포팅 매뉴얼

1. 배포

- Git Repository Clone

```
# 프로젝트 파일 생성
sudo mkdir /palette

# 프로젝트 파일 이동
cd /palette

# Git Init
sudo git init

# Git Remote
sudo git remote add origin https://lab.ssafy.com/s09-webmobile1-sub2/S09P12E103.git

# Git pull
sudo git pull origin prod
```

- Docker & docker-compose Install

```
# ubuntu update
sudo apt-get update

# docker install
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release

sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.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

# ubuntu user 권한 부여
sudo usermod -aG docker ubuntu

sudo reboot

# docker-compose install
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin

sudo curl -L https://github.com/docker/compose/releases/download/v2.1.0/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-

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

- java 17 install

```
sudo apt install openjdk-17-jdk openjdk-17-jre -y

sudo apt-get update
```

버전

BACKEND

- Springboot : `3.1.2`
    - Project Metadata
        - Group : com.palette
        - Artifact : palette
        - Name : palette
        - Package Name = com.palette.palette
- jdk : `17.0.8`
- QueryDSL : `5.0.0`
- Python : `3.9.0`
- Django : `4.0.6`
- djangorestframework : `3.13.1`
- OpenCV : `4.8.0.74`
- Dlib : `19.24.0`
- Scikit-learn : `1.1.1`
- Swagger : `2.0.2`
- Nginx : `1.25.2`

FRONTEND

- React: `18.2.0`
- Axios: `1.4.0`
- recoil-persistance: `5.1.0`
- VS code: `0.1.0`
- Dart: `3.0.6`

DATABASE

- MySQL : `8.1.0`
- Redis : `7.2.0`

DEVOPS

Docker : `24.0.5`

docker compose : `2.1.0`

WAS

- AWS
    - ec2
    - S3

환경 변수

BACKEND

- **Springboot**

```
sudo vim /palette/backend/palette-spring/src/main/resources/properties/env.properties
```

```
oauth2.google.client-id= 103846021246-78is58di7n3hvgml8u73i4g9ro66o2v1.apps.googleusercontent.com

oauth2.google.client-secret= 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

oauth2.google.redirect-uri= http://localhost:8080/login/oauth2/code/google

oauth2.google.token-uri=  https://oauth2.googleapis.com/token

oauth2.google.resource-uri= https://www.googleapis.com/oauth2/v2/userinfo

pgmodule.app-id= 1347818331771024

pgmodule.secret-key= hJTv8ZGkHvcN9EUv5ZRyhmvkiHj7ekn9rhCI2RdPCBTYrpOB6geDZiXYDS3t9ABDlLlJOY1CpUcyATh4
```

- **React**

```
sudo vim /palette/frontend/mon_palette/.env
```

```
REACT_APP_AWS_S3_ACCESS_ID = "AKIAXP6HE2GKABEEQE6O"
REACT_APP_AWS_S3_ACCESS_PW = "/6nyZAST6EB+RCYxhekLotMCo2cAUzjPMLWowVFQ"
REACT_APP_AWS_S3_REGION = "ap-northeast-2"
REACT_APP_AWS_S3_BUCKET = "ssafy9-monpalette"

REACT_APP_API=https://mon-palette.shop:8080
```

- **Django**

```
sudo vim /palette/backend/palette-django/.env
```

```
SECRET_KEY='django-insecure-=-l0gbb0y))-$7&@w#&3ohd_r!^p^m5xf0m9b$jxmt!k589zv4'
```

SSL

`/etc/letsencrypt/live/mon-palette.shop(도메인 네임)` 또는 `/data/certbot/conf/live/` 안에 4개의 키 들이 존재. 그 경로에서 아래 명령어 실행.

(만약 경로 접근 권한 문제가 생기면 `sudo chmod  +x {경로}` 명령어로 권한 부여 후 진행.

```
sudo openssl pkcs12 -export -in fullchain.pem -inkey privkey.pem -out keystore.p12 -name tomcat -CAfile chain.pem -caname root
```

위 명령어로 해당 경로에 공개키가 생긴다.

이 공개키를 자바 application.yml 경로로 복사한다.

