Progress Big Project untuk hari ini:

1. Saya mencoba mencari tahu source code yang cocok selain pesbuk, wordpress dan landing page

2. Saya mendpt source code berbasis ci ( codeigneter) ada beberapa contoh yang sesuai

3. Saya menjalankan secara localhost untuk aplikasi blog, aplikasi berbasis ci, dan profil sekolah. Ketiganya jalan dilokal.

4. Ketika saya mencoba mencoba membuat dockefile untuk dijalankan di EC2 dengan sistem container. 1 buah blog sudah jalan.

http://52.15.48.203:85/

MBLOG

MBLOG - Source Code untuk Blogger

Open Source source code untuk personal blog, simple, elegan, full responsif, dan mudah di customize.

Dockerfile

5. Dockerfile

FROM php:7.1.20-apache

RUN apt-get -y update --fix-missing

RUN apt-get upgrade -y

# Install useful tools

RUN apt-get -y install apt-utils nano wget dialog

# Install important libraries

RUN apt-get -y install --fix-missing apt-utils build-essential git curl libcurl3 libcurl3-dev zip

# Composer

RUN curl -sS https://getcomposer.org/installer | php -- --install-dir=/usr/local/bin --filename=composer

# Install xdebug

RUN pecl install xdebug-2.5.0

RUN docker-php-ext-enable xdebug

# Other PHP7 Extensions

RUN apt-get -y install libmcrypt-dev

RUN docker-php-ext-install mcrypt

RUN apt-get -y install libsqlite3-dev libsqlite3-0 mysql-client

RUN docker-php-ext-install pdo\_mysql

RUN docker-php-ext-install pdo\_sqlite

RUN docker-php-ext-install mysqli

RUN docker-php-ext-install curl

RUN docker-php-ext-install tokenizer

RUN docker-php-ext-install json

RUN apt-get -y install zlib1g-dev

RUN docker-php-ext-install zip

RUN apt-get -y install libicu-dev

RUN docker-php-ext-install -j$(nproc) intl

RUN docker-php-ext-install mbstring

RUN apt-get install -y libfreetype6-dev libjpeg62-turbo-dev libpng-dev

RUN docker-php-ext-configure gd --with-freetype-dir=/usr/include/ --with-jpeg-dir=/usr/include/

RUN docker-php-ext-install -j$(nproc) gd

# Enable apache modules

RUN a2enmod rewrite headers

COPY . /var/www/html

EXPOSE 80

Collaps

Progress Big Project untuk hari ini:

( Progress untuk hari Kamis 16 Mei 2019)

1. Saya mencoba membuat Dockerfile untuk 2 aplikasi yaitu mtechno & sneaky

Ini untuk pengganti landing page. Web dasar dengan html, css, js. Hanya perlu nginx saja.

Sneaky web

FROM nginx:alpine

COPY . /usr/share/nginx/html

Sebetulnya hampir sama dengan yang mblog.

2. Membuat database di RDS untuk mtechno. Simulasi untuk masukin komentar tercatat masuk di database( diakses dg Mysqlfront).

3. Build images untuk keduanya. Jadi sudah membuat 3 images custom.

4. Menjalan container untuk ketiga apps tersebut

5. Semua repository untuk 3 apss udah dibuat. Nanti dipakai untuk jenkins.

6. Akses untuk ke web nya : di port 86,91,92 di IP: 52.15.48.203

http://52.15.48.203:91/

http://52.15.48.203:92/

<http://52.15.48.203:86/>

Progress Big Project untuk hari ini:

( Progress untuk hari Kamis 17 Mei 2019):

1. Saya melakukan deploy web apps mblog, mtechno, dan sneaky lewat jenkins dan berhasil semua.

2. Awalnya ada kesulitan kenapa ketika web apps sudah online tidak bisa diakses. Ternyata ada di .htaccess

<IfModule mod\_rewrite.c>

Options -Indexes

RewriteEngine On

RewriteCond $1 !^(index\\.php|resources|robots\\.txt)

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule ^(.+)$ index.php?/$1 [L,QSA]

</IfModule>

3. Edit di bagian application/config/config.php >>

$config['base\_url'] = ((isset($\_SERVER['HTTPS']) && $\_SERVER['HTTPS'] == "on") ? "https" : "http");

$config['base\_url'] .= "://".$\_SERVER['HTTP\_HOST'];

