여행의 경험,
소셜미디어에 담다

개발:김도영,이정우,이태형

도메인:http://tripwaves.site

Git hub : https://github.com/doyoungking/TripSNS_project

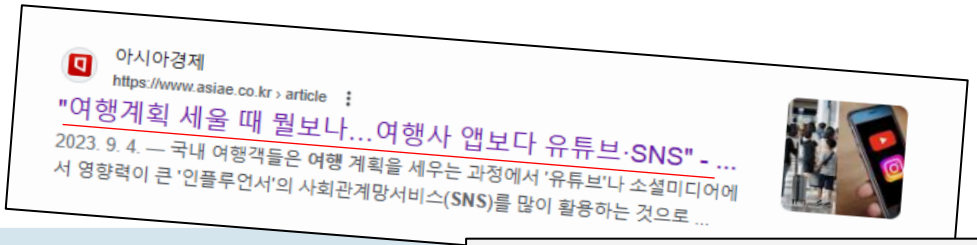
개발기간:2024.06.03~2024.07.18

김도영 /010-7688-8774

목차

1.주제 소개

"여행 정보 탐색시 신뢰성과 영향력을
급속히 상실 하고 있는 포털 사이트"



최근 여행 정보를 찾는 방식이 크게 변화하고 있습니다. 과거에는 기업과 공공기관이 제공하는 정보를 주로 활용했지만, 이제는 SNS와 유튜브 등 개인의 경험을 중심으로 한 매체의 영향력이 커지고 있습니다.

이에 맞춰 여행에 특화된 기능을 갖춘 SNS를 개발하면 소비자들이 국내 여행 정보를 더 쉽게 찾고, 개인의 경험을 중심으로 한 신뢰성 있는 정보를 얻으며,

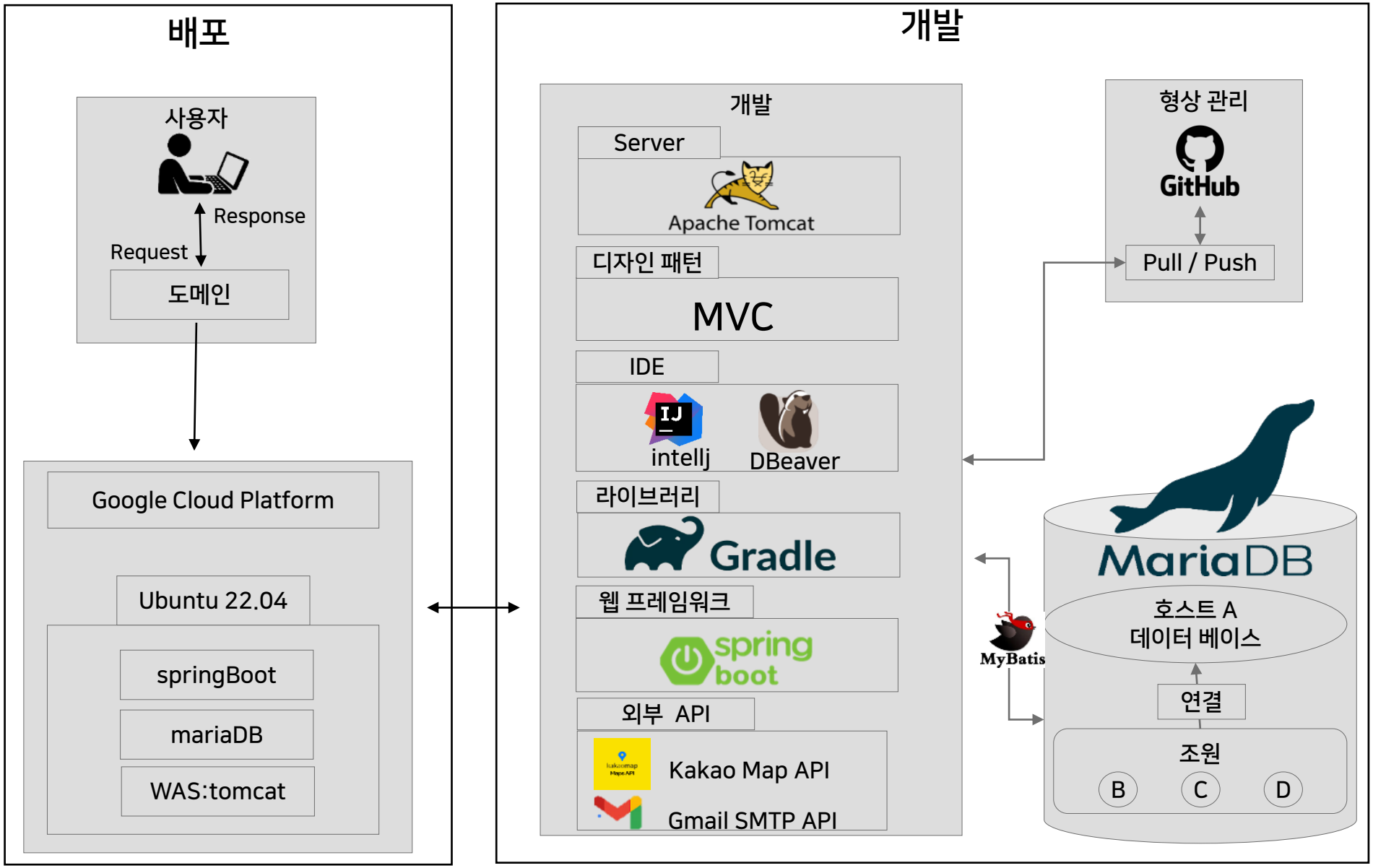
미처 알지 못했던 한국의 지역 명소를 접하는 데 용이해질 것입니다.

이를 통해 지역경제 활성화에도 크게 이바지할 수 있을 것으로 기대됩니다.

2.개발환경

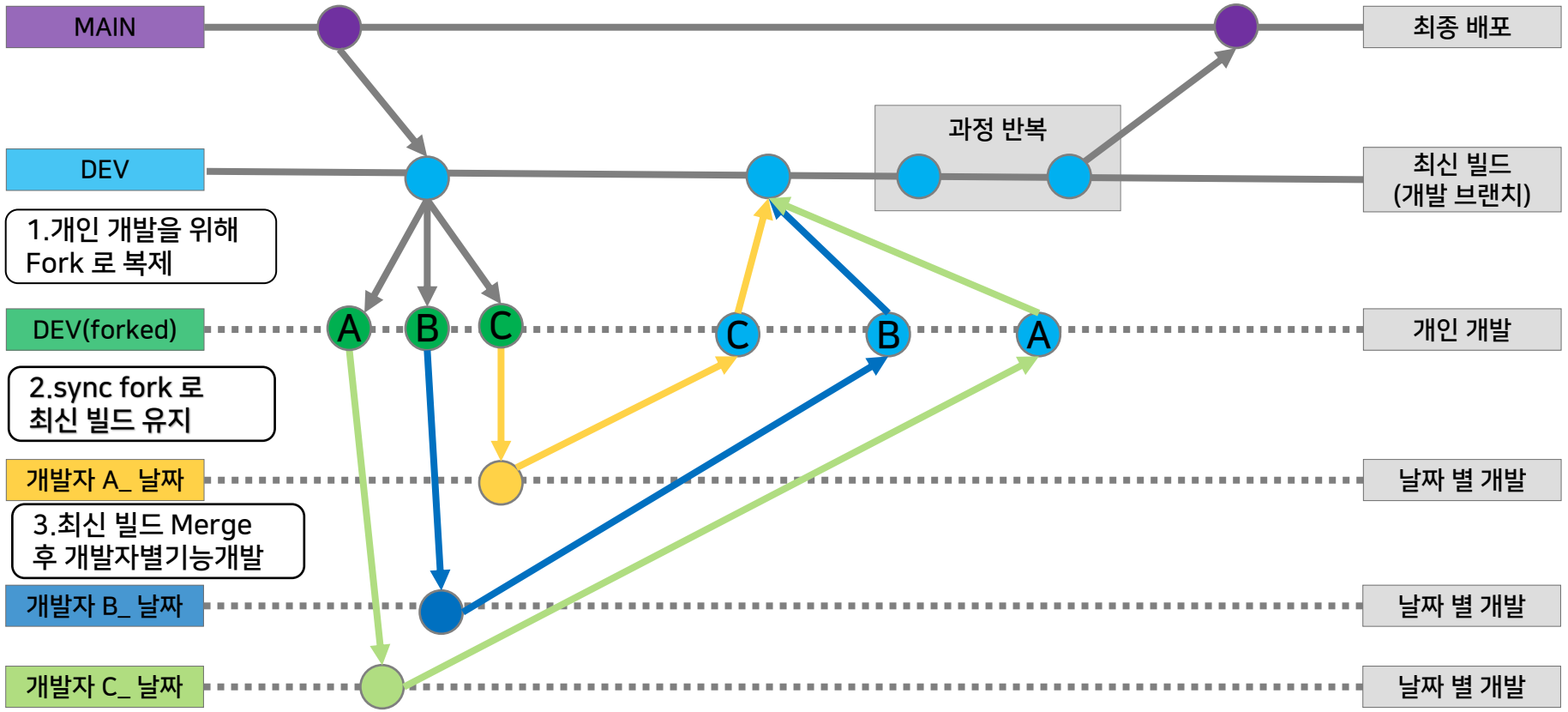
구분	언어/라이브러리	범위
FE	JavaScript ECMAScript 6+	동적 화면
	HTML5	화면 디자인
	CSS3	HTML에 스타일 적용
	Thymeleaf	받은 파라미터 값 문서에 활용 (뷰 템플릿)
	Jquery 3.4.1	AJAX 기능 구현 / DOM 제어
SERVER	JAR(Apache Tomcat)	애플리케이션 서버, 동적 콘텐츠 생성
BE	Spring Boot 3.9.11	로직 처리 및 애플리케이션 개발
	MyBatis	DB 로직 개발 및 SQL 쿼리 매핑
	JAVA 17	DAO, DTO, 인터페이스 제공
DB	MariaDB 11.4.2	데이터 저장 및 관리
TOOLS	IntelliJ	JAVA기반 통합 개발 환경
	DBeaver 24.0.5	DB 관리
	Visual Studio Code	FE개발
Build	Gradle	의존성 관리
API	Kakao Map API	카카오 지도 사용
	Gmail SMTP API	이메일 인증 기능
형상관리	GitHub	버전 관리 및 협업
배포	Linux Ubuntu 22.04	가상 컴퓨터
	Google Cloud Platform	클라우드 서버 배포(E2서버 활용)

3.개발구조


배포
사용자
Response
Request
도메인
Google Cloud Platform
Ubuntu 22.04
springBoot
mariaDB
WAS:tomcat
개발
개발
Server
Apache Tomcat
디자인 패턴
MVC
IDE
intellj
DBeaver
라이브러리
Gradle
웹 프레임워크
spring boot
외부 API
Kakao Map API
Gmail SMTP API
형상 관리
GitHub
Pull / Push
MyBatis
MariaDB
호스트 A
데이터 베이스
연결
조원
B
C
D

4.형상관리

효율적인 소스 관리를 위해 협업 도구로 Git hub를 사용 후
Main, dev, 개인작업브랜치 를 사용하는 브랜치 전략 활용



5. 업무 분배

- 팀원의 능력을 기준으로 업무 분배

기능 구분	팀원 기술 스택	업무 분배
1.회원 기능 2.게시글 기능 3.숏폼 동영상 기능 4.지도 api 기능	이정우,이태형,김도영 기술 스택 같음 Oracle HTML CSS JavaScript Java Spring MVC	기술 스택은 같았지만 팀 회의를 통해 서로의 개발 속도 및 각자 원하는 파트기준으로 업무 분배를 진행하였다. 1.회원 - 김도영 2.게시글,지도api –이태형 3.숏폼 - 이정우

