

# 고급웹프로그래밍 과제 3 \_ 최원빈

## DAMARA 서비스 (명지대학교 공동구매 플랫폼)

Architecture Documentation

Author: 최원빈

### 1. 서비스 개요 (Overview)

명지대학교 학생을 위한 공동구매 기반 커뮤니티 플랫폼으로,  
상품 등록, 참여, 이미지 업로드, 회원 관리 등 다양한 기능을 제공한다.  
DAMARA는 다음과 같은 기술 스택과 아키텍처 패턴을 따른다:

#### 기술 스택

Layer	Technology
Backend	Node.js, Express, TypeScript, ts-node
DB Layer	Sequelize ORM, MySQL
View Layer	HTML, Handlebars
Frontend Script	Vanilla JavaScript(ES6), Fetch API
Build Tool	tsc(TypeScript Compiler)

#### 아키텍처 구조

Router → Controller → Service → Repository → Model → DB

모든 API는 상기 흐름에 따라 일관된 구조로 작동하며,  
이는 유지보수성과 확장성, 테스트 용이성을 극대화한다.

### # 2. 주요 기능 상세 설명 (Feature Specifications)

각 기능은 다음의 6단계 공통 패턴으로 처리된다.

- ① 사용자 행동
- ② 프론트엔드 스크립트(fetch) 요청
- ③ Router → Controller (입력 검증)
- ④ Service (비즈니스 로직)
- ⑤ Repository (DB 접근)
- ⑥ 응답 → UI 렌더링 / 토스트 표시

## # 2.1 회원가입 (Create User)

### ① 기능 설명

사용자가 이메일, 비밀번호, 닉네임 등을 입력해 계정을 생성한다.

회원 데이터는 `users` 테이블에 저장되며, **Zod 스키마 기반 입력 검증**이 수행된다.

### ② 전체 흐름 요약

구분	처리 내용
사용자 인터랙션	users.html에서 입력 후 "등록" 클릭
프론트 요청	fetch('/api/users', POST)
Router/Controller	<code>UserController.createUser()</code> 에서 <code>parseReq</code> 로 검증
Service	<code>UserService.registerUser()</code> 비즈니스 로직 수행
Repository	<code>UserRepo.create()</code> → INSERT
응답/UI	201 응답 후 리스트 갱신, 토스트 표시

### ③ 흐름도

```
[사용자 입력]
↓
fetch('/api/users', POST)
↓
UserController.createUser()
↓
```

```

UserService.registerUser()
↓
UserRepo.create() → INSERT users
↓
201 Created
↓
UI: "회원 등록 완료" 알림 + 목록 갱신

```

## # 2.2 로그인 (Authenticate User)

### ① 기능 설명

학생번호 + 비밀번호 기반 인증을 수행한다.

성공 시 사용자 정보가 반환되며, 프론트에서는 로그인 상태를 유지한다.

### ② 전체 처리 과정

단계	설명
사용자 행동	로그인 모달에서 입력 후 제출
프론트	fetch('/api/users/login')
컨트롤러	UserController.login()
서비스	bcrypt.compare로 비밀번호 검증
레포지토리	findByStudentId
반환	성공: 사용자 정보 / 실패: 401

### ③ 로그인 흐름도

```

[로그인 입력]
→ POST /api/users/login
→ UserService.login()
→ UserRepo.findByStudentId()
→ bcrypt.compare()
→ 성공 → 사용자 정보 반환

```

→ UI: 로그인 영역 활성화

## # 2.3 게시물 전체 조회 (Read Posts)

### 기능 설명

모든 게시물 + 대표 이미지 1장을 카드 형태로 보여준다.

### 처리 흐름

```
posts.html → fetch('/api/posts')
  → PostController.getAllPosts()
  → PostRepo.list() (JOIN: posts + post_images)
  → 200 OK
  → Handlebars 렌더링
```

## # 2.4 게시물 등록 (Create Post)

DAMARA의 핵심 기능 중 하나.

게시글 등록은 **이미지 업로드** → **게시글 생성**의 이단계 방식이다.

