**DanStep**

**포팅 매뉴얼**

SSAFY 11기 서울 4반 A406

원민혁, 강다현, 김연욱, 윤상일, 윤정섭, 탁인혁

**목차**

1. **사용한 환경**
2. 프로젝트 도구
3. 개발 도구 및 환경
4. **빌드 및 실행**
5. EC2 서버에 접속
6. EC2 서버에 Docker 설치
7. EC2에 Nginx 설치
8. SSL – cerbot 설치
9. SSL - 인증서 발급
10. cerbot - 자동 갱신
11. nginx 재시작
12. EC2에 Docker MySQL 설정
13. Front 배포 (repository 내 front 디렉토리)
14. Backend 배포 (repository내 backend 디렉토리)
15. **배포 시 특이사항**
16. **프로젝트에 활용되는 주요 파일 목록**
17. frontend (Dockerfile)
18. frontend (nginx.conf)
19. backend (Dockerfile)
20. backend (application.properties)
21. backend (build.gradle)
22. backend (com.danstep.entity - 테이블 생성 용도)
23. **외부 서비스**
24. S3
25. IAM
26. CloudFront
27. **서비스 이용 방법(시연 시나리오)**

**1. 사용한 환경**

**1) 프로젝트 도구**

- Jira

- Figma

- Notion

- Mattermost

- Jenkins

**2) 개발 도구 및 환경**

- OpenJDK17

- IntelliJ IDEA 2024.2 (Ultimate Edition)

- Visual Studio Code 1.90.2

- AWS EC2

- AWS S3

- AWS CloudFront

**2. 빌드 및 실행**

**1) EC2 서버에 접속**

**2) EC2 서버에 Docker 설치**

2.1 패키지 목록 업데이트

$ sudo apt update

2.2 필수 패키지 설치

$ sudo apt install apt-transport-https ca-certificates curl gnupg-agent software-properties-common

2.3 Docker GPG 키 추가

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –

2.4 Docker 저장소 추가

$ sudo add-apt-repository "deb [arch=amd64] \

https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

2.5 Docker 설치

$ sudo apt update

$ sudo apt install docker-ce

2.6 Docker 서비스 시작

$ sudo systemctl start docker

$ sudo systemctl enable docker

2.7 Docker 설치 확인

$ docker –version

2.8 도커 로그인

$ sudo docker login -u [도커 허브 아이디]

2.9 컨테이너 간 네트워크 설정

$ docker network create danstep-network

**3. EC2에 Nginx 설치**

3.1 패키지 업데이트

$ sudo apt update

3.2 설치

$ sudo apt-get install nginx

3.3 80, 443번 포트개방

$ sudo ufw allow 'Nginx FULL'

3.4 ssh 접속을 위한 22번 포트 개방

$ sudo ufw allow ssh

3.5 방화벽 적용

$ sudo ufw enable

3.6 nginx 구동 상태 확인

$ sudo systemctl status nginx

3.7 서버 부팅시 자동 시작

$ sudo systemctl enable nginx

3.8 nginx 서비스 시작

$ sudo systemctl start nginx

**4) SSL - certbot 설치**

$ sudo apt install certbot python3-certbot-nginx

**5) SSL - 인증서 발급**

$ sudo certbot --nginx -d i11a406.p.ssafy.io

(이후 2번 선택하여 http요청을 https로 redirect하게 함**)**

**6) certbot - 자동 갱신**

$ sudo certbot renew --dry-run

**7) nginx 재시작**

7.1 설정 파일 검증

$ sudo nginx –t

7.2 Nginx 재시작

$ sudo systemctl restart nginx

**8) EC2에 Docker MySQL 설정**

8.1 docker 컨테이너 실행

$ docker run –d --name danstep-mysql-container -e TZ=Asia/Seoul \

-e MYSQL\_ROOT\_PASSWORD='tkrhddldutjt406!@#$' \

-e MYSQL\_DATABASE='danstep\_db' \

-e MYSQL\_USER='danstep406' \

-e MYSQL\_PASSWORD='tkrhddldutjt406' \

--network danstep-network \

mysql:latest

8.2 docker 컨테이너 쉘 접속

$ docker exec -it danstep-mysql-container /bin/bash

8.3 mysql 접속

$ mysql -u root –p

8.4 danstep406 계정 설정

mysql> GRANT ALL PRIVILEGES ON danstep\_db.\* TO 'danstep406'@'%';

mysql> FLUSH PRIVILEGES;

**9) Front 배포 (repository내 front 디렉토리)**

9.1 docker build (ec2 – frontend폴더 내 최상단)

$ sudo docker build -t danstep-frontend-image .

