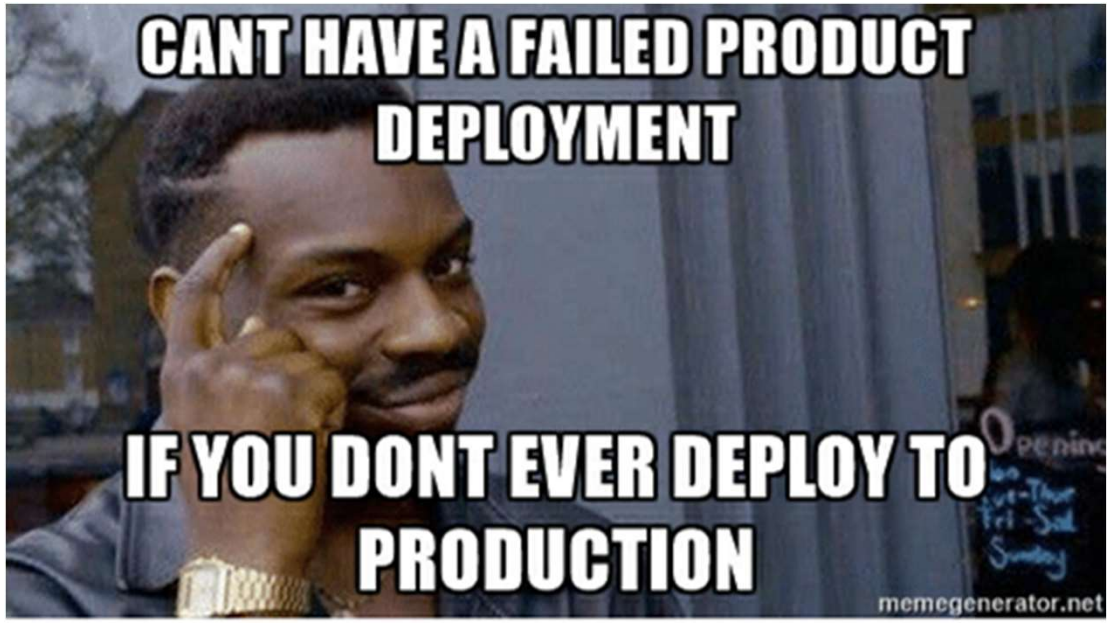
Continuous Integration & Continuous Delivery-Deployment

aryya.widigdha@yahoo.com

Scenario

You are developing web application for finance with 2 of your friends in separated location. There are a lot of features to be done in 15 days. The user want to get progress every 3 days. How will you manage to develop and deliver your project?


BRACE YOURSELVES
A DEPLOY TO PRODUCTION IS COMING
memegenerator.net


CANT HAVE A FAILED PRODUCT DEPLOYMENT
IF YOU DONT EVER DEPLOY TO PRODUCTION
memegenerator.net


WORKED FINE IN DEV
OPS PROBLEM NOW
memegenerator.net


WHAT DO WE SAY TO DEPLOYING TO PRODUCTION ON A FRIDAY
NOT TODAY

Agenda

- Continuous Integration & Delivery-Deployment
- Introduction to Containerization
- Introduction to Github Actions
  - Merge request
  - Automatic test
  - Autodeploy

Continuous Integration & Deployment

CI/CD is a method to frequently deliver apps to customers by introducing automation into the stages of app development. The main concepts attributed to CI/CD are continuous integration, continuous delivery, and continuous deployment.