```
sudo cp keystore.p12 /palette/backend/palette-spring/src/main/resources/
```

빌드 및 배포 문서

```
cd /palette
```

BACKEND

- SpringBoot

  ```
  sudo vim backend/palette-spring/Dockerfile
  ```

  ```
  FROM openjdk:17-alpine

  WORKDIR /usr/src/app

  ARG JAR_PATH=./build/libs

  COPY ./build/libs/palette-spring-0.0.1-SNAPSHOT.jar /build/libs/palette-spring-0.0.1-SNAPSHOT.jar

  CMD ["java","-jar","/build/libs/palette-spring-0.0.1-SNAPSHOT.jar"]
  ```

- Django

  ```
  sudo vim backend/palette-django/Dockerfile
  ```

  ```
  FROM python:3.9.0
  ENV PYTHONDONTWRITEBYTECODE 1
  ENV PYTHONUNBUFFERED=1

  COPY . /app/server/palette

  WORKDIR /app/server/palette

  RUN apt-get update && apt-get install -y cmake && apt-get -y install libgl1-mesa-glx && apt-get install -y --no-install-recommends
  RUN pip install --upgrade pip
  RUN pip install -r requirements.txt
  ```

FRONTEND

- React

  ```
  sudo vim frontend/mon_palette/Dockerfile
  ```

  ```
  # Dockerfile

  FROM node:alpine as builder
  WORKDIR /usr/src/app
  COPY package.json .
  RUN npm install
  COPY ./ ./
  RUN npm run build

  FROM nginx
  EXPOSE 3000
  COPY ./default.conf /etc/nginx/conf.d/default.conf
  COPY --from=builder usr/src/app/build  /usr/share/nginx/html
  ```

- react-nginx

```
sudo vim frontend/mon_palette/default.conf
```

```
server {
    listen 3000;

    location / {

        root /usr/share/nginx/html;
        index index.html index.htm;
        try_files $uri  $uri/ /index.html;

    }
}
```

- nginx

```
sudo vim conf/nginx.conf
```

```
worker_processes auto;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;

events{
    worker_connections 1024;
}


http{
                # upstream spring {
                #        server 3.213.187.68:8080;
                # }
                # upstream react {
                #        server react:3000;
                # }
                client_max_body_size 0;

                server {
                       listen 80;
                       client_max_body_size 0;

                       server_name mon-palette.shop;
                       return 301 https://mon-palette.shop$request_uri;   # http로 들어오면 https로 redirect 해주는 부분

                       # location /static/{
                       #     alias /static/;
                       # }

                       # location /media/{
                       #      alias /media/;
                       # }
                       location /media {
                              alias /home/ubuntu/static/media;
                       }
                }

                server {
                       listen 443 ssl;
                       client_max_body_size 0;
                       server_name mon-palette.shop;

                       # Certificate
                       ssl_certificate /etc/letsencrypt/live/mon-palette.shop/fullchain.pem;

                       # Private Key
                       ssl_certificate_key /etc/letsencrypt/live/mon-palette.shop/privkey.pem;


                       location /django {

                               proxy_pass http://django:8000; # 자신의 django app이사용하는 포트
                               proxy_set_header Host $host;
                               proxy_set_header X-Real-IP $remote_addr;
```

```
                                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                                proxy_set_header X-Forwarded-Proto $scheme;
                                }
                        location /api {

                                proxy_pass http://3.39.252.81:8080; # 자신의 springboot app이사용하는 포트
                                proxy_set_header Host $host;
                                proxy_set_header X-Real-IP $remote_addr;
                                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                                proxy_set_header X-Forwarded-Proto $scheme;
                                }
                         location / {
                                proxy_pass http://react:3000; # 자신의 springboot app이사용하는 포>트
                                proxy_set_header Host $host;
                                proxy_set_header X-Real-IP $remote_addr;
                                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                                proxy_set_header X-Forwarded-Proto $scheme;
                        }
                 }
```

- docker-compose.yml

```
services:

  mysql:
    image: mysql
    #        platform: linux/amd64
    container_name: mysql
    volumes:
      - ./:/app/server/mysql/
    environment:
      MYSQL_DATABASE: palette
      MYSQL_ROOT_PASSWORD: 1234
      TZ: "Asia/Seoul"
    ports:
      - "3306:3306"
    command:
      - --character-set-server=utf8mb4
      - --collation-server=utf8mb4_unicode_ci
      - --default-authentication-plugin=mysql_native_password  # 추가한 부분
    restart: always


  redis:
    image: redis
    container_name: redis
    ports:
      - 6379:6379
    restart: always


  spring:
    build:
      context: ./backend/palette-spring
      dockerfile: Dockerfile
    container_name: spring
    volumes:
      - ./:/app/server/palette/palette-spring/
    ports:
      - 8080:8080
    expose:
      - 8080
    environment:
      SPRING_DATASOURCE_URL: jdbc:mysql://mysql:3306/palette
      SPRING_DATASOURCE_USERNAME: root
      SPRING_DATASOURCE_PASSWORD: 1234
      TZ: Asia/Seoul

    depends_on:
      - mysql
      - redis
    links:
      - mysql
      - redis
    restart: always


