인생 N 컷

삼성 SW 청년 아카데미 대전 캠퍼스 7 기

자율 프로젝트 (6 주 2022/10/10 ~ 2022/11/21)

포팅 매뉴얼

담당 컨설턴트 : 김성준 컨설턴트

유일권(팀장), 김성수, 이정건, 최재현, 최주희, 최준혁

<<목차>>

1. 프로젝트 소개

"바쁘다 바빠 현대사회" 라는 말을 들어보신 적이 있나요?

현대 사회는 태어나는 순간부터 치열한 경쟁에서 뒤쳐지지 않으려면 열심히 사는것을 미덕으로 여기고 그에 맞는 행동을 요구합니다.

모두가 각자마다의 이유로 바쁘게 살고 있습니다. 그래서 현대인에게는 잠시 쉬어갈 시간, 한번쯤 자신을 돌아볼 시간이 필요하다고 생각했습니다.

이러한 이유로 다양한 질문을 통해 사용자의 인생을 되돌아보고 생각을 정리할 수 있는 시간을 부여하고, 회고를 통해 "성장의 밑거름" 혹은 "동기부여"가 이루어질 수 있도록 도와주자는 의도를 가지고 "**인생 N 컷**" 서비스를 기획하게 되었습니다.

"**인생 N 컷**"은 인생을 여러 장면들로 담아낸다는 의미를 이름에 담았습니다.

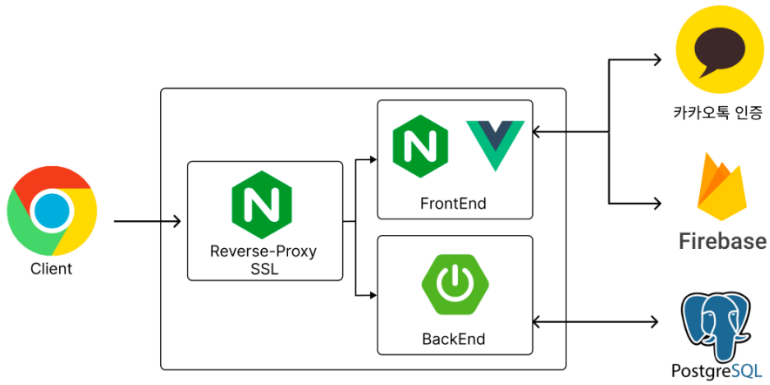
스토리텔링으로 진행되며, 여행이 끝나면 결과를 기록물로 제공하기 때문에 이러한 뜻을 서비스 이름에 담았습니다.

2. 기술 스택

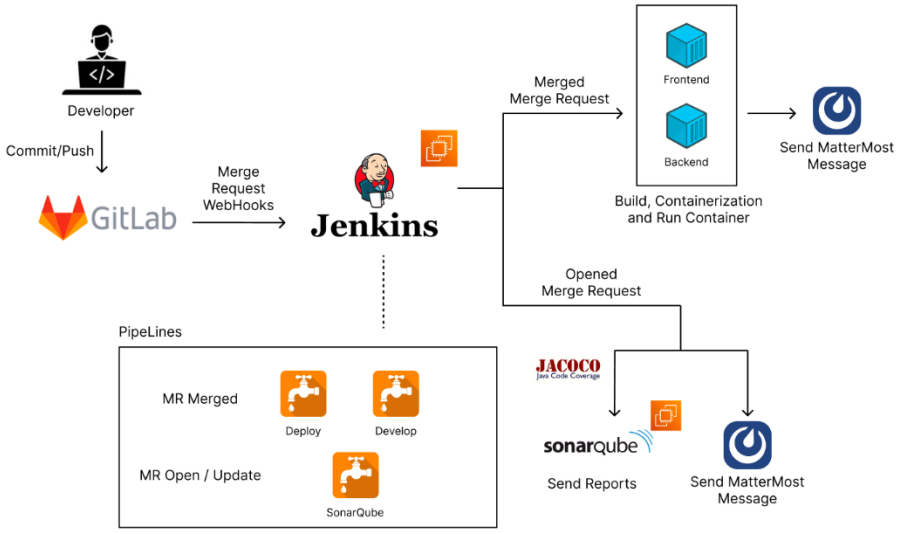
구분	기술 스택	상세 내용	버전
공통	형상 및 이슈 관리	GitLab, Jira	-
	커뮤니케이션	Mattermost, Notion	-
	화상 회의	Webex	-
Backend	DB	Postgresql	12.12
		H2	1.4.200
		JPA	2.7.5
	JDK	Zulu	8.33.0.1
	Spring	Spring	5.3.24
		Spring Boot	2.7.5
	IDE	IntelliJ	2022.1.2
	Build	Gradle	7.5.1
	Test	Junit5	5.9.1
	API Docs	Swagger2	3.0.0
Frontend	HTML5		-
	CSS3		
	JavaScript(ES6)		
	Vue	Vue	3.2.41
		Vue-router	4.1.5
		Pinia	2.0.23
		bootstrap-vue-3	0.3.12
	IDE	Visual Studio Code	1.70
Server	Server	AWS EC2	
	OS	Ubuntu	20.04.3 LTS
	배포	Docker	20.10.20
		Jenkins	2.361.3
	웹 서버	Nginx	1.23.1

