PNU SW학습공동체 중간 보고서

1. 프로젝트 소개

가. 배경 및 필요성

학교 근처에서 식사를 해결해야 할 일이 많지만, 대부분의 학생들이 한정된 몇몇 식당만 반복적으로 이용하는 것을 보고, "오늘 뭐 먹지?"라는 고민을 덜어줄 수 있는 서비스의 필요성을 느꼈습니다. 여기에 더해, AI API를 활용해보며 실제 서비스에 적용해보고 싶다는 개발 동기도 있었습니다. 이 두 가지를 결합해, 사용자 맞춤형 식당 추천 앱을 기획하게 되었습니다.

나. 개발목표 및 주요내용, 세부내용 등

• 개발 목표

본 프로젝트의 개발 목표는 학교 근처에서 식사를 해결해야 하는 학생들이 매일 겪는 "오늘 뭐 먹지?"라는 고민을 덜어줄 수 있는 맞춤형 식당 추천 서비스를 제공하는 것입니다. 특히 반복적으로 이용하는 몇몇 식당만 떠올리는 기존의 패턴에서 벗어나, AI 기반 추천 시스템을 통해 다양한 선택지를 제시하고 새로운 식당을 발견할 수 있는 기회를 제공하고자 합니다. 이 과정에서 Google의 Gemini API를 활용하여 실제 서비스에 AI API를 적용해보는 경험을 쌓으려 합니다. 이를 통해 실질적인 기술 역량을 향상시키고, AI 추천 서비스 개발 역량을 키우고자 합니다.

• 주요 기능

본 서비스는 크게 세 가지 핵심 기능으로 구성됩니다.

첫째, AI 기반 식당 추천 시스템을 통해 사용자의 취향에 맞는 식당을 추천합니다.

둘째, 식당 평가 시스템을 도입해 사용자가 추천받은 식당에 대해 직접 평가하고 피드백을 남기면, 해당 정보를 반영해 추천 정확도를 지속적으로 개선해 나갑니다.

마지막으로, 개인뿐만 아니라 여러 명이 함께 식사할 경우에도 인원과 구성원의 선호를 고려한 집단 추천 시스템을 제공하여, 소모임이나 동아리 활동 등 다양한 상황에서도 유용하게 사용할 수 있도록 설계할 예정입니다

• 세부 내용

본 서비스가 구현해야 할 핵심 기능적 요구사항으로는 먼저, 사용자 취향 등의 정보를 토대로 개인 맞춤형 식당을 추천하는 기능이 있습니다. 추천 결과는 Gemini API와 연동하여 AI 기반으로

도출되며, 이를 통해 사용자별로 최적화된 결과를 제공합니다.

또한, 추천받은 식당에 대해 사용자가 별점이나 한줄평 등의 평가를 남길 수 있는 피드백 시스템을 마련하여, 지속적인 서비스 개선과 추천 정확도 향상을 도모할 계획입니다. 아울러, 개인 추천 외에도 여러 명이 함께 식사하는 상황을 고려한 집단 추천 기능을 구현하여, 인원수와 구성원의 취향을 종합적으로 반영한 추천 결과를 제공할 수 있도록 할 예정입니다.

비기능적 요구사항으로는 서비스의 신뢰성과 빠른 응답 속도가 중요한 요소로 고려됩니다. 특히, 사용자의 만족도를 높이기 위해 편리한 UI/UX 설계가 필수적입니다. 아울러, 사용자의 개인정보와 위치 정보 등 민감한 데이터의 보안을 확보하는 것 역시 중요한 요구사항입니다.

저희 프로젝트의 개발 환경은 웹 서비스를 중심으로 구성하려고 합니다. 백엔드는 Java Spring Boot를 기반으로 개발하여 안정성과 확장성을 확보할 예정이며, 프론트엔드는 Next.js를 활용해 사용자 친화적이고 반응형 웹 인터페이스를 구현할 계획입니다. 데이터베이스는 PostgreSQL을 사용하여 식당 정보, 사용자 정보, 평가 데이터 등 다양한 데이터를 효율적으로 관리하고, AI 추천 시스템과 연동하기 위해 Google Gemini API를 도입합니다. 이를 통해 보다 정교하고 개인화된 식당 추천 서비스를 제공할 수 있도록 설계할 예정입니다.

