2. 이더리움 및 DB 구축

노트북: 블록체인
만든 날짜: 2019-08-26 오전 10:11 **수정한 날짜:** 2019-10-09 오후 4:28
작성자: 감자
태그: Docker
URL: http://whyitworld.blogspot.com/2016/03/docker-2.html

AWS VM 활용 이더리움 네트워크 구축

-3개의 AWS를 이용하여 프로젝트를 운영할것임
-1번 AWS : Ethereum
-2번 AWS : Hyperledger fabric
-3번 AWS : Front-End, Back-End, DB

Frontend

```
-지갑생성, 지갑열람, 코인충전
```

Backend

```
-MySQL 도커 이미지 다운로드 및 테이블 구조 확인

-이더리움 네트워크 연동, 지갑 정보 조회, 코인충전구현
```

Docker

-Docker를 이용하여 웹 서버 구축
- http://whyitworld.blogspot.com/2016/03/docker-2.html
- http://whyitworld.blogspot.com/2016/02/docker-aws-ec2.html
-https://blog.cosmosfarm.com/archives/248/%EC%9A%B0%EB%B6%84%ED%88%AC-18-04-%EB%8F%84%EC%BB%A4-docker-%EC%84%A4%EC%B9%98-%EB%B0%A9%EB%B2%95/

<환경>

- node.js
- mysql
- AWS EC2(amazon OS)

<개요>

1. node.js설치
2. 도커 이미지 생성
3. mysql이미지 다운
4. mysql 컨테이너 실행
5. node.js 컨테이너로 mysql 컨테이너에 연결