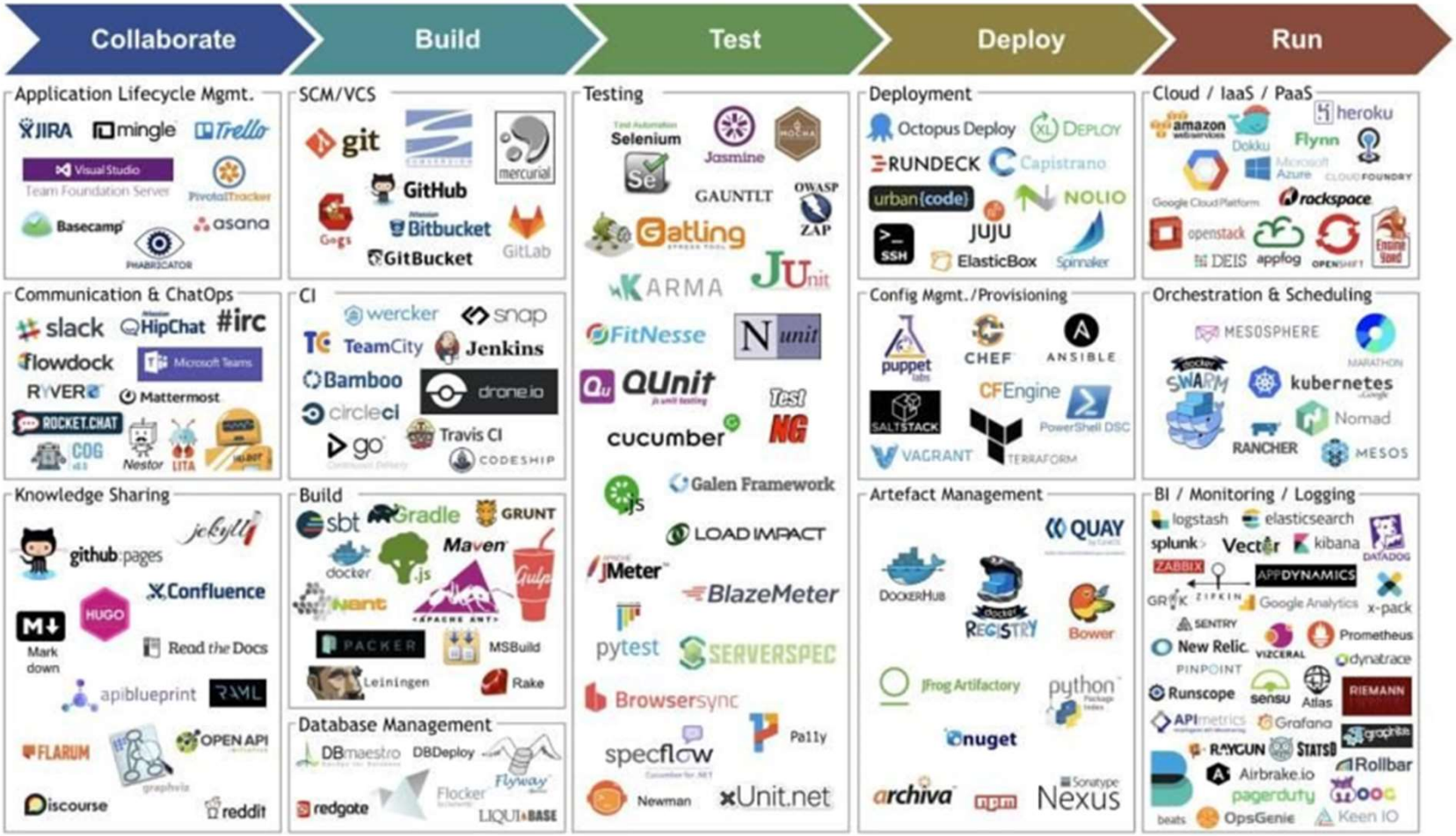
Benefit of CI/CD

- Faster software delivery
- More time to focus on development
- Reduce testing cost
- Reduce mistake on deploying software on production

What You Need of CI/CD

- Additional flow
- Additional configuration
- Convention
- Additional worker to do the “manual job”