9.2 docker 컨테이너 실행

$ sudo docker run -d --name danstep-frontend-container -e TZ=Asia/Seoul -p 80:80 -p 443:443 \

-v /etc/letsencrypt/live/i11a406.p.ssafy.io/fullchain.pem:/etc/letsencrypt/live/i11a406.p.ssafy.io/fullchain.pem \

-v /etc/letsencrypt/live/i11a406.p.ssafy.io/privkey.pem:/etc/letsencrypt/live/i11a406.p.ssafy.io/privkey.pem \

--network danstep-network \

danstep-frontend-image

**10) backend 배포 (repository내 backend 디렉토리)**

10.1 docker build (ec2 - backend폴더 내 최상단)

$ sudo docker build –t danstep-backend-image .

10.2 docker 컨테이너 실행

$ sudo docker run -d --name danstep-backend-container -e TZ=Asia/Seoul \

--network danstep-network \

danstep-backend-image

**3. 배포 시 특이사항**

- Jenkins와 Gitlab을 활용하였으며, Webhooks를 통해 Jenkinsfile에 정의된 내용으로 Docker를 활용하여 자동 배포가 되도록 하였습니다

- 그러다 보니 Gitlab repository내에 있는 frontend와 backend 파일을 서버에 직접 업로드 후 명령어를 입력하는 형식으로 바꿔서 문서를 작성해 보았습니다

- 중간에 잘 안 되는 경우가 있을 수 있으며, Gitlab과 Jenkins를 Webhooks를 통해 연결하여 띄우는 것을 권장 드립니다

- AWS S3와 AWS CloudFront를 추가로 사용합니다 (추가 비용 발생 가능)

**4. 프로젝트에 활용되는 주요 파일 목록**

**1) frontend (Dockerfile)**

# 단계 1: 빌드 이미지 생성

# 최신 LTS 버전 사용

FROM node:lts AS build

# 작업 디렉토리 설정

WORKDIR /app

# 패키지 파일 복사

COPY package.json ./

COPY package-lock.json ./

# 의존성 설치

RUN npm install

# 애플리케이션 소스 코드 복사

COPY . .

# 애플리케이션 빌드

RUN npm run build

# 단계 2: Nginx 이미지 생성

FROM nginx:alpine

# Nginx 설정파일 복사

COPY nginx.conf /etc/nginx/conf.d/i11a406.p.ssafy.io.conf

# 빌드된 React 애플리케이션을 Nginx의 기본 HTML 디렉토리로 복사

COPY --from=build /app/build /usr/share/nginx/html

# Nginx 실행

CMD ["nginx", "-g", "daemon off;"]

**2) frontend (nginx.conf)**

server {

listen 80;

server\_name i11a406.p.ssafy.io;

return 301 https://$host$request\_uri; # HTTP 요청을 HTTPS로 리디렉션

}

server {

listen 443 ssl;

server\_name i11a406.p.ssafy.io;

client\_max\_body\_size 10M; # 요청 크기 10MB로 설정

ssl\_certificate /etc/letsencrypt/live/i11a406.p.ssafy.io/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/i11a406.p.ssafy.io/privkey.pem;

location / {

root /usr/share/nginx/html; # 웹사이트의 루트 디렉토리

index index.html index.htm;

try\_files $uri /index.html; # React Router를 사용하는 경우

}

location /api {

proxy\_pass http://danstep-backend-container:8080; # Spring Boot API와 연동 (docker의 danstep-backend-container)

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;

}

}

**3) backend (Dockerfile)**

# 1단계: 빌드 환경

FROM openjdk:17-jdk-slim AS builder

# 애플리케이션 소스 코드 복사

COPY . /app

WORKDIR /app

# Gradle Wrapper에 실행 권한 부여

RUN chmod 755 gradlew

# Gradle Wrapper를 사용하여 애플리케이션 빌드

# RUN ./gradlew build

RUN ./gradlew clean build -x test

# 2단계: 실행 환경

FROM openjdk:17-slim

# 빌드된 JAR 파일을 실행 환경으로 복사

COPY --from=builder /app/build/libs/\*.jar app.jar

# EXPOSE는 문서화로 있어도 없어도 상관 없음

# EXPOSE 8080

# JAR 파일을 실행

CMD ["java", "-jar", "app.jar"]

