CampingLog

유우진

염윤호

이재원

이창훈

황서이

2025.4.30.수 발표자: 유우진



목차

- 1.프로젝트 개요
- 2.기획 및 설계
- 3.개발환경
- 4.구현 및 결과
- 5.회고

1. 프로젝트 개요

1.1 서비스 소개

"캠핑로그(CampingLog)"는 캠핑을 좋아하는 사람들을 위한 커뮤니티 중심의 웹 서비스입니다.

> 캠핑 관련 리뷰 작성 좋아요/댓글/북마크 활동 내역과 프로필을 관리

1.2 주제 선정 이유

| 웹사이트명 | 장점 | 단점 | 특징 | 총평 | |
|--------------|--------------------------|---|------------------------------|------------------------------------|--|
| 애플뮤직 | 심플한 디자인 앨범아트가 한눈에 들어옴 | 기능이 별로 없음 | 사용자(개개인)에 초점 | 가시성이 떨어짐 | |
| 스포티파이 | 직관적이고 화려한 UI | UI 가 좌에서 우 기준 -> 가시성이 떨어짐 | UI 가 가장 맘에 듦 | 벤치마킹한 사이트중 UI 는 가장 마음에 듦 | |
| 멜론 | 직관적 | 올드한 UI | 매거진(커뮤니티)가 활성화되어있음 | 매거진(게시판, 좋아요, 댓글) 기능이 마음에 듦 | |
| 스페이스 클라우드 | 키워드 기반 게시판 구성 | UI가 가독성이 떨어짐 | 게시판이 잘되어있음 챗봇(채널톡)이 있음 | 가독성이 떨어짐 | |
| 에어비앤비 | 후기에 필터링 기능 | 구현할 기능이 너무 많음 | 호스트정보, 사용자정보가 잘 분리되어있음 | 잘 만든 웹페이지 | |

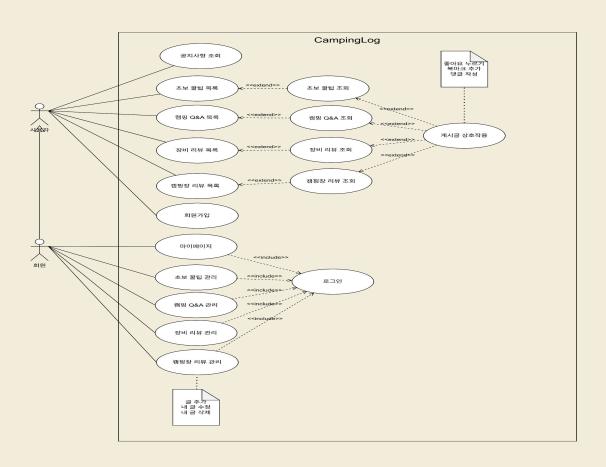
1.3 페르소나

| 이름 | 김캠핑 | 캠핑경력 | 5년 |
|-----|---------|------|--|
| 나이 | 36세 | 성향 | 남에게 정보를 설명하고 공유하는 것 잘함. 캠핑 초보자에게 도움을 주는 것을 보람으로 느낌 |
| 직업 | 교사 | 문제점 | 캠핑정보 공유 사이트가 없음 |
| 거주지 | 서울시 강남구 | 니즈 | 자신의 정보를 체계적으로 남길 수 있는 공간이 필요함 |