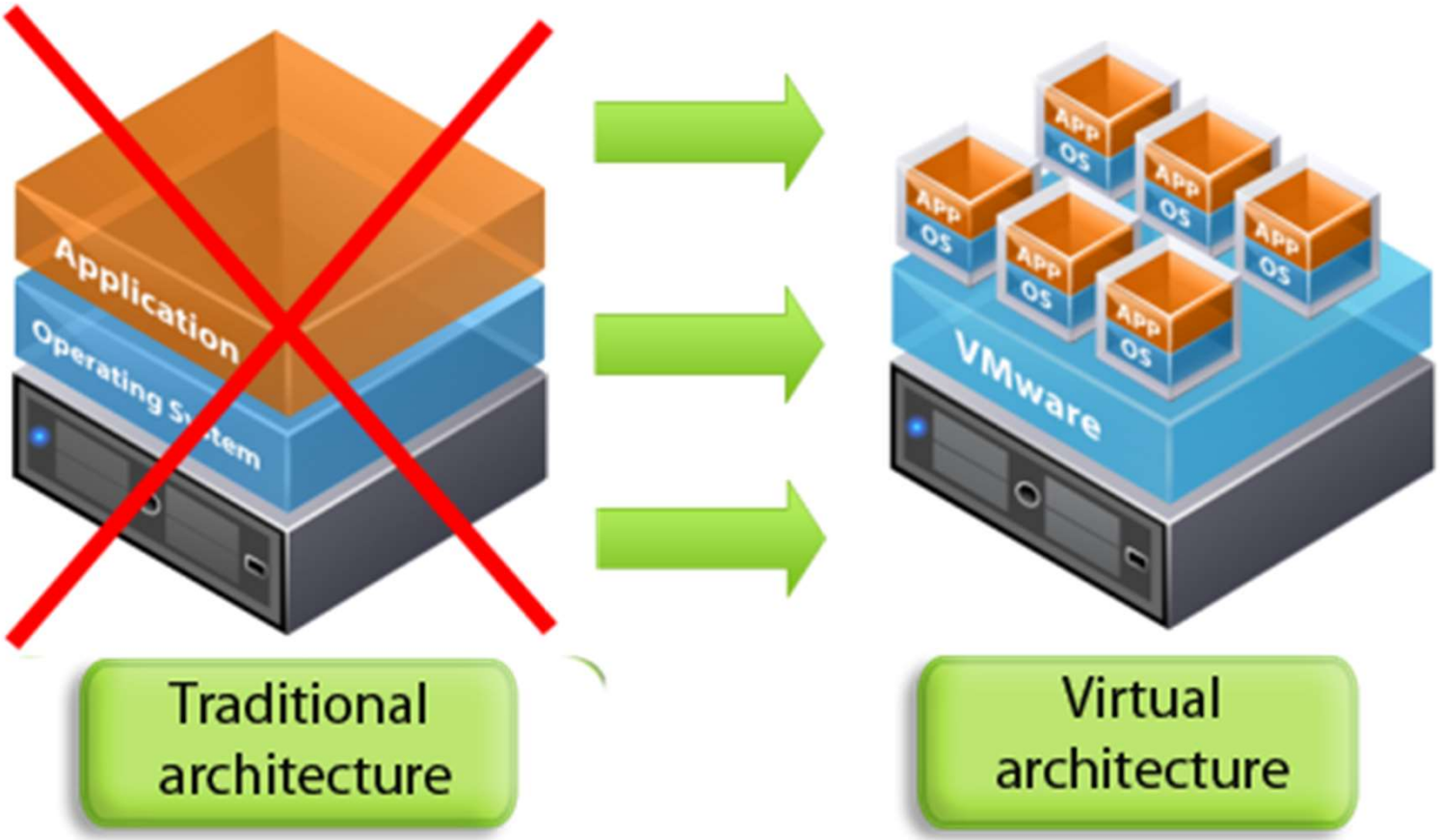
CI/CD Stack Landscape



Deployment Style

- Bare metal
- Over VM
- Container

Virtualization

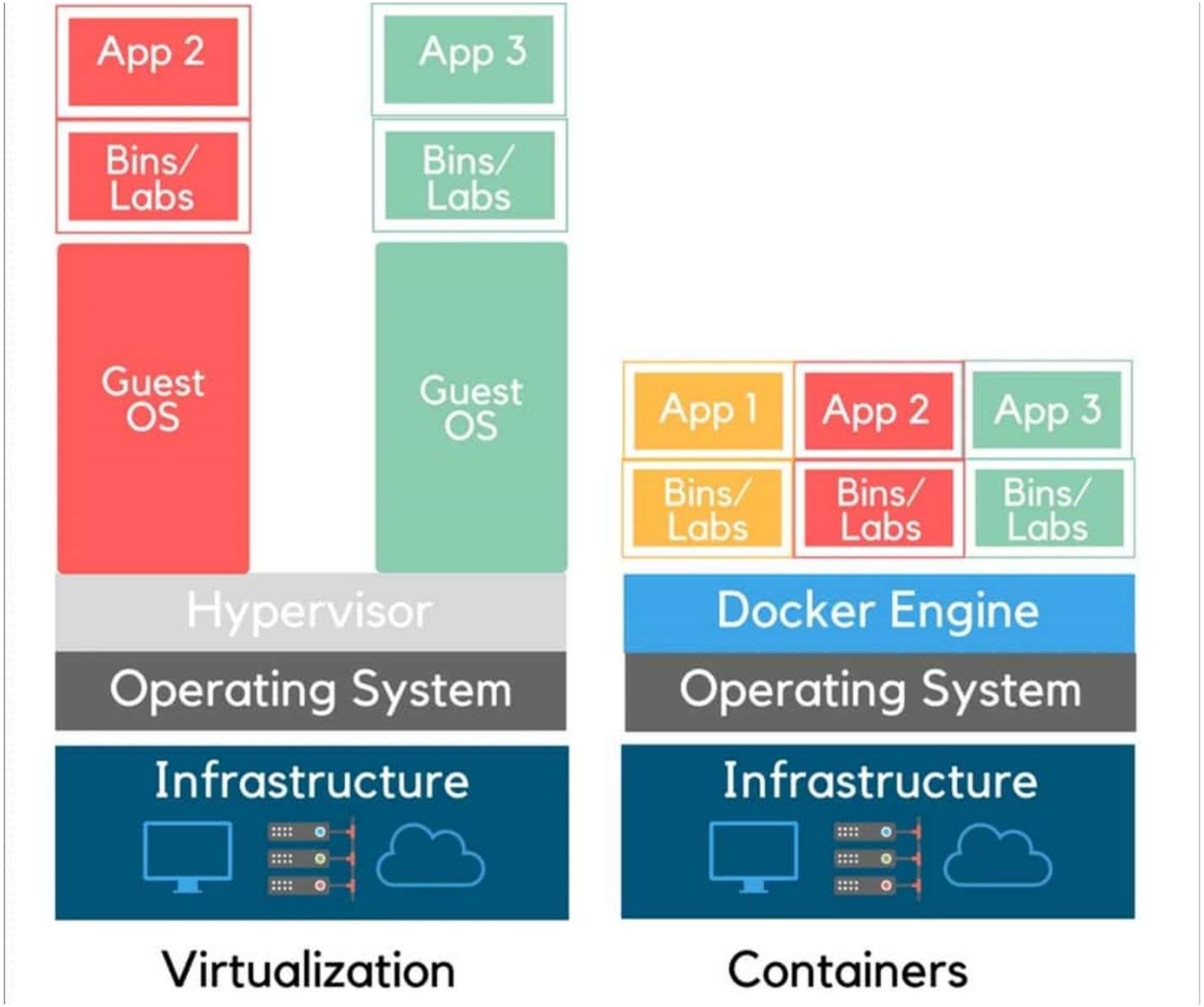


http://www.oldanygroup.com/vmware-virtualization-about-58/

Why Virtualization

- Resource utilization
- Isolation
- Replicable environment

Container



https://www.parkmycloud.com/blog/application-containerization/