**4) backend (application.properties)**

spring.application.name=danstep

server.port=8080

# datasource

spring.datasource.hikari.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.hikari.jdbc-url=jdbc:mysql://danstep-mysql-container:3306/danstep\_db?serverTimezone=UTC&characterEncoding=UTF-8

spring.datasource.hikari.username=danstep406

spring.datasource.hikari.password=tkrhddldutjt406

spring.datasource.hikari.mapper-locations=classpath:/mapper/\*\*/\*.xml

#jwt

spring.jwt.secret=a4940f306c63e1e78d58002607a232700a95ec50c977af5d01be4e4f8789e8c2a0d5ba28bf5781ffe338b7d8c59ea6a0e5d0c4cf71183937e2690da4a3d98daa

# jpa

spring.jpa.hibernate.ddl-auto=update

# 업로드 파일 용량

spring.servlet.multipart.maxFileSize=50MB

spring.servlet.multipart.maxRequestSize=50MB

# S3

cloud.aws.region.static=ap-southeast-2

cloud.aws.stack.auto-=false

# admin (public, private 다 접근 가능)

cloud.aws.credentials.admin.accessKey=AKIAQFLZDJEWJPZYG47A

cloud.aws.credentials.admin.secretKey=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

# user (public read만 가능)

cloud.aws.credentials.user.accessKey=AKIAQFLZDJEWB34TCWNQ

cloud.aws.credentials.user.secretKey=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

# aws bucket name

cloud.aws.s3.bucket=danstep406

# AWS CloudFront

cloud.aws.cloudfront.domain=d3gnfa5yiksyfe.cloudfront.net

cloud.aws.cloudfront.url=https://d3gnfa5yiksyfe.cloudfront.net

# user\_info regex

username.regex=^[a-zA-Z0-9]{4,20}$

nickname.regex=^[a-zA-Z0-9]{2,6}$

password.regex=^[a-zA-Z0-9!@#$%^&\*()\_+=\\[\\]{};':\"\\\\|,.<>/?`~]{8,20}$

**5) backend (build.gradle)**

plugins {

id 'java'

id 'org.springframework.boot' version '3.3.2'

id 'io.spring.dependency-management' version '1.1.6'

}

group = 'com'

version = '0.0.1-SNAPSHOT'

jar {

enabled = false

}

java {

toolchain {

languageVersion = JavaLanguageVersion.of(17)

}

}

configurations {

compileOnly {

extendsFrom annotationProcessor

}

}

repositories {

mavenCentral()

}

dependencies {

implementation 'org.springframework.boot:spring-boot-starter-jdbc'

implementation 'org.springframework.boot:spring-boot-starter-data-jpa'

implementation 'org.springframework.boot:spring-boot-starter-web'

implementation 'org.mybatis.spring.boot:mybatis-spring-boot-starter:3.0.3'

compileOnly 'org.projectlombok:lombok'

developmentOnly 'org.springframework.boot:spring-boot-devtools'

runtimeOnly 'com.mysql:mysql-connector-j'

annotationProcessor 'org.projectlombok:lombok'

testImplementation 'org.springframework.boot:spring-boot-starter-test'

testImplementation 'org.mybatis.spring.boot:mybatis-spring-boot-starter-test:3.0.3'

testRuntimeOnly 'org.junit.platform:junit-platform-launcher'

// Spring Security

implementation 'org.springframework.boot:spring-boot-starter-security'

// jwt

implementation 'io.jsonwebtoken:jjwt-api:0.12.3'

implementation 'io.jsonwebtoken:jjwt-impl:0.12.3'

implementation 'io.jsonwebtoken:jjwt-jackson:0.12.3'

// AWS S3

implementation 'org.springframework.cloud:spring-cloud-starter-aws:2.2.6.RELEASE'

implementation 'com.amazonaws:aws-java-sdk-s3:1.12.100'

}

tasks.named('test') {

useJUnitPlatform()

