

파이널 프로젝트

TRIP JAVA

✓
**목차
페이지**

01

사이트 소개

02

사용 기술
및
개발 환경

03

팀원 소개

04

플로우 차트

05

ERD

06

시연

07

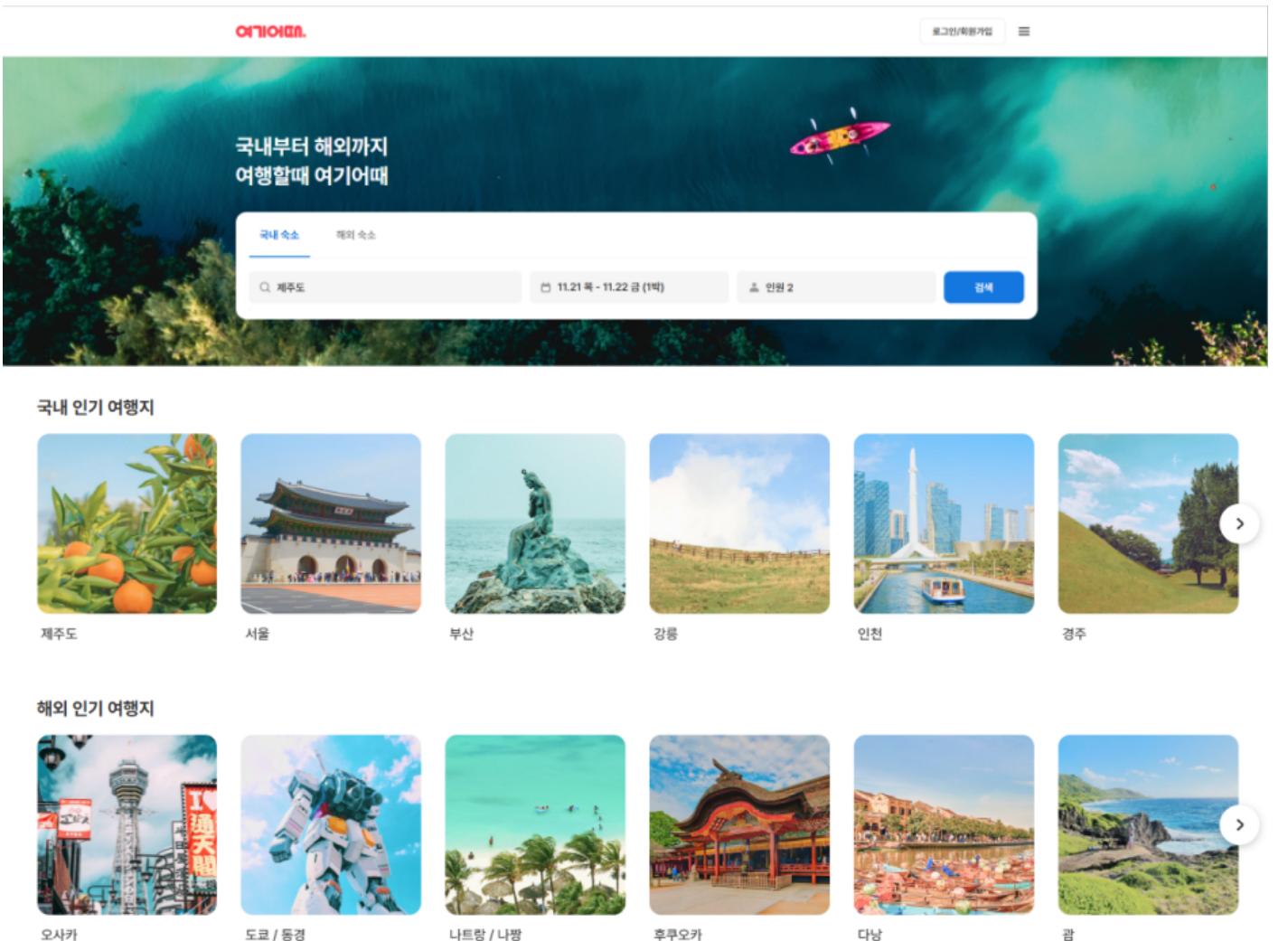
코드 리뷰

08

소감

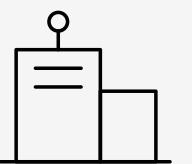
TRIP JAVA

01 | 사이트 소개



TRIP JAVA 국내 여행 숙박 예약 플랫폼

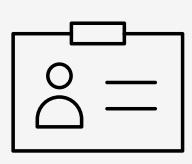
- '여기 어때' 사이트를 참조
- 국내 여행자들을 위한 숙박 예약을 도와주는 웹사이트
- 사용자 친화적인 인터페이스를 통해 다양한 숙소 검색 및 예약 가능
- 예약 및 결제 정보의 실시간 동기화