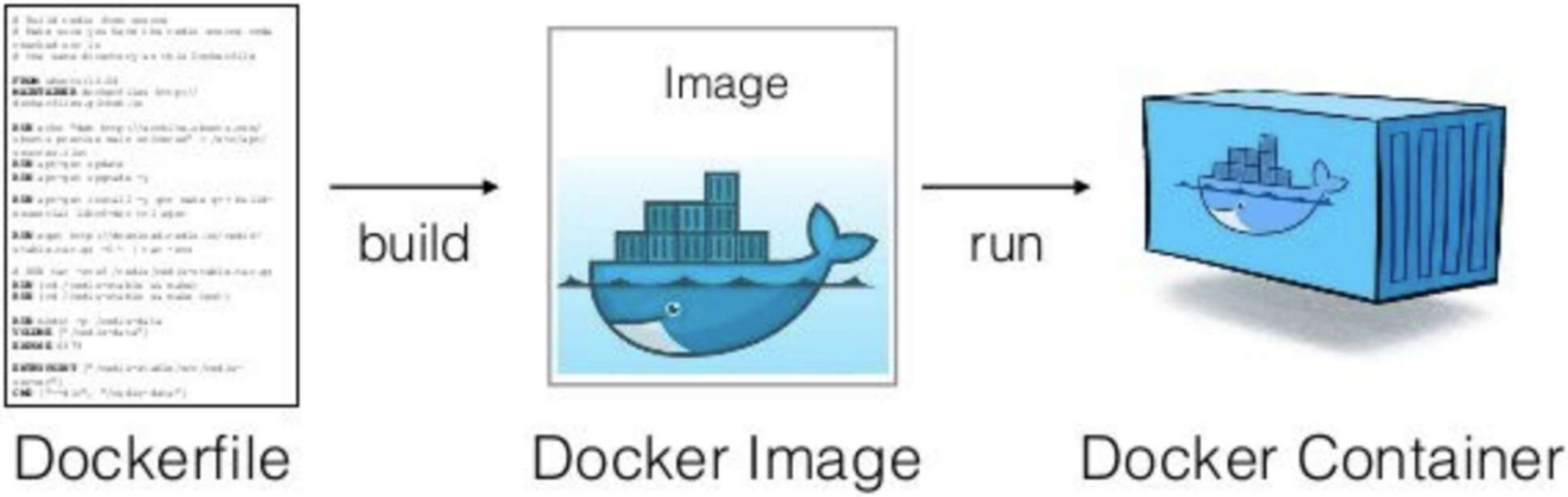
Docker

Docker is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels

Docker Terminology

- Docker registry
- Docker daemon
- Docker image
- Docker container
- Dockerfile

Understanding Docker Life Cycle



https://medium.com/platformer-blog/practical-guide-on-writing-a-dockerfile-for-your-application-89376f88b3b5