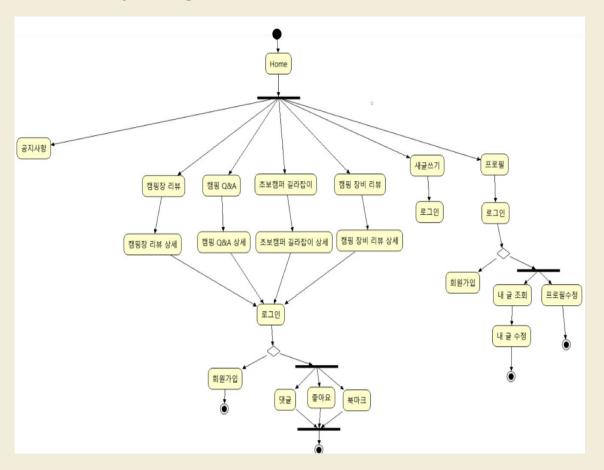
| 이름 | 나초보 | 캠핑경력 | 없음 |
|-----|---------|------|--|
| 나이 | 28세 | 성향 | 자연을 좋아하고 주말에 휴식이 필요함 남들에게 추천받는 것을 선호함, 캠핑에 남다른 로망을 가지고 있음, 캠핑경력은 없지만 SNS를 통한 간접 경험이 풍부함 |
| 직업 | 사무직회사원 | 문제점 | 캠핑정보 활성화된 커뮤니티가 없음 기존 블로그 및 카페는 광고성 게시글이 많아 신뢰하기 어려움 |
| 거주지 | 서울시 구로구 | 니즈 | 추천기반으로쉽게 고를 수 있는 캠핑장 정보 자유롭게 질문하고 답변할 수 있는 커뮤니티, 초보자를 위한 필터링된 후기 |

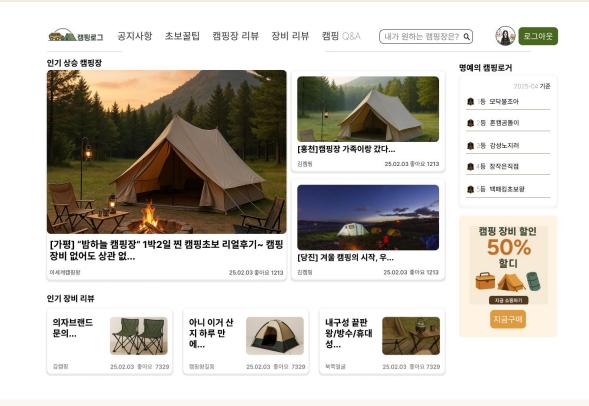
2. 기획 및 설계

2.1 Use Case & Activity Diagram

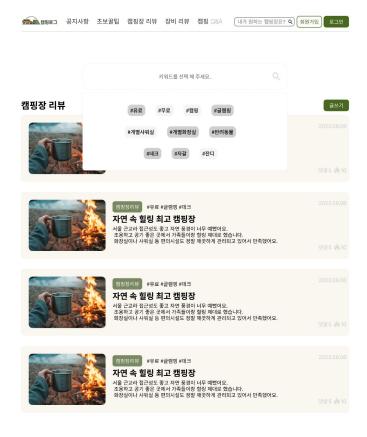


2.1 Use Case & Activity Diagram





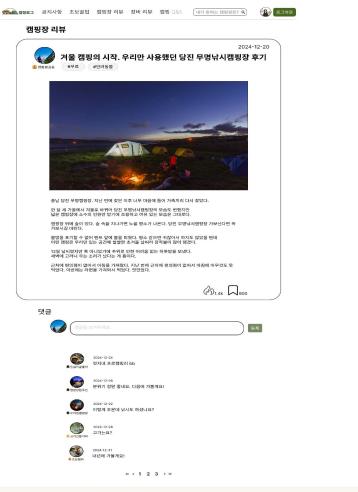




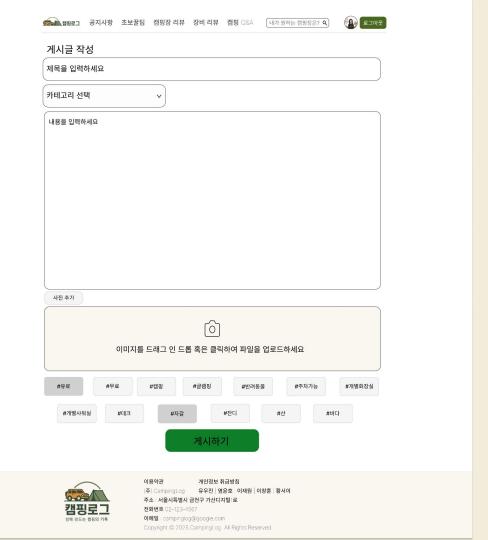
« < 1 2 3 > »