  django:
    build:
      context: ./backend/palette-django
      dockerfile: Dockerfile
    container_name: django
    command:
      - bash
      - -c
      - |
        cd ./backend/palette-django
        python manage.py makemigrations
        python manage.py migrate
        python manage.py runsslserver -6 [::]:8000 --certificate /data/certbot/conf/live/mon-palette.shop/fullchain.pem --key /dat
    volumes:
      - ./:/app/server/palette/
      - /data/certbot/conf/live/mon-palette.shop/privkey.pem:/data/certbot/conf/live/mon-palette.shop/privkey.pem
      - /data/certbot/conf/live/mon-palette.shop/fullchain.pem:/data/certbot/conf/live/mon-palette.shop/fullchain.pem
    ports:
      - "8000:8000"
    expose:
      - 8000
    env_file:
      - ./backend/palette-django/.env
    depends_on:
      - mysql
      - redis
    restart: always


  react:
    container_name: front
    build:
      context: ./frontend/mon_palette
      dockerfile: Dockerfile
    ports:
      - 3000:3000
    expose:
      - 3000


  nginx:
    container_name: nginx
    image: nginx:latest
    restart: always
    volumes:
      - ./conf/nginx.conf:/etc/nginx/nginx.conf
      - /data/certbot/conf:/etc/letsencrypt
      - /data/certbot/www:/var/www/certbot
    ports:
      - 80:80
      - 443:443
    depends_on:
      - spring
      - react
      - django
```

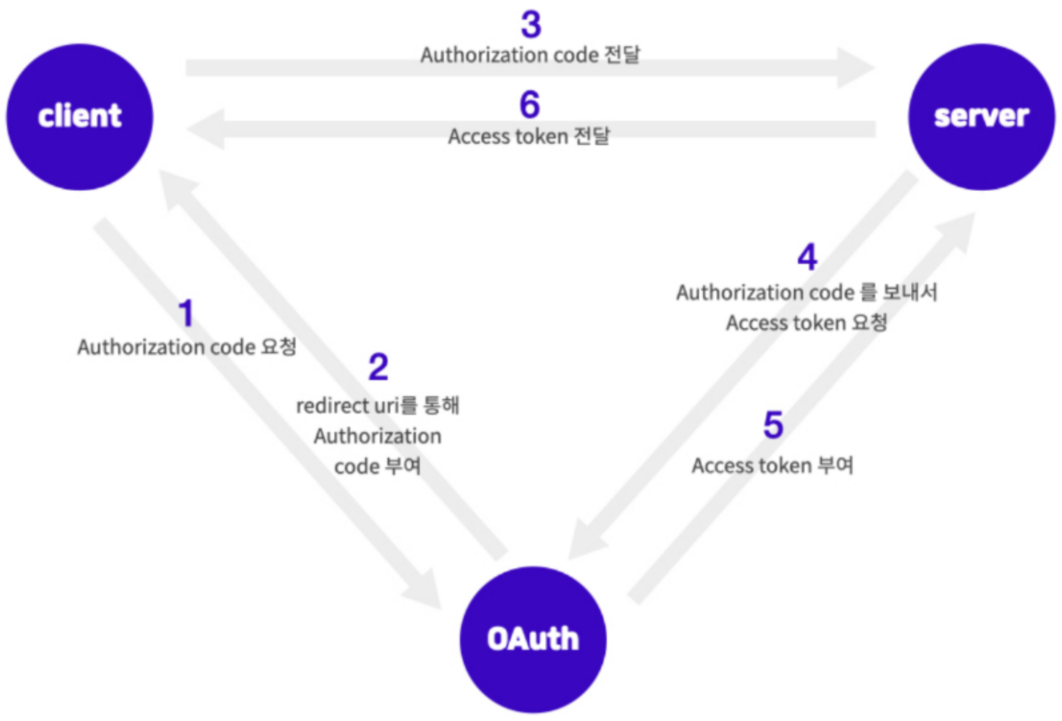
- 실행