Common Docker Command

- Docker pull
- Docker build
- Docker run
- Docker stop
- Docker exec
- Docker ps
- Docker images
- etc

Benefit of using Docker

- Consistency & isolation
- Cost effective and fast deployment
- Able to run anywhere. Name some cloud provider?
- Repeatability & automation
- Collaboration, Modularity, and Scalability

Let’s Start

- Run simple web application
- Understanding docker life cycle

Docker Hub

DockerCon Live is coming this May 27th! Sign up to learn more.

docker hub | nginx | Explore | Pricing | Sign In | Sign Up

Explore > nginx

NGINX

nginx

Docker Official Images

Official build of Nginx.

1B+

Container | Linux | IBM Z | mips64le | ARM | ARM 64 | PowerPC 64 LE | 386 | x86-64

Application Infrastructure | Official Image

Copy and paste to pull this image

```
docker pull nginx
```

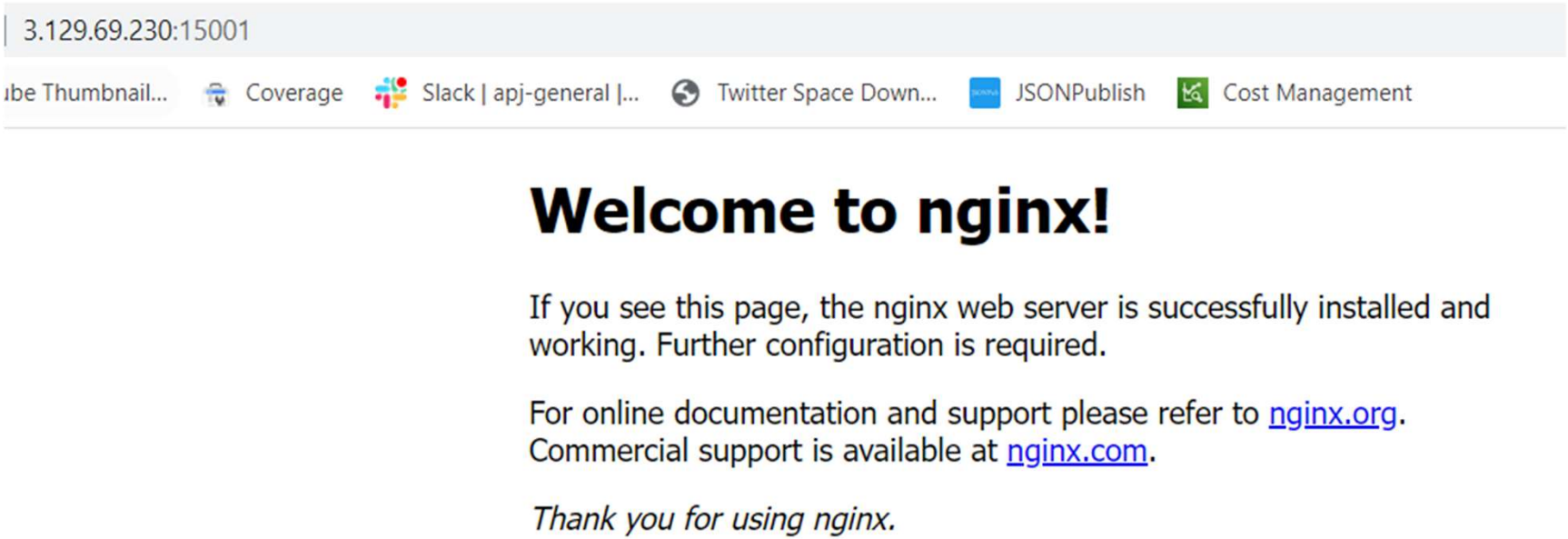
View Available Tags

Description | Reviews | Tags

Run Simple Webapp

https://hub.docker.com/_/nginx

- Docker pull nginx
- docker run -p 15001:80 nginx



Dockerfile

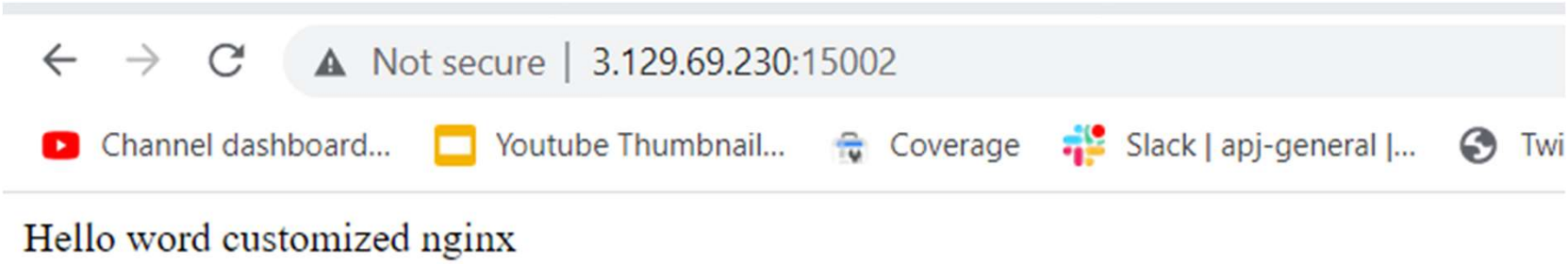
```
FROM nginx
COPY static-html-directory
/usr/share/nginx/html
```

Docker Build

Docker build -t customized_nginx .

Docker Run

docker run -p 15002:80 customized_nginx



Docker Compose

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application’s services. Then, with a single command, you create and start all the services from your configuration. To learn more about all the features of Compose, see the list of features.