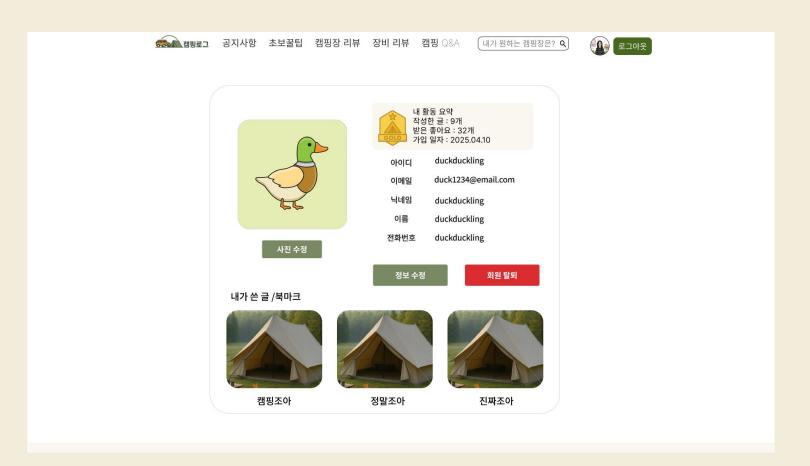


이용약관 기반(정보 취급방점 (주) (ampingtion) 유무진 (영호호 이재원 이상훈 | 황서이 주소 서울시독병시 급천구 가산디지방 로 전화반호 02 123 4567 이메일 (ampingtog@google.com