또한, GitHub을 통해 버전 관리를 체계적으로 진행하고, 협업 도구로 Notion, Figma를 활용해 원활한 커뮤니케이션과 작업 공유가 이루어질 수 있도록 할 계획입니다.

다. 사회적가치 도입 계획 및 활용 계획

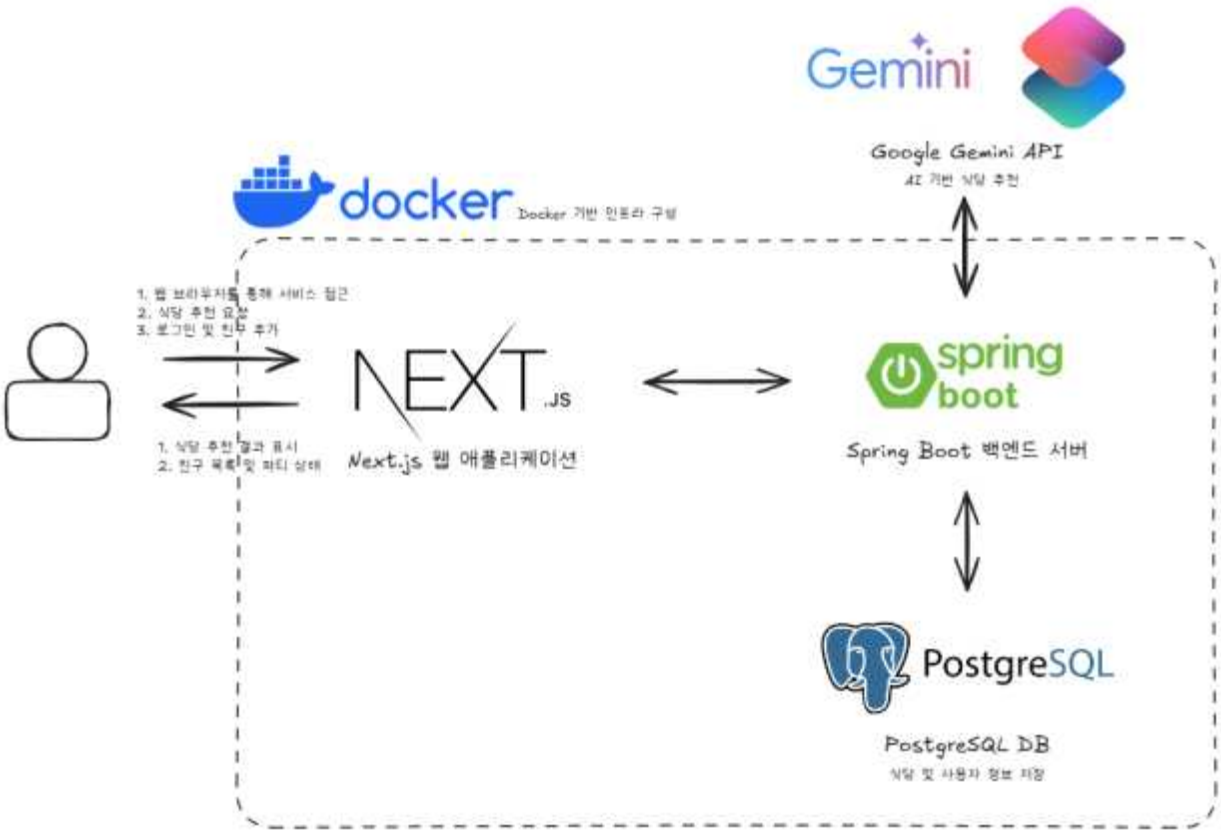
이 앱은 부산대학교 학생들을 주요 대상으로 하여, 학생들이 일상적으로 겪는 식사 선택의 고민을 덜어주고, 다양한 식당을 쉽게 접할 수 있도록 도와주는 서비스입니다. 특히 자취생이나 기숙사 생처럼 학교 근처에서 끼니를 자주 해결해야 하는 학생들에게 유용하게 쓰일 수 있습니다.