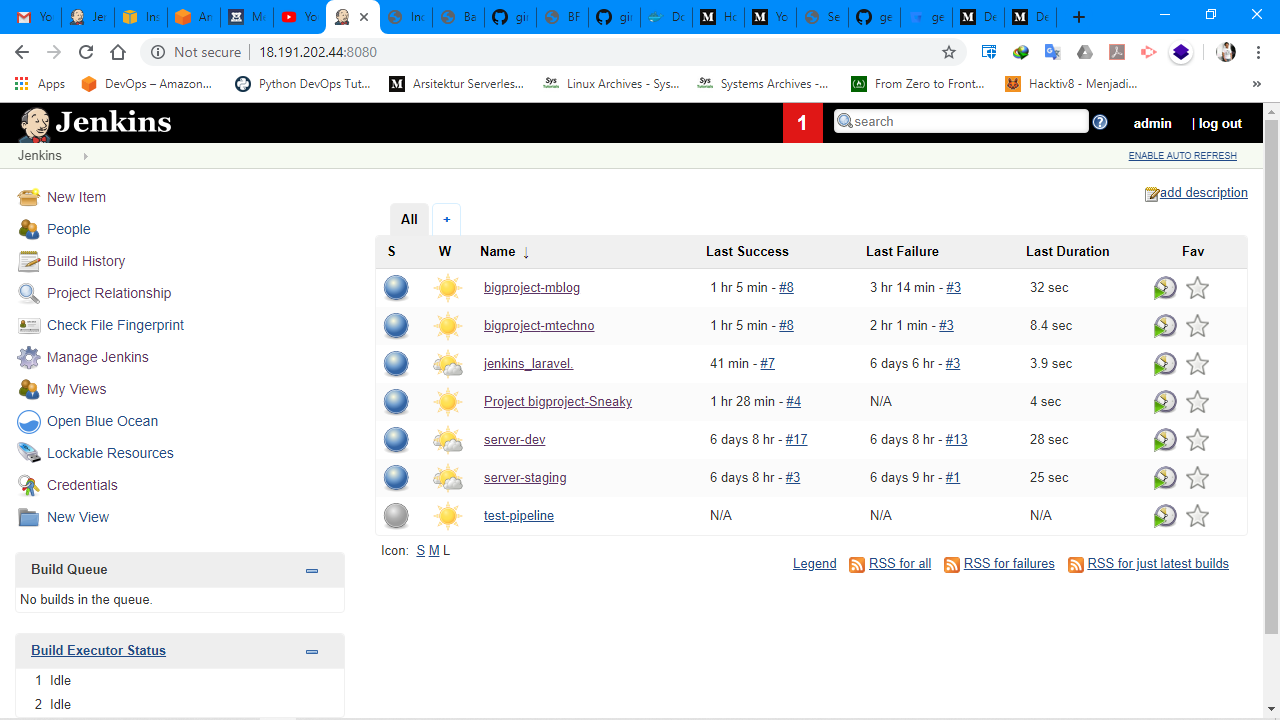
$config['base\_url'] .= str\_replace(basename($\_SERVER['SCRIPT\_NAME']),"",$\_SERVER['SCRIPT\_NAME']);

