## 프로젝트 사용 도구

* **이슈 관리** : JIRA
* **형상 관리** : Gitlab
* **실시간 모니터링** : Prometheus, Grafana
* **CI/CD :** Jenkins
* **커뮤니케이션** : Notion, Discord, Mattermost
* **UCC** : CapCut

## 개발환경

* **VS Code** : 1.90.2
* **IntelliJ** : 2024.1.4 (Ultimate)
* **JVM** : 17.0.2
* **SpringBoot** : 3.3.1
* **Node.js** : 20.15.0
* **React**: 18.3.0
* **Python** : 3.10
* **SERVER** : AWS EC2 Ubuntu 20.04.6 LTS
* **DB** : MariaDB(11.4.2), redis(7.4.0)

## 외부 서비스

* **GPT** : [API key 입력]

## 4. 환경 변수 설정

## Backend(application.yaml)

|  |
| --- |
| sspring:application:name: appdata:redis:host: redisport: 6379datasource:driver-class-name: org.mariadb.jdbc.Driverusername: rootpassword: ssafyurl: jdbc:mariadb://mariadb:3306/aicoupjpa:hibernate:ddl-auto: noneproperties:hibernate:format\_sql: truedatabase-platform: org.hibernate.dialect.MariaDBDialectsql:init:schema-locations: classpath:schema.sqldata-locations: classpath:data.sqlencoding: UTF-8mode: alwayslogging:level:# org.hibernate.SQL: debugorg.hibernate.type: traceaiot:url: http://192.168.137.191:8000startApi: /api/boardgame-start?player\_num=4realApi: /api/board/game-statusgpt:openai:model1: ft:gpt-3.5-turbo-0125:personal::9o0wQY8K #action APImodel2: ft:gpt-3.5-turbo-0125:personal:10none:9olewIAr #challenge APImodel3: ft:gpt-3.5-turbo-0125:personal:counteractionbasic:9oPJtrPR #counter action APImodel4: ft:gpt-3.5-turbo-0125:personal:ca-challengebasic:9qCYIZev #counter action challenge APImodel5: gpt-3.5-turbo-0125 #dialog APIsecret-key: sk-proj-979P2KmZftRu0cUt9sR2T3BlbkFJrr3TRyORsZ2pMqjuE8Vgapi-url: https://api.openai.com/v1/chat/completionsfrontend:url: <https://i11a206.p.ssafy.io>#Jetson nano용 빌드/배포#url: http://192.168.137.191cookie:name: aiCoup |

## React (environment.env)

|  |
| --- |
| VITE\_BACKEND\_SERVER='https://i11a206.p.ssafy.io'VITE\_FRONTEND\_SERVER='https://i11a206.p.ssafy.io'VITE\_WEBSOCKET\_SERVER='wss://i11a206.p.ssafy.io/api/game'# Jetson nano용 빌드/배포# VITE\_BACKEND\_SERVER='http://192.168.137.191:8081# VITE\_FRONTEND\_SERVER='http://192.168.137.191:3000'# VITE\_WEBSOCKET\_SERVER='ws://192.168.137.191:8081/api/game' |

## 5. CI/CD, 모니터링 구축

## Docker 설치하기

## 우분투 시스템 패키지 업데이트

## sudo apt-get update

## sudo apt-get install ca-certificates curl

## sudo install -m 0755 -d /etc/apt/keyrings

## Docker의 공식 GPG키를 추가

## sudo curl -fsSL [https://download.docker.com/linux/ubuntu/gpg -o etc/apt/keyrings/docker.asc](https://download.docker.com/linux/ubuntu/gpg%20-o%20etc/apt/keyrings/docker.asc)

## sudo chmod a+r /etc/apt/keyrings/docker.asc

## Docker의 공식 apt 저장소를 추가

## echo \ "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docke $(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \ sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

## sudo apt-get update

1. **Docker 패키지 설치**

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

1. **설치확인**

sudo docker run hello-world

* **docker-compose 설치**

1. **docker-compose 설치**

sudo curl -SL https://github.com/docker/compose/releases/download/v2.28.1/docker-compose-linux-x86\_64 -o /usr/local/bin/docker-compose