6. mysql연결 확인
7. test페이지 작성 및 DB연결 테스트

(1) 3번 AWS에 DB 네트워크를 구축하는 방법

1.AWS에서 인스턴스 생성 후 Puttygen으로 .pem키를 .ppk로 바꾼 후 putty에서 서버 구동 후 아래와 같이 docker 설치

```
#설치된 패키지 업그래이드 : 설치되어 있는 패키지를 모두 새버전으로 업그래이드 합니다.
$ sudo apt-get upgrade

#패키지 설치
$ sudo apt-get install yum

##docker설치
$ sudo apt update
$ sudo apt install apt-transport-https ca-certificates curl software-
properties-common
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add
-
$ sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu bionic stable"
$ sudo apt update
$ apt-cache policy docker-ce

#여기까지 하면 도커설치 완료
$ sudo apt install docker-ce

#도커가 정상적으로 실행중인지 확인 -> active(running)
$ sudo systemctl status docker

##Docker실행
#docker 서비스 실행
$ sudo service docker start

##Docker 접속
#설치되어 있는 이미지 확인
$ sudo docker images

#이미지 검색(우분투)
$ sudo docker search ubuntu

#이미지 다운로드(기본값으로 latest라는 태그가 붙어있다.)
$ sudo docker pull ubuntu

#다운 된 이미지 확인(반복)
$ sudo docker images

#다운 된 이미지 실행 (상기 커맨드를 에러 없이 입력을 마쳤다면 컨테이너 내부의 우분투에
접속된 상태가 된다.)
#컨테이너 생성 및 접속
$ sudo docker run -i -t ubuntu:latest /bin/bash

~/# apt-get update
~/# apt-get install -y curl
~/# apt-get -y install sudo
~/# curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
~/# sudo apt-get install -y nodejs
~/# apt-get install -y build-essential libssl-dev

#환경 변수가 재 설정되므로 다시 로그인 해준다.
~/# exit

#컨테이너 확인 -> 이름 등등을 확인할 수 있다.
$ sudo docker ps -a
```

```
#컨테이너가 종료되어 있으므로 실행
$ sudo docker start [실행시킬컨테이너이름]

#컨테이너에 재 접속
$ sudo docker attach hopeful_darwin

#node.js버전 확인
~/# node -v

#npm을 이용하여 express설치
~/# cd srv
~/# npm install -g express-generator

#express를 이용하여 테스트 프로젝트 생성 및 실행 확인 (근데 파일없으니까 이 과정은 걍 넘어가자 ㅎ)
~/# express –ejs example01
~/# cd exsample01
~/# npm install
~/# node bin/www
[Ctrl]+[c]
~/# rm -rf example01
~/# exit

#컨테이너 확인
$ sudo docker ps -a

#밑줄은 내 이미지 이름
sudo docker commit hopeful_darwin ubuntu-nodejs:nodejs

#이미지 확인
sudo docker images

##Mysql 이미지 다운로드
#mysql 이미지 검색
$ sudo docker search mysql

# mysql 이미지 다운로드
$ sudo docker pull mysql:latest

#이미지 확인
$ sudo docker images

##mysql 실행(컨테이너 생성)
# mysql 데몬으로 실행 및 컨테이너 생성 (password는 알아서 설정)
$ sudo docker run -d –name db01 -e MYSQL_ROOT_PASSWORD=[password] mysql
$ sudo docker ps

##node.js 실행(컨테이너 생성) (참고: 컨테이너 삭제 $ sudo docker rm 컨테이너명)
#node.js 실행 및 컨테이너 생성 (웹 서버쪽에서 연결하고 싶은 mysql서버에 링크를 건다.)
$ sudo docker run –name web01 -i -t -p 8010:3000 -v /srv:/srv –link db01:db01
ubuntu-nodejs:nodejs /bin/bash

#컨테이너에서 나오기
~/# [Ctrl]+[p]와 [Ctrl]+[q]

#실행 중인 컨테이너 확인 -> nodejs, mysql 컨테이너가 둘 다 잘 있나 확인
$ sudo docker ps

#컨테이너의 링크가 제대로 걸려 있는지 확인
$ sudo docker inspect --format "{{.HostConfig.Links}}" web01

##web01 -> db01mysql 연결 확인
#web01컨테이너에 접속
$ sudo docker attach web01

#mysql접속 확인
~/# mysql -uroot -p -hdb01
-> 여기서 bash: mysql: command not found가 뜬다면 mysql을 설치하지 않아서 발생하는 에러이므로 mysql설치
```

```
~/# apt-get install mysql-client
~/# mysql -uroot -p -hdb01

=> 만약 여기서 에러가 난다면 mysql을 여러번 설치하여 발생한 것일수도 있으니 아래를 실행 후 다시 mysql 작동
~# ps -A|grep mysql
~# sudo pkill mysql
~# ps -A|grep mysqld
~# sudo pkill mysqld
~# service mysql restart
~# mysql -u root -p

~#mysql --user=root

##Test페이지 작성 및 DB연결
#DB 접속 테스트를 위해 테스트용 DB와 테이블을 생성
mysql> create database example01;
mysql> use example01;
mysql> create table test01(printtext varchar(100) primary key) ENGINE=InnoDB;
mysql> insert into test01 value(“welcome!”);
mysql> select * from test01;

->결과화면
+-----------+
| printtext |
+-----------+
|  welcome! |
+-----------+

mysql> exit;

#test페이지는 node.js의 express를 이용할 예정이다.
#npm 설치안되어있을경우 아래두줄 해주셈
~/# curl -sL https://deb.nodesource.com/setup_5.x | sudo -E bash -
~/# npm install -g express-generator

~/# cd /srv
~/srv# express –ejs example01
~/srv# cd example01
~/srv/example01# npm install
~/srv/example01# npm install -s mysql
~/srv/example01# cd routes
~/srv/example01/routes# vi index.js => vim에디터가 잘 동작하지 않는다면 설치
~/srv/example01/routes# apt-get install -y vim

#index.js에 아래 입력
var express = require('express');
var router = express.Router();


var mysql=require('mysql');
var connection=mysql.createConnection({
        host: 'db01',
        port: '3306',
        user: 'root',
        password: 'anrnDGhk82',
        database: 'example01'
});


/* GET home page. */
router.get('/', function(req, res, next) {


  connection.query('select * from test01', function(err, rows, fields){
    if (err) callback(err, null);
    console.log(rows[0].printtext);
    res.render('index', { title: 'Express', printtext: rows[0].printtext });
  });
});
```

```
#index.ejs 편집
~/srv/example01/routes# cd ../views/
~/srv/example01/views# vi index.ejs

#index.ejs
<!DOCTYPE html>
<html>
<head>
<title><%= title %></title> <link rel='stylesheet'
href='/stylesheets/style.css' />
</head>
<body>
<h1><%= title %></h1>
</h3><%= printtext %></h3>
</body>
 </html>

#node.js실행
~/srv/example01/views# cd ..
~/srv/example01# nohup node bin/www &
~/srv/example01# exit;
```