ImTeller

포팅메뉴얼

A509

포팅메뉴얼

개발환경

OS

- window 10
- Ubuntu 20.04 LTS

IDE

- IntelliJ IDE 2022.1.3
- Visual Studio Code 1.70

Datebase

- MySQL workbench 8.0.28

Docker

- Docker 20.10.12

EC2 기본

Docker 설치

```
# 오래된 버전 삭제하기
sudo apt-get remove docker docker-engine docker.io containerd runc

# repository 설정하기
sudo apt-get update
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release

# Docker의 Official GPG Key 를 등록
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

#stable repository 를 등록
echo \
  "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" |  tee /etc/apt/sources.list.d/docker.list > /dev/null

# Docker Engine 설치하기
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io

# 설치확인
docker --version
```

Jenkins 설치

- Docker로 Jenkins 설치

```
docker pull jenkins/jenkins

docker run -d -p 8064:8080 -p 50000:50000 --name jenkins jenkins/jenkins
```

jenkins 컨테이너 안에 docker 설치

- 컨테이너 형태로 설치된 젠킨스 안에서 docker 명령어를 실행하기 위해서 docker를 설치
- 설치를 위해 먼저 컨테이너 안으로 접근, 쉘을 실행.

```
docker exec -it jenkins bash
```

- 도커 설치 → 위의 도커 설치 과정 참고

Docker Image pull

- mysql 5.7

```
docker pull mysql:8
```

- node:16-alpine

```
docker pull node:16-alpine
```

- openjdk:11

```
docker pull openjdk:11
```

DB

mysql 설치 ~ 초기 세팅

```
docker pull mysql:8

docker run --name mysql-imteller -e MYSQL_ROOT_PASSWORD=ssafy -e MYSQL_DATABASE=imteller -e MYSQL_USER=classic  -e MYSQL_PASSWORD=im50

docker exec -it mysql-container bash

mysql -u classic -p im509teller!

CREATE DATABASE imteller CHARACTER SET utf8 DEFAULT COLLATE utf8_general_ci;
```

mysql 테이블 계정 설정

```
docker exec -it mysql-imteller bash // docker 내 mysql 접속

mysql -u root -p // 루트 계정으로 접속 후 비밀번호 입력

SHOW DATABASES; // 데이터베이스 확인

// 개발용 데이터베이스 생성
CREATE DATABASE imteller_dev CHARACTER SET utf8 DEFAULT COLLATE utf8_general_ci;

// 유저 생성
CREATE USER 'classic_dev'@'%' IDENTIFIED BY '비밀번호';

// 유저 확인
SELECT user FROM user;

// 권한 부여
GRANT ALL PRIVILEGES ON imteller_dev.* TO classic_dev@'%' IDENTIFIED BY '비밀번호';
FLUSH PRIVILEGES;

// 권한 확인
SHOW GRANTS FOR classic_dev@'%';
```

S3

springboot 설정

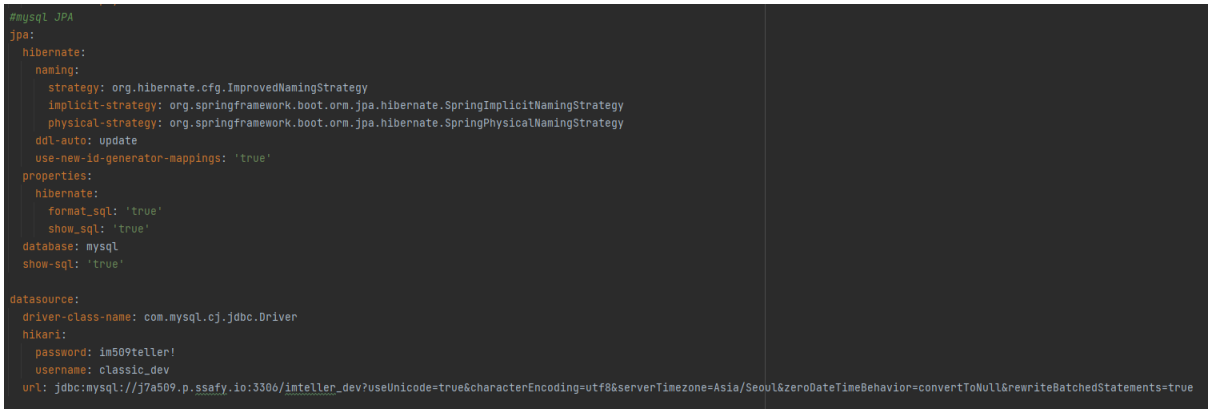
aws.yml

```
#S3
cloud:
  aws:
    credentials:
      accessKey: AKIA3MW7LJZMYVSJJHGJ
      secretKey: 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
    s3:
      bucket: imtellercard
    region:
      static: ap-northeast-2
    stack:
      auto: false

spring:
  servlet:
    multipart:
      max-file-size: 10MB
      max-request-size: 10MB
```