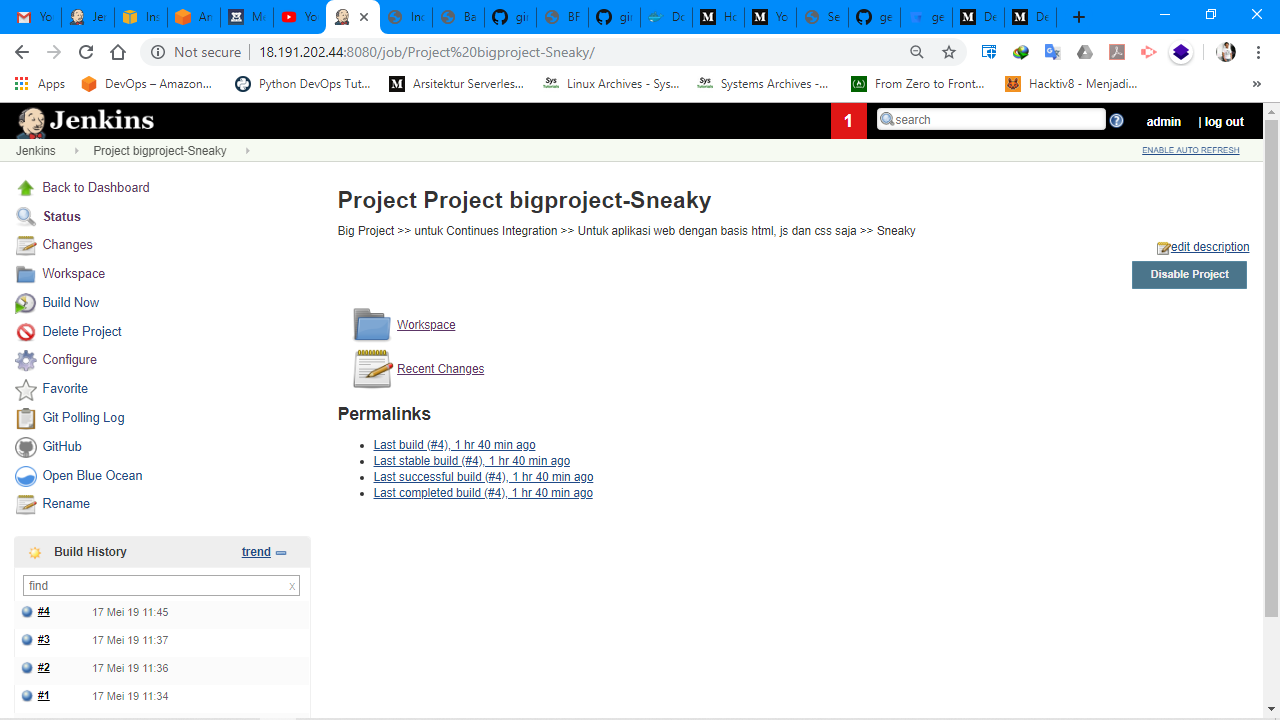
3. Membuat free project untuk setiap web apps di jenkins. Membuat repo di github selayaknya developer push/pull code ke github.

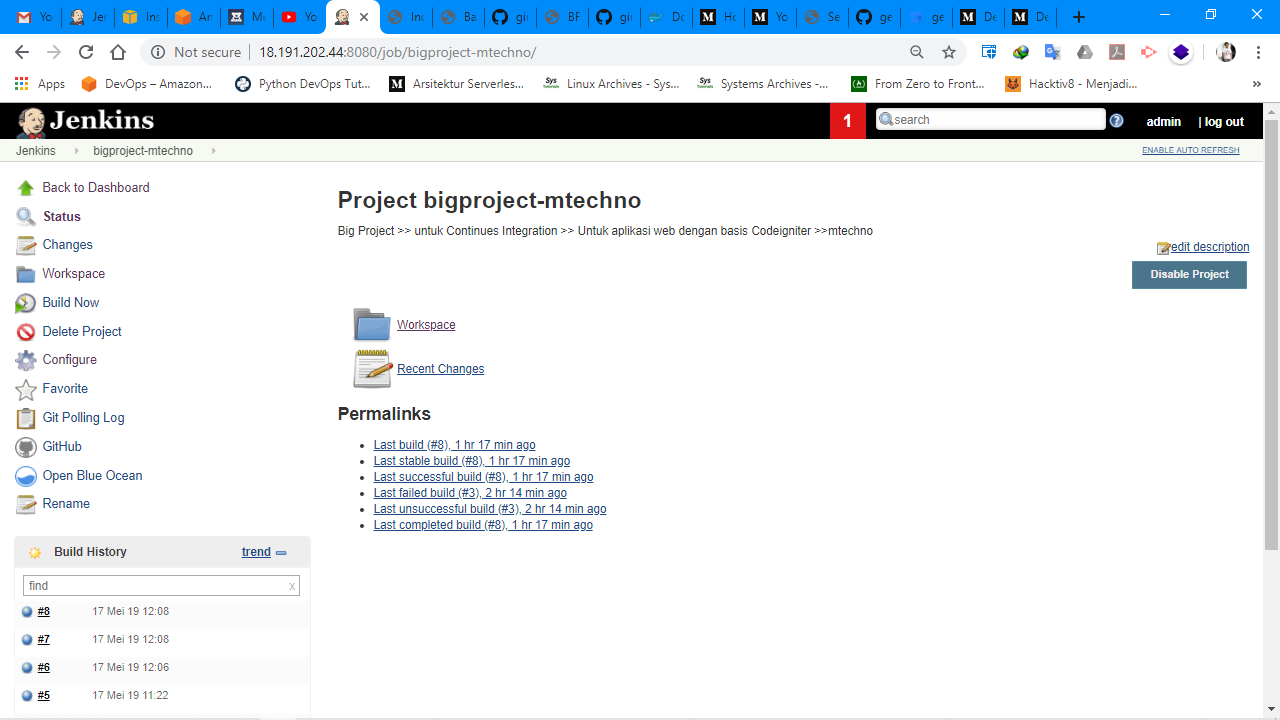
4. Konfigurasi SCM pool untuk trigger deployment saat ada push code baru ke github.