6. 기능 명세서

-데이터베이스,게시글,쇼츠

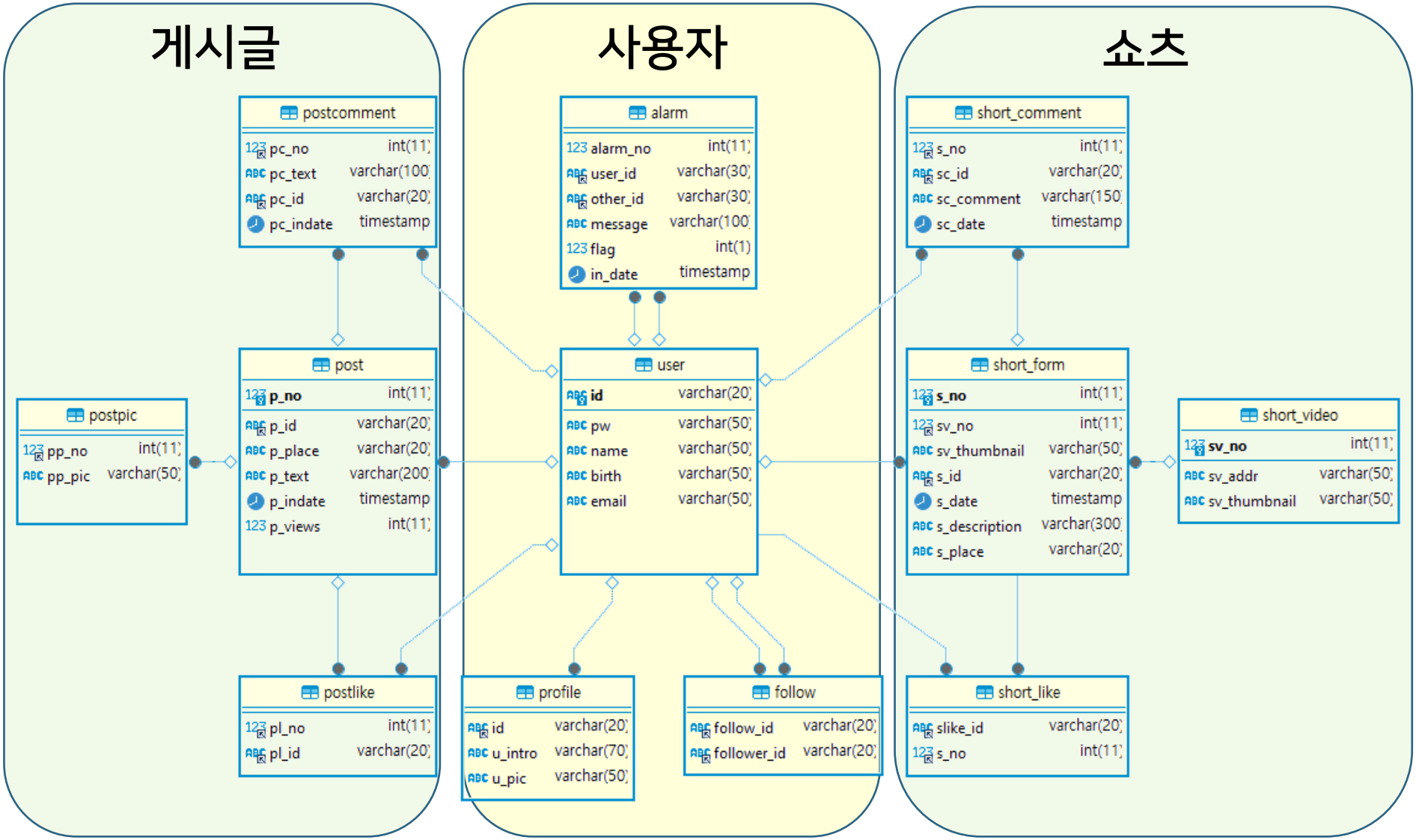
구분	요구사항 명	요구사항 상세 설명	구현 여부	담당자
데이터베이스	게시글 데이터 관리	글 작성시 시퀀스를 이용한 고유번호 자동 생성	Y	이태형
		좋아요를 누른 이용자와 게시글의 아이디, 고유번호 저장	Y	
		댓글을 작성한 이용자와 게시글의 아이디, 고유번호 저장	Y	
	숏폼 데이터 관리	시퀀스로 고유번호 생성,비디오 파일과 썸네일 파일의 이름을 저장	Y	이정우
		시퀀스로 고유번호 생성,전체보기를 위해 썸네일 파일 이름 저장	Y	
		댓글 입력시 글번호,댓글 작성자 id도 저장하게 함	Y	
		좋아요 한 아이디와 글 번호 저장	Y	
	회원 데이터관리	회원가입시 사용자의 아이디,비밀번호,이름,생일,이메일 저장	Y	김도영
		사용자의 아이디,한줄소개,프로필사진 파일 이름저장	Y	
		팔로우아이디,팔로워 아이디 저장	Y	
		시퀀스로 고유번호 생성,사용자아이디,상대방아이디,메세지내용,알람확인여부,날짜저장	Y	
프로그램	게시글 작성	게시글 작성 버튼 누르면 작성 폼으로 이동 후 작성 저장 가능	Y	이태형
	게시글 전체 보기	게시글의 사진, 내용, 작성자, 조회수, 지역이 나옴	Y	
	게시글 자세히 보기	자세히 보기 클릭시 자세히 보기 뷰로 이동	Y	
		수정 버튼 클릭시 지역, 글 내용 수정 가능	Y	
		삭제 버튼 클릭시 댓글, 게시글 전부 삭제	Y	
		좋아요 버튼 클릭시 버튼 모양 변경	Y	
		댓글 작성, 삭제 가능	Y	
	지도 API	게시글의 지역명 클릭시 해당 지역을 검색값으로 지도 페이지 이동	Y	
		카카오맵에서 제공하는 카테고리 기능으로 숙박, 식당, 관광지등의 정보를 표시 가능	Y	
	관리자	모든 게시글의 정보 확인가능	Y	
		직접 삭제하거나, 체크박스를 통해 일괄삭제 가능	Y	
		엑셀형식으로 다운로드 기능	Y	
	숏폼 동영상 저장	동영상 파일을 업로드하면 자동으로 썸네일 파일을 생성하여 사용자가 저장할 수 있게 함	Y	이정우
	숏폼 게시글 저장	동영상 파일 이름을 저장한 고유 번호를 가져와 게시물 내용과 함께 저장	Y	
	숏폼 전체보기	동영상 썸네일을 가져와 앨범식 구성으로 화면에 보여지도록 한다.	Y	
	숏폼 자세히 보기	자세히 보기에서는 등록한 동영상이 자동으로 실행되고 글내용,댓글이 보여지도록한다.	Y	
		수정버튼과 삭제버튼으로 게시글 update,delete 작업을 한다. 글 작성자만 가능	Y	
		게시물에 댓글을 달 수 있으며, 댓글 작성자만 삭제할 수 있게 한다.	Y	
		게시물에 좋아요를 누르면 좋아요 버튼의 색이 변경	Y	
	검색	입력된 검색어로 게시글,쇼츠,회원 검색결과를 각각 tab으로 출력	Y	
	관리자 페이지	모든 쇼츠 게시물들의 목록이 보여진다,검색으로 특정 게시물들만 볼 수 있음	Y	
		직접 버튼을 눌러 삭제하거나 체크박스를 통해 일괄 삭제 가능	Y	
		쇼츠 게시물들의 목록을 Excel 파일로 다운로드 구현	Y	
	프로필 화면	회원이 작성한 쇼츠 목록 확인 가능	Y	

6. 기능 명세서

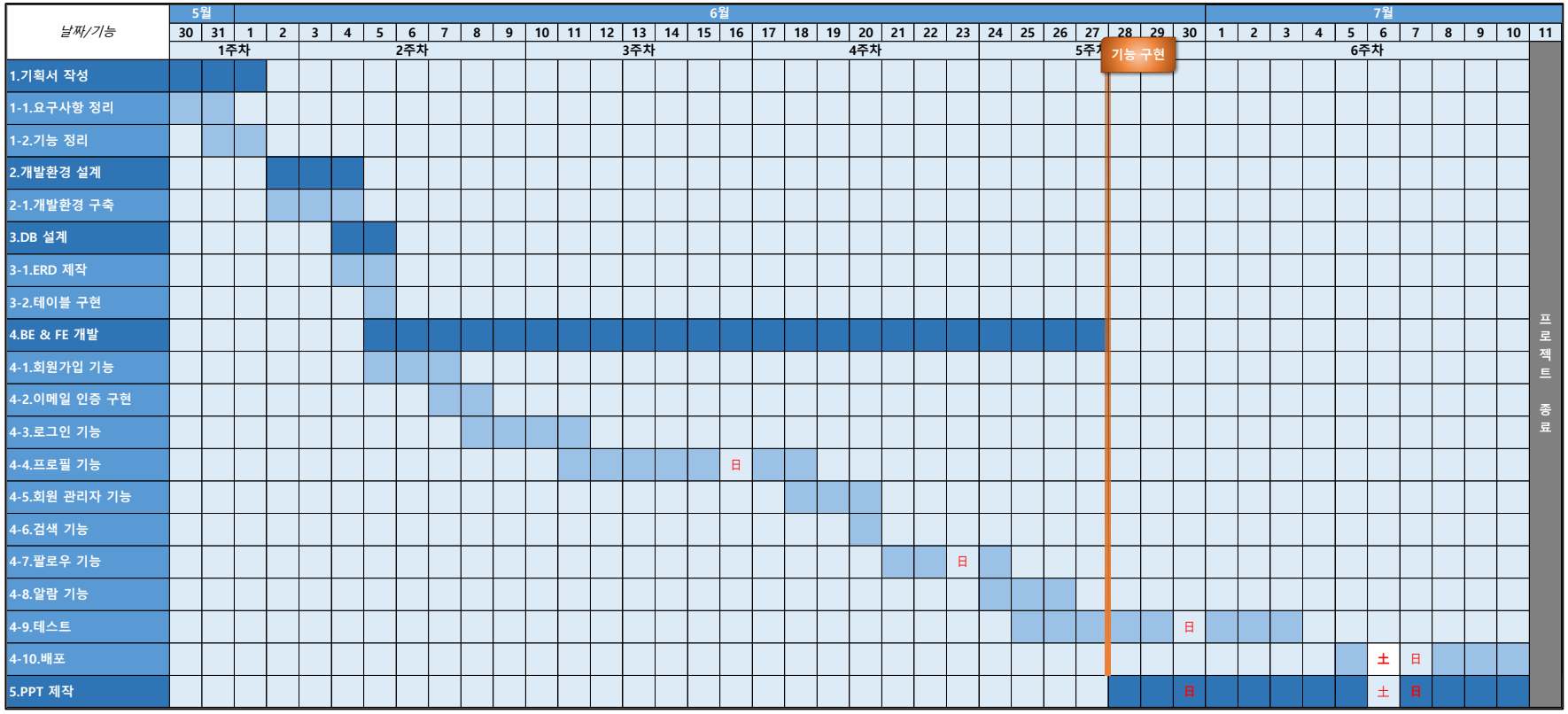
-회원관리

구분	요구사항 명	요구사항 상세 설명	구현 여부	담당자
프로그램	사이트 접속시 로그인 처리	로그인 시 필요한 모든값이 입력되어야 로그인 가능	Y	김도영
		로그인 시 입력한 아이디가 불일치할경우 로그인불가	Y	
		로그인 시 입력한 아이디가 동일하지만 비밀번호가 불일치할경우 로그인불가	Y	
		로그인 성공시 게시글이 보이는 메인화면으로 이동한다	Y	
		로그인 시 1시간 동안 유효 한 세션이 발생되며 사용자에게 쿠키를 전달한다	Y	
	회원가입	회원가입시 아이디 중복확인 후 회원가입 가능	Y	
		비밀번호 는 영어,숫자,특수기호 포함 최소 4자 최대 16자 가능,	Y	
		회원가입시 비밀번호 확인 후 회원가입 가능	Y	
		생년월일 선택시 해당 달의 마지막 날짜 구하는 기능 (윤년여부 판단) Date함수 이용하여 마지막 날짜 선택 기능 구혀	Y	
		이메일 인증 발송시 랜덤한 숫자가 적힌 메일이 발송됨	Y	
		이메일 주소가 형식과 다를 경우 회원가입 불가	Y	
		이메일 인증번호가 틀렸을 경우 회원가입 불가	Y	
		회원가입에 필요한 모든값이 입력되어야 회원가입 가능	Y	
	프로필	네비게이션바 하단의 프로필 버튼 누르면 프로필 폼으로 이동	Y	
		자신의 프로필 화면에서는 프로필사진, 한줄소개,게시물 수,쇼츠 수 ,팔로워 수, 팔로우 수,편집,탈퇴 버튼이 나옴	Y	
		상대방의 프로필 화면 에서는 프로필사진, 한줄소개,게시물 수,쇼츠 수 ,팔로워 수, 팔로우 수, 팔로우 버튼이 나옴	Y	
	프로필 및 개인정보 수정	프로필 화면의 편집 버튼 클릭시 수정 화면으로 이동	Y	
		수정화면에서 변경 버튼 클릭시 자신의 프로필정보,개인정보 수정가능	Y	
		개인정보 수정시 새 비밀번호 확인후 변경 가능	Y	
	탈퇴	탈퇴버튼 클릭시 탈퇴화면으로 이동	Y	
		비밀번호가 일치할경우 회원탈퇴 가능	Y	
	로그아웃	로그아웃시 로그인 페이지로 이동	Y	
		로그아웃시 로그인때 발급 된 세션과 쿠키 삭제	Y	
	팔로우	팔로우 버튼 클릭시 팔로우 가능 후 언팔로우 버튼으로 변경	Y	
		언팔로우 버튼 클릭시 언팔로우 가능 후 팔로우 버튼으로 변경	Y	
	팔로우목록	프로필 화면의 팔로우,팔로워 수 클릭시 팔로우,팔로워 목록으로 이동	Y	
		팔로우,팔로워 목록에 표시된 아이디 클릭시 상대 프로필 화면으로 이동	Y	
	전체 검색	입력된 검색어로 게시글,쇼츠,회원 검색결과를 각각 tab으로 출력	Y	
	관리자 페이지	모든 회원들의 목록이 보여짐	Y	
		직접 버튼을 눌러 삭제하거나 체크박스를 통해 일괄 삭제 가능	Y	
		회원들의 목록을 Excel 파일로 다운로드 구현	Y	
		아이디 또는 이름 검색으로 특정 회원을 볼수 있음	Y	
	알람	팔로우,언팔로우 버튼 을 클릭시 상대방에게 메시지 보내짐	Y	
		네비게이션바 의 알람 버튼 클릭시 메시지 보관함으로 이동	Y	
		메시지 보관함에서 확인 버튼 클릭시 상대 프로필 화면으로 이동 후 메시지 안보임	Y	
		DB event scheduler 활용하여 현재 시간에서 7일 이전인 경우 메시지 db삭제 처리	Y	

7.ERD


게시글
사용자
쇼츠
postcomment
pc_no int(11)
pc_text varchar(100)
pc_id varchar(20)
pc_indate timestamp
alarm
alarm_no int(11)
user_id varchar(30)
other_id varchar(30)
message varchar(100)
flag int(1)
in_date timestamp
short_comment
s_no int(11)
sc_id varchar(20)
sc_comment varchar(150)
sc_date timestamp
post
p_no int(11)
p_id varchar(20)
p_place varchar(20)
p_text varchar(200)
p_indate timestamp
p_views int(11)
user
id varchar(20)
pw varchar(50)
name varchar(50)
birth varchar(50)
email varchar(50)
short_form
s_no int(11)
sv_no int(11)
sv_thumbnail varchar(50)
s_id varchar(20)
s_date timestamp
s_description varchar(300)
s_place varchar(20)
postpic
pp_no int(11)
pp_pic varchar(50)
short_video
sv_no int(11)
sv_addr varchar(50)
sv_thumbnail varchar(50)
postlike
pl_no int(11)
pl_id varchar(20)
profile
id varchar(20)
u_intro varchar(70)
u_pic varchar(50)
follow
follow_id varchar(20)
follower_id varchar(20)
short_like
slike_id varchar(20)
s_no int(11)