3. 빌드 상세 내용

저희 **“인생 N 컷”** 서비스의 배포환경 및 CI/CD 배포 자동화 흐름도 입니다.



서비스 구조도



CICD 흐름도

팀원들이 GitLab 에 코드를 푸쉬하고, MR 을 올리게 되면 Jenkins 를 통해 Jacoco 를 통해 테스트 코드 커버리지를 확인하고 70%가 넘지 못하면 빌드를 중지하고, 넘는 경우 SonarQube 에 코드 분석을 의뢰합니다. 이후 코드의 악취, 버그 등을 확인하고, 리팩토링을 진행합니다. 이후 MR 이 머지 되면 Jenkins 를 통해 자동으로 빌드를 하고 Docker 컨테이너를 만들어 실행 시킵니다.

▸ Docker

- Docker 및 Docker-Compose 설치

아래 명령어를 순서대로 입력하여 Docker 를 설치한다.

```
$ sudo apt-get update

$ sudo apt-get install ca-certificates curl gnupg lsb-release

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

$ sudo apt-get update

$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

명령어를 순서대로 입력하고 설치가 되었다면 아래 명령어로 확인한다.

```
$ sudo docker --version
```

Docker 를 설치했다면, Docker-compose 를 설치한다.

```
$ sudo curl -L
"https://github.com/docker/compose/releases/download/1.29.2/docker-
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose$ sudo
apt-get install ca-certificates curl gnupg lsb-release

$ sudo chmod +x /usr/local/bin/docker-compose
```

Docker-compose 설치가 다되었다면, 아래 명령어로 확인한다.

```
$ docker-compose -version
```

FrontEnd

프로젝트를 GitLab 에서 Clone 받은 후. 아래 명령어를 통해 Docker Container Image 를 생성한다.

빌드를 위한 Dockerfile 은 프로젝트 내에 작성되어 있다.

```
$ docker build -t mylifencut_fe ./frontend
```

컨테이너 이미지 이름은 “mylifencut_fe”로 설정하였다.

BackEnd

프로젝트의 Backend 폴더에서 Gradle 을 통해 빌드를 진행한다.

```
$ /gradlew clean build
```

이후 아래 명령어를 통해 Docker Container Image 를 생성한다.

빌드를 위한 Dockerfile 은 프로젝트 내에 작성되어 있다.

```
$ docker build -t mylifencut _be ./backend
```

방화벽 설정

인생 N 컷을 EC2 에서 실행하기 위해서 리눅스 방화벽을 열어주어야한다.

만약 개인 EC2 를 사용한다면, EC2 보안그룹에서도 제외해주어야 한다.

```
$ sudo ufw allow <포트 번호>
```

방화벽을 해제해야하는 포트들

```
5432 : PostgreSQL
80 : HTTP
443 : HTTPS
1521 : h2
8000: SonarQube
```

Docker-Compose

프로젝트의 최상단에서 Docker-Compose 를 통해서 3 개의 컨테이너를 한번에 실행 시킨다.

```
$ docker-compose up -d      #-d 옵션으로 백그라운드 실행
```

Docker-Compose 의 여러 명령어를 통해 컨테이너를 관리 한다.

```
$ docker-compose down      #docker-compose 종료
$ docker-compose logs      #docker-compose 로그 확인
```

H2

백엔드 테스트 코드 실행을 위하여 H2 Database 도 Docker 를 통해 설치한다.