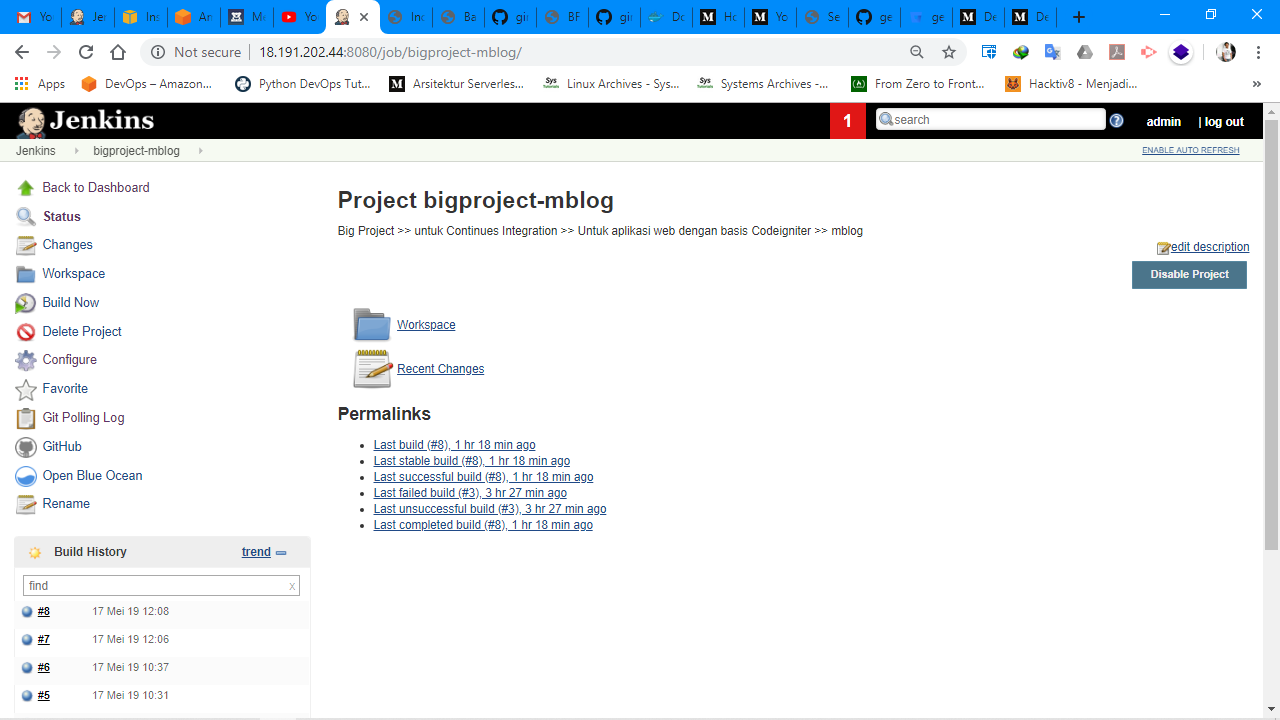
5. Semua apps bisa berjalan dengan lancar

3 Apps di Jenkins









Progress Report for today 24 May 2019:

A close up of a map

Description generated with very high confidence

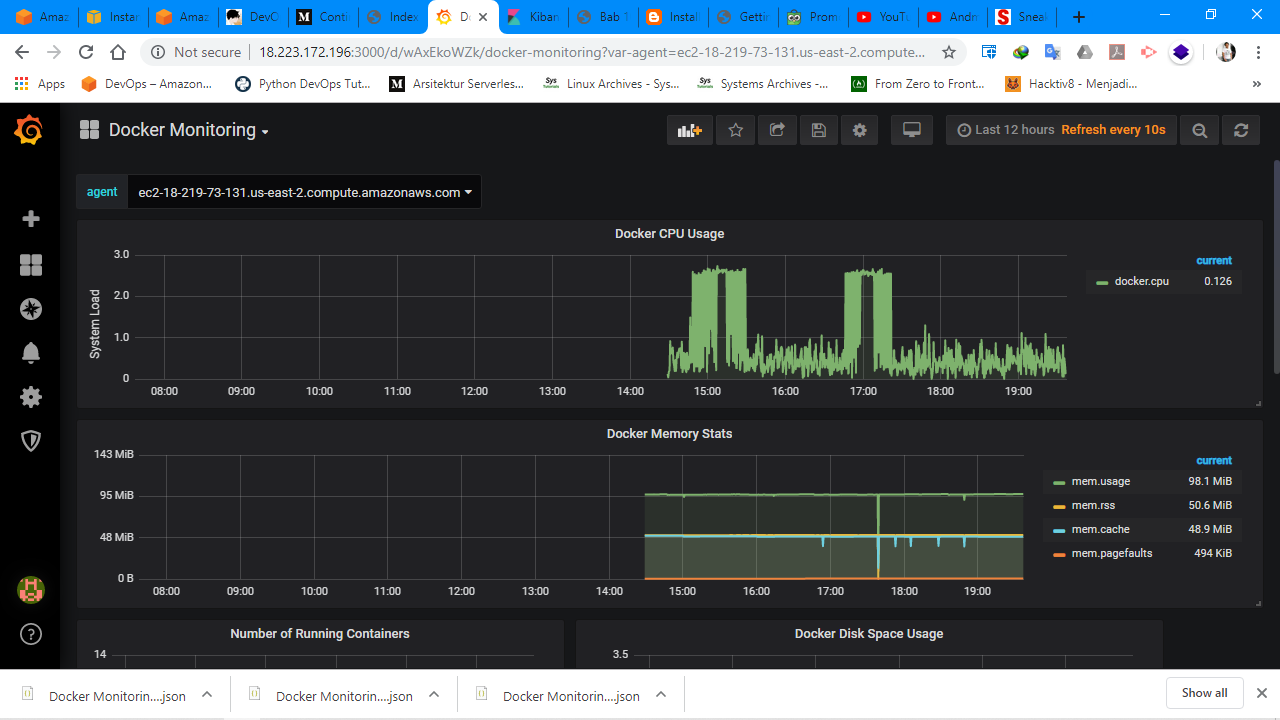
1. Melakukan konfigurasi di repo dengan 3 branch: master > staging > production.
2. Ketiga branch ini akan digunakan untuk deploy CI CD di Jenkins sesuai branchnya.
3. Melakukan deploy staging dari repo staging ke Jenkins kemudian diarahkan ke ec2 (ECS) di cluster jenkinsdev-giri .
4. Melakukan deploy dari production dari repo production dengan trigger webhook.
5. Konfigurasi DNS di route53