}

**6) backend (com.danstep.entity – 테이블 생성 용도)**

**5. 외부 서비스**

**1) S3**

- 버킷 생성

- private 폴더 생성 : /private/games, /private/users

- public 폴더 생성 : /public/games, /public/users

**2) IAM**

- S3 읽기 전용 IAM 생성

- S3 Full Access IAM 생성

- 각각 해당하는 Access Key, Secret Key 필요

**3) CloudFront**

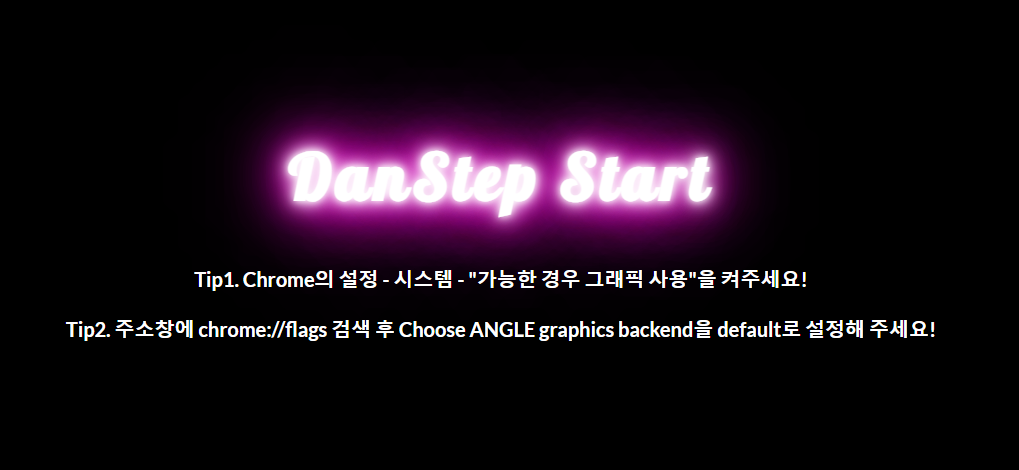
- S3에 연결

- 동작 편집에서 뷰어 프로토콜 정책 HTTP & HTTPS 선택

- 동작 편집에서 허용된 HTTP 방법 GET, HEAD 선택

**6. 서비스 이용 방법**

1. 시작 화면에서 DanStep Start 버튼 클릭