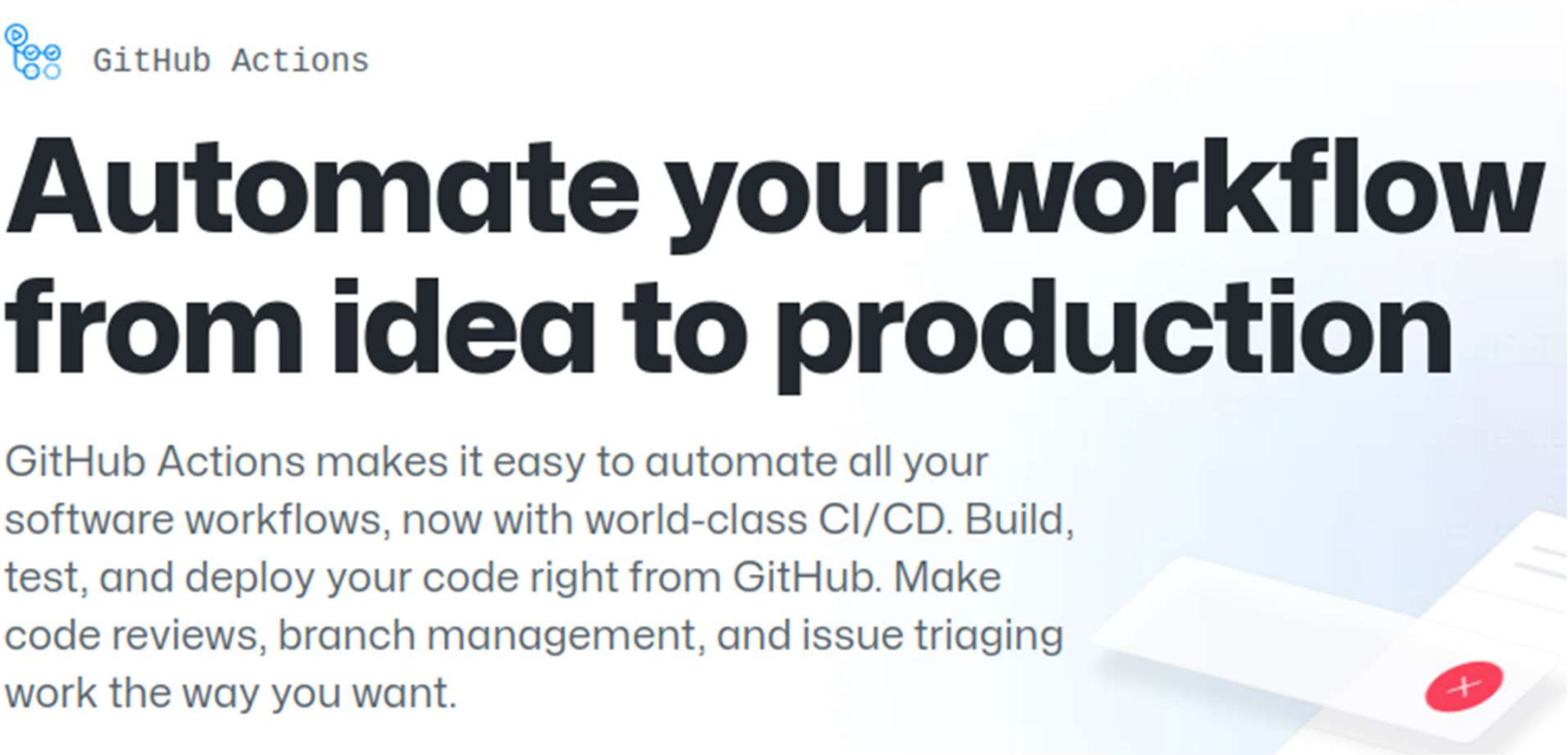
Docker Compose Three Step

- Define your app's environment with a Dockerfile so it can be reproduced anywhere.
- Define the services that make up your app in docker-compose.yml so they can be run together in an isolated environment.
- Run docker-compose up and Compose starts and runs your entire app.

Docker Compose Example

```yaml
version: "3.9"  # optional since v1.27.0
services:
  web:
    build: .
    ports:
      - "5000:5000"
    volumes:
      - .:/code
      - logvolume01:/var/log
    links:
      - redis
  redis:
    image: redis
volumes:
  logvolume01: {}
```

Github Action



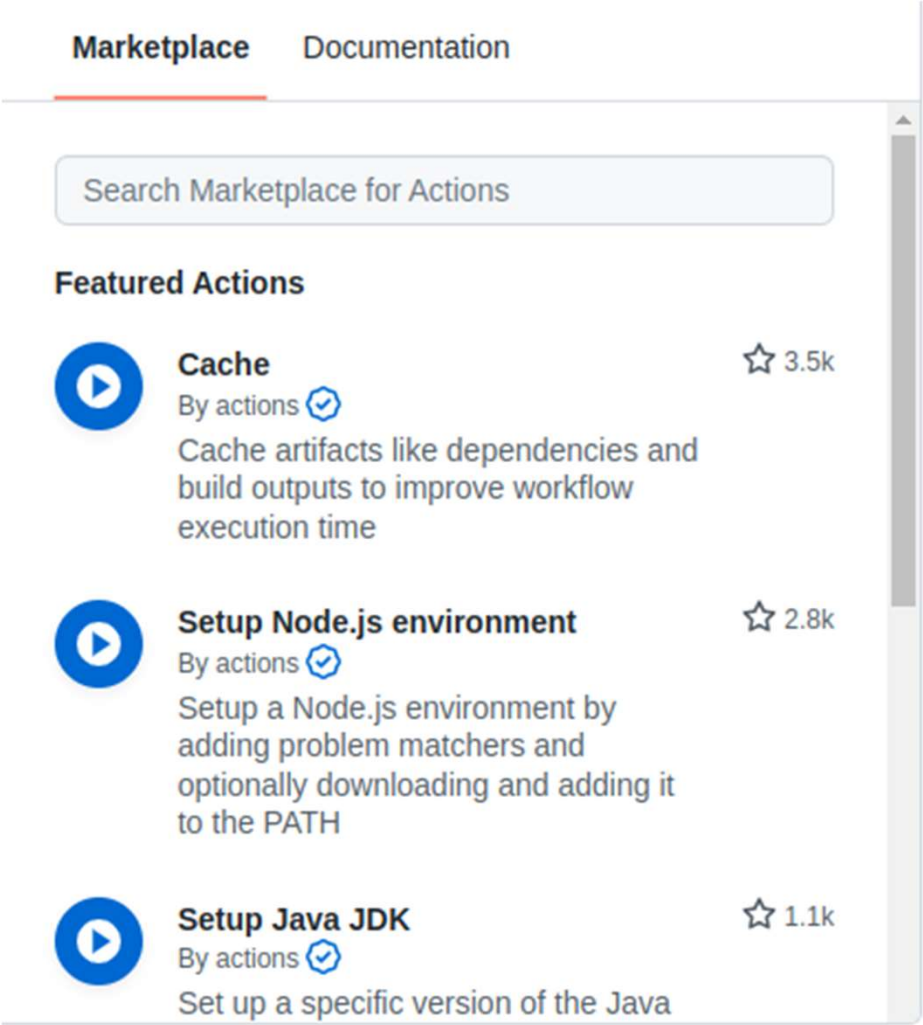
Github Action Components

Github Action defined by conventional file called .github/workflows/main.yml This file consist of:

1. When to trigger workflow
   a. merge
   b. push
   c. schedule
2. What action in sequence
3. Secrets

https://olex.biz/2020/01/deploying-to-docker-nginx-via-gitlab-ci/

Github Action Marketplace



Secret

Secrets | Variables

New repository secret

Environment secrets

Manage environments

There are no secrets for this repository's environments.

