Contents

1 Introduction

The purpose of the deployment team; the creation of an appropriate working environment by meeting the project requirements.We use docker and some python scripts for realize our aims.We will be talking about project tools and our methodologies at next sections.

2 What is DevOps ?

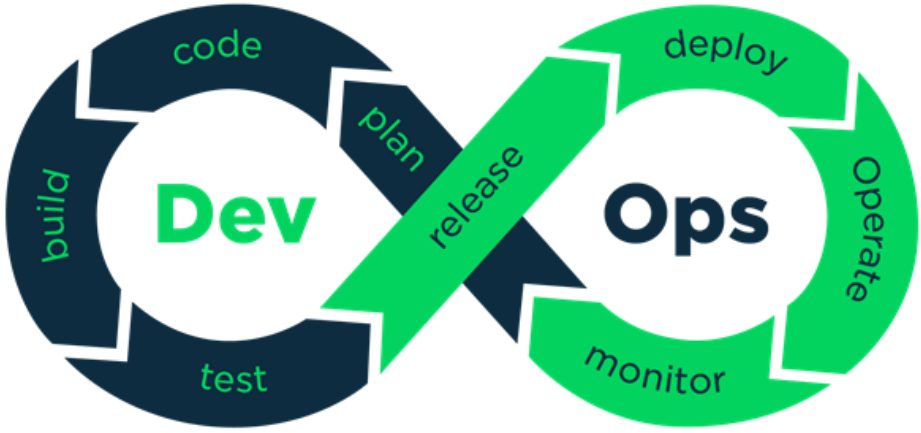

Figure 1: DevOps process

DevOps is a set of practices that automates the processes between software development and IT teams, in order that they can build, test, and release software faster and more reliably.DevOps is a culture, a movement, a philosophy. The DevOps culture puts a focus on creating a fast and stable work flow through development and IT operations. One main goal of DevOps is to deploy features into production quickly and to detect and correct problems when they occur, without disrupting other services.

2.1 Dev

Dev is one part of DevOps which describes the Software Development process, development stage.It has some important subsections.

2.1.1 Plan

Every project needs a vision that indicates the ultimate work goal to its participants. Defining a set of functionalities giving each iteration value, the criteria to be fulfilled and the end – for each project phase. A Living Product Stack which continuously supports a process of gardening – from a business point of view.

- Trello :
- MuleSoft Anypotint Studio :

2.1.2 Development Build

This phase is where the project gets built. Writing code, designing infrastructure, automating processes, defining tests or implementing security. It's where, at present, you'll find the most important effort in automating repetitive or complex actions.

2.1.3 Test

Software testing in the DevOps requires an automated test process that provides feedback at every checkpoint. Integration testing is needed on the build server. Functional testing, regression testing and more are needed on the test server. Deployment testing is needed on the staging server. If the build passes, it advances to the next step. If it fails, it's time to stop and fix the issue.

- Maven :

2.2 Ops

Ops is other part of DevOps which describes the operations on DevOps process. It has more wide area than Dev stage. Ops contain, System Admins,System Engineers,Database Admins,Network Engineers, Security Experts etc.

2.2.1 Monitor

Monitoring provides feedback from production. Monitoring delivers information about an application's performance and usage patterns.

In our project, Monitor group using the tools below for monitoring information :

- Zabbix :

2.2.2 Deploy

Deploy includes all the process required for preparing a software application to run and operate in a specific environment. It involves installation, configuration, testing and making changes to optimize the performance of the software. It can either be carried out manually or through automated systems.

2.2.3 Operate

2.3 Integration

3 What is Deployment?

3.1 Release

The release activity follows from the completed development process, and is sometimes classified as part of the development process rather than deployment process. It includes all the operations to prepare a system for assembly and transfer to the computer system(s) on which it will be run in production. Therefore, it sometimes involves determining the resources required for the system to operate with tolerable performance and planning and/or documenting subsequent activities of the deployment process.