****

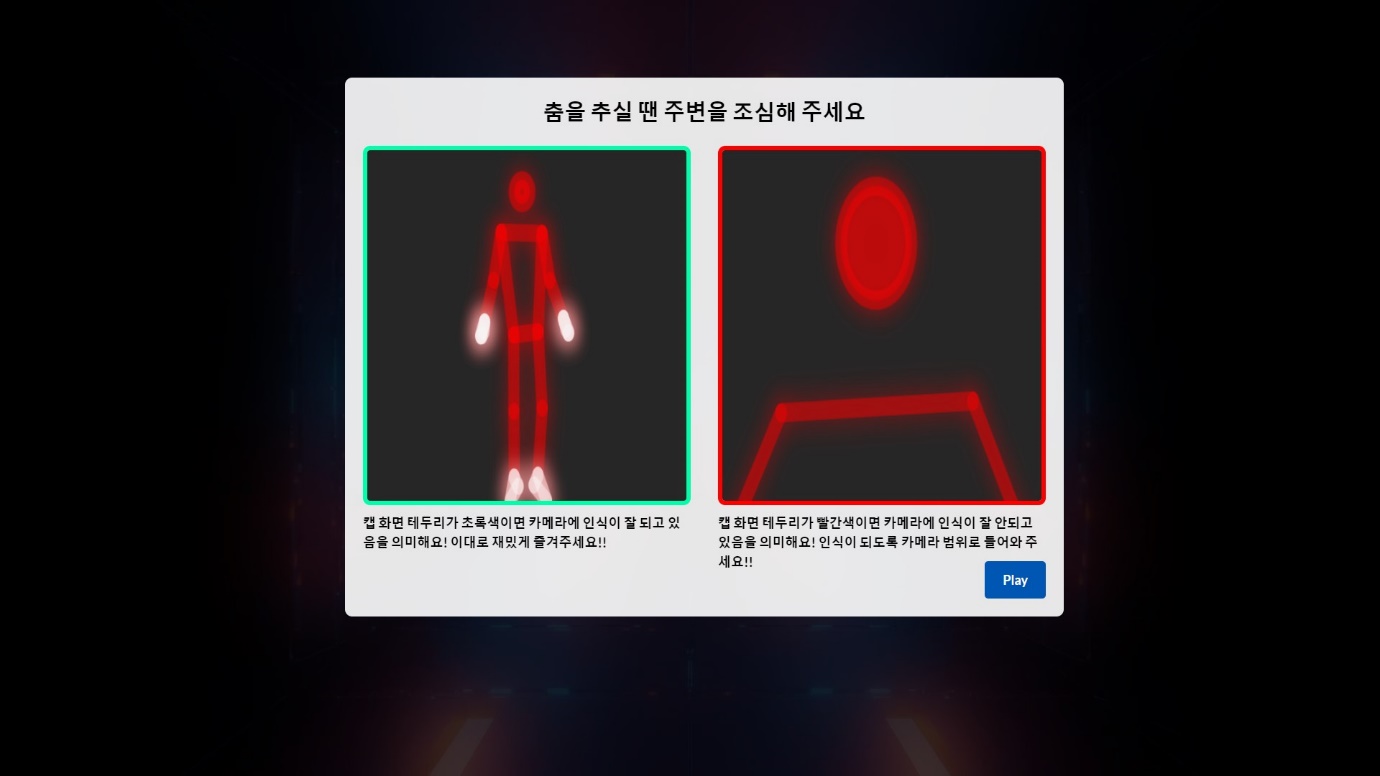
1. 메인 화면 이동 후 마우스 휠을 이용하여 게임 탐색

****

1. 사용자는 원하는 게임을 선택 후 게임 페이지로 이동

****

1. 게임 페이지 이동 시 캠 인식 정보 및 플레이 방법 가이드 창 출력

****

1. 사용자의 캠이 인식되면 Loading창 종료

****

1. 사용자의 위치에 따른 메시지 유도 (Go Back, Go Front, Inside Frame)

****

1. 사용자의 위치가 알맞을 시 색과 텍스트를 통해 알림 (초록색, Hands Up)



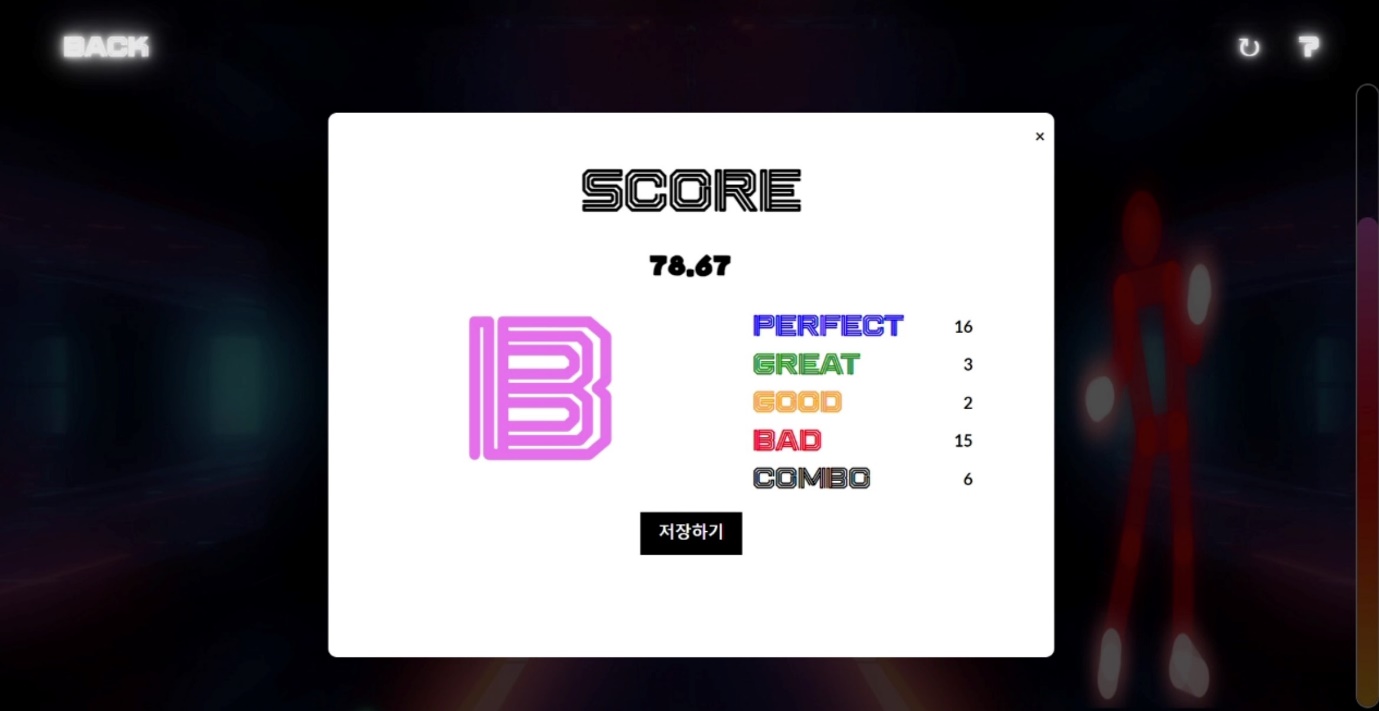
1. 사용자가 양손을 들어 게임을 시작할 시 주변 선 제거 및 카운트 다운 시작
2. 카운트 다운 중 좌측 화면에 사용자가 제일 처음 시작해야할 자세 출력