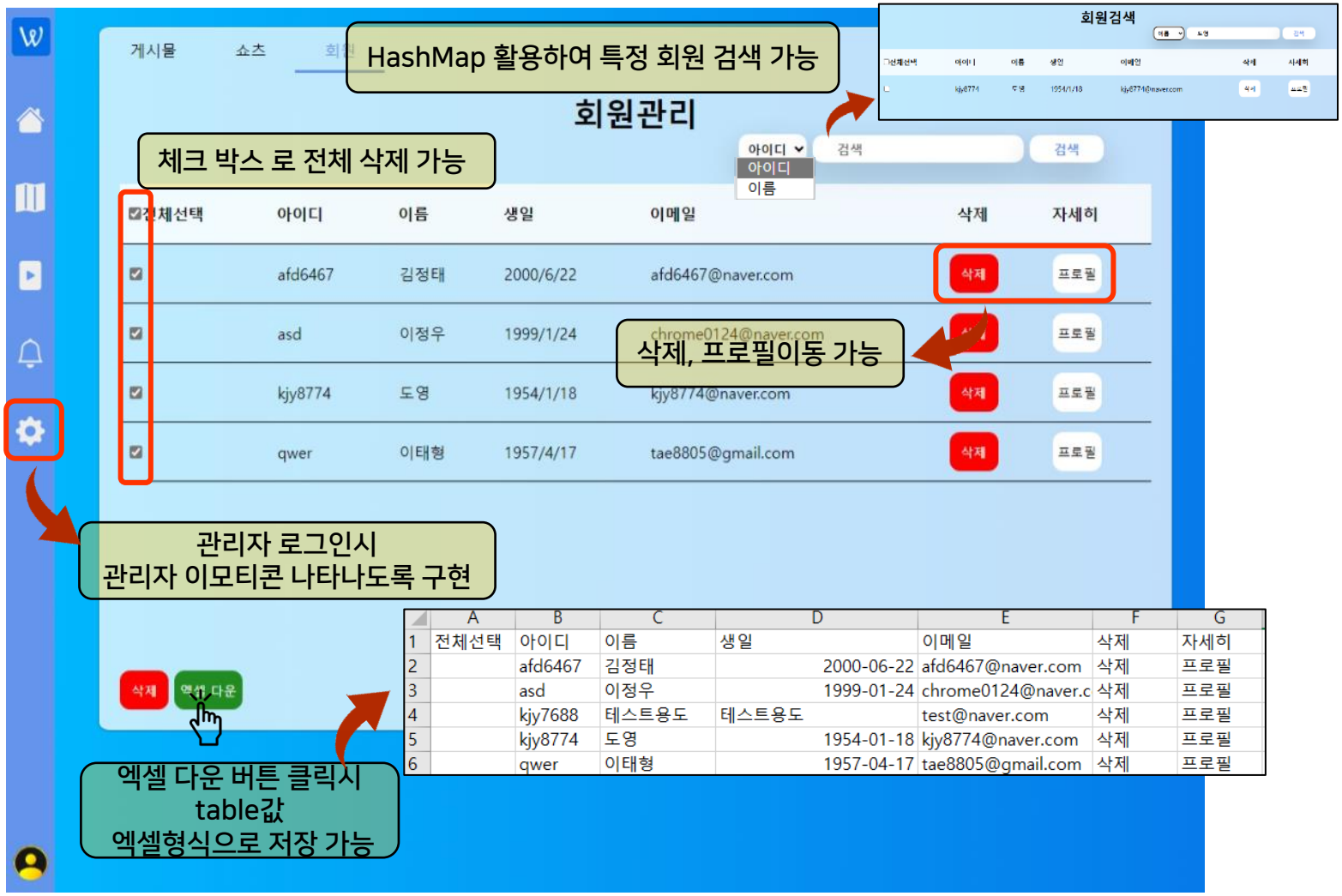
8.타임라인


날짜/기능
5월
6월
7월
30 31 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 1 2 3 4 5 6 7 8 9 10 11
1주차
2주차
3주차
4주차
5주차
6주차
기능 구현
1.기획서 작성
1-1.요구사항 정리
1-2.기능 정리
2.개발환경 설계
2-1.개발환경 구축
3.DB 설계
3-1.ERD 제작
3-2.테이블 구현
4.BE & FE 개발
4-1.회원가입 기능
4-2.이메일 인증 구현
4-3.로그인 기능
4-4.프로필 기능
4-5.회원 관리자 기능
4-6.검색 기능
4-7.팔로우 기능
4-8.알람 기능
4-9.테스트
4-10.배포
5.PPT 제작
日
土
프로젝트 종료

9.주요기능

- 회원 관리자 화면

-사용기술 : html,css,javascript,HashMap,SheetJS(엑셀 다운로드 api)

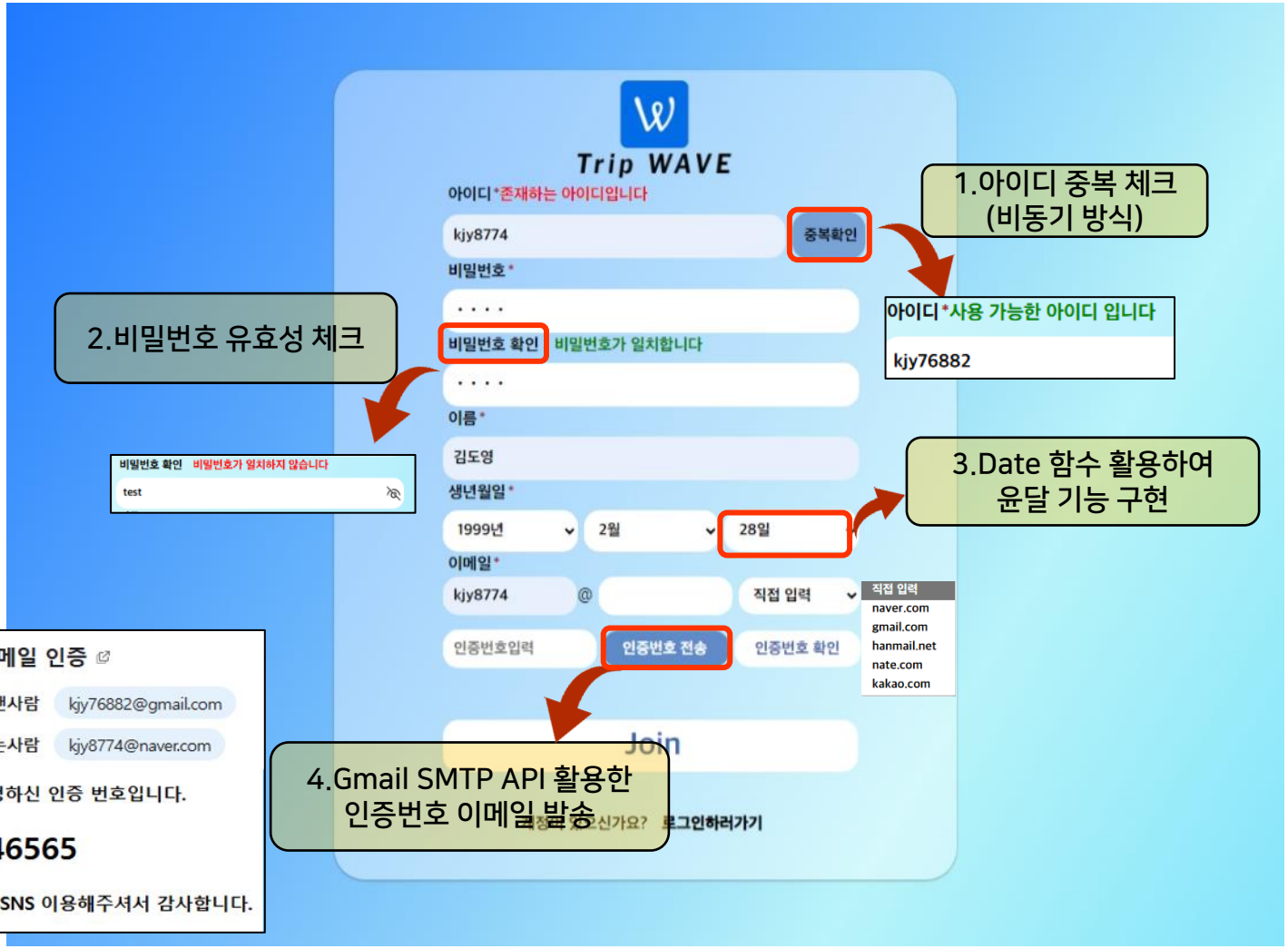


	A	B	C	D	E	F	G
1	전체선택	아이디	이름	생일	이메일	삭제	자세히
2		afd6467	김정태	2000-06-22	afd6467@naver.com	삭제	프로필
3		asd	이정우	1999-01-24	chrome0124@naver.c	삭제	프로필
4		kjy7688	테스트용도	테스트용도	test@naver.com	삭제	프로필
5		kjy8774	도영	1954-01-18	kjy8774@naver.com	삭제	프로필
6		qwer	이태형	1957-04-17	tae8805@gmail.com	삭제	프로필