1. **docker-compose 권한 부여**

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

1. **docker-compose 심볼릭 링크 지정**

sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

1. **정상 설치 확인**

docker-compose –v

* **Jenkins 설치하기**

1. **Jenkins 다운 및 실행**

docker run -d --name jenkins \

-e TZ=Asia/Seoul \

-u root \

-p 9090:8080 \

-v /var/jenkins\_home:/var/jenkins\_home \

-v /var/run/docker.sock:/var/run/docker.sock \

-v /usr/bin/docker:/usr/bin/docker \

jenkins/jenkins:latest-jdk17

* **Jenkins Plugin 추가 설치**

Prometheus metrics plugin버전780.v7c50a\_d288424

NodeJS버전1.6.1

Pipeline Graph Analysis Plugin버전216.vfd8b\_ece330ca\_

Pipeline: Build Step버전540.vb\_e8849e1a\_b\_d8

Pipeline: Stage Step버전312.v8cd10304c27a\_

Pipeline: Stage Tags Metadata버전2.2205.vc9522a\_9d5711

Pipeline: Stage View Plugin버전2.34

* **Jenkins Prometheus, GitLab, NodeJS 설정**

첨부이미지 참조

* **Prometheus 설정파일 (Prometheus.yml)**

|  |
| --- |
| global:  scrape\_interval: 15s  evaluation\_interval: 15s  scrape\_timeout: 10s  alerting:  alertmanagers:  - static\_configs:  - targets:  - alertmanager:9093  rule\_files:  scrape\_configs:  - job\_name: 'prometheus'  static\_configs:  - targets: ['localhost:9091']  - job\_name: 'jenkins'  metrics\_path: /jenkins/prometheus/  static\_configs:  - targets: ['i11a206.p.ssafy.io'] |

* **Grafana, Prometheus 설치**

|  |
| --- |
| sudo docker run -d -p 3000:3000 --name=grafana grafana/grafana-oss  docker run -p 9091:9091 –v home/ubuntu/prometheus.yml:/etc/prometheus/prometheus.yml prom/Prometheus  sudo docker run -p 9091:9091 -v /home/ubuntu/prometheus.yml:/etc/prometheus/prometheus.yml prom/prometheus --config.file=/etc/prometheus/prometheus.yml --web.external-url=<https://i11a206.p.ssafy.io/> |

* **Jenkins data 확인**

<https://i11a206.p.ssafy.io/jenkins/prometheus/>

* **Grafana, Prometheus 연동**

Home -> Connections -> Datasources -> Add new

첨부 이미지 참조

* **포트 허용**

|  |
| --- |
| sudo ufw enable  sudo ufw allow 9091  sudo ufw allow 3000  sudo ufw reload  # Jetson nano용 빌드/배포  # sudo ufw enable  # sudo ufw allow 22  # sudo ufw allow 8081  # sudo ufw allow 3000  # sudo ufw reload |

1. **빌드하기**

* **Backend 서버**

|  |
| --- |
| cp /var/jenkins\_home/application/application.yaml /var/jenkins\_home/workspace/gitlab-jenkins/CC/BE/app/src/main/resources/application.yaml  cd /var/jenkins\_home/workspace/gitlab-jenkins/CC/BE/app  chmod +x gradlew  ./gradlew -x test clean build  # Jetson nano용 빌드 |

* **Frontend 서버**

cd /var/jenkins\_home/workspace/gitlab-jenkins/CC/FE

yarn install

yarn run lint

yarn run build

**7. 배포하기**

* **Backend dockerfile (Dockerfile)**

|  |
| --- |
| FROM openjdk:17  EXPOSE 8081  COPY app-0.0.1-SNAPSHOT.jar deploy/app-0.0.1-SNAPSHOT.jar  ENTRYPOINT java -jar deploy/app-0.0.1-SNAPSHOT.jar |

* **Backend docker compose (docker-compose.yml)**

|  |
| --- |
| services:  spring:  container\_name: spring-hey  build:  context: .  dockerfile: Dockerfile  environment:  - SPRING\_DATASOURCE\_URL=jdbc:mariadb://mariadb:3306/aicoup  - SPRING\_DATASOURCE\_USERNAME=root  - SPRING\_DATASOURCE\_PASSWORD=ssafy  - SPRING\_REDIS\_HOST=redis  - SPRING\_REDIS\_PORT=6379  - SERVER\_PORT=8081  ports:  - 8081:8081  volumes:  - /var/jenkins\_home/workspace/gitlab-jenkins/CC/BE/app/build/libs:/deploy  depends\_on:  redis:  condition: service\_healthy  mariadb:  condition: service\_healthy  redis:  container\_name: redis  ports:  - 6379:6379  image: redis:latest  healthcheck:  test: ["CMD", "redis-cli", "ping"]  interval: 5s  retries: 10  mariadb:  container\_name: mariadb  image: mariadb:11.4.2  ports:  - 3306:3306  environment:  MYSQL\_ROOT\_PASSWORD: ssafy MYSQL\_DATABASE: aicoup  MYSQL\_USER: ssafyy  MYSQL\_PASSWORD: ssafy  healthcheck:  test: [ "CMD", "healthcheck.sh", "--su-mysql", "--connect", "--innodb\_initialized" ]  interval: 5s  retries: 10 |