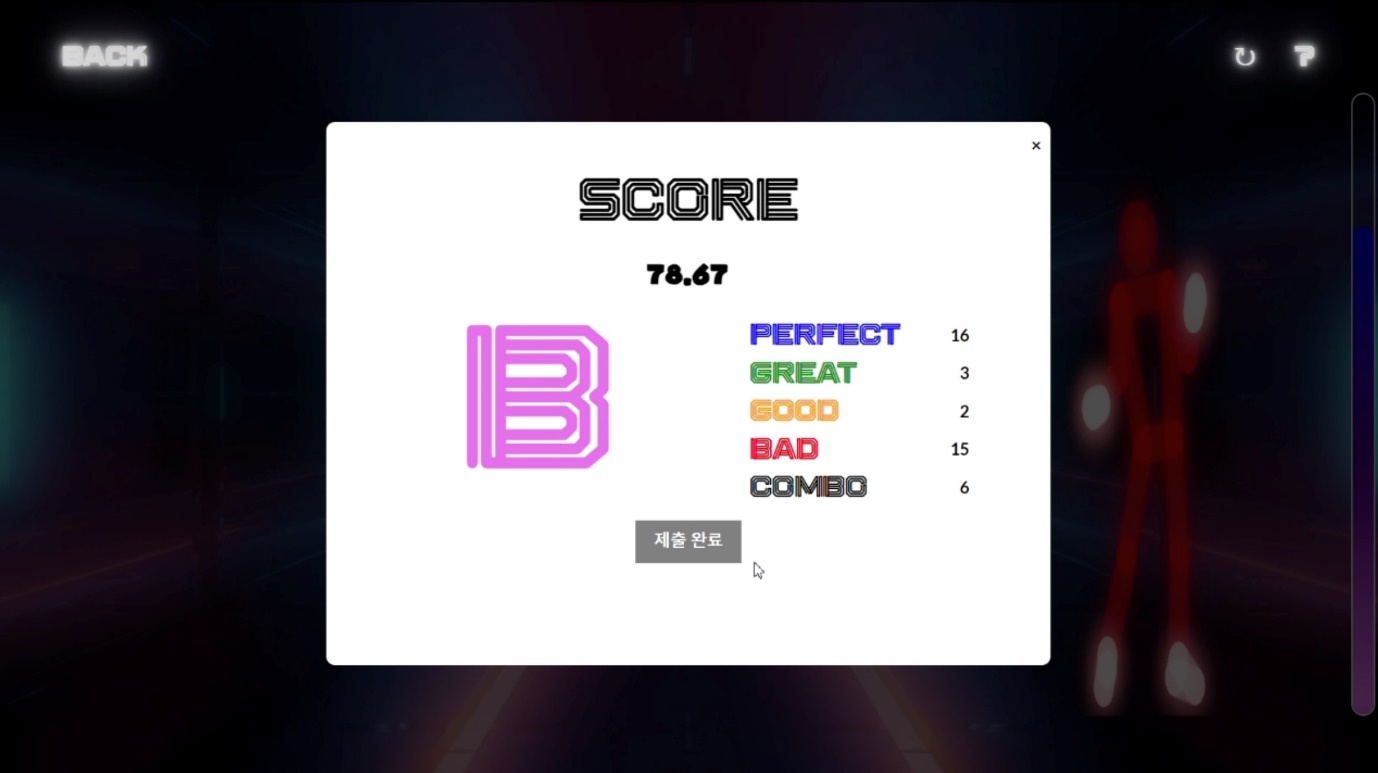
1. 게임이 시작되면 현재 사용자와 안무의 일치율을 실시간으로 출력(Perfect, great, good, bad)하며 일치율이 bad가 되지 않을 시 combo 누적