지역사회와의 연계 가능성도 고려할 수 있습니다. 예를 들어, 부산대 인근 식당들과 제휴를 통해 프로모션이나 쿠폰을 제공하고, 앱을 통해 해당 식당의 노출 기회를 늘려 지역 상권 활성화에 기여할 수 있습니다. 장기적으로는 추천 정확도를 바탕으로 식당별 매출 증대와 사용자 유입 증대를 연결해 수익화를 노릴 수 있으며, 이후에는 타 대학 혹은 지역으로 확장하는 것도 가능합니다.

수집한 데이터를 바탕으로, 단순한 랜덤 추천이 아닌 사용자 맞춤형 필터링 추천이 가능해집니다. 또한 사용자의 추천 반응(예: 추천 식당 클릭 여부, 방문 여부, 선호도 평가 등)을 수집하고 분석함으로써 추천 알고리즘의 성능을 지속적으로 개선할 수 있습니다.

2. 상세설계

가. 시스템 구성도



나. 사용기술

• Frontend

Next.js, TypeScript, Tailwind CSS : CSR/SSR 하이브리드 렌더링 기반 UI 구현

Zustand : 전역 상태 관리

Lottie, Custom Font : 애니메이션 효과 및 사용자 경험 향상

• Backend

Java 17, Spring Boot, Gradle : REST API 서버 구현, 의존성 관리

Spring Security, JWT : 사용자 인증 및 권한 관리

JPA (Hibernate), PostgreSQL : ORM 기반 데이터 저장 및 쿼리 처리

Swagger / SpringDoc : API 문서 자동화

• DB

PostgreSQL : 관계형 데이터베이스

• AI

Google Gemini API : AI 기반 식당 추천 로직 실행

• Infra

Docker, Docker Compose : 전체 서비스 컨테이너화 및 실행

Nginx (리버스 프록시, 정적파일 제공) : 웹 요청 라우팅 및 보안

• 협업 도구

GitHub, Notion, Figma : 형상관리, 문서 협업, UI 설계

3. 개발 중간 보고

가. 디렉토리 구조

```
# Back
├── EatThisBack
│   ├── .cursorrules
│   ├── .gitattributes
│   ├── .gitignore
│   ├── .gradle
│   ├── .idea
│   │   ├── .gitignore
│   │   ├── compiler.xml
│   │   ├── gradle.xml
│   │   ├── jarRepositories.xml
│   │   ├── misc.xml
│   │   ├── uiDesigner.xml
│   │   ├── vcs.xml
│   │   ├── workspace.xml
│   ├── Dockerfile
│   ├── build
```

```
│   │   ├── reports
│   │   │   ├── problems
│   │   │   │   ├── problems-report.html
│   ├── build.gradle
│   ├── gradle
│   │   ├── wrapper
│   │   │   ├── gradle-wrapper.jar
│   │   │   ├── gradle-wrapper.properties
│   ├── gradlew
│   ├── gradlew.bat
│   ├── pom.xml
│   ├── settings.gradle
│   ├── src
│   │   ├── main
│   │   │   ├── java
│   │   │   │   ├── com
│   │   │   │   │   ├── ydyjj
│   │   │   │   │   │   ├── eatthisback
│   │   │   │   │   │   │   ├── EatThisBackApplication.java
│   │   │   │   │   │   │   ├── common
│   │   │   │   │   │   │   │   ├── ApiResponse.java
│   │   │   │   │   │   │   ├── config
│   │   │   │   │   │   │   │   ├── OpenApiConfig.java
│   │   │   │   │   │   │   │   ├── SecurityConfig.java
│   │   │   │   │   │   │   │   ├── WebConfig.java
```

```
│   │   │   │   │   │   │   ├── controller
│   │   │   │   │   │   │   │   ├── AuthController.java
│   │   │   │   │   │   │   │   ├── DataFavoriteController.java
│   │   │   │   │   │   │   │   ├── DataRestaurantController.java
│   │   │   │   │   │   │   │   ├── DataUserController.java
│   │   │   │   │   │   │   │   ├── DataUserPlaceStatusController.java
│   │   │   │   │   │   │   │   ├── FriendController.java
│   │   │   │   │   │   │   │   ├── GeminiController.java
│   │   │   │   │   │   │   │   ├── UserController.java
