[A306] 포팅 매뉴얼 (dekku)

Project Skill Stack Version

Skill	Version
Java	17
SpringBoot	3.3.2
MySQL	9.0.1
Redis	7.4.0
MongoDB	7.0.12
Node	18.17.0
NPM	10.7.0
Next	14.2.5
Docker	27.1.1
Jenkins	2.462

EC2 포트 번호

Name	Version
ssh	22
Nginx	443, 80/tcp
Jenkins	9090
Mysql	3306
MongoDB	27017
Next.js	3000

외부 API

KAKAO OAuth	회원가입 & 로그인용

배포 환경

EC2 세팅

EC2 서버에 gitlab 레퍼지토리 클론

```
git clone https://lab.ssafy.com/s11-webmobile2-sub2/S11P12A30
```

빌드 방법

1. Backend

```
# S11P12A306 spring-dekku 폴더로 이동
cd spring-dekku
docker build -t dekku-backend .
```

2. Frontend

```
# S11P12A306 next-dekku 폴더로 이동
cd next-dekku
npm ci
npm run build
docker build -t dekku-frontend .
```

3. docker-compose

```
# S11P12A306 최상단 위치로 이동
docker compose down
docker compose up -d
```

▼ docker-compose.yml 내용

```
version: '3'
services:
  dekku-mysql:
    container_name: dekku-mysql
```

```
    image: mysql
    environment:
      - MYSQL_ROOT_PASSWORD=Dekku1234
      - MYSQL_ROOT_HOST=%
    ports:
      - 3306:3306
    volumes:
      - mydata:/var/lib/mysql
    restart: on-failure
    networks:
      - dekku-network

  dekku-redis:
    container_name: dekku-redis
    image: redis
    ports:
      - '6379:6379'
    volumes:
      - redisdata:/data
    restart: on-failure
    networks:
      - dekku-network

  dekku-mongo:
    container_name: dekku-mongo
    image: mongo
    ports:
      - 27017:27017
    volumes:
      - mongodata:/data
    environment:
      - MONGO_INITDB_ROOT_USERNAME=root
      - MONGO_INITDB_ROOT_PASSWORD=Dekku1234
    restart: on-failure
    networks:
      - dekku-network

  dekku-backend:
```

```
    container_name: dekku-backend
    build: ./spring-dekku
    image: sanghupark/dekku-backend
    depends_on:
      - dekku-mysql
      - dekku-redis
      - dekku-mongo
    ports:
      - '8080:8080'
    volumes:
      - /home/ubuntu/yml/application.yml:/app/config/applic
    restart: on-failure
    networks:
      - dekku-network

  dekku-frontend:
    user: root
    container_name: dekku-frontend
    build: ./next-dekku
    image: sanghupark/dekku-frontend
    depends_on:
      - dekku-backend
    ports:
      - '3000:3000'
    volumes:
      - type: bind
        source: /etc/letsencrypt
        target: /etc/letsencrypt
    restart: on-failure
    networks:
      - dekku-network

volumes:
  mydata:
  redisdata:
  mongodata:
```

```
networks:
  dekku-network:
```

