Day: 1

What is Docker?

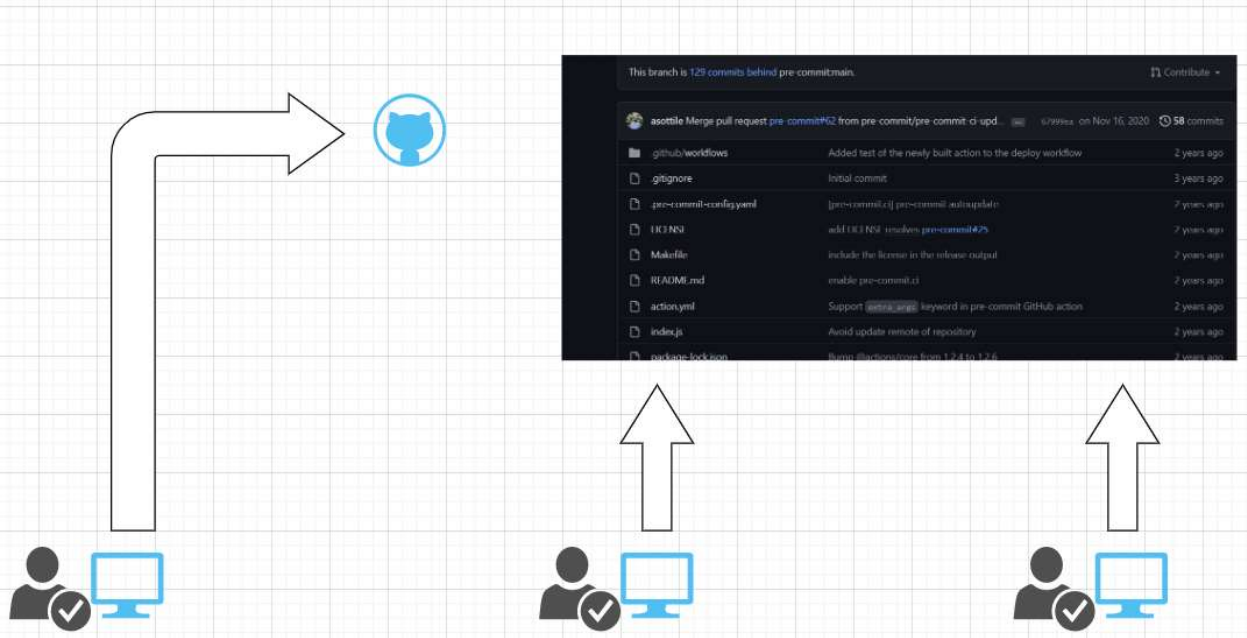
Docker is a **software packaging tool to packages the software as a container** which uses the OS virtualization. Docker was Introduced in 2013 by Docker Inc.

Docker can package an application and its dependencies in a virtual container that can run on any Linux, windows, or macOS computer.

Why Docker? What Problem does it solve?

In a Realtime, we have the various environments namely Dev, QA, UAT and Production. Usually, Developers will write the source code and all the Developers source code will be merged to an integrated environment that is called the Dev, QA, UAT PROD environment.

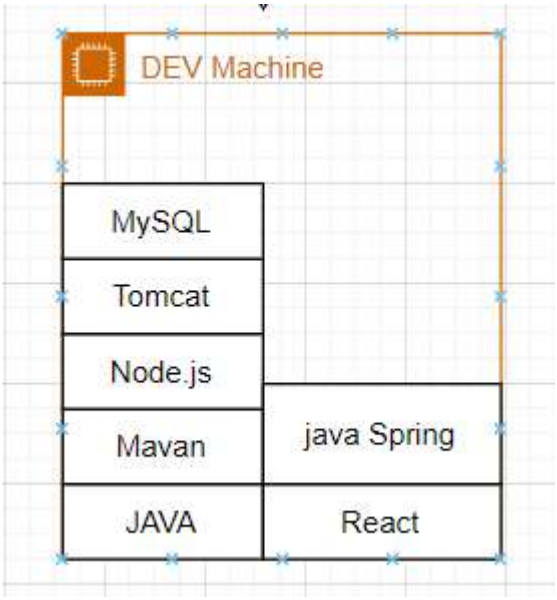
Usually, developers will write the code in their local machine-like Laptop, PC and code will be pushed to the feature specific Branch on which feature the developer is working on.



Let's assume the developers working for a project called Retail Ecommerce, which is following the N-Tier Architecture in client-server model and using the following technical stack:

When all the developers written code will be merged on to dev environment, we need to do the following activities:

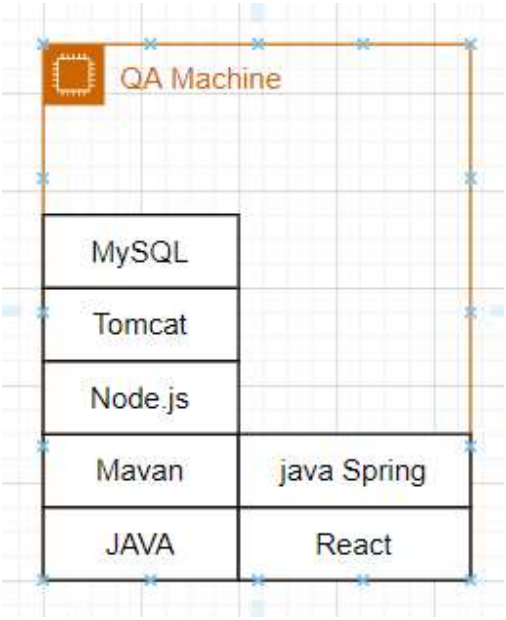
- Create a Dev Environment Machine
- Install the required technical stack set i.e., JDK, Tomcat, Node.js, MySQL etc.
- Pull the Source Code from the dev-release branch
- Build the Source Code
- Deploy the war/jar file came in Tomcat Server which came as an outcome of the source code build.