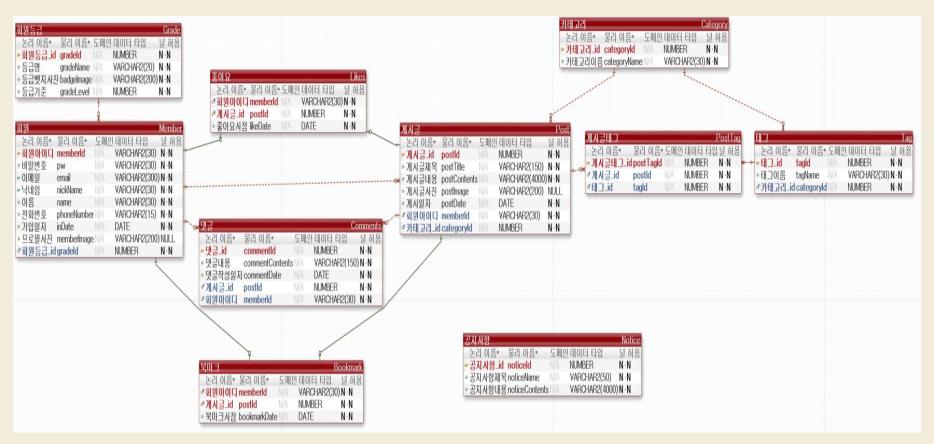


캠핑로그





2.3 DB 모델링

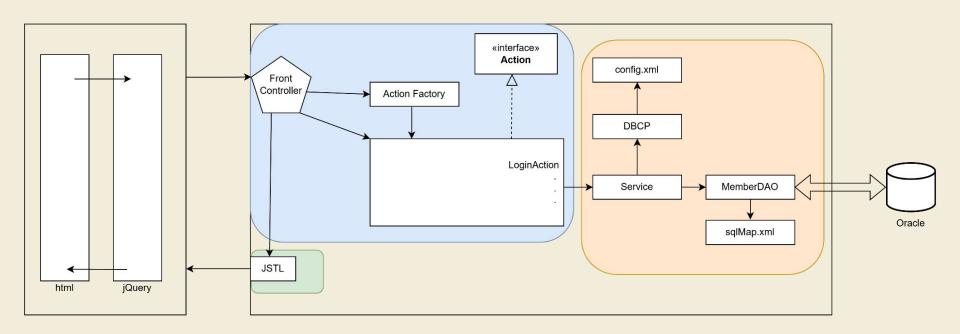


2.4 업무 상세 설계

| | | | | 업무상서 | 설계서 | | | | | |
|------|-----------------------|----|-----------------------|--|---|---|--|---|-----|-------------------------------|
| | | · | | Control | ller 기준 | DAO / ¹ | VO 기준 | | | |
| 업무번호 | 업무 | 구분 | Action | input | output | input | output | View | | |
| 1 | 회원가입 | 처리 | SignUpAction | id, pw, email, nickName, name, phoneNumber | setAttribute : loginUser, result, nickName | MemberVO | boolean | signUpResult.jsp, signUp.html | 유우진 | 업무 8개, Action 8개, View 12개 |
| 2 | 회원가입 시 중복값 확인 | 처리 | CheckDuplicateAction | field, value | setAttribute : error, isDuplicate | value(String memberId email nickName) | boolean | checkDuplicateResult.jsp, signUp.html | 이창훈 | 업무9개, Action9개, View15개 |
| 3 | 비밀번호 입력 일치 확인 | 처리 | PasswordCheckAction | pw | setAttribute : verified | MemberVO | boolean | passwordCheckResult.jsp, signUp.html | 황서이 | 업무5개,Action7개, View 7개 |
| 4 | 로그인 | 처리 | LoginAction | id, pw | setAttribute : loginSuccess, loginUser, nickName | MemberVO | boolean | login.jsp, login.html | 이재원 | 업무 6개, Action8개, View 4개 |
| 5 | 로그인 체크 | 처리 | LoginCheckAcion | - | setAttribute : loggedIn, profileImage | | - | | 염윤호 | 업무 5개, Action6개, View6개 |
| 6 | 로그아웃 | 처리 | LogoutAction | - | | - | - | mainUI.html | | |
| 7 | 회원탈퇴 | 처리 | DeleteMemberAction | loginUser, pw | setAttribute : result | String memberId, String pw | boolean | deleteMember.jsp, mainUI.html | | |
| 8 | 마이페이지 - 내 정보 조회 | 처리 | GetMyInfoAction | loginUser | setAttribute : error, loginUser, memberInfo | String memberId | MemberVO | getMyInfo.jsp, myPage.html | | |
| 9 | 마이페이지 - 내가 쓴 글 리스트 조회 | 처리 | MyPagePostsAction | loginUser | setAttribute : error, posts | String memberId | List <hashmap<string, object="">></hashmap<string,> | myPagePosts.jsp, myPage.html | | |
| 10 | 마이페이지 - 북마크한 글 리스트 조회 | 처리 | MyPageBookmarksAction | loginUser | setAttribute : error, bookmarks | String memberId | List <hashmap<string, object="">></hashmap<string,> | myPageBookmarks.jsp, myPage.html | | |
| 11 | 프로필 사진 조회 | 처리 | GetMyInfoAction | loginUser | setAttribute : loginUser, memberInfo | String memberId | boolean | getMyInfo.jsp, myPage.html | | |
| 12 | 프로필 사진 등록 및 수정 | 처리 | AddProfilePhotoAction | loginUser | setAttribute : result, fileName | String memberId, String fileName | MemberVO | addProfileImg.jsp, myPage.html | | |
| 13 | 마이페이지 수정 전 비밀번호 확인 | 처리 | PasswordCheckAction | loginUser | setAttribute : verified | MemberVO | boolean | passwordOkResult.jsp, myPage.html | | |
| 14 | 마이페이지 정보 수정 | 처리 | UpdateMemberAction | MemberVO | setAttribute:Json (boolean) | memberld | MemberVO | updateMember.jsp, myPageEdit.html | | |
| 15 | 비밀번호 수정 | 처리 | UpdatePasswordAction | loginUser | setAttribute : result | String memberld, String newPw | boolean | updatePasswordResult.jsp , passwordEdit.html | | |
| 16 | 인기상승 캠핑 조회 | 화면 | MainPopularListAction | categoryld | Json | categoryld | Collection <postvo></postvo> | mainUI.html, mainPopularList.jsp | | |