Backend

application.yml → DB연결



```
#mysql JPA
jpa:
  hibernate:
    naming:
      strategy: org.hibernate.cfg.ImprovedNamingStrategy
      implicit-strategy: org.springframework.boot.orm.jpa.hibernate.SpringImplicitNamingStrategy
      physical-strategy: org.springframework.boot.orm.jpa.hibernate.SpringPhysicalNamingStrategy
    ddl-auto: update
    use-new-id-generator-mappings: 'true'
  properties:
    hibernate:
      format_sql: 'true'
      show_sql: 'true'
  database: mysql
  show-sql: 'true'

datasource:
  driver-class-name: com.mysql.cj.jdbc.Driver
  hikari:
    password: im509teller!
    username: classic_dev
  url: jdbc:mysql://j7a509.p.ssafy.io:3306/imteller_dev?useUnicode=true&characterEncoding=utf8&serverTimezone=Asia/Seoul&zeroDateTimeBehavior=convertToNull&rewriteBatchedStatements=true
```

Dockerfile

```
FROM openjdk:8-jre-alpine
EXPOSE 8080
ARG JAR_FILE=build/libs/backend-0.0.1-SNAPSHOT.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

Frontend

Dockerfile

```
FROM node:16

WORKDIR /app
```

```
COPY package*.json .
copy yarn.lock .

RUN yarn

COPY . .

RUN yarn build

EXPOSE 3000

CMD ["yarn", "start" ]
```

Jenkins

젠킨스 접속

- http://localhost:9090/로 접속 (위에서 포트를 9090으로 설정.)
- 젠킨스 초기 비밀번호 확인

```
docker logs jenkins
또는
sudo docker exec -it jenkins cat /var/jenkins_home/secrets/initialAdminPassword
```

- Plugin 다운로드 GitLab, generic WebhookTrigger, GitLab API, GitLab Authentication, Docker, Docker Comons, Docker Pipeline, Docker API

- 프로젝트 생성
  - 소스코드 관리: Git
    - Repository URL:
    - Credentials : gitlab id, password로 추가
  - Branchs to build: */master
  - Build Step: Execute shell

Backend Execute shell

```
cd backend
docker rm spring -f
chmod +x gradlew
./gradlew bootJar
docker build . -t pingpongclass:latest
docker run -d -p 8080:8080 --link mysql-pingpong --name spring pingpongclass:latest
```

Frontend Execute shell

```
cd client
docker rm client -f
docker image prune -a --force
docker build . -t imteller_client:latest
docker run -d -p 3000:3000 --name client imteller_client:latest
```

Letsencrypt - cetbot

```
sudo apt update
sudo apt install certbot -y
sudo apt install python3-certbot-dns-cloudflare -y
sudo certbot certonly --dns-cloudflare --preferred-challenges dns-01 --dns-cloudflare-propagation-seconds 20 --dns-cloudflare-credenti
```

```
/etc/letsencrypt/live/[도메인명]/ 위치에 발급
```

Nginx

- fullchain.pem과 private.pem을 default.conf 의 ssl_config 부분에서 경로 잘 쓰기

default.conf

```
upstream frontend{
  server j7a509.p.ssafy.io:3000;
}

upstream backend{
  server j7a509.p.ssafy.io:8080;
}


server {
    listen 80;
    listen [::]:80;
    server_name  j7a509.p.ssafy.io;

    return ^(.*) https://j7a509.p.ssafy.io:443$1 permanent;
}

server{
    listen 443 ssl;
    listen [::]:443 ssl;
    server_name  i7a403.p.ssafy.io;

    #ssl config
    ssl_certificate fullchain.pem;
    ssl_certificate_key private.pem;

    #Proxy
    proxy_connect_timeout 300;
    proxy_send_timeout 300;
    proxy_read_timeout 300;
    send_timeout 300;

    location / {
        proxy_pass http://frontend;
    }
    location /api {
        proxy_pass http://backend;
    }

}
```

```
# conf 파일 수정 후 수정사항을 반영
$ docker container exec <container> nginx -s reload
<>에는 현재 가동중인 nginx 컨테이너명 또는 ID를 작성
확인이 필요하다면 docker ps 로 확인
```

SmartContract

- remix 로 ssafynet 배포

```
remixd -s ./ -u https://remix.ethereum.org/
```