│   │   │   │   │   │   │   │   ├── VisitController.java
│   │   │   │   │   │   │   ├── dto
│   │   │   │   │   │   │   │   ├── FavoriteDTO.java
│   │   │   │   │   │   │   │   ├── FriendRequest.java
│   │   │   │   │   │   │   │   ├── LoginResponseDTO.java
│   │   │   │   │   │   │   │   ├── RestaurantDTO.java
│   │   │   │   │   │   │   │   ├── TokenDTO.java
│   │   │   │   │   │   │   │   ├── UserDTO.java
│   │   │   │   │   │   │   │   ├── UserLoginDTO.java
│   │   │   │   │   │   │   │   ├── UserSignupDTO.java
│   │   │   │   │   │   │   │   ├── UserVisitDTO.java
│   │   │   │   │   │   │   │   ├── VisitStatusDTO.java
│   │   │   │   │   │   │   ├── entity
│   │   │   │   │   │   │   │   ├── Allergy.java
│   │   │   │   │   │   │   │   ├── Favorite.java
│   │   │   │   │   │   │   │   ├── Friend.java
```

```
│   │   │   │   │   │   │   ├── Restaurant.java
│   │   │   │   │   │   │   ├── Restaurants.java
│   │   │   │   │   │   │   ├── Role.java
│   │   │   │   │   │   │   ├── User.java
│   │   │   │   │   │   │   ├── UserFavorites.java
│   │   │   │   │   │   │   ├── UserPlaceStatus.java
│   │   │   │   │   │   │   ├── UserVisits.java
│   │   │   │   │   │   │   ├── VisitStatus.java
│   │   │   │   │   │   ├── exception
│   │   │   │   │   │   │   ├── GlobalExceptionHandler.java
│   │   │   │   │   │   ├── repository
│   │   │   │   │   │   │   ├── FavoriteRepository.java
│   │   │   │   │   │   │   ├── FriendRepository.java
│   │   │   │   │   │   │   ├── RestaurantKRepository.java
│   │   │   │   │   │   │   ├── RestaurantRepository.java
│   │   │   │   │   │   │   ├── UserFavoriteRepository.java
│   │   │   │   │   │   │   ├── UserPlaceStatusRepository.java
│   │   │   │   │   │   │   ├── UserRepository.java
│   │   │   │   │   │   │   ├── UserVisitRepository.java
│   │   │   │   │   │   ├── security
│   │   │   │   │   │   │   ├── CustomUserDetailsService.java
│   │   │   │   │   │   │   ├── JwtAuthenticationFilter.java
│   │   │   │   │   │   │   ├── JwtTokenProvider.java
│   │   │   │   │   │   ├── service
│   │   │   │   │   │   │   ├── AuthService.java
```

```
│   │   │   │   │   │   │   │   ├── FriendService.java
│   │   │   │   │   │   │   │   ├── UserService.java
│   │   │   │   │   │   │   │   ├── impl
│   │   │   │   │   │   │   │   │   ├── AuthServiceImpl.java
│   │   │   │   │   │   │   │   │   ├── GeminiServiceImpl.java
│   │   │   │   │   │   │   │   │   ├── HttpRequestServiceImpl.java
│   │   │   │   │   │   │   │   │   ├── RestaurantServiceImpl.java
│   │   │   │   │   │   │   │   │   ├── UserFavoritesServiceImpl.java
│   │   │   │   │   │   │   │   │   ├── UserPlaceStatusServiceImpl.java