```
$ docker pull oscarfonts/h2:1.4.200

$ docker run  -d -p 1521:1521 oscarfonts/h2
```

SonarQube

코드 정적 분석을 위해 SonarQube 실행 파일을 만든다.

이후 docker-compose 를 통해 SonarQube 를 실행한다.

```
$ vim sonarqube/docker-compose.yml
```

Docker-compose.yml 파일 내용은 아래와 같이 작성한다.

서버의 Postgresql 에 미리 sonar 테이블을 만들고 접속한다.

```
version: "3"

services:
  sonarqube:
    image: sonarqube
    environment:
      SONAR_JDBC_URL: jdbc:postgresql://k7b105.p.ssafy.io:5432/sonar
      SONAR_JDBC_USERNAME: sonar
      SONAR_JDBC_PASSWORD: sonar
    volumes:
      - sonarqube_data:/opt/sonarqube/data
      - sonarqube_extensions:/opt/sonarqube/extensions
      - sonarqube_logs:/opt/sonarqube/logs
    ports:
      - "8000:9000"

volumes:
  sonarqube_data:
  sonarqube_extensions:
  sonarqube_logs:
```

이후 서버의 8000 번 포트로 접속 하게 되면 소나큐브에 접속이 가능하다.

4. DB 설치 및 설정

인생 N 컷 서비스를 사용하기 위해서 DB 를 설치한다.

```
$ sudo apt-get update
$ sudo apt-get install -y postgresql postgresql-contrib
```

설치가 완료되면 Postgresql 서비스에 접속하고, 새 유저를 생성한다.

```
$ sudo su postgres    #postgres 유저로 접속
$ psql                #posgresql 서비스 접속
$ CREATE USER {user_name} WITH PASSWORD '{user_password}'; # 유저 생성
```

새 유저가 생성 되었으면 확인하고 테이블 생성 후 권한을 부여한다

```
$ ₩du                 #유저 모두 나열
$ ALTER USER {user_name} WITH SUPERUSER;     #유저 권한 부여
```

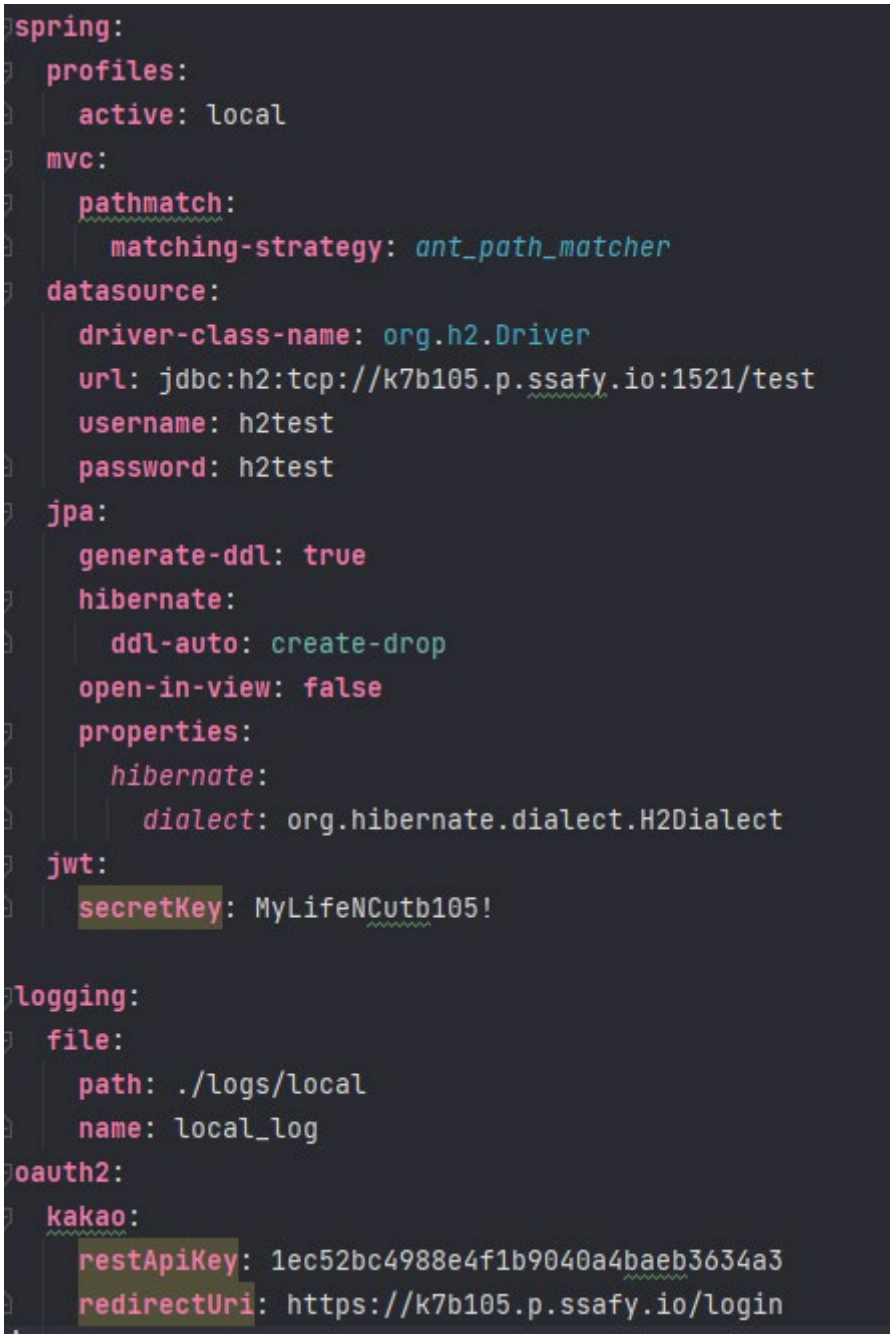
이후 서버에서 접속하여 작업을 진행한다.

5. 프로퍼티 정의