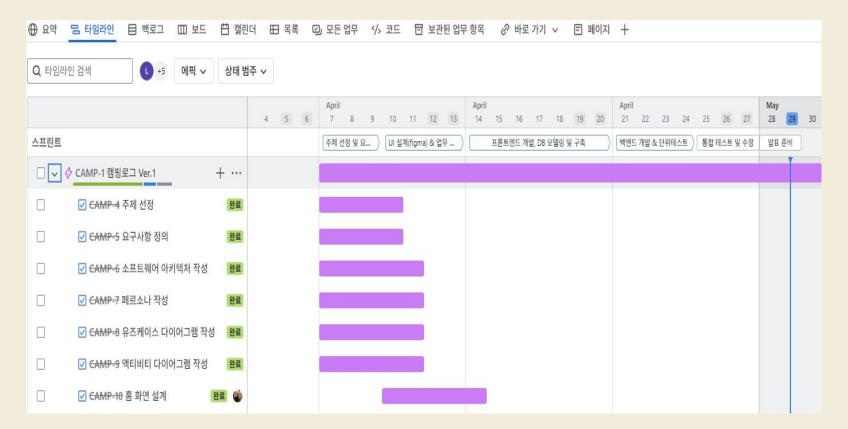
3. 개발 환경

3.1 S/W 아키텍처



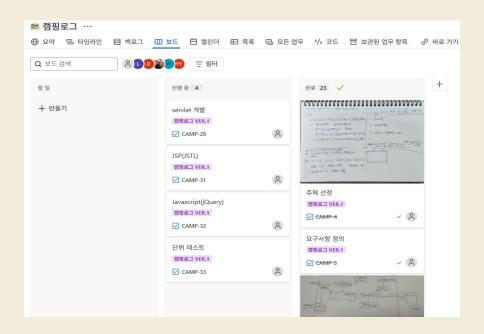
3.2 업무 관리

Jira & Confluence



3.2 업무 관리

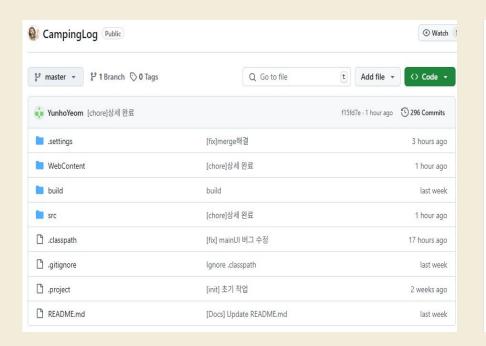
Jira & Confluence





2.4 업무 상세 설계

- Git, Github



커밋 컨벤션

| 태그 이름 | 설명 | | | | | |
|--|--------------------------------------|--|--|--|--|--|
| [Feat] | 기능 (새로운 기능) | | | | | |
| [Fix] | 버그 (버그 수정) | | | | | |
| [Design] | CSS 등 사용자 UI 디자인 변경 | | | | | |
| [Refact] | 코드 리팩토링 | | | | | |
| [Comment] | 필요한 주석 추가 및 변경 | | | | | |
| [style] | 스타일 (코드 형식, 세미콜론 추가: 비즈니스 로직에 변경 없음) | | | | | |
| [Docs] | cs] 문서 수정 (문서 추가, 수정, 삭제, README) | | | | | |
| [test] 테스트 (테스트 코드 추가, 수정, 삭제: 비즈니스 로직에 변경 | | | | | | |
| [Chore] 기타 변경사항 (빌드 스크립트 수정, assets, 패키지 매니저 | | | | | | |
| [Init] 브랜치 초기화 및 초기셋팅 관련된 설정일 경우 | | | | | | |
| [rename] 파일 혹은 폴더명을 수정하거나 옮기는 작업만 한 경우 | | | | | | |

4. 구현 및 결과

- DAO, VO(Post)

G com.camp.model.PostVO postld: int postTitle: String postContents: String postimage: String postDate: String memberld: String a categoryld: int nickName: String memberlmage: String badgelmage: String likeCount: int bookmarkCount: int a categoryName: String p commentCount: int tagList: String