9.주요기능

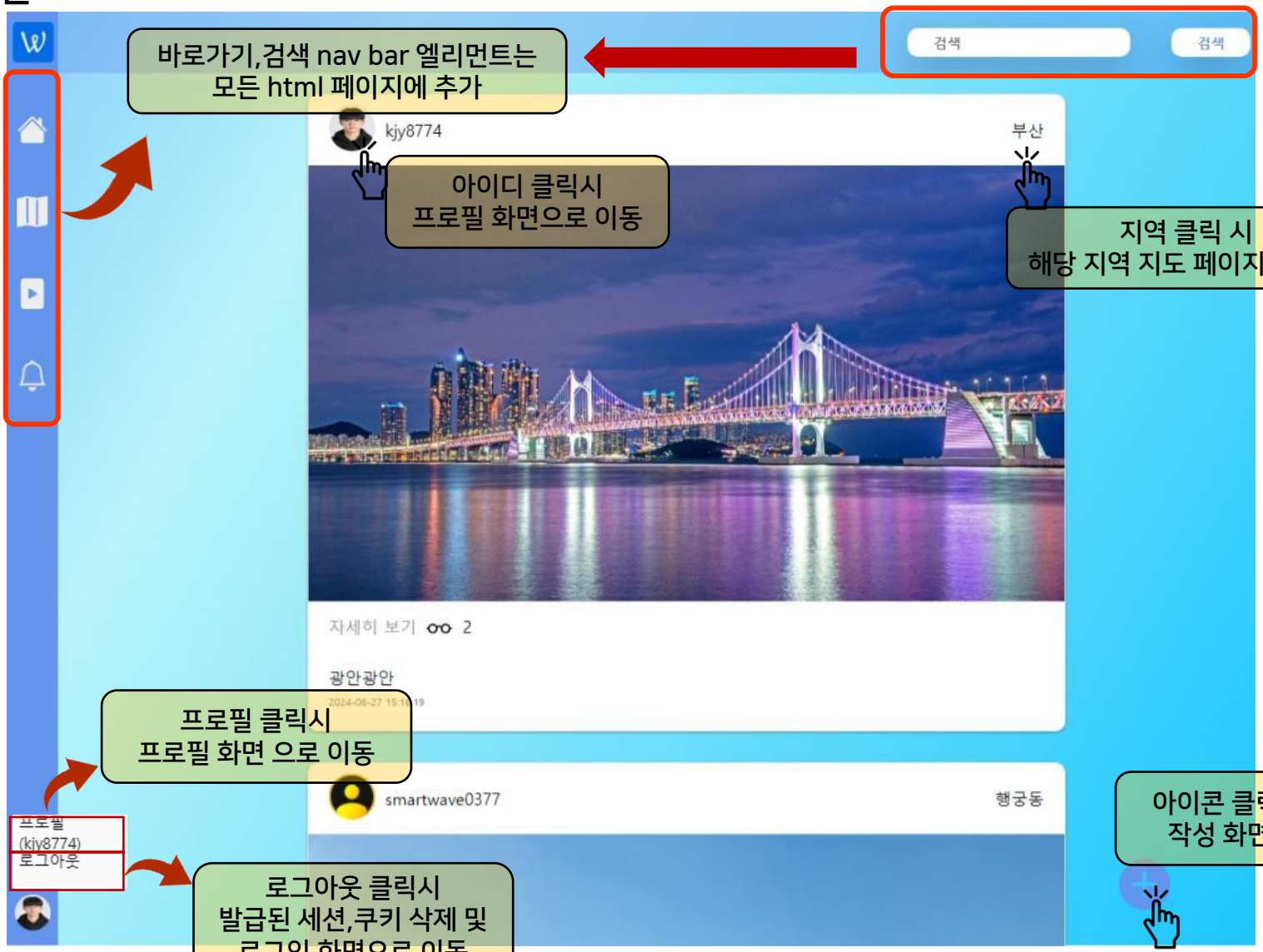
-회원가입 화면

-사용기술 : html,css,javascript,ajax, SMTP api



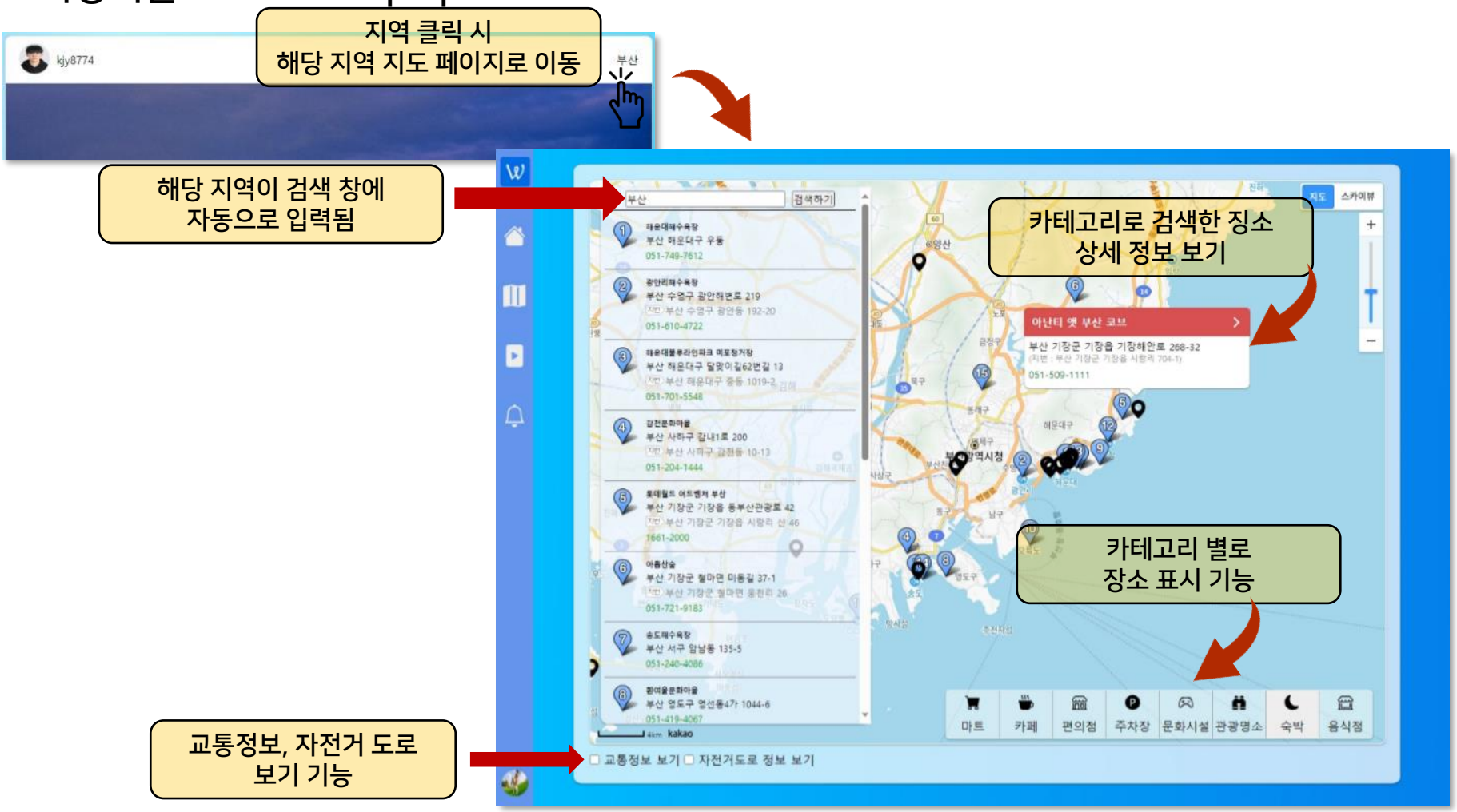
9.주요기능

-메인화면


바로가기,검색 nav bar 엘리먼트는
모든 html 페이지에 추가
검색
검색
kjy8774
부산
아이디 클릭시
프로필 화면으로 이동
지역 클릭 시
해당 지역 지도 페이지로 이동
자세히 보기 2
광안광안
프로필 클릭시
프로필 화면 으로 이동
smartwave0377
행궁동
프로필
(kjy8774)
로그아웃
로그아웃 클릭시
발급된 세션,쿠키 삭제 및
로그인 화면으로 이동
아이콘 클릭시 게시글
작성 화면으로 이동

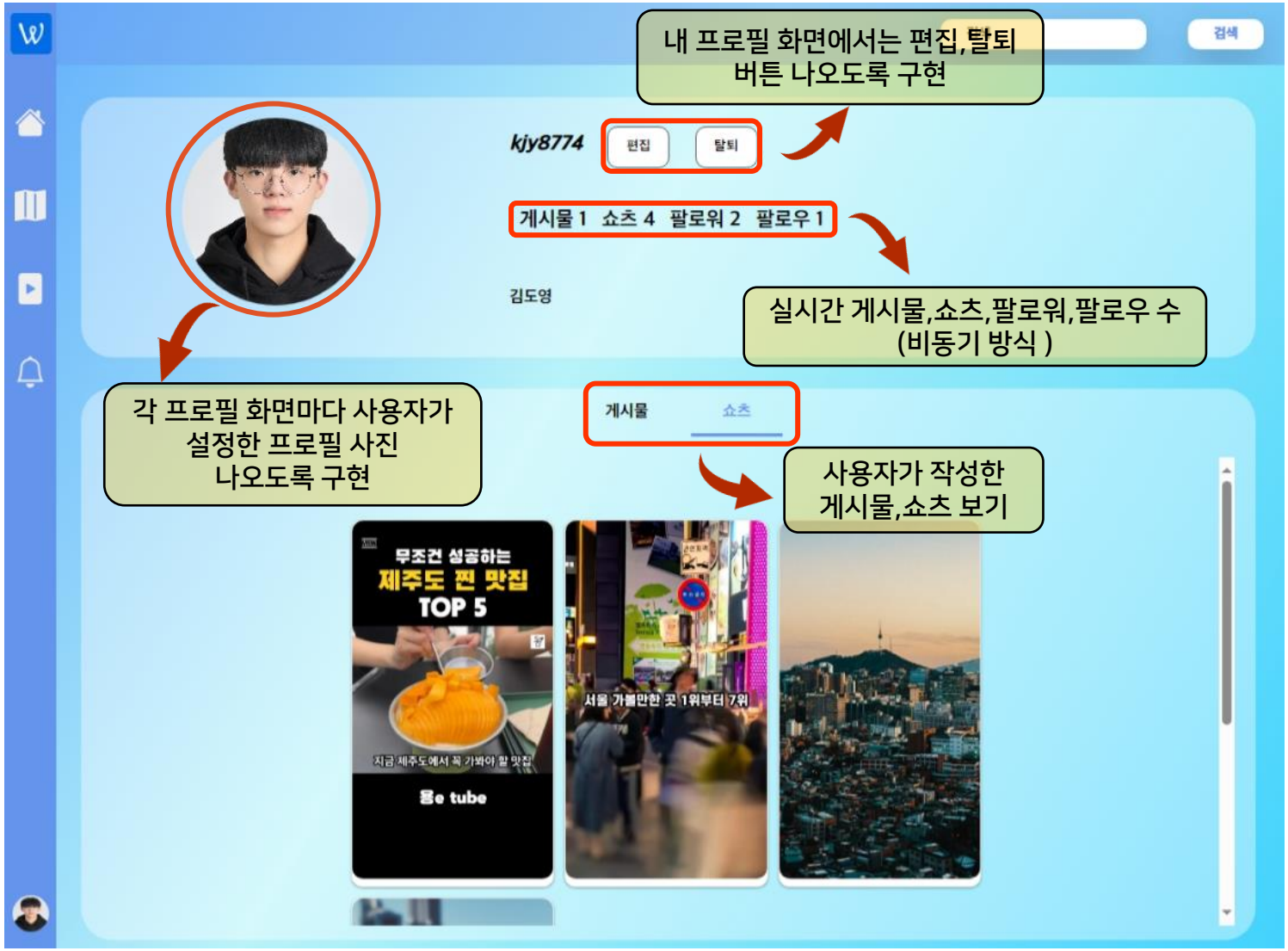
9.주요기능

-지도 기능
-사용기술 : kakao map api



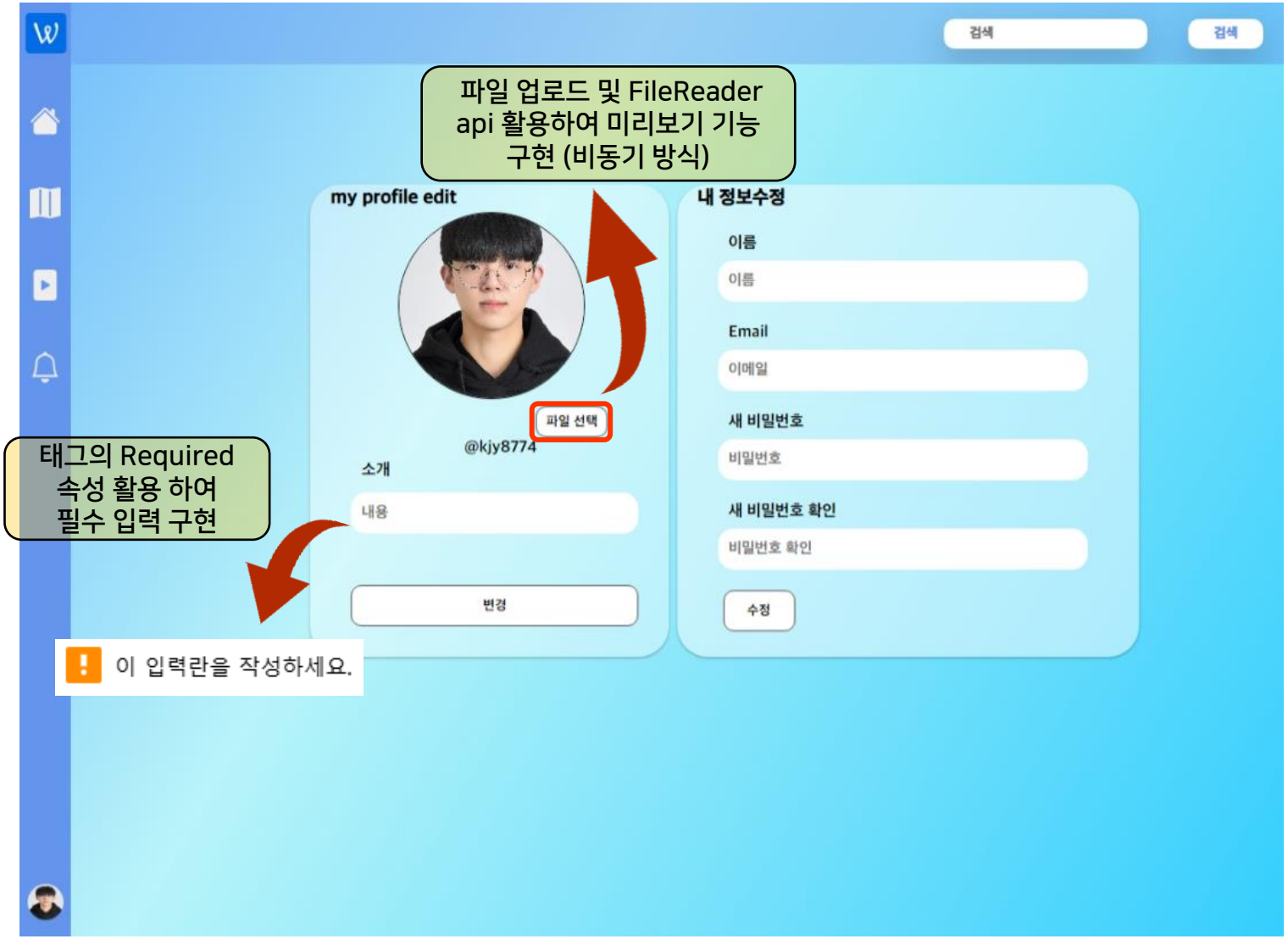
9.주요기능

- 프로필 화면


내 프로필 화면에서는 편집,탈퇴 버튼 나오도록 구현
검색
kjy8774
편집
탈퇴
게시물 1 쇼츠 4 팔로워 2 팔로우 1
김도영
실시간 게시물,쇼츠,팔로워,팔로우 수 (비동기 방식)
각 프로필 화면마다 사용자가 설정한 프로필 사진 나오도록 구현
게시물
쇼츠
사용자가 작성한 게시물,쇼츠 보기
무조건 성공하는
제주도 찐 맛집
TOP 5
지금 제주도에서 꼭 가봐야 할 맛집
e tube
서울 가볼만한 곳 1위부터 7위