│   │   │   │   │   │   │   │   │   ├── VisitServiceImpl.java
│   │   │   ├── resources
│   │   │   │   ├── application.properties
│   │   ├── test
│   │   │   ├── java
│   │   │   │   ├── com
│   │   │   │   │   ├── ydyjj
│   │   │   │   │   │   ├── eatthisback
│   │   │   │   │   │   │   ├── EatThisBackApplicationTests.java


# Front
├── EatThisFront
│   ├── .cursorrules
│   ├── .env.development
│   ├── .env.production
│   ├── .gitignore
```

```
│   ├── Dockerfile
│   ├── README.md
│   ├── app
│   │   ├── favicon.ico
│   │   ├── find
│   │   │   └── page.tsx
│   │   ├── friends
│   │   │   └── page.tsx
│   │   ├── globals.css
│   │   ├── layout.tsx
│   │   ├── login
│   │   │   └── page.tsx
│   │   ├── main
│   │   │   └── page.tsx
│   │   ├── page.tsx
│   │   ├── result
│   │   │   └── page.tsx
│   │   ├── signup
│   │   │   └── page.tsx
│   │   ├── visits
│   │   │   └── page.tsx
│   ├── components
│   │   ├── logo-section.tsx
│   │   ├── logo.tsx
│   │   ├── nav-bar.tsx
```

```
│   │   ├── providers
│   │   │   ├── auth-provider.tsx
│   │   ├── ui
│   │   │   ├── button.tsx
│   │   │   ├── custom-input.tsx
│   │   │   ├── input.tsx
│   ├── eslint.config.mjs
│   ├── hooks
│   │   ├── use-auth.ts
│   ├── lib
│   │   ├── api.ts
│   │   ├── utils.ts
│   ├── next-env.d.ts
│   ├── next.config.js
│   ├── next.config.ts
│   ├── package-lock.json
│   ├── package.json
│   ├── postcss.config.mjs
│   ├── public
│   │   ├── file.svg
│   │   ├── fonts
│   │   │   ├── Pretendard-Black.woff
│   │   │   ├── Pretendard-Bold.woff
│   │   │   ├── Pretendard-ExtraBold.woff
│   │   │   ├── Pretendard-ExtraLight.woff
```

```
│   │   │   ├── Pretendard-Light.woff
│   │   │   ├── Pretendard-Medium.woff
│   │   │   ├── Pretendard-Regular.woff
│   │   │   ├── Pretendard-SemiBold.woff
│   │   │   ├── Pretendard-Thin.woff
│   │   │   ├── logintitle.woff
│   │   ├── globe.svg
│   │   ├── images
│   │   │   ├── camera.svg
│   │   │   ├── logoBackground.png
│   │   │   ├── logoStar.png
│   │   │   ├── mainLogo.png
│   │   │   ├── photoUpload.png
│   │   ├── lottie
│   │   │   ├── cookingLottie.json
│   │   │   ├── desertLottie.json
│   │   ├── next.svg
│   │   ├── vercel.svg
│   │   ├── window.svg
│   ├── store
│   │   ├── use-auth-store.ts
│   │   ├── use-scroll-store.ts
│   ├── tailwind.config.ts
│   ├── tsconfig.json
```

Infra

```
├── EatThisInfra
│   ├── .DS_Store
│   ├── README.md
│   ├── db
│   │   ├── init.sql
│   ├── docker-compose.yml
│   ├── nginx
│   │   ├── nginx.conf
```

4. 팀 소개 (소속, 구성원별 역할)

- 김해중

: 정보컴퓨터공학부, 3학년

: 프론트앤드 개발

- 김동윤

: 정보컴퓨터공학부, 4학년

: 백앤드 개발

- 권윤재

: 정보컴퓨터공학부, 3학년

: AI 추천 시스템 개발

- 박준이

: 정보컴퓨터공학부, 3학년

: 프로젝트 총괄 및 배포