G com.camp.model.PostDAO getPostPage(start: int, end: int, searchTerm: String): List<PostVO> getPostPageByCategory(categoryId: int, tagList: String, start: int, end: int): List<PostVO> getTotalPostCountByCategory(categoryld: int, tagList: String): int getTotalPostCount(searchTerm: String): int getCommentCount(session: SqlSession, postld: int): int getCategoryName(): List<PostVO> getDetailContents(session: SqlSession, postid: int): PostVO getDetailTag(session: SglSession, postId: int): List<String> getMemberImage(session: SqlSession, memberId: String): String insertPost(session: SqlSession, vo: PostVO): int getLastPostId(session: SqlSession): int insertPostTag(session: SqlSession, postId: int, tagId: int): int getPostById(session: SqlSession, postId: int): PostVO getSelectedTaglds(session: SglSession, postId: int): List<Integer> getTagldsByPostId(session: SglSession, postId: int): List<Integer> updatePost(session: SqlSession, vo: PostVO): int deleteTagsByPostId(session: SqlSession, postId: int): int deletePost(session: SqlSession, postId: int): int insertLike(postid: int, memberid: String): boolean InsertBookmark(session: SqlSession, postld: int, memberld: String): boolean isAlreadyBookmarked(session: SqlSession, memberld: String, postId: int); boolean isAlreadyLiked(conn: SalSession, memberld: String, postId: int); boolean

- DAO, VO(Member)

addMember(vo: MemberVO): boolean isDuplicateId(value: String): boolean isDuplicateEmail(email: String): boolean isDuplicateEmailExcludingSelf(email: String, memberld: String): boolean G com.camp.model.MemberVO isDuplicateNickName(nickName: String): boolean memberid: String isDuplicateNickNameExcludingSelf(nickName: String, memberId: String): boolean pw: String getMembers(): List<MemberVO> memail: String pwCheck(vo: MemberVO): boolean nickName: String updateProfile(vo: MemberVO): boolean name: String updatePw(memberld: String, pw: String, newPw: String): boolean phoneNumber: String getMemberlmage(memberld: String): String inDate: String setMemberImage(memberId: String, memberImage: String): boolean memberlmage: String updateMemberImage(memberId: String, newImageFileName: String, uploadDirPath: String): boolean gradeld: int deleteAndSetDefaultImage(vo: MemberVO, uploadDirPath: String): boolean badgelmage: String getMyInfo(memberId: String): MemberVO postCount: int getMyPosts(memberld: String): List<HashMap<String,Object>> likeCount: int getMyBookmarks(memberId: String): List<HashMap<String,Object>> getYourInfo(memberId: String): MemberVO updateProfileImage(memberid: String, fileName: String): boolean updateMemberInfo(vo: MemberVO): boolean deleteMember(memberld: String, pw: String): boolean getPasswordByld(memberld: String): String updatePassword(memberld: String, newPw: String): boolean

G com.camp.model.MemberDAO

Service

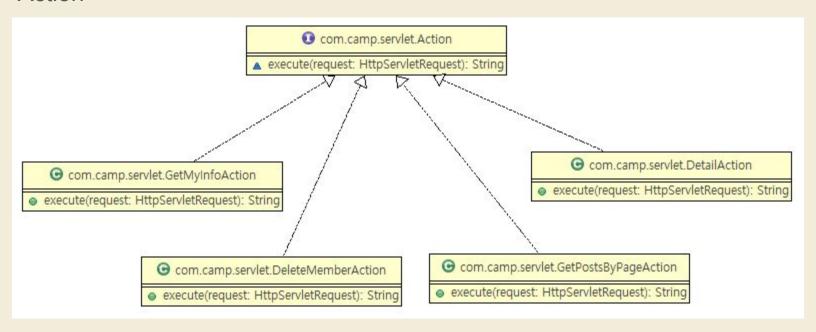
G com.camp.service.PostService

- dao: PostDAO
- registerPostWithTags(vo: PostVO, taglds: String[]): boolean
- getPostDetail(postId: int): Map<String,Object>
- updatePostWithTags(vo: PostVO, taglds: String[]): boolean
- deletePostWithTags(postId: int): boolean

