B207 포팅 메뉴얼

1. 프로젝트 개요

초등학교 저학년을 위한 등굣길은 항상 안전할까요? 맞벌이 부부가 늘어나면서 아이는 보호자 없이 혼자 혹은 아이들끼리 등교하는 경우가 대다수입니다.

아이들이 실종되었을 때, 3시간 이후로 기점으로 발견 건수는 급락한다고 합니다.

아이가 연락이 안되었던 지점을 바로 안다면? 그 근처 cctv 위치도 바로 알고 있다면? 빠른 속도로 찾을 수 있게 될 것입니다.

그래서 cctv를 중심으로 경찰청, 아동안전지킴이집 위치를 고려한 안전 경로를 제공해주는 바래다줄게를 기획하게 되었습니다.

바래다줄게는 아이와 부모 앱으로 구분되어 아이는 부모가 지정해준 목적지 중 하나를 도착치로 하여 아이가 출발을 누른다면 그때부터 부모님들은 아이의 실시간 위치를 볼 수 있으며, 아이의 출발,도착, 경로이탈에 대한 알림 정보를 받을 수 있습니다.

2. 프로젝트 사용 도구

- 이슈 관리 : JIRA, Git
- 형상 관리 : GitLab, Gerrit
- 커뮤니케이션 : Notion, Mattermost, Discord
- 디자인 : Figma
- UCC :
- CI / CD : Jenkins, Docker, DockerCompose

3. 개발 환경

- VisualStudioCode
- Flutter

- IntelliJ IDEA Ultimate 2023.3
- Java 17
- SpringBoot 3.1.9
- Gradle 8.5

4. 기술 세부 스택 <BackEnd>

- IntelliJ : 2023.03.04
- Java : 17.0.10
- Springboot : 3.1.9
- Spring Security
- Spring Data JPA
- Gradle 8.5
- Junit5
- h3 4.1.1

5. 기술 세부 스택 <FrontEnd>

- flutter 3.19.3
- shared_preferences : 로컬 저장소 key : value 형식으로 저장
- flutter_secure_storage : shared_preferences와 같은 형식이지만 보안적으로 우수한 저장 라이브러리
- provider : 상태 저장 라이브러리(모든 화면에서 같은 변수를 사용할 수 있다거나.. static 같은거임)
- geolocator : 현재 위치 불러올 수 있는 라이브러리
- firebase_messaging : FCM 알림을 보낼 수 있는 라이브러리

6. 기술 세부 스택 <Database, Infra>

- MySQL 8.0.36
- AWS S3
- Ubuntu 20.04 LTS
- Nginx 1.18.0(Ubuntu)
- Docker 25.0.4
- Docker Compose 2.24.7
- Jenkins

7. 외부 서비스

```
- Kakao OAuth
- Naver OAuth
- S3 Bucket
- Tmap API
```

8. 빌드

1. 환경 변수 (백엔드)

- 서버에서 cd /var/jenkins_home/workspace/bada-pipeline/bada-back/src/main/resources/ 하위에 파일 생성

```
# ../main/resources에 위치
# application-secret.yml
EC2_HOST: j10b207.p.ssafy.io
DATABASE_NAME : badadb
EC2_USERNAME : bada
EC2_PASSWORD: bada207

# aws s3
S3_ACCESS_KEY : AKIA47CR3O27STM2MQFT
S3_SECRET_KEY : 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
```

```
S3_BUCKET_NAME : bada-bucket
S3_REGION_STATIC : ap-northeast-2

#Tmap API appKey
TMAP_APPKEY: Yc1psCM8oa5WonxszvnAJ9QHXZN8EnPz4ivOiDEf

#jwt
jwt:
  secret:
    key: c2lsdmVybmluZS10ZWNoLXNwcmluZy1ib290LWp3dC10dXRvcmlh
```

```
// ../main/resources/firebase에 위치
// firebase_service_key.json
{
  "type": "service_account",
  "project_id": "bada-1c6f7",
  "private_key_id": "0939157261d4aa456b20fb3d62c40ef8eed4c5f7
  "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEvgIBADANBg
  "client_email": "firebase-adminsdk-mbydv@bada-1c6f7.iam.gse
  "client_id": "118284616563655270625",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/
  "client_x509_cert_url": "https://www.googleapis.com/robot/v
  "universe_domain": "googleapis.com"
}
```

2. 환경 변수 (프론트엔드)

```
# 프로젝트 폴더 제일 바깥에 위치
# .env
KAKAO_MAP_API = '22d1399bb5e84d5d41af148a6d94c904'
KAKAO_SEARCH_API = 'be8a38ff76c199cc88b459e8c29957be'
```

3. 빌드하기

- Back

```
chmod +x gradlew
./gradlew clean
./gradlew build
```

- DockerFile

```
FROM openjdk:17

# 인자 설정 부분과 jar 파일 복제 부분 합쳐서 진행해도 무방
COPY ./*.jar app.jar

# 실행 명령어
ENTRYPOINT ["java", "-jar", "app.jar"]
```

4. 배포하기

1. Nginx

```
// cd /etc/nginx/conf.d/에 위치
// default.conf
server {
        listen 80;
        server_name j10b207.p.ssafy.io;
        return 301 https://j10b207.p.ssafy.io$request_uri;
}

server {
        listen 443 ssl;

        server_name j10b207.p.ssafy.io;

        location /api {
                proxy_pass http://localhost:8080;
                #proxy_redirect uri;
                charset utf-8;

                client_max_body_size 20m;
                proxy_buffer_size 4096k;
```

```
                proxy_buffers 4 4096k;

                proxy_http_version 1.1;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection "upgrade";
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_ad
                proxy_set_header X-Forwarded-Proto $scheme
        }

 ssl_certificate /etc/letsencrypt/live/j10b207.p.ssafy.io/
    ssl_certificate_key /etc/letsencrypt/live/j10b207.p.ss
    include /etc/letsencrypt/options-ssl-nginx.conf; # man
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;
}
```

2. Jenkins 빌드 스텝

```
// 빌드
chmod +x gradlew
./gradlew clean
./gradlew build

// 도커 이미지 생성
FROM openjdk:17

# 인자 설정 부분과 jar 파일 복제 부분 합쳐서 진행해도 무방
COPY ./*.jar app.jar

# 실행 명령어
ENTRYPOINT ["java", "-jar", "app.jar"]
```

docker jar 이미지 빌드 후 컨테이너로 배포

```
#파일 위치 및 이름: /home/ubuntu/BadaServer/build.sh
```

```
##########################################################
####################       back       ####################
##########################################################
cd /home/ubuntu/BadaServer

# remove if process exist
if docker ps -a | grep bada-back; then
    echo "Stop and remove the existing 'bada-back' contain
    docker stop bada-back
    docker rm bada-back
fi



# remove the existing images
IMAGE_ID=$(docker images | grep 'docker-springboot' | awk

if [ -n "$IMAGE_ID" ]; then
    echo "remove the existing 'docker-springboot' image. I
    docker rmi $IMAGE_ID
else
    echo "image with name 'docker-springboot' not found"
fi
sudo docker images

# build the new image and docker-compose up
sudo docker build -t docker-springboot .
sudo docker images

sudo docker run -d --name bada-back -p 8080:8080 docker-sp
```