```yaml
server:
  context-path: /api
spring:
  profiles:
    active: local
  mvc:
    pathmatch:
      matching-strategy: ant_path_matcher
  datasource:
    driver-class-name: org.postgresql.Driver
    url: jdbc:postgresql://k7b105.p.ssafy.io:5432/mylifencut
    username: b105
    password: MyLifeNCutb105
  jpa:
    hibernate:
      ddl-auto: update
    properties:
      hibernate:
        format_sql: true
        database: postgresql
        database-platform: org.hibernate.dialect.PostgreSQLDialect
  sql:
    init:
      platform: postgres
  jwt:
    secretKey: MyLifeNCutb105
logging:
  file:
    path: ./logs/local
    name: local_log

oauth2:
  kakao:
    restApiKey: 1ec52bc4988e4f1b9040a4baeb3634a3
    redirectUri: https://k7b105.p.ssafy.io/login
```

src/main/resources/application.yml



```yaml
spring:
  profiles:
    active: local
  mvc:
    pathmatch:
      matching-strategy: ant_path_matcher
  datasource:
    driver-class-name: org.h2.Driver
    url: jdbc:h2:tcp://k7b105.p.ssafy.io:1521/test
    username: h2test
    password: h2test
  jpa:
    generate-ddl: true
    hibernate:
      ddl-auto: create-drop
    open-in-view: false
    properties:
      hibernate:
        dialect: org.hibernate.dialect.H2Dialect
  jwt:
    secretKey: MyLifeNCutb105!

logging:
  file:
    path: ./logs/local
    name: local_log
oauth2:
  kakao:
    restApiKey: 1ec52bc4988e4f1b9040a4baeb3634a3
    redirectUri: https://k7b105.p.ssafy.io/login
```

src/test/resources/application.yml

```
VUE_APP_FIREBASE_API_KEY=AIzaSyAHJcsJyT3RheO00d6WmktAdYiKxpY7nmA
VUE_APP_FIREBASE_AUTH_DOMAIN=mylifencut.firebaseapp.com
VUE_APP_FIREBASE_DATABASE_URL=https://mylifencut-default-rtdb.firebaseio.com
VUE_APP_FIREBASE_PROJECT_ID=mylifencut
VUE_APP_FIREBASE_STORAGE_BUCKET=mylifencut.appspot.com
VUE_APP_FIREBASE_MESSAGING_SENDER_ID=669910488578
VUE_APP_FIREBASE_APP_ID=1:669910488578:web:69cf9926f7ba67776f391b
VUE_APP_FIREBASE_MEASUREMENT_ID=G-FQMFCYL7CT
VUE_APP_HI=hihi
VUE_APP_KAKAO_JS_SDK=e5c4f9986658dbe9fe3a4f071804122f
VUE_APP_KAKAO_API_KEY=1ec52bc4988e4f1b9040a4baeb3634a3
VUE_APP_KAKAO_REDIRECT_URI=https://k7b105.p.ssafy.io/login
```

.env