프로젝트 환경변수 설정

- application.yml 으로 통일함

```
spring:
  data:
    mongodb:
      host: dekku.co.kr
      port: 27017
      database: dekkudb
      username: root
      authentication-database: admin
      password: Dekku1234
    redis:
      host: dekku.co.kr
      port: 6379
  datasource:
    url: jdbc:mysql://dekku.co.kr:3306/dekkudb?serverTimez
    username: root
    password: Dekku1234
    driver-class-name: com.mysql.cj.jdbc.Driver
  jpa:
    hibernate:
      ddl-auto: update
    #    show-sql: true
    properties:
      hibernate:
        dialect: org.hibernate.dialect.MySQLDialect
  # oauth2
  security:
    oauth2:
      client:
        registration:
          naver:
            client-name: naver
```

```
            client-id: Rsx5vAptmUrSISgUkmDO
            client-secret: 5FDZm8VeAB
            authorization-grant-type: authorization_code
            redirect-uri: http://localhost:8080/login/oaut
            scope: name, email, age, gender
            client-authentication-method: client_secret_po
          kakao:
            client-name: kakao
            client-id: 49e8b661f0b102fb6d48af8f9d51ae58
            client-secret: QP32lMzg40d7z0PZblH7UQMnUJATkTq
            authorization-grant-type: authorization_code
            redirect-uri: http://localhost:8080/login/oaut
            scope: profile_image, profile_nickname, accoun
            client-authentication-method: client_secret_po

        provider:
          naver:
            authorization-uri: https://nid.naver.com/oauth
            token-uri: https://nid.naver.com/oauth2.0/toke
            user-info-uri: https://openapi.naver.com/v1/ni
            user-name-attribute: response
          kakao:
            authorization-uri: https://kauth.kakao.com/oau
            token-uri: https://kauth.kakao.com/oauth/token
            user-info-uri: https://kapi.kakao.com/v2/user/
            unlink-uri: https://kapi.kakao.com/v1/user/unl
            user-name-attribute: kakao_account
  jwt:
    secret: vmfhaltmskdlstkfkdgodyroqkfwkdbalroqkfwkdbalax
    accessExpiredTime: 3_600_000 # 1시간
    refreshExpiredTime: 86_400_000 # 1일

# mustache 한글 깨짐
server:
  servlet:
    encoding:
      force-response: true
```

```
cloud:
  aws:
    credentials:
      accessKey: AKIAVRUVV4FVT2KWAOET
      secretKey: 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
    s3:
      bucket: dekku-bucket
    region:
      static: ap-northeast-2
    stack:
      auto: false

# 얘는 logger 를 통해 log 를 남기는거니까 SQL 을 얘로 확인
logging.level:
  org.hibernate.SQL: debug
  org.hibernate.type: trace # 쿼리 에서 value 에 ? ? 였던 부분
  org.springframework.security: debug
  org.springframework.web: debug
```

Nginx 세팅

`/etc/nginx/sites-available/default`

```
server {

        listen 80 default_server;
        listen [::]:80 default_server;

        root /var/www/html;

        # Add index.php to the list if you are using PHP
        index index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                # First attempt to serve request as file, the
                # as directory, then fall back to displaying
```

```
                try_files $uri $uri/ =404;
        }

}

server {
        root /var/www/html;

        # Add index.php to the list if you are using PHP
        index index.html index.htm index.nginx-debian.html;
    server_name dekku.co.kr; # managed by Certbot

    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/dekku.co.kr/fullcha
    ssl_certificate_key /etc/letsencrypt/live/dekku.co.kr/pri
    include /etc/letsencrypt/options-ssl-nginx.conf; # manage
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed

    location / {

        proxy_pass http://localhost:3000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forward
        proxy_set_header X-Forwarded-Proto $scheme;

    }

    location /api {
        proxy_pass http://localhost:8080;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forward
        proxy_set_header X-Forwarded-Proto $scheme;
    }

}
```

```
server {

    if ($host = dekku.co.kr) {
        return 301 https://$host$request_uri;
    } # managed by Certbot


        listen 80 ;
        listen [::]:80 ;
    server_name i11a306.p.ssafy.io;
    # server_name dekku.co.kr
    return 404; # managed by Certbot

}
```

Https 설정

```
apt-get update
apt-get upgrade
apt-get install python3-certbot-nginx -y
certbot certonly --nginx -d dekku.co.kr
sudo systemctl reload niginx
sudo nginx -t
```

DB 덤프 파일

dekkudb_deskterior_posts_images.sql

dekkudb_deskterior_posts_products_info.sql

dekkudb_file_paths.sql

dekkudb_follow.sql

dekkudb_members.sql

dekkudb_members_liked_posts_info.sql

dekkudb_products.sql

dekkudb_refresh.sql