### 전체 흐름

단계	설명
Step 1	<code>/api/upload/images</code> 로 이미지 업로드
Step 2	업로드된 이미지 URL 포함해 <code>/api/posts</code> 로 게시물 생성

### 상세 흐름도

```
[사용자 입력: 제목/내용/이미지]
↓
uploadImages()
↓ POST /api/upload/images
```

```

multer 저장 → 이미지 URL 리스트 반환
↓
POST /api/posts
↓
PostService.createPost()
↓
PostRepo.create() → INSERT posts
PostImageRepo.save() → INSERT post_images
↓
UI 업데이트

```

## # 2.5 게시물 수정 (Update Post)

단계	설명
프론트	PUT /api/posts/:id
컨트롤러	updatePost()
서비스	유효성 검증 후 업데이트
레포	PostRepo.update
UI	수정된 내용 반영

수정 버튼 → inline edit form → PUT 요청 → 성공 → 토스트 + UI 갱신

## # 2.6 게시물 삭제 (Delete Post)

```

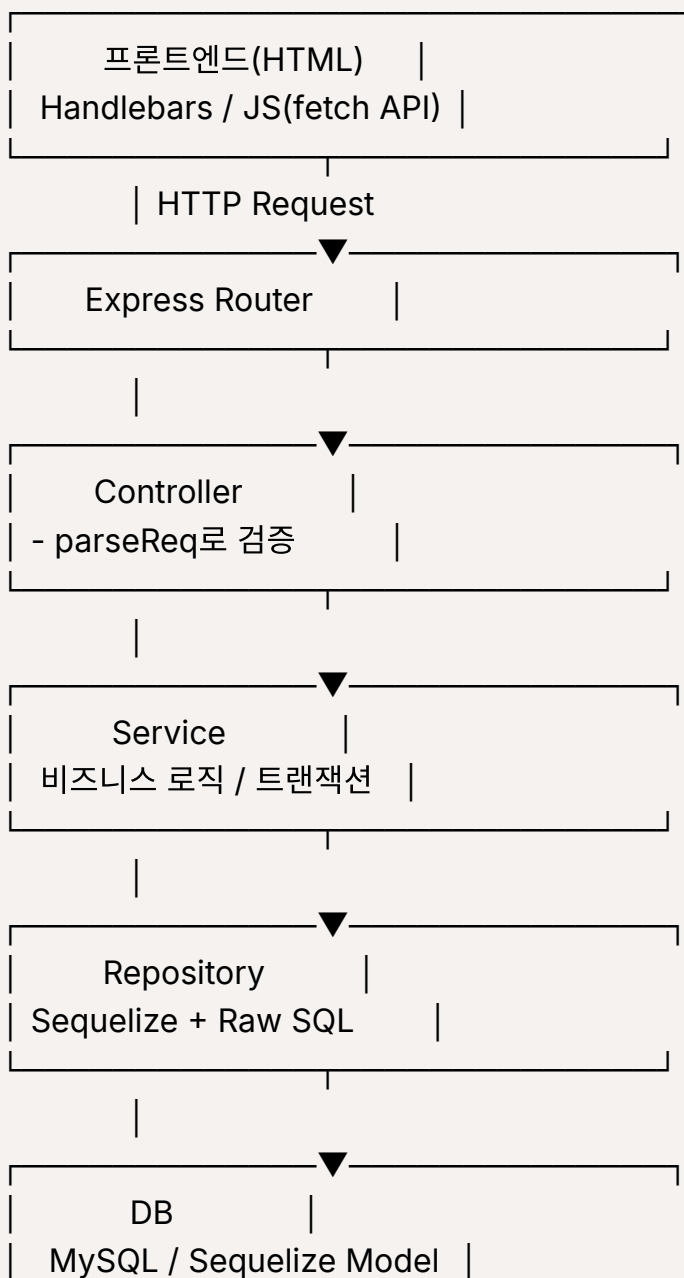
삭제 클릭
→ DELETE /api/posts/:id
→ PostController.deletePost
→ PostRepo.delete + PostImageRepo.deleteByPostId
→ 204 No Content
→ UI에서 카드 제거

```

## # 2.7 사용자 관리 (User CRUD)

기능	HTTP	설명
전체 조회	GET /api/users	users.html 테이블 렌더링
수정	PUT /api/users/:id	UserRepo.update
삭제	DELETE /api/users/:id	UserRepo.delete

## # 3. 전체 시스템 아키텍처



## # 4. DB 도메인 모델 (Entity & Relationship)

### 4.1 users 테이블

#### 스키마 요약

컬럼	타입	설명
id	UUID PK	자동 생성
email	varchar	unique
password_hash	varchar	bcrypt 해싱
nickname	varchar	
department	varchar	nullable
student_id	varchar	unique
avatar_url	varchar	nullable
createdAt / updatedAt	timestamp	Sequelize 관리

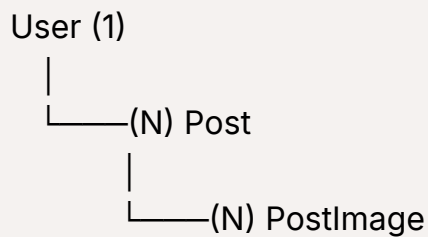
### 4.2 posts 테이블

컬럼	타입	설명
id	UUID	PK
author_id	UUID FK	users.id 참조
title	varchar	제목
content	text	내용
price	decimal	가격
min_participants	int	최소 인원
current_quantity	int	현재 인원
status	ENUM	open/closed/cancelled
deadline	DATE	모집 마감일
pickup_location	varchar	nullable

