup, we will plug other devops tools into jenkins to complete your software delivery pipeline. The other tools which are using is github and Maven. Github acts as source code repository and Maven acts as Building tool. This guide will show you how to create a very simple pipeline that pulls code from a source and automatically deploys it an amazon ec2 instances

3.CONTEXT FLOW DIAGRAM

A context diagram is a diagram that defines the boundary between the system. And its environment, showing the entities that interact with it. This diagram is a high-level view of system.

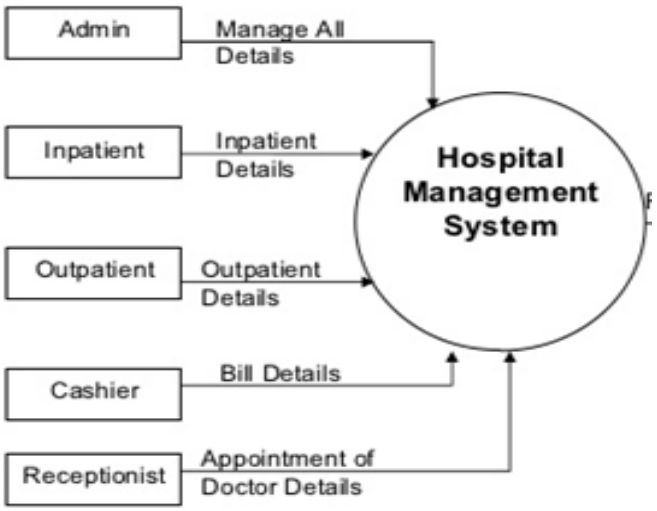

FIGURE 3-CONTEXT FLOW DIAGRAM

4.PROPOSED SYSTEM

continuous integration is a process in which describes all development work integrated as early as possible. the resulting objects are automatically created and tested. this process allows to identify errors as early as possible. here git tool is used for storing source code and maven for building the project and jenkins for setting up the pipeline for the project. with continuous integration help detecting errors more quickly and release code much faster than had done before .these days many dev teams have moved beyond ci to cd, which stands for either continuous delivery or continuous deployment._whatever the designation, cd is a process that moves software from code check-in to staging, or even production deployments.

5. DESIGN ARCHIECTURE

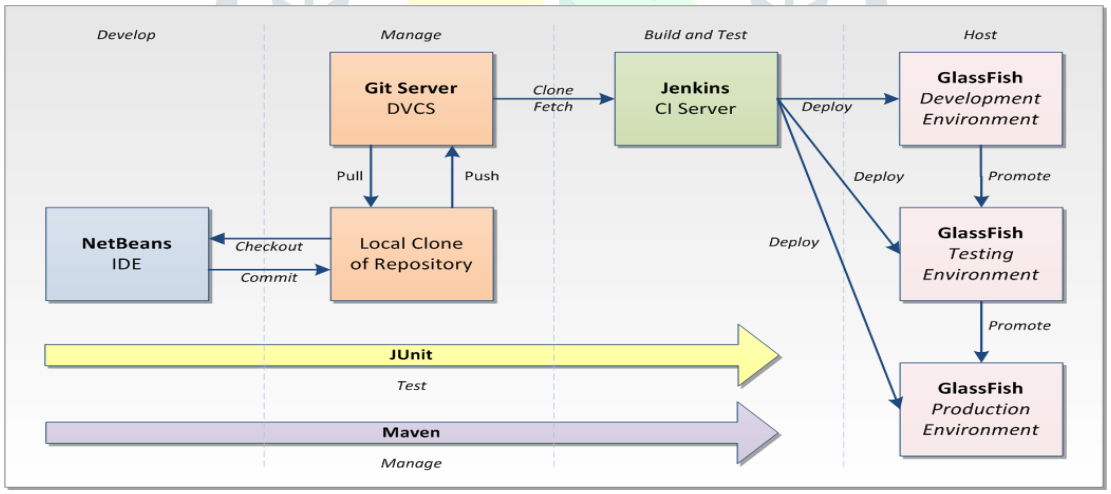

FIGURE 6- DESIGN ARCHITECTURE

GitHub:

Git is a one of the free, open source distributed version control system

And it is tool designed to handle everything from small to big projects with speed and efficiency.

Maven

Maven is a building tool and also it is a one of the powerful project management tools that is based on project object modelit is used projects build, dependency and documentation.

6. IMPLEMENTAION

The goal of coding or programming activity is to implement design in the best possible way. The coding activity effects on both testing and maintenance profoundly; the time spent in coding is much lesser then that of the total software cost, while testing and maintenances consume the major percentage. Thus, it should be clear that the goal during coding shouldn't be reduced implementation cost, but the goal is to reduce the cost of later phases, even if it means that the In other words, the goal during the phase is not the job of the programmer. Rather, the goal should be the job of the tester and maintainer

This distinction is important, as programmers are often concerned about how to finishing their job quickly, without keeping the later phases in mind. During coding, it should be kept in mind that the program should not be constructed easy to write, so that they are easy to read and understand, program is read more often and by lot more people during later phases. There are many different criteria for judging a program, including read ability, size of the program, execution time and required memory. Having readability and understand ability as a clear objective of the coding can itself help in producing software that is more maintainable.

7.RESULT

From the results, it's clear that our custom Docker PHP apache container greatly outperformed the official PHP images with close performance to our current aws setup. We use images like this in all of our PHP Docker deployments as the small performance loss from awash is within a small enough tolerance for us to be happy and is offset by the advantages listed at the beginning of this post. as mentioned, we are are able to successfully orchestrate the application using docker and hosted in aws servers and databases.

8. CONCLUSION

In this project, we have demonstrated each one step by step how to configuring and we are fully working PHP application. We built custom images for PHP and apache, and configured docker to orchestrate our containers. setup reflects a real-life scenario. In this guide, we have built and tagged our images locally and pushed it to docker hub. we may push to the official docker hub or even setup your own docker registry. In any case, this will allow us to build your images on one host and use them on another. Depending on your application requirements and the PHP framework we use, we may want to add more extensions. This can easily be done by modifying the Docker file used to build our custom PHP image. However, some extensions need extra dependencies to be installed in the container.

9. FUTURE ENHANCEMENT

In our future enhancements, we can host the application in kubernetes

- Kubernetes allows mounting is a variety of storage systems, including local storage, network storage and public cloud
- Kubernetes makes its very easier to deploy and update app configuration and secrets. Users don't have to rebuild the entire images, helping then safeguard secrets in their stack configuration.

10. REFERENCES

1. https://dzone.com/articles/learn-how-to-setup-a-cicd-pipeline-from-scratch
2. https://codefresh.io/continuous-integration/continuous-integration-delivery-pipeline-important/
3. https://www.docker.com/solutions/cid
4. https://blog.codeship.com/continuous-integration-and-delivery-with-docker/