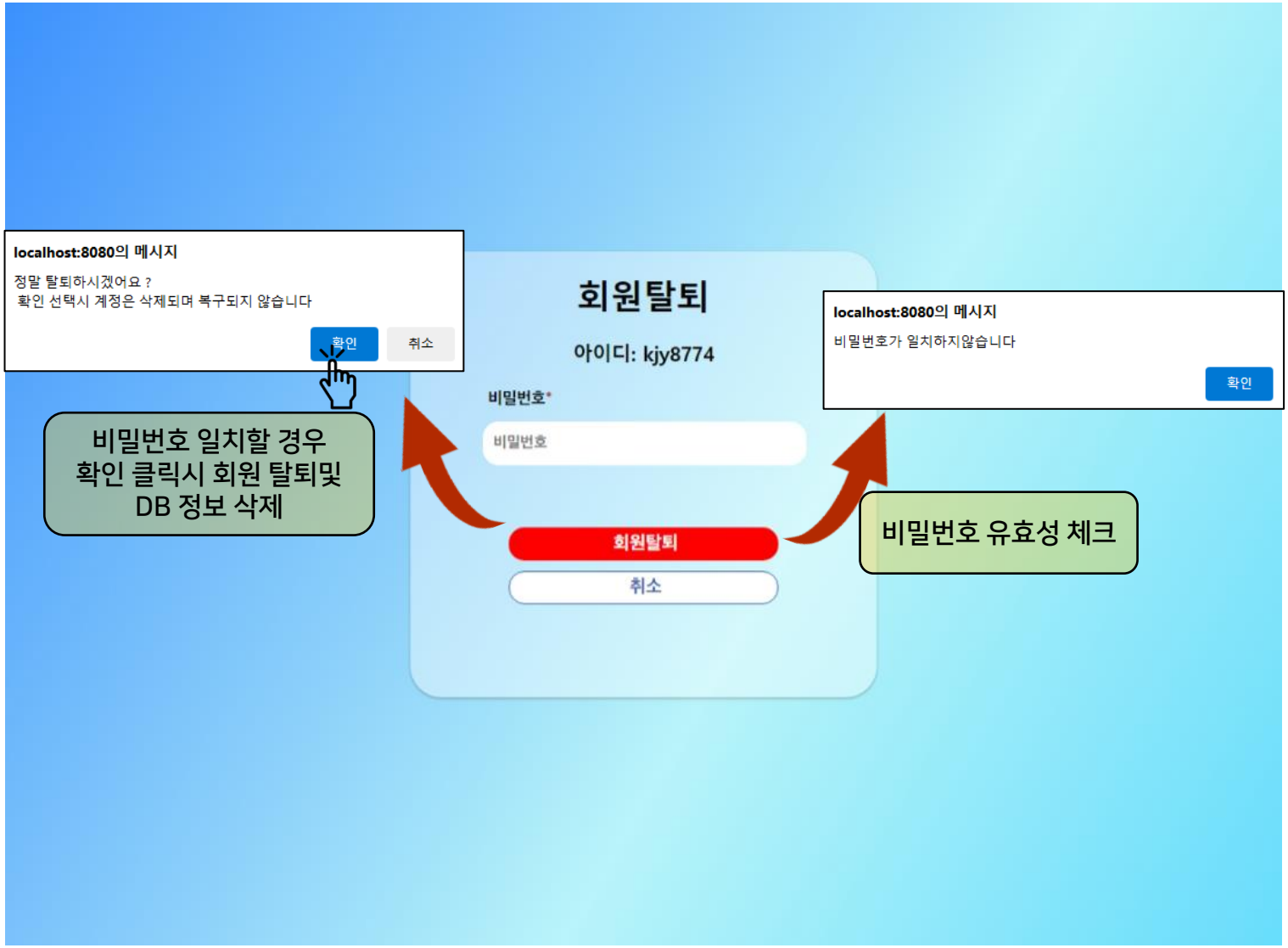
9.주요기능

- 프로필 수정 화면



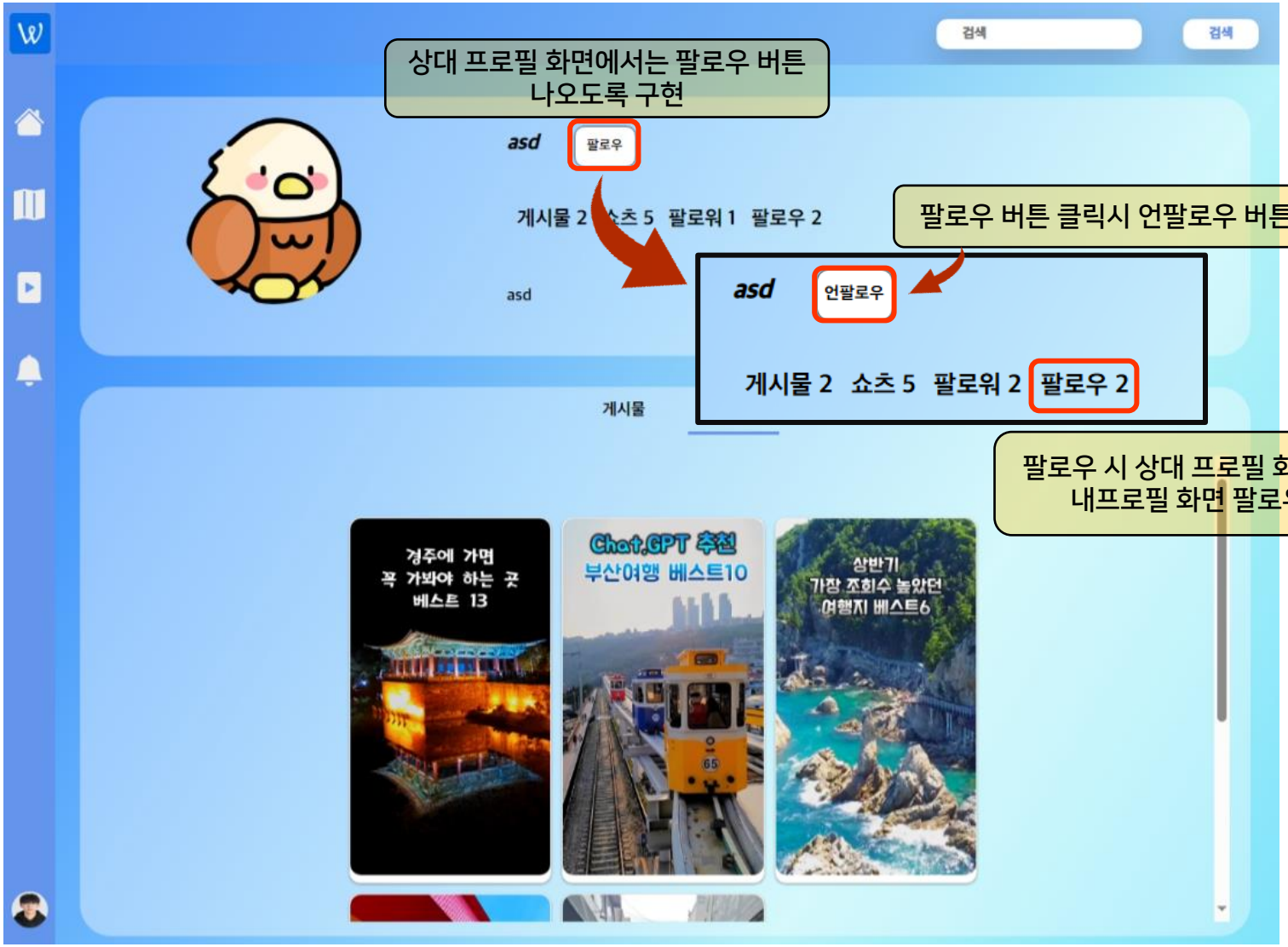
9.주요기능

- 탈퇴 화면


localhost:8080의 메시지
정말 탈퇴하시겠어요 ?
확인 선택시 계정은 삭제되며 복구되지 않습니다
확인
취소
비밀번호 일치할 경우
확인 클릭시 회원 탈퇴및
DB 정보 삭제
회원탈퇴
아이디: kjy8774
비밀번호*
비밀번호
회원탈퇴
취소
localhost:8080의 메시지
비밀번호가 일치하지않습니다
확인
비밀번호 유효성 체크

9.주요기능

- 팔로우,언팔로우 기능



9.주요기능

- 팔로우,팔로워 목록 화면

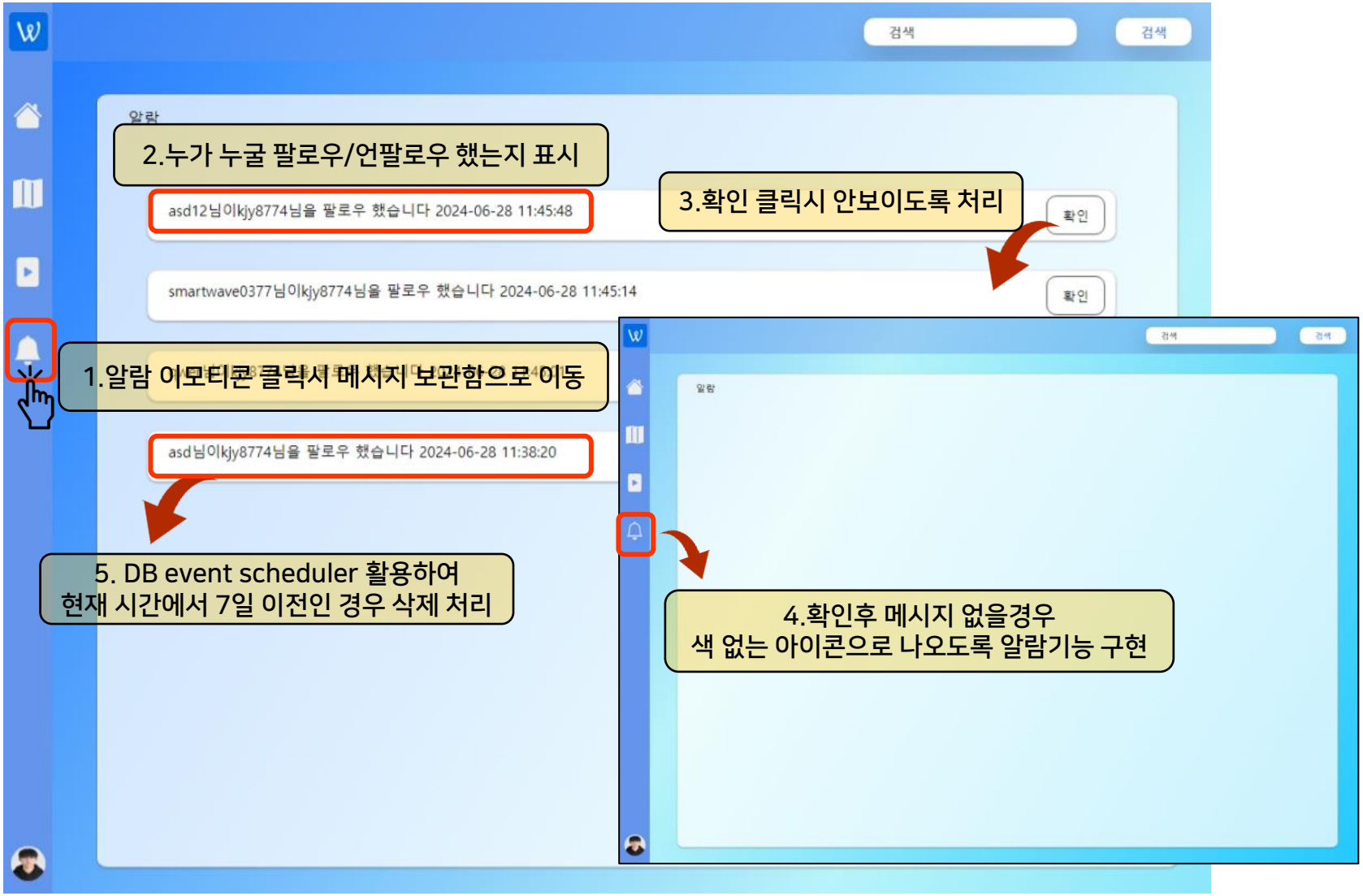
1.팔로워,팔로우수 클릭시 목록으로 이동

2.팔로워,팔로우목록
tab 기능 활용하여 구현

3.아이디 클릭시 상대 프로필
화면으로 이동

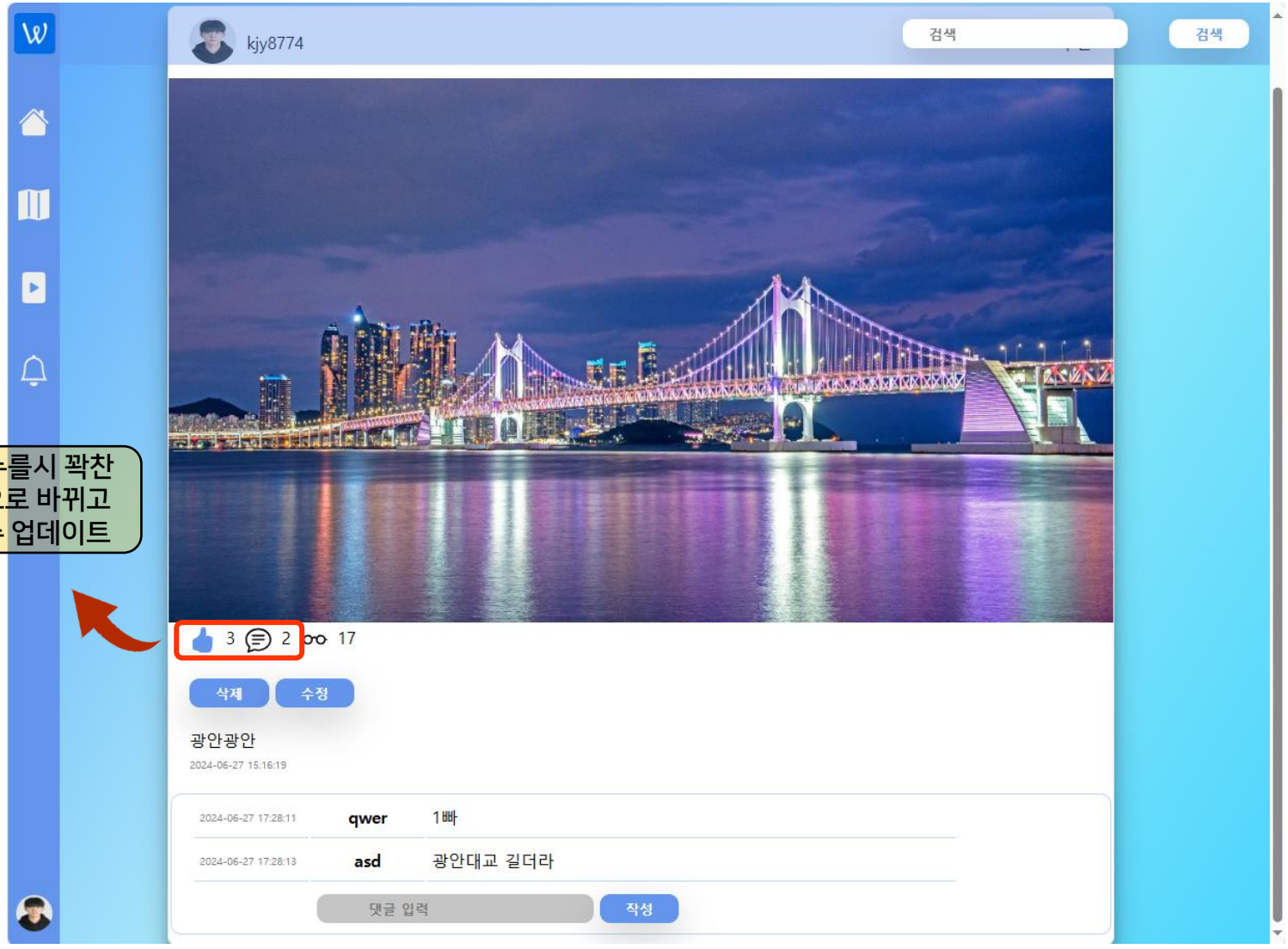
9.주요기능

- 알람 기능


검색
검색
알람
2.누가 누굴 팔로우/언팔로우 했는지 표시
asd12님이kjy8774님을 팔로우 했습니다 2024-06-28 11:45:48
3.확인 클릭시 안보이도록 처리
확인
smartwave0377님이kjy8774님을 팔로우 했습니다 2024-06-28 11:45:14
확인
1.알람 이모티콘 클릭시 메시지 보관함으로 이동
asd님이kjy8774님을 팔로우 했습니다 2024-06-28 11:38:20
5. DB event scheduler 활용하여
현재 시간에서 7일 이전인 경우 삭제 처리
4.확인후 메시지 없을경우
색 없는 아이콘으로 나오도록 알람기능 구현