3.2 Installation and activation

For simple systems, installation involves establishing some form of command, shortcut, script or service for executing the software (manually or automatically). For complex systems it may involve configuration of the system – possibly by asking the enduser questions about its intended use, or directly asking them how they would like it to be configured – and/or making all the required subsystems ready to use. Activation is the activity of starting up the executable component of software for the first time (not to be confused with the common use of the term activation concerning a software license, which is a function of Digital Rights Management systems.) In larger software deployments on servers, the main copy of the software to be used by users - "production" - might be installed on a production server

in a production environment. Other versions of the deployed software may be installed in a test environment, development environment and disaster recovery environment. In complex continuous delivery environments and/or software as a service systems, differently-configured versions of the system might even exist simultaneously in the production environment for different internal or external customers (this is known as a multi-tenant architecture), or even be gradually rolled out in parallel to different groups of customers, with the possibility of cancelling one or more of the parallel deployments. For example, Twitter is known to use the latter approach for A/B testing of new features and user interface changes. A "hidden live" group can also be created within a production environment, consisting of servers that are not yet connected to the production load balancer, for the purposes of blue-green deployment.

3.2.1 Deactivation

Deactivation is the inverse of activation, and refers to shutting down any already-executing components of a system. Deactivation is often required to perform other deployment activities, e.g., a software system may need to be deactivated before an update can be performed. The practice of removing infrequently used or obsolete systems from service is often referred to as application retirement or application decommissioning.

3.3 Uninstallation

Uninstallation is the inverse of installation. It is the removal of a system that is no longer required. It may also involve some reconfiguration of other software systems in order to remove the uninstalled system's dependencies.

3.4 Update

The update process replaces an earlier version of all or part of a software system with a newer release. It commonly consists of deactivation followed by installation. On some systems, such as on Linux when using the system's package manager, the old version of a software application is typically also uninstalled as an automatic part of the process. (This is because Linux package managers do not typically support installing multiple versions of a software application at the same time, unless the software package has been specifically designed to work around this limitation.)

3.5 Built-in update

Mechanisms for installing updates are built into some software systems (or, in the case of some operating systems such as Linux, Android and iOS, into the operating system itself). Automation of these update processes ranges from fully automatic to user initiated and controlled. Norton Internet Security is an example of a system with a semi-automatic method for retrieving and installing updates to both the antivirus definitions and other components of the system. Other software products provide query mechanisms for determining when updates are available.

4 Deployment Module Requirements

4.0.1 Requests

4.0.2 Environments

5 Deployment Module Scope and Tools

5.1 Docker

Figure 2: Docker

Docker is a tool designed to make it easier to create, deploy, and run

applications by using containers. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package. By doing so, thanks to the container, the developer can rest assured that the application will run on any other Linux machine regardless of any customized settings that machine might have that could differ from the machine used for writing and testing the code.

In a way, Docker is a bit like a virtual machine. But unlike a virtual machine, rather than creating a whole virtual operating system, Docker allows applications to use the same Linux kernel as the system that they're running on and only requires applications be shipped with things not already running on the host computer. This gives a significant performance boost and reduces the size of the application.