## 4.3 post\_images 테이블

컬럼	타입	설명
id	UUID	PK
post_id	UUID	posts.id FK
image_url	varchar	이미지 URL
sort_order	smallint	순서

## Relation Cardinality



## # 5. 기능별 API 표준 명세 (RESTful Specification)

기능	Method	Endpoint	설명
회원조회	GET	/api/users	모든 사용자
회원등록	POST	/api/users	신규 회원 생성
회원수정	PUT	/api/users/:id	사용자 수정
회원삭제	DELETE	/api/users/:id	사용자 삭제
게시글조회	GET	/api/posts	전체 게시글
게시글조회	GET	/api/posts/:id	특정 게시글
게시글등록	POST	/api/posts	게시글 생성
게시글수정	PUT	/api/posts/:id	수정
게시글삭제	DELETE	/api/posts/:id	삭제
이미지업로드	POST	/api/upload/images	다중 업로드

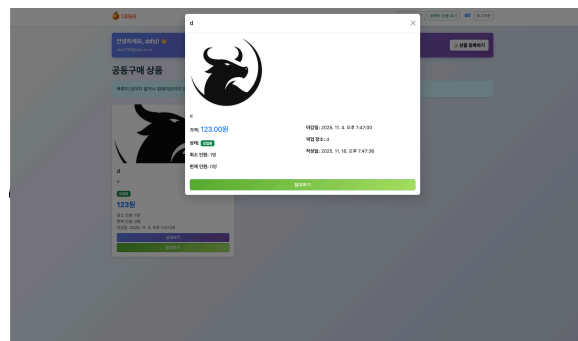
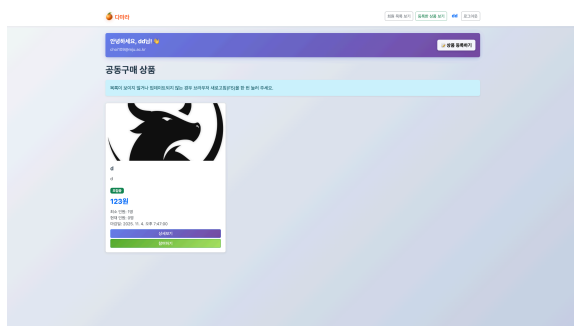


## # 6. 전체 흐름 요약 (Final Summary Flow)

HTML 화면 로딩  
↓  
JS(fetch API) 요청  
↓  
Express Controller  
↓  
Service  
↓  
Repository (Sequelize)  
↓  
DB 처리  
↓  
결과 반환  
↓  
Handlebars 템플릿 렌더링  
↓  
UI 업데이트 + Toast 알림

## #7. UI

### 메인페이지



# 상품 관리 페이지

내가 올린 상품 관리

로그인한 사용자 상품 등록 페이지로 이동. 이 페이지에서 상품 등록을 할 수 있습니다. 상품 등록을 완료하면 해당 상품이 등록됩니다.

작성자 확인으로 조회

작성자 확인

로그인

로그인한 사용자 상품 등록 페이지로 이동. 이 페이지에서 상품 등록을 할 수 있습니다.

내가 올린 상품 관리

로그인

내가 올린 상품 관리

로그인한 사용자 상품 등록 페이지로 이동. 이 페이지에서 상품 등록을 할 수 있습니다. 상품 등록을 완료하면 해당 상품이 등록됩니다.

작성자 확인으로 조회

작성자 확인

로그인

로그인한 사용자 상품 등록 페이지로 이동. 이 페이지에서 상품 등록을 할 수 있습니다.

내가 올린 상품

작성자 확인으로 조회

로그인

로그인한 사용자 상품 등록 페이지로 이동. 이 페이지에서 상품 등록을 할 수 있습니다.

로그인

# 회원 관리 페이지

## 회원 관리

### 회원 등록

이메일

example@mju.ac.kr

비밀번호 (8자 이상)

비밀번호 입력

닉네임

닉네임 입력

학과 (선택사항)

예: 컴퓨터공학

학번 (선택사항)

예: 20241234

프로필 이미지 URL (선택사항)

https://example.com/avatar.jpg

등록

### 회원 목록

새로고침

dd	
이메일: cho109@mju.ac.kr	ID: d16a0d22-a735-4cb0-8d73-4cc1ae585da2
학과: dd	가입일: 2025. 11. 18. 오후 7:47:18
학번: 12	수정일: 2025. 11. 18. 오후 7:47:18
수정 삭제	