9.주요기능

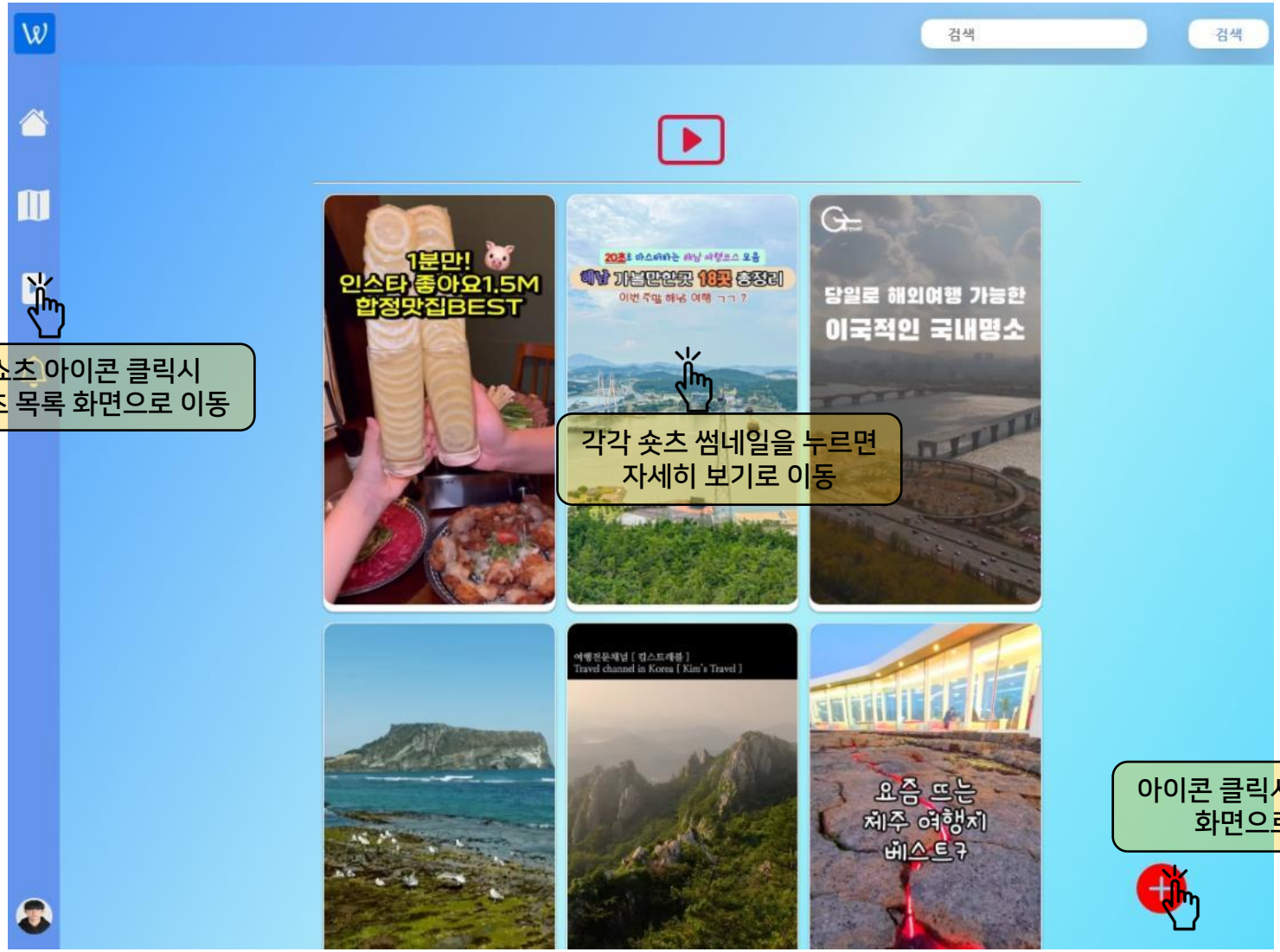
-게시글 자세히보기 화면



좋아요 누를시 꽉찬 아이콘으로 바뀌고 좋아요수 업데이트

9.주요기능

-쇼츠 목록 화면

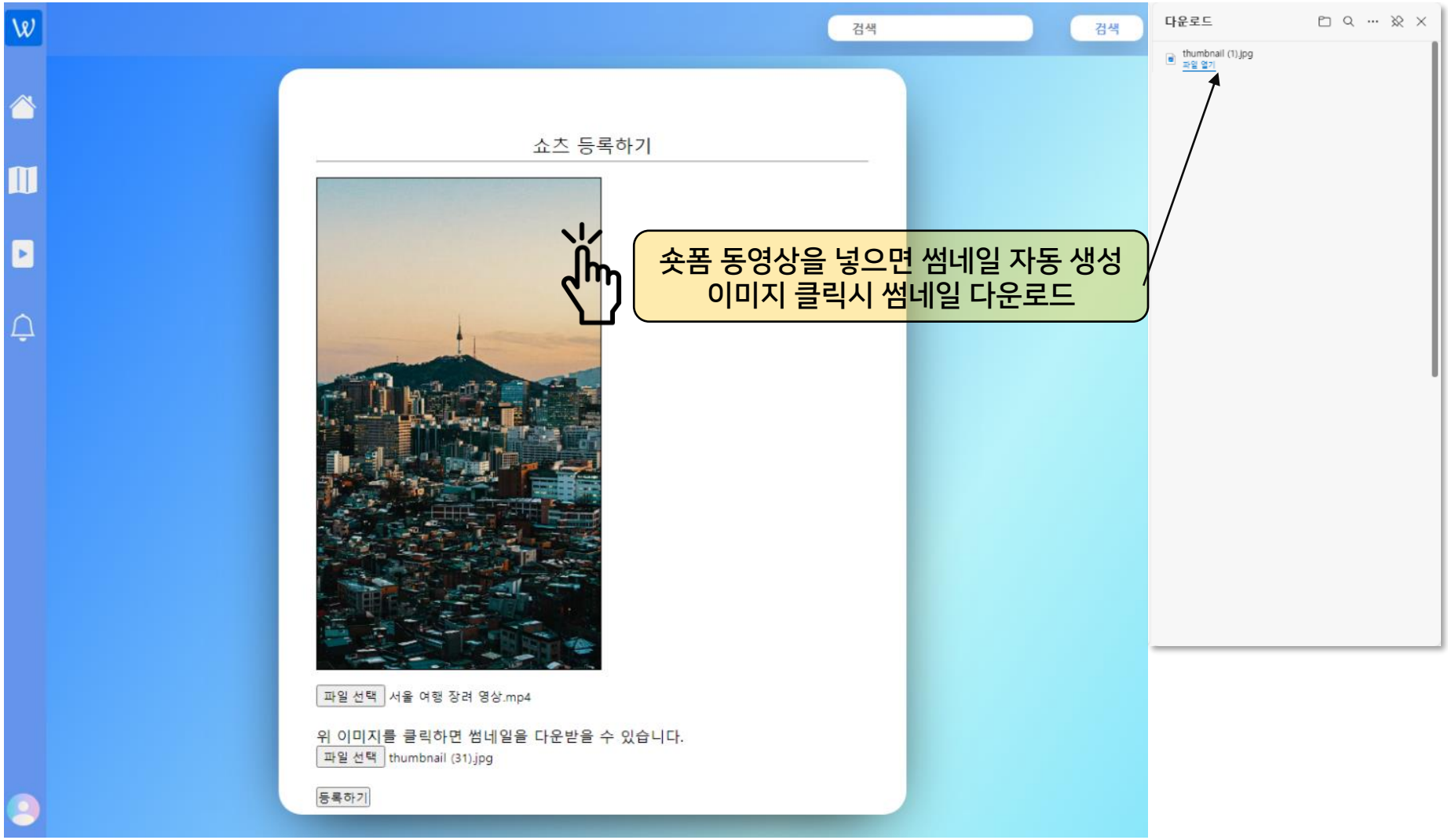


9.주요기능

-쇼츠 자세히 보기 화면

9.주요기능

-쇼츠 작성 화면



10.시연 영상

https://drive.google.com/file/d/1BHSzn_VJkQQXqH1hD1nm249WB3fjUDLM/view?usp=drive_link

11.마무리

- 강점 및 기대효과

강점

사용자 생성 콘텐츠

사용자들의 실시간 리뷰와 경험 공유를 통해
신뢰성 높은 정보를 제공

다양한 기능 통합

숙박, 관광 명소, 맛집 등 다양한 여행 정보를 한
곳에서 검색하고 계획할 수 있는 기능 제공

유비쿼터스 환경

구글 클라우드 플랫폼(GCP) 을 이용하여
웹 서비스 를 언제 어디서든 서비스를 이용가능

기대효과

협약을 통한 광고 수익 기대

지역명소에서 숙박 예약 등을 제공하는 회사들과 협약을 맺어 ,여행 정보와 관련된 특가나 예약 사이트로 이어지는 광고 배너를 통한 수익을 창출할 수 있을 것으로 기대됩니다. 이를 통해 지역 경제 활성화에도 크게 이바지 할 수 있을 것으로 기대됩니다 .

구독 시스템

사용자가 구독을 할 경우 광고를 중단시키고 ,
계정의 신뢰성을 높여주는 계정 인증을 관리자가 직접 인증해주는 방식을 통해 이익을 창출할 수 있을 것으로 기대됩니다 .

11.마무리

- 느낀 점

협업

팀 회의를 통해 업무를 배정하고 ,각자 맡은 기능을 구현한 후 합치는 과정에서 공유 파일을 사용하는 방식이 비효율적으로 느껴졌었습니다 .이러한 문제를 해결하기 위해 Git을 사용하게 되었습니다. 형상관리 도구를 사용하면서 충돌 이슈를 경험하고 파일이 사라지는 등의 다양한 오류를 겪었지만 팀원들과의 소통 및 백업 파일을 통해 오류를 해결해 나갔고 이후에 브랜치 전략을 탐색하여 main, dev, 개인 작업 브랜치를 사용하는 전략을 적용했습니다. 이로 인해 충돌 패턴을 이해하게 되었고, 그 이후부터는 충돌 없이 효율적으로 작업할 수 있었습니다.
이러한 경험을 통해 형상 관리의 중요성을 이해하고, 팀 내 소통과 협업의 중요성을 깨닫는 계기가 되었습니다.

개발 환경

이전에는 Spring과 Oracle DB를 주로 사용했습니다. 이번에는 Spring Boot와 Maria DB를 사용하여 개발 환경을 구축하면서 새로운 환경에서 프로젝트를 진행 했습니다.
시행착오가 많았지만 스프링 부트 의 내장된 웹 서버와 간편한
의존성 설정 덕분에 결과적으로 더 간편하게 작업할 수 있었습니다.
이를 통해 새로운 환경 에서도 시간이 주어진다면 공부를 통해서 적응 할 수 있다는 자신감과 유연성을 키우는 계기가 되었습니다.

기술

서비스를 위해 회원 관리의 절차를 이해하고 이를 CRUD 통해 구현하였습니다.
이를 통해 데이터 조작의 업무를 수행할 수 있는 역량을 키우게 되었습니다.
또한 Ajax를 사용하여 비동기 방식을 통해 웹 페이지를 부분적으로 로딩 하면서
아이디 중복 체크, 게시물 수 조회, 팔로우 기능 등을 구현했습니다.
이 과정에서 새로운 기술을 도입하고 적용하는 데 자신감을 얻었으며,코드의 일부분만 보는 것이 아닌 전체 흐름을 이해하는 시야를 넓히게 되었습니다.

배포

Google Cloud Platform(GCP)을 이용하여 Ubuntu 22.04 기반의 e2 서버에 배포 작업을 진행했습니다. 우선, 리눅스 환경에서 테스트 배포를 수행한 후, GCP를 활용하여 실제 배포를 완료했습니다. 자바 소스를 JAR 파일로 배포하고, MariaDB의 데이터를 덤프 하는 과정을 통해 클라우드 서비스의 개념을 학습했습니다.
이를 통해 TUI 기반의 OS 를 경험하면서 배포 과정의 흐름을 이해할수 있었습니다.
특히, 퍼미션 설정 및 사용자 계정 관리의 중요성 을 깨닫게 되었습니다.