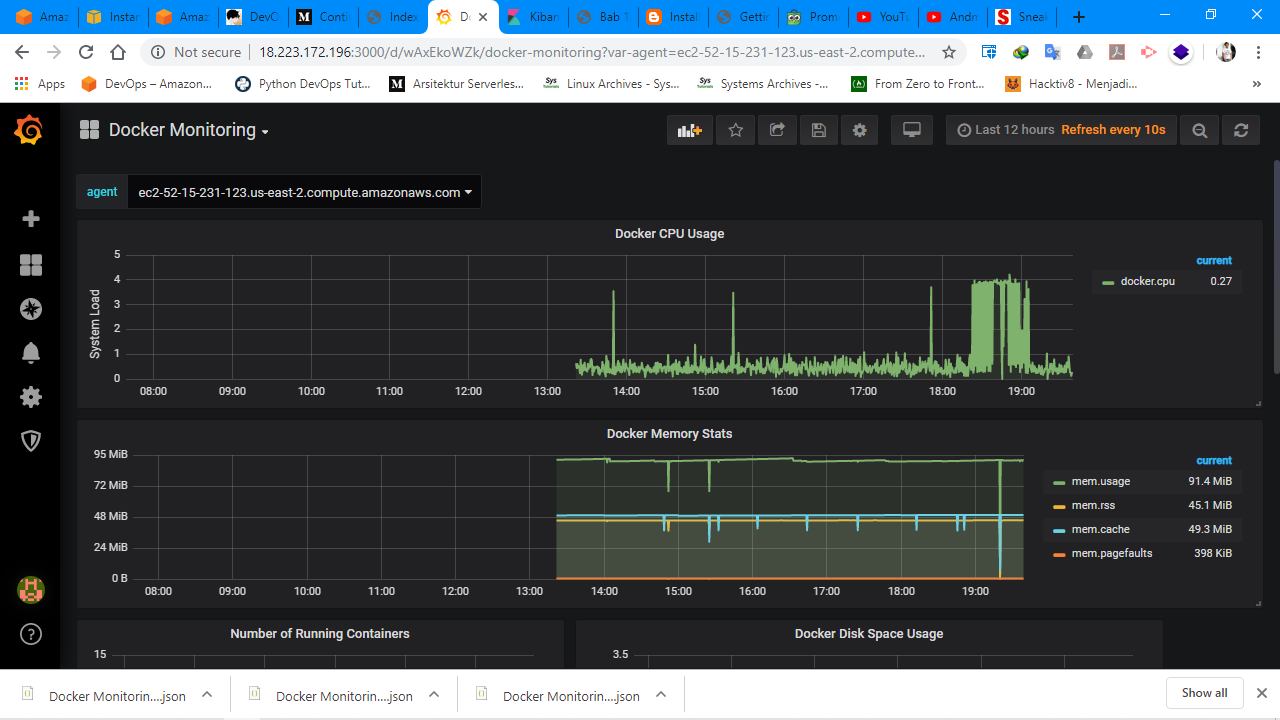
* Untuk staging, saya berikan domain awalan dev : <http://devsneaky.sdcilsy-alpha.web.id>,

<http://devmtechno.sdcilsy-alpha.web.id>, <http://devmblog.sdcilsy-alpha.web.id>

* Untuk production, saya berikan nama domain <http://mtechno.sdcilsy-alpha.web.id>, <http://mblog.sdcilsy-alpha.web.id>, <http://sneaky.sdcilsy-alpha.web.id>
* Ada backend login di kedua web mblog dan mtechno

1. Melakukan konfigurasi TIG untuk monitoring server, di server t2.medium Giri-Monitoring-TIG-ELK <http://18.223.172.196:3000> bisa diakses disini. Telegraf agent berhasil diinstall di kedua EC2 ECS ECS Instance - EC2ContainerService-jenkinspro-giri & ECS Instance - EC2ContainerService-jenkinsdev-giri
2. Untuk monitoring log, masih belum berhasil di kibana. Belum bisa Couldn't find any Elasticsearch data
3. Monitoring untuk cluster production & Staging:





Progress per task:

1. Memberikan desain topologi, sistem dan flow seperti apa yang sebaiknya dibangun. Kelebihan dan kekurangannya apa. >>>>>>>>>>>> DONE

2. Local development (laptop programmer masing-masing) >>>>>>>>>>>> DONE

3. Environment Staging untuk memastikan bahwa benar-benar sudah siap dan tanpa cacat sebelum masuk ke production. >>>>>>>>>> DONE

4. Dan environment Production environment yang tampil ke user. >>>>>>>> DONE

5. Seluruh sistem harus berbasis Container dan saling berkesinambungan proses development dan deploynya dari sisi programmer. mulai dari development di lokal, masuk ke environment staging, hingga production. >>>>>>>> DONE

6. Environment berbasis di Cloud AWS dan berbasis infrastructure as a code (jadi bisa

dibuat ulang dengan mudah infrastrukturnya). >>>>>> DONE

7. Diterapkannya sistem CI/CD (Jenkins atau semacamnya) >>>> DONE walau tidak pakai pipeline

8. Terdapat sistem sentral monitoring untuk seluruh container dan server (TIG atau semacamnya) >>>> DONE

9. Terdapat sistem sentral logging untuk seluruh container dan server (ELK atau semacamnya) >>>>> Couldn't find any Elasticsearch data

10. Code-code terkait environment (docker, dll) yang dibuat harus scalable dan bisa digunakan ulang secara bersama-sama oleh tim lain dengan mudah. Misalnya ada programmer baru yang masuk, maka programmer itu tinggal pull seluruh code dari github tanpa ubah apapun, dia sudah bisa langsung start kerja. >>>>>> DONE

11. Biaya environment seefisien mungkin.

12. Sistem harus High availability 24x7x365