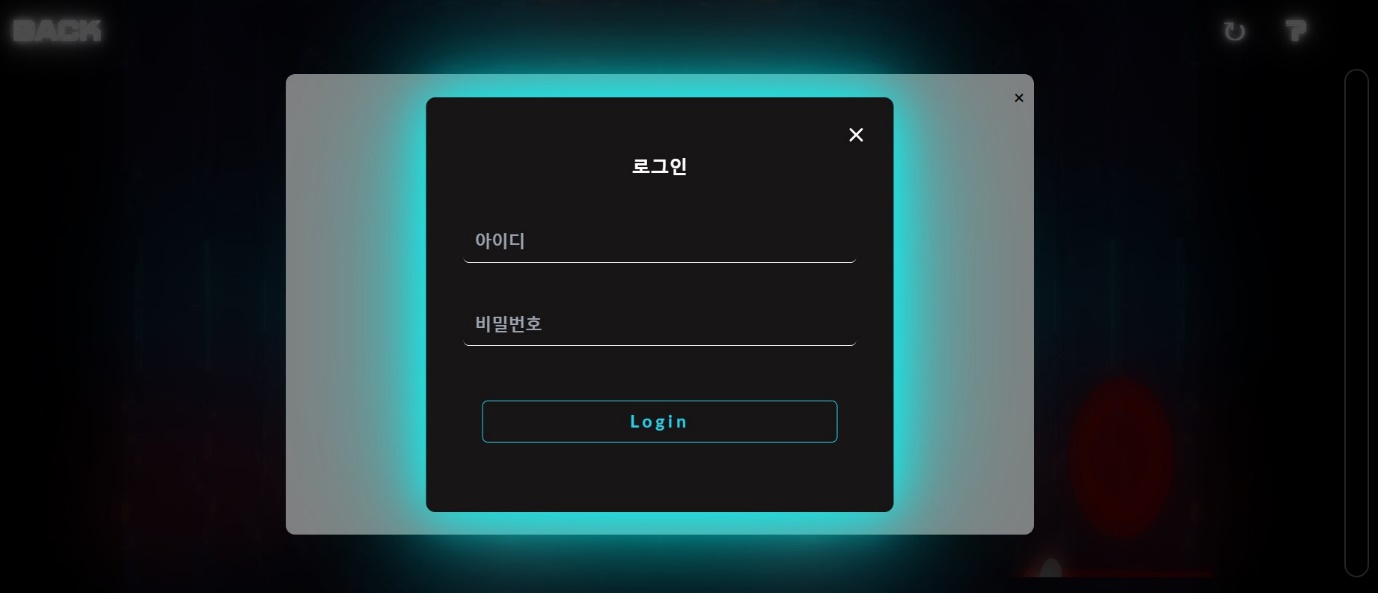
1. 게임 종료 후 사용자의 점수와 등급을 출력



1. 사용자에게 우측 상단의 새로 고침 아이콘을 클릭하여 리플레이 기능 제공 및 플레이 저장 기능 제공



1. 로그인이 되어있지 않다면 저장 시 로그인 화면 출력



1. 메인페이지에서 우측 상단의 프로필 아이콘을 클릭 시 메인페이지로 이동

****

1. 메인페이지에서는 사용자에게 플레이 기록, 정보 수정, 메인 메뉴 이동 기능 및 로그아웃 기능 제공
2. 플레이 기록은 최신 기록 순으로 정렬 되며 다시보기, 삭제 기능 제공

****

1. 재생 버튼을 클릭 하면 이전에 플레이 했던 춤의 과정, 점수 출력, 결과 확인 가능