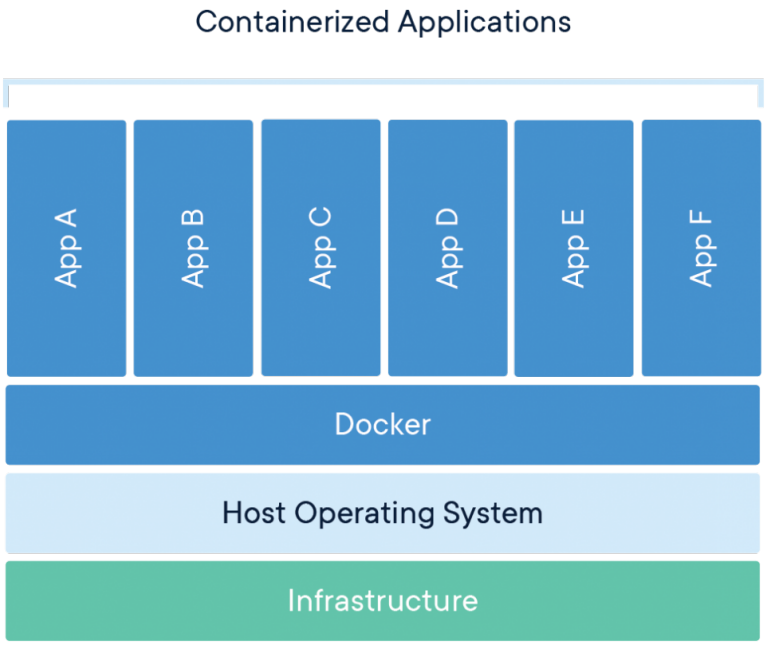

Figure 3: Container Architecture

5.1.1 Containers

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

5.1.2 Dockerfile

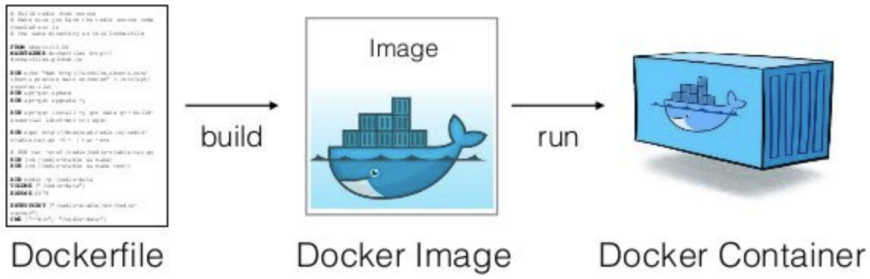

Figure 4: What can does Dockerfile ?

Docker can build images automatically by reading the instructions from a Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.

5.1.3 Docker Compose

5.1.4 Docker - Mule Connection

5.2 JSon

JSON is a lightweight format for storing and transporting data. It stands for JavaScript Object Notation. To communicate with other modules, JSON

data format used in this project.

Specifications and properties of JSON formats for communication explained below ;

5.3 Python

6 What We Did?

7 Workflow by Procedures

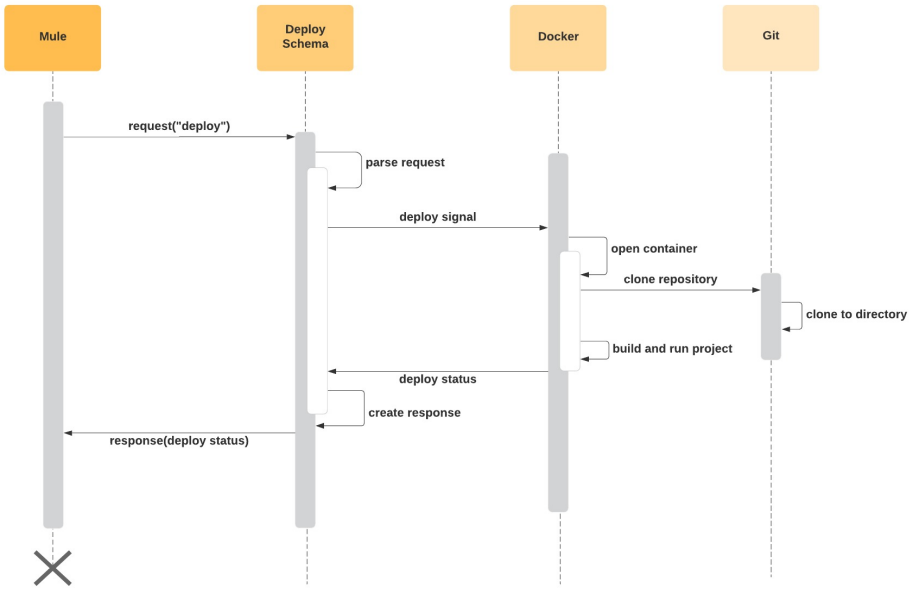

Figure 5: Deploy Diagram

8 Measurement

adsaksldaskldj

9 References

adsaksldaskldj