When the code is integrated on dev environment and unit/integration testing done by the developers on all these merged features, it will be released to the QA environment.

In Order to test the Integrated features on QA environment by the testers we need to the following activities:

- Create a QA Environment Machine
- Install the required technical stack set i.e., JDK, Tomcat, NodeJS, MySQL etc.
- Merge the Dev release branch code into QA release Branch Code
- Pull the Source Code from the QA-release branch
- Build the Source Code
- Deploy the war/jar file came in Tomcat Server which came as an outcome of the source code build



If we Observer

When any new environment is added, we need to set up the environment specific machine with all the required technical software stack installing on it.

When any new environment is added, we need to set up the environment specific machine all the required technical software stack installing on it.

When any environment is changing then we are pulling the source code, building the code and deploying the code. This process we are doing iteratively which causes the below problems:

1. Setting up environment with the required technical stack installation will takes times and it's a repetitive process when any new environment is added.
2. As we are pulling the source code, building it again and again on each and every environment, there is a chance of build issues/deployment issues if the stable code is

not pulled, not build properly. That's the reason may be the dev-release branch code might works well on the dev machine but the qa-release branch code might have some issues on QA machine.

3. We are violating the concepts of the cloud Agnostic if we would like to make our code as cloud agnostic. Means the same code should have to run on any cloud environment without building it again and again.

Cloud Agnnostic will suggest the following Principles:

1. WORA
2. Build Once and the same build should be deployable on anywhere and any environment.
3. KIS

The about 3 problems will be solved and cloud Agnostic benefits we can achieve using the Docker.

Solution

Docker solved the Problem with the Concept Called **Containerization**.

DEV Environment

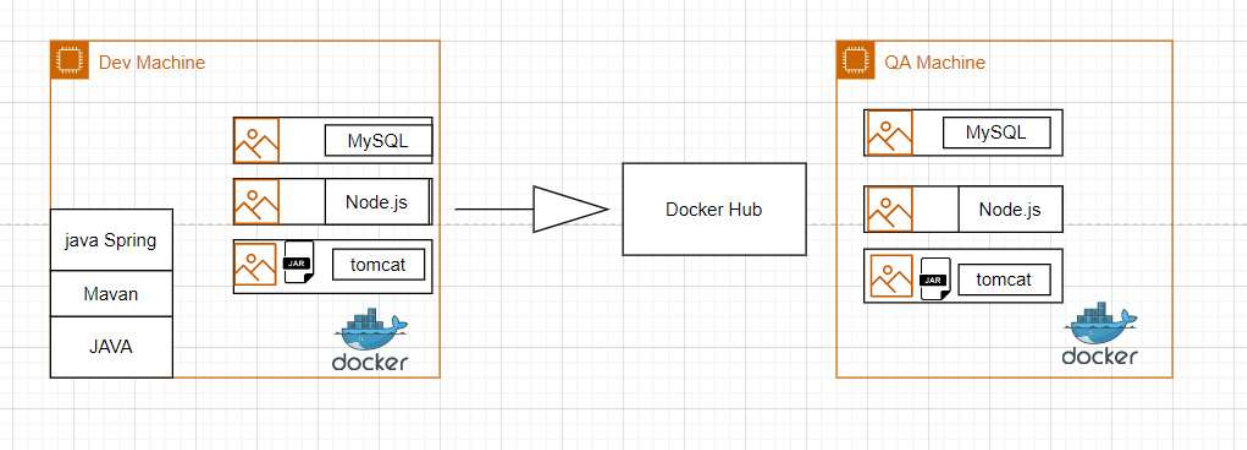
When all the developers written code will be merged on to dev environment, we need to do the following activities:

- Create a Dev Environment Machine
- Install the required technical stack set i.e, JDK, Maven.
- Pull the Source Code from the dev-release branch
- Build the Source Code
- Install the docker
- Create the docker image for the spring application with the technical stack OpenJDK, Tomcat, MySQL
- Create the docker image for the React Application with the technical stack NodeJS
- Run the Spring application and react application images as containers.
- Access the application running inside the container.
- Push the docker images of the spring application and react application to the docker registry.

QA Environment

In Order to test the Integrated features on QA environment by the testers we need to the following activities:

- Create a QA Environment Machine
- Install the docker
- Pull the spring application and react application images from docker registry
- Run the spring application and react application images as containers.
- Access the application running inside the container.



H/W

- Read EC2, Docker, Docker Vs Virtual Machine and prepare the Pdf file and push to GitHub
- Download Mobaxterm

  https://mobaxterm.mobatek.net/download-home-edition.html

- Download Docker Desktop
- https://www.docker.com/