감사합니다

개발:김도영,이정우,이태형

도메인:http://tripwaves.site

Github : https://github.com/doyoungking/TripSNS_project

개발기간:2024.06.03~2024.07.18

김도영 /010-7688-8774

12.기능별 주요소스

- 아이디 중복체크

Javascript:
Ajax 활용하여 비동기 방식으로 아이디 확인

```
function chkid() : void  {// 유효성 아이디 중복체크  Show usages  doyoung *
    let sendid = $('#newid').val();// 입력한 id 값 저장
    var iderrmsg : HTMLElement  = document.getElementById( elementId: 'idmsg');
    // 아이디 중복체크후 가능 여부 확인 메세지 출력용
    $.ajax( url: {
        url: "/idchk",//컨트롤러에서 요청받을 주소
        type: "post",//post 방식으로 전달
        data: {"originid": sendid},
        dataType: "text",
        //dataType: "json",
        success: function (cnt) : boolean  {// 컨트롤러에서 넘어온 cnt 값 받는다
            console.log(typeof cnt);
            if (cnt == '0') {// cnt 가 0 이면 DB에 없음 -> 사용 가능 아이디
                //if(cnt == 0)
                iderrmsg.style.color = "green";
                iderrmsg.textContent = "사용 가능한 아이디 입니다";
                return true;
            } else {
                $('#newid').val( value: "");
                $('#newid').focus();
                iderrmsg.style.color = "red";
                iderrmsg.textContent = "존재하는 아이디입니다";
                return false;
            }
        }
    })

}
```

Controller:
DB에서 받은 값을 ajax로 리턴

```
@ResponseBody  doyoung
@PostMapping(value = "idchk")//중복체크 버튼
public int idchk(@RequestParam("originid") String id) throws Exception {
    int cnt = userservice.idchk(id);
    return cnt;
}
```

Mapper

```
<!--아이디 중복체크-->
<select id="idchk" parameterType="String" resultType="int">
    select count(id)
    from user
    where id = #{id}
</select>
```

12.기능별 주요소스

- 이메일 api

Javascript:

Ajax 활용하여 비동기 방식으로 이메일형식 보냄

```
function sendNumber() : void  {  Show usages  doyoung
    let nowmail = $('#nowmail').val();
    let inputmail = $('#domain-txt').val();
    let fullmail : string  = nowmail + "@" + inputmail;
    let cnt : number  = 0;
    $.ajax( url: {
        url: "/mail",
        type: "post",
        dataType: "json",
        data: {"mail": fullmail},
        success: function (data) : void  {
            alert("인증번호가 발송되었습니다");
            $("#Confirm").attr( name: "value", data);
        }
    });
}
```

Controller:

service단에서 생성했던 랜덤 숫자 Ajax로 리턴

```
@Controller  doyoung *
@RequiredArgsConstructor
public class MailController {
    private final MailService mailService;
    @ResponseBody  doyoung *
    @PostMapping("/mail")//ajax 로 받을 식별자
    public String MailSend(String mail){

        int number = mailService.sendMail(mail);// 받은 이메일 매개변수로 활용

        String num = "" + number; // 랜덤 숫자

        return num;
    }
```

Service:

랜덤 숫자 생성후 매개변수로 받은 이메일로 인증번호 전송

```
@Service  2 usages  doyoung *
@RequiredArgsConstructor
public class MailService {
    private final JavaMailSender javaMailSender;
    private static final String senderEmail = "kjy76882@gmail.com";//보낸사람 이메일 내이메일  1 usage
    private static int number;  3 usages
    public static void createNumber() {  1 usage  new *
        number = (int) (Math.random() * (90000)) + 100000;// 100000부터 189999 사이의 정수를 생성
    }
    public MimeMessage CreateMail(String mail) {  1 usage  doyoung *
        createNumber();
        MimeMessage doyoungMessage = javaMailSender.createMimeMessage();
        try {
            doyoungMessage.setFrom(senderEmail);
            doyoungMessage.setRecipients(MimeMessage.RecipientType.TO, mail);
            doyoungMessage.setSubject("이메일 인증");
            String body = "";
            body += "<h3>" + "요청하신 인증 번호입니다." + "</h3>";
            body += "<h1>" + number + "</h1>";
            body += "<h3>" + "TripSNS 이용해주셔서 감사합니다." + "</h3>";
            doyoungMessage.setText(body,  charset: "UTF-8",  subtype: "html");
        } catch (MessagingException e) {
            e.printStackTrace();
        }
        return doyoungMessage;
    }
    public int sendMail(String mail) {  1 usage  doyoung *
        MimeMessage doyoungMessage = CreateMail(mail);
        javaMailSender.send(doyoungMessage);
        return number;
    }

}
```

12.기능별 주요소스

- 이메일 api(2)

Javascript:

Ajax 통해 받은 랜덤숫자와 사용자가 입력한 인증번호 확인

```
function sendNumber() : void  {  Show usages  doyoung
    let nowmail = $('#nowmail').val();
    let inputmail = $('#domain-txt').val();
    let fullmail : string  = nowmail + "@" + inputmail;
    let cnt : number  = 0;
    $.ajax( url: {
        url: "/mail",
        type: "post",
        dataType: "json",
        data: {"mail": fullmail},
        success: function (data) : void  {
            alert("인증번호가 발송되었습니다");
            $("#Confirm").attr( name: "value", data);
        }
    });
}
```

```
function confirmNumber() : boolean  {  Show usages  doyoung
    var number1 = $("#number").val();
    var number2 = $("#Confirm").val();
    if (number1 == number2) {
        alert("인증되었습니다.");
        return true;
    } else {
        alert("인증번호가 다릅니다 다시입력해주세요.");
        return false;
    }
}
```

12.기능별 주요소스

- 세션

Controller:

로그인 버튼 클릭했을시 로그인 유효성 확인후 로그인 아이디에 세션 값 부여

```
@PostMapping("loginsave")//로그인 버튼  doyoung +1 *
public String loginsave(@RequestParam("id") String id, @RequestParam("pw") String pw
        , HttpSession session, RedirectAttributes rt) throws Exception {
    //model 객체가 redirect 하는순간 사라지기 때문에 값을 넘기기위해  RedirectAttributes
    // 공부하여  리다이렉트 페이지로 데이터 넘기기위해 적용 ,
    // 임시로 저장하는 방식인 flashattribute 적용 하여 세션에 저장되어 사용된 뒤에 자동으로 삭제되도록함
    // -> 임시로 사용되는 데이터 다루기위해 사용(리다이렉트 직전 클래시에 저장하는 메서드, Redirect 이후에는 소멸)
    UserVO uvo = userservice.login(id);
//       System.out.println("아이디:"+uvo.getId()+"비번:"+uvo.getPw());
    // 리턴받아온 db 의 pw 와 login.html에서 post 방식으로 받아온 pw 일치확인
    if (uvo != null) {
        if (uvo.getPw().equals(pw)) {
            //비밀번호가 일치한후 접속시 쿠키값을 서버에 전송하고
            // 서버에서는 쿠키값을 참고하여 세션에 등록된 변수값을 가져오도록 설정
            if (session.getAttribute( s: "userid") != null) {// userid값이 존재한다면 -> 쓰레기값이 있다면
                session.removeAttribute( s: "userid");
            }
            session.setAttribute( s: "userid", uvo.getId());
        } else {
            //비밀번호 틀릴경우
            rt.addFlashAttribute( attributeName: "mesg",  attributeValue: "비밀번호가 일치하지않습니다");
            return "redirect:/login";
        }
    } else {
        // 아이디 없을경우
        rt.addFlashAttribute( attributeName: "mesg",  attributeValue: "존재하지않는 아이디 입니다");
        return "redirect:/login";
    }
    return "redirect:main";
}
```

Controller:

로그아웃시 세션값 삭제

```
@GetMapping(value = "logout")//로그아웃  doyoung +1
public String logout(HttpSession session) {
    session.invalidate();
    return "redirect:/login";
}
```

application.yml:

로그인시 1시간 동안 유효한 세션 발생

```
server:
  servlet:
    encoding:
      enabled: 'true'
    session:
      timeout: 3600
```

12.기능별 주요소스

- 인터셉터

interceptor:

세션으로 부 터 값을 가져온 후 세션 값 없을 경우 로그인 화면으로 가지도록 인터셉터 처리

```
public class IDchkInterceptor implements HandlerInterceptor {  3 usages  doyoung *
    //  HandlerInterceptor를 상속받음
    //   * preHandle()-컨트롤러 진입전에 실행
    //   * postHandle()-컨트롤러 수행 후에 실행
    //   * afterCompletion()-화면으로 가기 직전에 수행
    @Override  1 usage  doyoung *
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws
            Exception {
        HttpSession session = request.getSession();
        Object user = session.getAttribute( s: "userid");//세션으로부터 값을 가져온다
        if (user == null) {//세션값이 없을경우 -> 로그인x
            response.sendRedirect( s: "/login");//로그인화면으로 보낸다
            return false;//컨트롤러 실행 x
        }
        // 로그인 해서 세션값 있으면 그대로 실행
        return HandlerInterceptor.super.preHandle(request, response, handler);
    }


}
```

config:

식별자 인터셉터 등록,예외처리

```
@Override  no usages  TaehyungLee +2 *
public void addInterceptors(InterceptorRegistry registry) {
    registry.addInterceptor(idchkInterceptor())//애플리케이션 내에 인터셉터를 등록
            .addPathPatterns("/main")// 게시판 작성가능한 메인화면 인터셉터 등록
            .addPathPatterns("/post")// 게시글 등록 인터셉터 등록
            .addPathPatterns("/posting")// 게시글 저장 인터셉터 등록
            .addPathPatterns("/detail") // 자세히보기 인터셉터 등록
```

⋮

```
.excludePathPatterns("/login*") //로그인 요청은 예외처리
.excludePathPatterns("/join*");
```

12.기능별 주요소스

- 팔로우,언팔로우 기능

HTML:

로그인시 자신의 프로필은 편집,탈퇴 버튼
상대방 프로필은 팔로우 버튼이 뜨도록 처리

```
<th:block th:if="${session.userid==mainsuser} and ${session.userid!='smartwave0377'}">
    <button class="editbtn" th:onclick="|location.href='@{/proupdate}'|">편집</button>
    <button class="editbtn" th:onclick="|location.href='@{/withdrawform}'|">탈퇴</button>
    <input type="hidden" th:value="${mainsuser}" id="myid">
    <input type="hidden" th:value="${session.userid}" id="seid">
</th:block>
<th:block th:unless="${session.userid==mainsuser}">
    <button class="followbtn" id="followlike">팔로우</button>
</th:block>
```

Javascript:

팔로우 버튼 클릭시 Ajax 실행

```
$('#followlike').click(function () {//팔로우 버튼 클릭했을시 함수 호출
    let followid = $('#otherid').val();
    let followerid = $('#currentid').val();
    let followbtn = document.getElementById('followlike');
    let message;// 알림 메세지 저장
    console.log("팔로우당한사람 아이디" + followid);
    console.log("팔로우 누른아이디 팔로워수 추가 " + followerid);
    $.ajax({
        url: "/follow",
        type: "post",
        data: {"follow_id": followid, "follower_id": followerid},
        dataType: "json",
        success: function (cnt) {
            if (cnt == 0) {
                followrCount();// 팔로우 버튼 클릭되엇을시 ajax 통해서 불러온 팔로워 숫자 나타나도록
                followCount();// 팔로우 버튼 클릭되엇을시 ajax 통해서 불러온 팔로우 숫자 나타나도록
                followbtn.innerText = '팔로우';
            } else {
                followrCount();
                followCount();
                followbtn.innerText = '언팔로우';
            }
        }
    })
```

Controller:

서비스단 메서드 호출후 리턴받은 값 ajax리턴

```
// 팔로우
@ResponseBody  doyoung *
@PostMapping(value = "follow")
public int follow(@ModelAttribute FollowVO fvo) throws Exception {
    int cnt =  userservice.follow(fvo);
    // 서비스단에서 팔로우 판단여부에 따른 true,false 값을 0,1 로 리턴받아 ajax 로 리턴
    return cnt;
}
```

Service

팔로우 여부 판단하여 추가,삭제 후 true,false 리턴

```
// 팔로우  (팔로우 삽입 , 삭제)
@Override  1 usage  doyoung *
public int follow(FollowVO fvo) throws Exception {
    if (sdao.selectFollow(fvo) == null) {// 팔로우 했던 사람이 아니면, db에 값이 없다면
        sdao.follow(fvo);// 팔로우 insert
        return 1;
    } else {
        sdao.followdel(fvo);// 값있으면 중복 x -> 삭제
        return 0;
    }
}
```

Mapper

팔로우 여부 구별, 팔로우 컬럼 추가, 삭제

```
<insert id="followinsert" parameterType="com.smartwave.tripsns.vo.FollowVO">
    insert into follow values (#{follow_id}, #{follower_id})
</insert>
<!--팔로우 조회 select 팔로우 안상태인지 여부 판단 -->
<select id="followSelect" resultType="com.smartwave.tripsns.vo.FollowVO">
    select *from follow where follow_id = #{follow_id} and follower_id = #{follower_id}
</select>
<!--팔로우 취소-->
<delete id="followdel" parameterType="com.smartwave.tripsns.vo.FollowVO">
    delete from follow where follow_id = #{follow_id} and follower_id = #{follower_id}
</delete>
```

12.기능별 주요소스

- 팔로우,언팔로우 기능(2)

Javascript:

페이지가 load 시 팔로우 유무 판단후 팔로우,언팔로우 처리 및 팔로우,팔로워수 불러오기

```
$('document').ready(function () {
    followchk();// 화면 로딩될때 팔로우 상태인지 확인 여부(팔로우수,팔로워수 함께 체크)
    // ajax 비동기 방식으로 화면 로딩 되자마자 판단 하기위한 함수
})

function followchk() {// 화면 로딩될때 팔로우 상태인지 확인 여부  Show usages  doyoung
    let followid = $('#otherid').val();
    let followerid = $('#currentid').val();
    let followbtn = document.getElementById('followlike');
    $.ajax({
        url: "/followchk",
        type: "post",
        data: {"follow_id": followid, "follower_id": followerid},
        dataType: "json",
        success: function (cnt) {
            if (cnt == 0) {
                followrCount();// 화면 로딩시
                followCount();
                followbtn.innerText = '팔로우';
            } else {
                followrCount();
                followCount();
                followbtn.innerText = '언팔로우';
            }
        }
    });
};
```

```
function followrCount() {//팔로워수(로그인한 사람이 팔로우 버튼 클릭했을시 상대 프로필 팔로워수+)
    let followid = $('#otherid').val();
    let followerscount = document.getElementById("followercnts")
    $.ajax({
        url: "/followerCount",
        type: "get",
        data: {"follow_id": followid},
        dataType: "json",
        success: function (followercnt) {
            followerscount.textContent = followercnt;
        }
    })
}
```

```
function followCount() {//팔로우수(로그인한 사람이 팔로우 버튼 클릭했을시 내 프로필 팔로워수 +)
    let followerid = $('#otherid').val();
    let followscount = document.getElementById("followcnts")
    $.ajax({
        url: "/followCount",
        type: "get",
        data: {"follower_id": followerid},
        dataType: "json",
        success: function (followcnt) {
            followscount.textContent = followcnt;
        }
    })
}
```

12.기능별 주요소스

- 회원관리자 검색

HTML:
아이디or이름 으로 조건 검색 가능

```
<form action="userSearchList" method="post">
    <select class="usearch" name="searchitem">
        <option value="id">아이디</option>
        <option value="name">이름</option>
    </select>
    <input class="usersearch" name="usersearch" placeholder="검색" type="search">
    <input class="submit" type="submit" value="검색">
</form>
```

HTML:
검색 결과를 검색결과 페이지에서 보여줌

```
<tr th:each="userList:${userList}">
    <td><input type="checkbox" name="userchk" class="uchk" th:value="${userList.id}"></td>
    <td th:text="${userList.id}"></td>
    <td th:text="${userList.name}"></td>
    <td th:text="${userList.birth}"></td>
    <td th:text="${userList.email}"></td>
    <td id="dels">
        <button class="delbtn"
                th:onclick="|location.href='@{/withdrawalManger(id=${userList.id})}'|">삭제
        </button>
    </td>
    <td>
        <button class="probtn"
                th:onclick="|location.href='@{/otherprofile(p_id=${userList.id})}'|">프로필
        </button>
    </td>
</tr>
```

Controller:
검색조건 , 검색어 파라미터 받은후 hashmap 활용 및
검색 결과 list 로 받아 리턴

```
//관리자 회원검색  > 검색페이지로 이동
@PostMapping(value = "userSearchList")  doyoung *
public String userSearchList(@SessionAttribute("userid") String u_id,
                             @RequestParam("searchitem") String usercolumn,
                             @RequestParam("usersearch") String usersearch, Model model) throws Exception {
    HashMap<String, String> userselect = new HashMap<>();
    userselect.put("usercolumn", usercolumn);
    userselect.put("usersearch", usersearch);
    List<UserVO> selectUserLists = userservice.selectUserList(userselect);
    model.addAttribute( attributeName: "userList", selectUserLists);
    //프로필 사진 불러오기
    ProfileVO prodetail = userservice.getProfile(u_id);
    model.addAttribute( attributeName: "profiledetail", prodetail);
    return "userManageSearch";
}
```

Service:
검색결과를 list 로 리턴받음

```
// 관리자용 회원 검색목록 select
@Override  1 usage  doyoung
public List<UserVO> selectUserList(HashMap<String, String> userselect) throws Exception {
    return sdao.selectUserList(userselect);
}
```

Mapper:
아이디or이름 조건검색 결과 select

```
<!--관리자 select 박스에 따른 검색 제어 id만 , name만 -->
<select id="selectUserList" parameterType="hashmap" resultType="com.smartwave.tripsns.vo.UserVO">
    select *
    from user
    where ${usercolumn} like concat('%', #{usersearch}, '%')
</select>
```

12.기능별 주요소스

- 회원관리자 삭제

Javascript:

체크박스 활용하여 전체 or 선택 삭제 가능

```
$(document).ready(function () {// 문서가 준비되었을시 매개변수로 넣은 콜백함수를 실행 {제이쿼리 이벤트 }
    $('#userAll').click(function () {
        if ($('#userAll').is(":checked")) {//.is(':checked') 를 이용하여 전체선택이 클릭되어 체크되었을때
            $("input[name =userchk]").prop("checked", true);
            // 전체선택의 리턴값이 true 면 아래의 체크박스 전부 true
        } else {
            $("input[name =userchk]").prop("checked", false);
            // 전체선택의 리턴값이 false 면 아래의 체크박스 전부 false
        }
    });
    $("input[name =userchk]").click(function () {
        var totalcnt = $("input[name =userchk]").length;// 체크박스 숫자
        var checkedcnt = $("input[name =userchk]:checked").length;// 체크박스가 체크된 숫자
        if (totalcnt != checkedcnt) {
            // 체크박스의 전체 숫자와 체크된 숫자가 다를 경우 > 한개라도 체크 해제 할시 전체선택 체크 수랑 다를경우
            $('#userAll').prop("checked", false)
            // 전체 선택의 체크된 리턴값을 false> 한개라도 체크 해제하면 전체선택 체크 해제
        } else {
            $('#userAll').prop("checked", true)
            // 전체 체크된 수와 동일 할경우 >전체 선택의 체크된 리턴값을 true
        }
    });
});
```

```
$('#chkdel').click(function () {
    //체크된박스를
    var cnt = $("input[name=userchk]:checked").length;
    if (cnt == 0) {//체크박스 유효성
        alert("선택된 글이 없습니다.");
    }
    let arr = new Array();
    let total = document.querySelectorAll('.uchk:checked');
    for (let i = 0; i < total.length; i++) {
        arr[i] = total[i].value;
        // console.log(arr[i]);
    }
    location.href = "delchk?id=" + arr;//파라미터값을 식별자 delchk로 넘김
});
```

12.기능별 주요소스

- 회원관리자 엑셀 다운

Javascript:
엑셀api 활용하여 테이블값 엑셀다운 가능

```
//엑셀다운
function s2ab(s) {  Show usages  new *
    var buf = new ArrayBuffer(s.length); //convert s to arrayBuffer
    var view = new Uint8Array(buf);
    for (var i = 0; i < s.length; i++) view[i] = s.charCodeAt(i) & 0xFF; //convert to octet
    return buf;
}

function userexportExcel() {  Show usages  new *
    // 1. workbook 생성
    var wb = XLSX.utils.book_new();
    // 2. 시트 만들기
    var newWorksheet = excelHandler2.getWorksheet2();
    // 3. workbook 에 새로만든 워크시트에 이름 주고 붙임
    XLSX.utils.book_append_sheet(wb, newWorksheet, excelHandler2.getSheetName2());
    // step 4. 엑셀 파일 만들기
    var wbout = XLSX.write(wb, {bookType: 'xlsx', type: 'binary'});
    // step 5. 엑셀 파일 내보내기
    saveAs(new Blob([s2ab(wbout)], {type: "application/octet-stream"}), excelHandler2.getExcelFileName2());
}
```

```
$(document).ready(function () {
    $('#excelFileExport').click(function () {
        userexportExcel();
    })
});
var excelHandler2 = {
    getExcelFileName2: function () {
        return '회원목록.xlsx';
    },
    getSheetName2: function () {
        return '회원목록 Sheet'
    },
    getExcelData2: function () {
        return document.getElementById('userTable')
    },
    getWorksheet2: function () {
        return XLSX.utils.table_to_sheet(this.getExcelData2())
    }
}
```

12.기능별 주요소스

-알람 기능

HTML:

하얀색 알람 과 검은색 알람 아이콘

```
<li>
    <a href="alarm">
        <div id="alarmwhite">
            <svg xmlns="http://www.w3.org/2000/svg" width="30" height="30" fill="currentColor" class="bi bi-bell" viewBox="0 0 16
        </div>
        <div id="alarmblack" style="display: none">
            <svg xmlns="http://www.w3.org/2000/svg" width="30" height="30" fill="currentColor" class="bi bi-bell-fill" viewBox="0
        </div>
    </a>
</li>
```

Javascript:

팔로우 버튼클릭시 ajax 활용하여 비동기방식으로 메시지 내용 넘김

```
$('#followlike').click(function () {//팔로우 버튼 클릭했을시 함수 호출
    let followid = $('#otherid').val();
    let followerid = $('#currentid').val();
    let followbtn = document.getElementById('followlike');
    let message;// 알림 메세지 저장
```

```
// 메세지 내용
if (followbtn.innerHTML=='언팔로우'){
    message= followerid+"님이" +followid+"님을 언팔로우 했습니다"
}else{
    message= followerid+"님이" +followid+"님을 팔로우 했습니다"
}
//Ajax 활용하여 메시지 내용 넘기기
$.ajax({
    url:"alarmcontent",
    type:"post",
    data: {"message": message,"user_id":followid,"other_id":followerid}
})
```

12.기능별 주요소스

-알람 기능(2)

Controller:

파라미터를 Ajax통해넘겨 받는 메서드, 알람처리 메서드 , 알람 체크 메서드

```
// 메세지 받기 위한 메서드
@ResponseBody  new *
@PostMapping(value = "alarmcontent")
public void alarmcontent(@ModelAttribute AlarmVO avo, Model model) throws Exception {
    userservice.alarmcontent(avo);// ajax 통한 메세지 db insert
}

// 알람 내용 select
@GetMapping(value = "alarm")  doyoung *
public String alarm(Model model, @SessionAttribute("userid") String u_id) throws Exception {
    String id = u_id;// 세션아이디
    List<AlarmVO> message = userservice.alarmlist(id);
    model.addAttribute( attributeName: "message", message);
    ProfileVO prodetail = userservice.getProfile(u_id);
    model.addAttribute( attributeName: "profiledetail", prodetail);
    return "alarm";
}

//페이지 로딩시 메세지 알림 체크
@ResponseBody  doyoung *
@PostMapping(value = "messagechk")
public int messagechk(@SessionAttribute("userid") String user_id) throws Exception {
    int chk = userservice.messagechk(user_id);
    return chk;
}
```

Javascript:

모든 페이지에 배포된 페이지 연결시 알람 체크 함수

```
<script type="text/javascript" src="/js/alarmchk.js"></script>
```

```
document.addEventListener( type: 'DOMContentLoaded', listener: function () : void {  doyoung
    alarmchk();
});
function alarmchk() : void {  Show usages  doyoung *
    let alarmwhite = $("#alarmwhite");//하얀색 알람 아이콘
    let alarmblack = $("#alarmblack");//검은색 알람 아이콘
    $.ajax( url: {// 알람체크 리턴값을 받기위한 ajax
        url:"messagechk",
        type:"post",
        dataType:"json",
        success: function (chk) : void  {
            if (chk > 0) {//알람함에 메시지가 있을경우
                alarmblack.css( name: "display",  value: "block");
                alarmwhite.css( name: "display",  value: "none");
            } else {//없을경우
                alarmblack.css( name: "display",  value: "none");
                alarmwhite.css( name: "display",  value: "block");
            }
        }
    })
}
```

12.기능별 주요소스

-알람 기능(3)

Controller:

알람 확인 클릭 시 flag 변경하여 안보이도록 처리 후 상대 프로필 화면으로 이동

```
// 상대 프로필 기본정보 html 보내기 (한줄소개 코멘트, 프로필 기본사진)
@GetMapping(value = "otherprofile")  doyoung +1 *
public String otherprofile(@RequestParam("p_id") String u_id, Model model,
                           @SessionAttribute("userid") String myid, @ModelAttribute AlarmVO avo) throws Excep
    String otheruser = u_id;
    String mainsuser = myid;
    if (mainsuser.equals(otheruser)) {// 파라미터 값이 세션아이디(로그인아이디)와 동일할경우 내프로필로 리다이렉트
        return "redirect:profile";
    } else {// 아닐경우 상대프로필 정보 포함후 프로필 화면으로 이동
        model.addAttribute( attributeName: "otheruser", otheruser);// 파라미터 로 받은 아이디를 팔로우 아이디
        model.addAttribute( attributeName: "followuser", otheruser);
        ProfileVO prodetail = userservice.getProfile(u_id);
        ProfileVO myprodetail = userservice.getProfile(myid);
        model.addAttribute( attributeName: "profiledetail", prodetail);
        model.addAttribute( attributeName: "profiledetail2", myprodetail);

        //게시글 목록
        List<PostVO> postVOList = sService.postSelectAll(u_id);
        model.addAttribute( attributeName: "postVOList", postVOList);
        //쇼츠 목록
        List<ShortVO> shortVOList = sService.userShortList(u_id);
        model.addAttribute( attributeName: "shortVOList", shortVOList);

        // 알람확인시 flag 변경 -> 안보이도록 처리 및 상대 프로필 화면으로 이동
        avo.setUser_id(myid);
        userservice.alarmchk(avo);
        return "profile";
    }

}
```

12.기능별 주요소스

-알람 기능(4)

Mapper:

알람 메시지 insert,select,update,화면로딩시 체크하는 쿼리문

```
<!--알람 메세지 insert-->
<insert id="inmessage" parameterType="com.smartwave.tripsns.vo.AlarmVO">
    insert into alarm values (nextval(alarmcnt),#{user_id},#{other_id},#{message},0,default)
</insert>
<!--알람메세지 select-->
<select id="getmessage" parameterType="String" resultType="com.smartwave.tripsns.vo.AlarmVO">
    select * from alarm where user_id =#{id} and flag = 0 order by in_date desc;
</select>
<!--알람 확인상태 flag 변경-->
<update id="alarmchk" parameterType="com.smartwave.tripsns.vo.AlarmVO">
    update alarm set flag = 1 where user_id =#{user_id} and alarm_no = #{alarm_no};
</update>
<!--화면 로딩시 알람 체크 -->
<select id="messagechk" parameterType="String" resultType="int">
    select count(*) from alarm where user_id =#{user_id} and flag= 0;
</select>
```