* **Backend 서버 배포**

sudo cp /var/jenkins\_home/workspace/gitlab-jenkins/CC/BE/app/build/libs/app-0.0.1-SNAPSHOT.jar .

sudo docker compose up --build –d

* Jetson nano용 Backend 서버 배포

|  |
| --- |
| cd dockerfiles  mkdir spring-dockerfile  chown orin spring-dockerfile  scp -r ubuntu@i11a206.p.ssafy.io:/home/ubuntu/spring-dockerfile/ /home/orin/dockerfiles/spring-dockerfile/  cd spring-dockerfile  sudo docker compose up --build –d |

* **NginX config 파일 (default.conf)**

|  |
| --- |
| server {  listen [::]:80;  listen 80;  server\_name i11a206.p.ssafy.io;  location ~ /.well-known/acme-challenge {  allow all;  root /var/www/certbot;  }  location / {  return 301 https://i11a206.p.ssafy.io$request\_uri;  }  }  server {  listen 443 ssl;  server\_name i11a206.p.ssafy.io;  root /usr/share/nginx/html;  index index.html;  include snippets/error\_pages.conf;  ssl\_certificate /etc/nginx/ssl/archive/i11a206.p.ssafy.io/fullchain1.pem;  ssl\_certificate\_key /etc/nginx/ssl/archive/i11a206.p.ssafy.io/privkey1.pem;  location /jenkins {  proxy\_pass http://54.180.9.157:9090;  proxy\_redirect default;  proxy\_http\_version 1.1;  # Required for Jenkins websocket agents  proxy\_set\_header Connection "Upgrade";  proxy\_set\_header Upgrade $http\_upgrade;  proxy\_set\_header Host $http\_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;  proxy\_max\_temp\_file\_size 0;  #this is the maximum upload size  client\_max\_body\_size 10m;  client\_body\_buffer\_size 128k;  proxy\_connect\_timeout 90;  proxy\_send\_timeout 90;  proxy\_read\_timeout 90;  proxy\_request\_buffering off;  }  location /api{  proxy\_pass http://54.180.9.157:8081;  proxy\_http\_version 1.1;  proxy\_set\_header Connection "Upgrade";  proxy\_set\_header Upgrade $http\_upgrade;  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;  proxy\_set\_header Accept-Encoding gzip;  }  location / {  try\_files $uri /index.html;  }  } |

* **NginX docker compose file(docker-compose.yml)**

|  |
| --- |
| services:  web:  image: nginx:latest  container\_name: nginx  restart: always  volumes:  - ./data/conf.d:/etc/nginx/conf.d  - ./data/certbot/conf:/etc/nginx/ssl  - ./data/certbot/data:/var/www/certbot  - ./data/html:/usr/share/nginx/html  ports:  - 80:80  - 443:443  certbot:  image: certbot/certbot:latest  command: certonly --webroot --webroot-path=/var/www/certbot --agree-tos --no-eff-email -d i11a206.p.ssafy.io  volumes:  - ./data/certbot/conf:/etc/letsencrypt  - ./data/certbot/logs:/var/log/letsencrypt  - ./data/certbot/data:/var/www/certbot  sudo docker compose up --build –d |

* **NginX error page 설정**

|  |
| --- |
| sudo docker exec –it nginx bash  mkdir -p /srv/http/default  git clone https://github.com/denysvitali/nginx-error-pages /srv/http/default  mkdir /etc/nginx/snippets/  ln -s /srv/http/default/snippets/error\_pages.conf /etc/nginx/snippets/error\_pages.conf  ln -s /srv/http/default/snippets/error\_pages\_content.conf /etc/nginx/snippets/error\_pages\_content.conf |

* **Frontend 배포**

|  |
| --- |
| sudo cp -a /var/jenkins\_home/workspace/gitlab-jenkins/CC/FE/dist/. /home/ubuntu/data/html  sudo docker restart nginx' |

* **Jetson Nano Frontend Dockerfile(Dockerfile.pro)**

|  |
| --- |
| FROM node:20.15.0-alpine  WORKDIR /app  ADD . /app  RUN yarn install  ENV VITE\_BACKEND\_SERVER=http://192.168.137.191:8081  ENV VITE\_FRONTEND\_SERER=http://192.168.137.191:3000  ENV VITE\_WEBSOCKET\_SERVER=WS://192.168.137.191:8081/api/game  EXPOSE 3000  RUN yarn build  RUN yarn global add serve  CMD ["serve", "-s", "dist"] |

sudo docker compose up --build –d