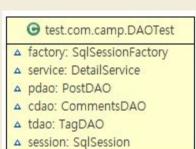
G com.camp.service.DetailService

- factory: SqlSessionFactory
- postDAO: PostDAO
- commentsDAO: CommentsDAO
- tagDAO: TagDAO
- getPostDetails(postId: int): PostVO
- getTagsForPost(postId: int): List<String>
- getCommentsForPost(postId: int): List<CommentsVO>
- getCommentsByPagination(postId: int, startRow: int, endRow: int): List<CommentsVO>
- getCommentCount(postId: int): int
- addComment(comment: CommentsVO): boolean
- getMemberlmage(memberld: String): String
- InsertBookmark(postId: int, memberId: String): boolean

- Action



4.2 단위테스트(Junit4)



단위_테스트_시작(): void
 TagDAO_테스트(): void
 PostDAO_테스트(): void

● CommentsDAO_테스트(): void

● test.com.camp.ServiceTest

△ ds: DetailService

● 단위_테스트_시작(): void

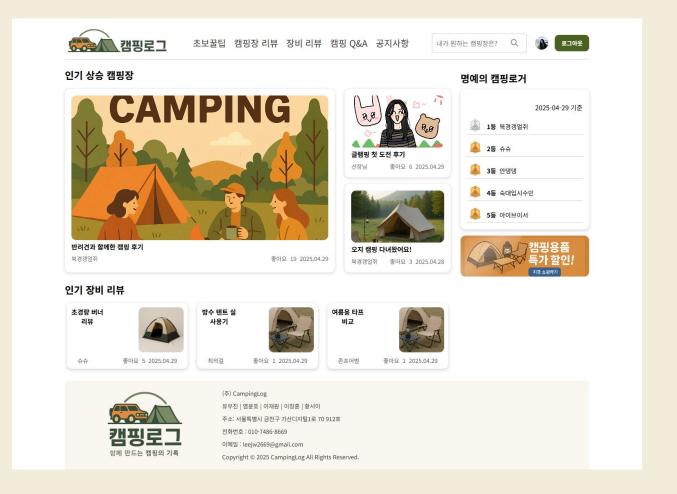
● 서비스_게시글_조회_성공(): void

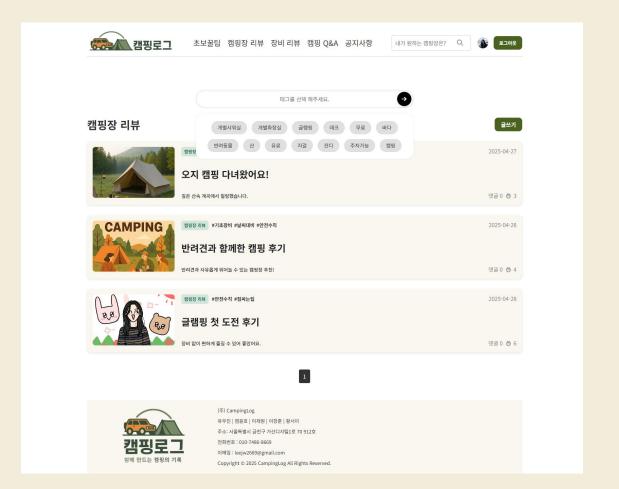
● 서비스_게시글_조회_실패(): void

● 서비스_게시글_댓글_조회_성공(): void

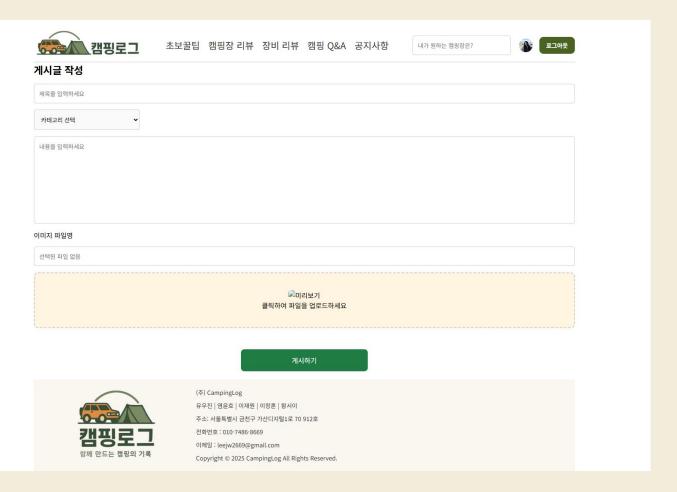
● 서비스_게시글_댓글_조회_실패(): void

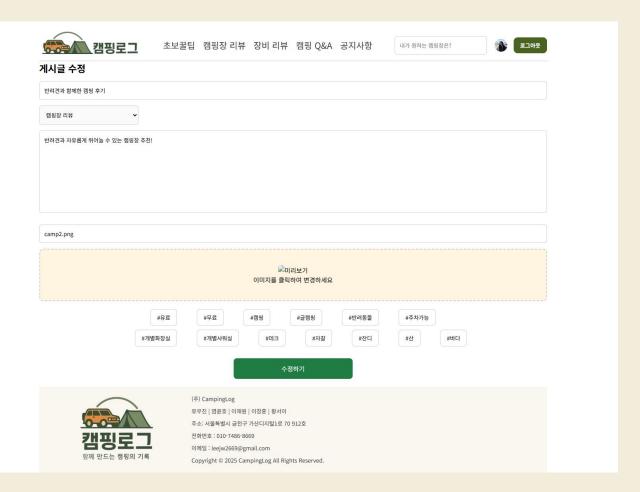
test.com.camp.ActionTest
 DetailTest(): void
 testGetMyInfo(): void
 testGetTotalPostCount_withEmptySearchTerm(): void

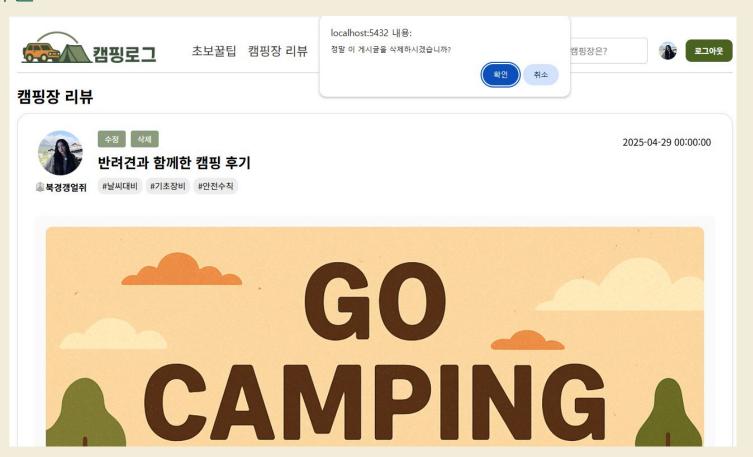


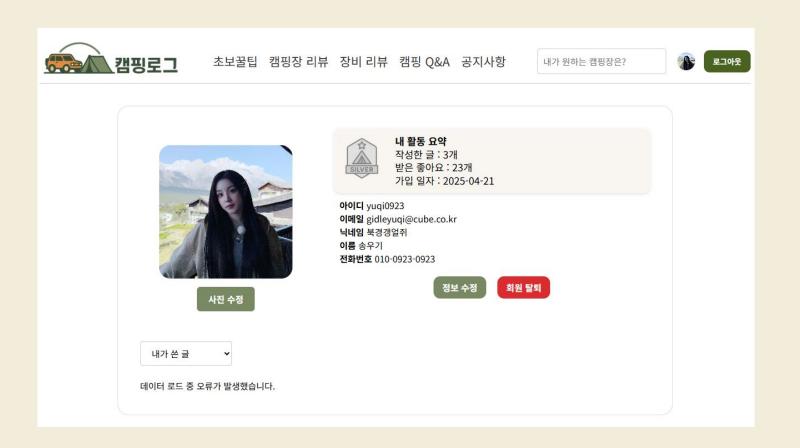














5. 회고

5.1 어려웠던 점과 해결방법



5.2 향후 개선 방향 또는 추가 기능