```
docker-compose up -d
```

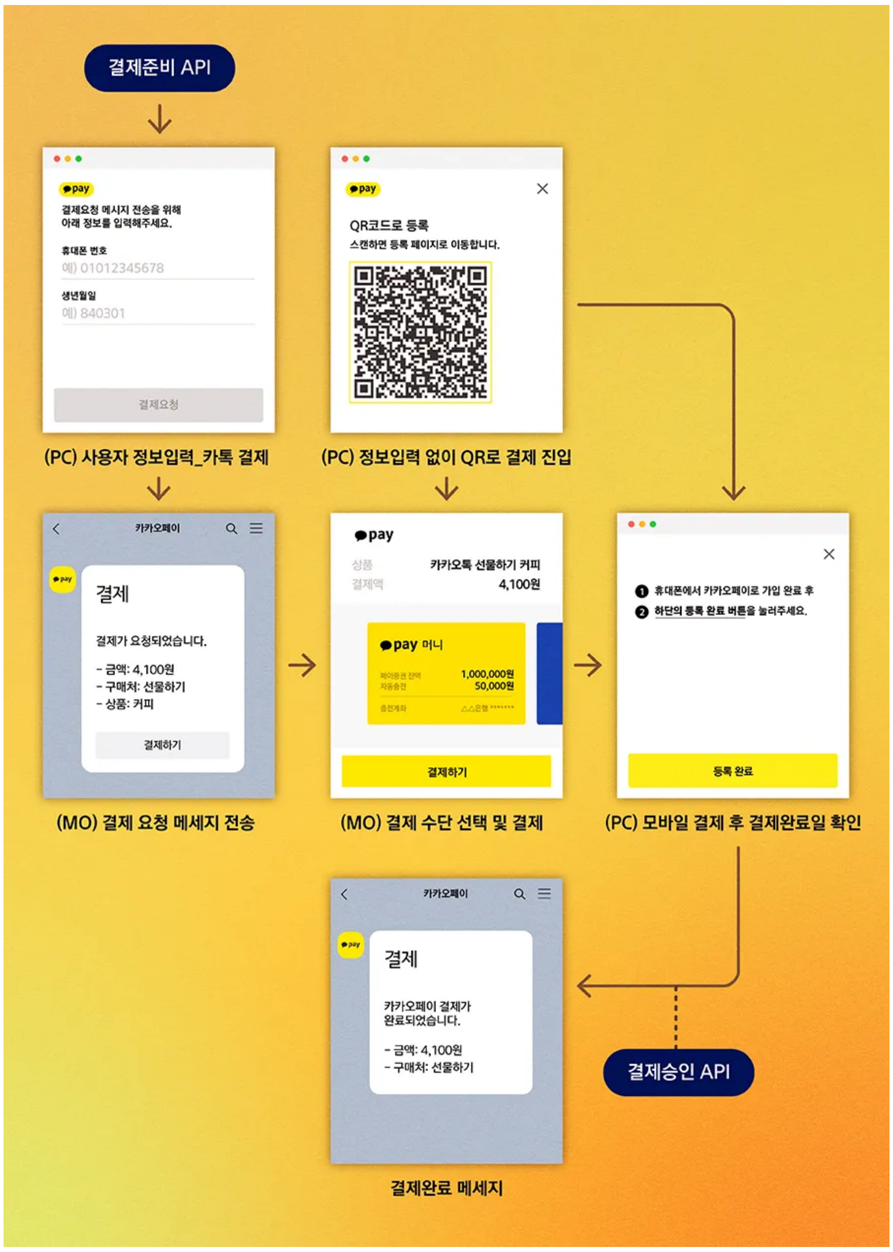
- springboot build

```
# 빌드 경로 이동
cd /palette/backend/palette-spring

# build 파일 권한 부여
sudo chmod 777 ./gradlew
sudo chown 777 ./gradlew

# 빌드
sudo ./gradlew --debug clean build -x test
```

2. OAuth 연동


client
server
OAuth
3
Authorization code 전달
6
Access token 전달
1
Authorization code 요청
2
redirect uri를 통해 Authorization code 부여
4
Authorization code 를 보내서 Access token 요청
5
Access token 부여

3. Portone


결제준비 API
pay
결제요청 메시지 전송을 위해
아래 정보를 입력해주세요.
휴대폰 번호
예) 01012345678
생년월일
예) 840301
결제요청
(PC) 사용자 정보입력_카톡 결제
pay
QR코드로 등록
스캔하면 등록 페이지로 이동합니다.
(PC) 정보입력 없이 QR로 결제 진입
카카오페이
결제
결제가 요청되었습니다.
- 금액: 4,100원
- 구매처: 선물하기
- 상품: 커피
결제하기
(MO) 결제 요청 메세지 전송
pay
상품 카카오톡 선물하기 커피
결제액 4,100원
pay 머니
1,000,000원
50,000원
결제하기
(MO) 결제 수단 선택 및 결제
❶ 휴대폰에서 카카오페이로 가입 완료 후
❷ 하단의 등록 완료 버튼을 눌러주세요.
등록 완료
(PC) 모바일 결제 후 결제완료일 확인
카카오페이
결제
카카오페이 결제가
완료되었습니다.
- 금액: 4,100원
- 구매처: 선물하기
결제승인 API
결제완료 메세지

- **기존의 카카오 페이 흐름**

  [준비단계 → 인증단계 → 인증완료응답 → 결제승인 단계]

- **포트원 카카오 페이 흐름**
  - 포트원은 결제창 호출을 위한 함수 호출과 콜백(또는 redirect_url)을 통한 최종결과 수신으로 위 과정을 축약