Repository secrets

HOST	Updated 3 minutes ago
PASSWORD	Updated 3 minutes ago
PORT	Updated 3 minutes ago
USERNAME	Updated 3 minutes ago

Github Action Example

36 lines (36 sloc) | 1.13 KB

Raw Blame

```yaml
name: "RPL-2023-CICD Demo"
on:
  push:
    branches:
    - main
jobs:
  build:
    name: Build
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@master
    - name: executing remote ssh commands using password
      uses: appleboy/ssh-action@v0.1.8
      with:
        host: ${{ secrets.HOST }}
        username: ${{ secrets.USERNAME }}
        password: ${{ secrets.PASSWORD }}
        port: ${{ secrets.PORT }}
        script: mkdir -p /home/rplcicd/13512043
    - name: copy file via ssh password
      uses: appleboy/scp-action@master
      with:
        host: ${{ secrets.HOST }}
        username: ${{ secrets.USERNAME }}
        password: ${{ secrets.PASSWORD }}
        port: ${{ secrets.PORT }}
        source: "app.py"
        target: "/home/rplcicd/13512043/"
    - name: executing remote ssh commands using password
```

Github Action Execution

All workflows

Showing runs from all workflows

Filter workflow runs

4 workflow runs

Event Status Branch Actor

Send process to background RPL-2023-CICD Demo #4: Commit 4528ac3 pushed by aryyaid	main	5 days ago 26s	...
Redirect log to background RPL-2023-CICD Demo #3: Commit 429e605 pushed by aryyaid	main	5 days ago 1m 54s	...
Giving initial action RPL-2023-CICD Demo #2: Commit 159bb74 pushed by aryyaid	main	5 days ago 1m 37s	...
Giving initial action RPL-2023-CICD Demo #1: Commit 159bb74 pushed by aryyaid	main	5 days ago 20s	...

CONTINUOUS INTEGRATION

Code Check on Commit

```yaml
  build:
    name: Build
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@master
    - name: Do python linter test check
      run: |
        sudo apt install python3-pip -y
        python -m pip install pylint
        pip install flask
        pylint app.py --disable=C0114,C0116
```

CONTINUOUS DELIVERY & DEPLOYMENT

Deploy to EC2

1. Create folder for your student id
2. Run flask application using port based on your student id
3. Check in browser

Pipeline Definition

36 lines (36 sloc) | 1.13 KB

Raw Blame

```yaml
name: "RPL-2023-CICD Demo"
on:
  push:
    branches:
    - main
jobs:
  build:
    name: Build
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@master
    - name: executing remote ssh commands using password
      uses: appleboy/ssh-action@v0.1.8
      with:
        host: ${{ secrets.HOST }}
        username: ${{ secrets.USERNAME }}
        password: ${{ secrets.PASSWORD }}
        port: ${{ secrets.PORT }}
        script: mkdir -p /home/rplcicd/13512043
    - name: copy file via ssh password
      uses: appleboy/scp-action@master
      with:
        host: ${{ secrets.HOST }}
        username: ${{ secrets.USERNAME }}
        password: ${{ secrets.PASSWORD }}
        port: ${{ secrets.PORT }}
        source: "app.py"
        target: "/home/rplcicd/13512043/"
    - name: executing remote ssh commands using password
```

Exercise 1

1. Create new repository and follow reference on https://github.com/aryyaid/RPL-2023-CICD
2. Configure the code and workflow definition to deploy to certain folder and port based on your NIM. e.g /home/rplcicd/13512043/ and flask port 12043

Form link:
K1: https://s.id/1EexU
K2: https://s.id/1Eey6
K3: https://s.id/1Eeyo

Exercise 2

From **exercise 1**, add additional workflow to run **linter test** using **pylint** on another branch except master

Question?

Reference

- https://vexxhost.com/resources/tutorials/how-to-install-and-use-docker-on-ubuntu-14-04/
- https://docs.gitlab.com/runner/shells/index.html#shell-profile-loading
- https://docs.docker.com/compose/compose-file/
- https://docs.gitlab.com/runner/install/linux-manually.html