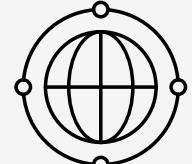
숙소 검색



숙소 예약



숙소 리뷰



소셜 로그인

02 | 사용 기술 및 개발 환경

사용 기술

Spring Boot (Java)

Spring JPA (JPQL, QueryDSL)

Oracle DB

React

개발 환경

IDE : Eclipse, VSCode

협업 도구 : GitHub

외부 API : 카카오 로그인, 네이버 지도, 토스 페이, 한국관광공사

03 | 팀원 소개

이름

역할

김창범님



ERD & 숙소 예약 및 결제 (결제 API)

박보희님



플로우 차트 & 리뷰 시스템 및 숙소 상세 페이지 (부팀장)

박선우님



ERD & 숙소 검색 및 필터링 (지도 API)

반선우님



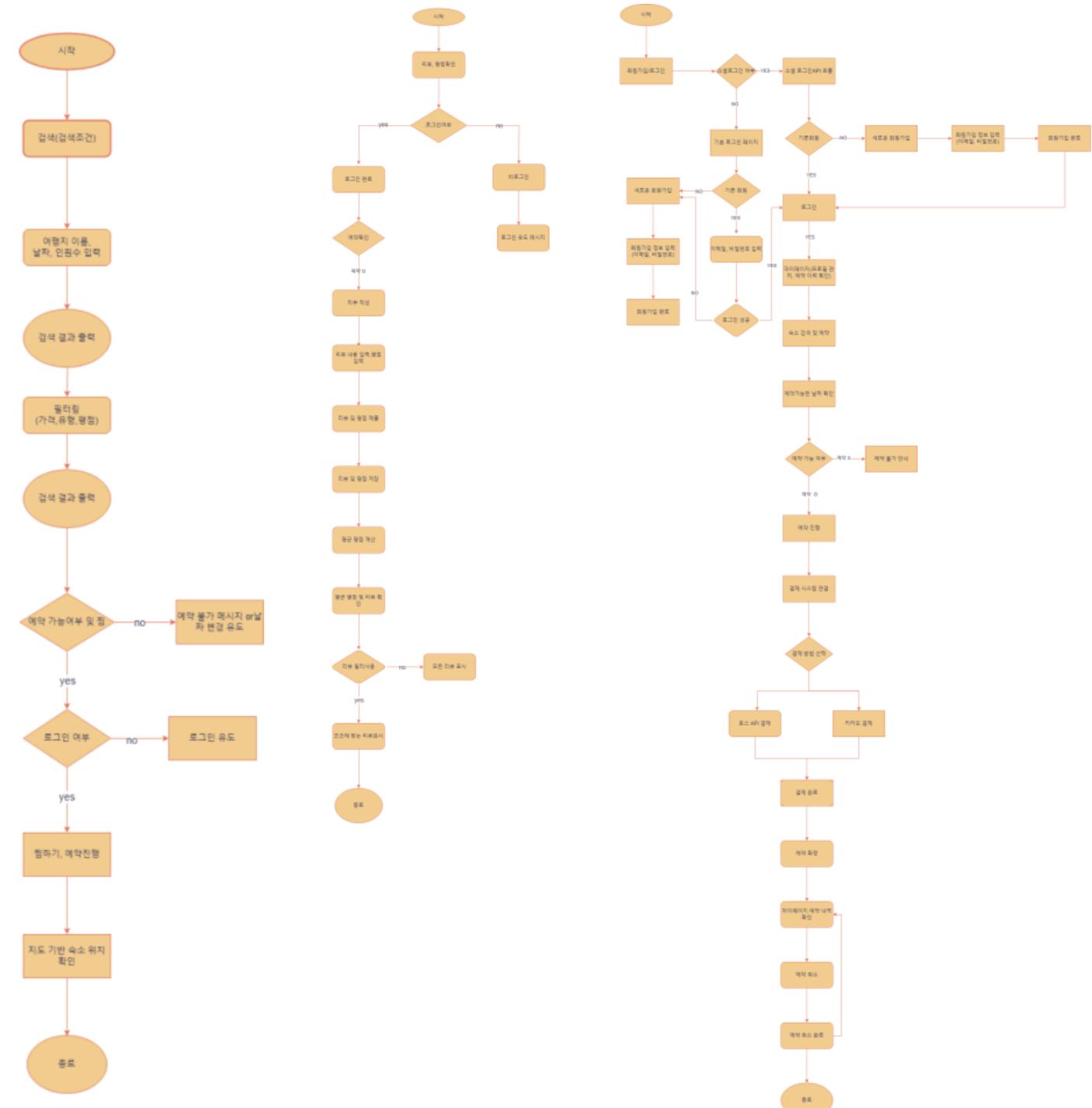
플로우 차트 & 회원가입, 로그인 및 마이페이지 (소셜 로그인)

한서진님

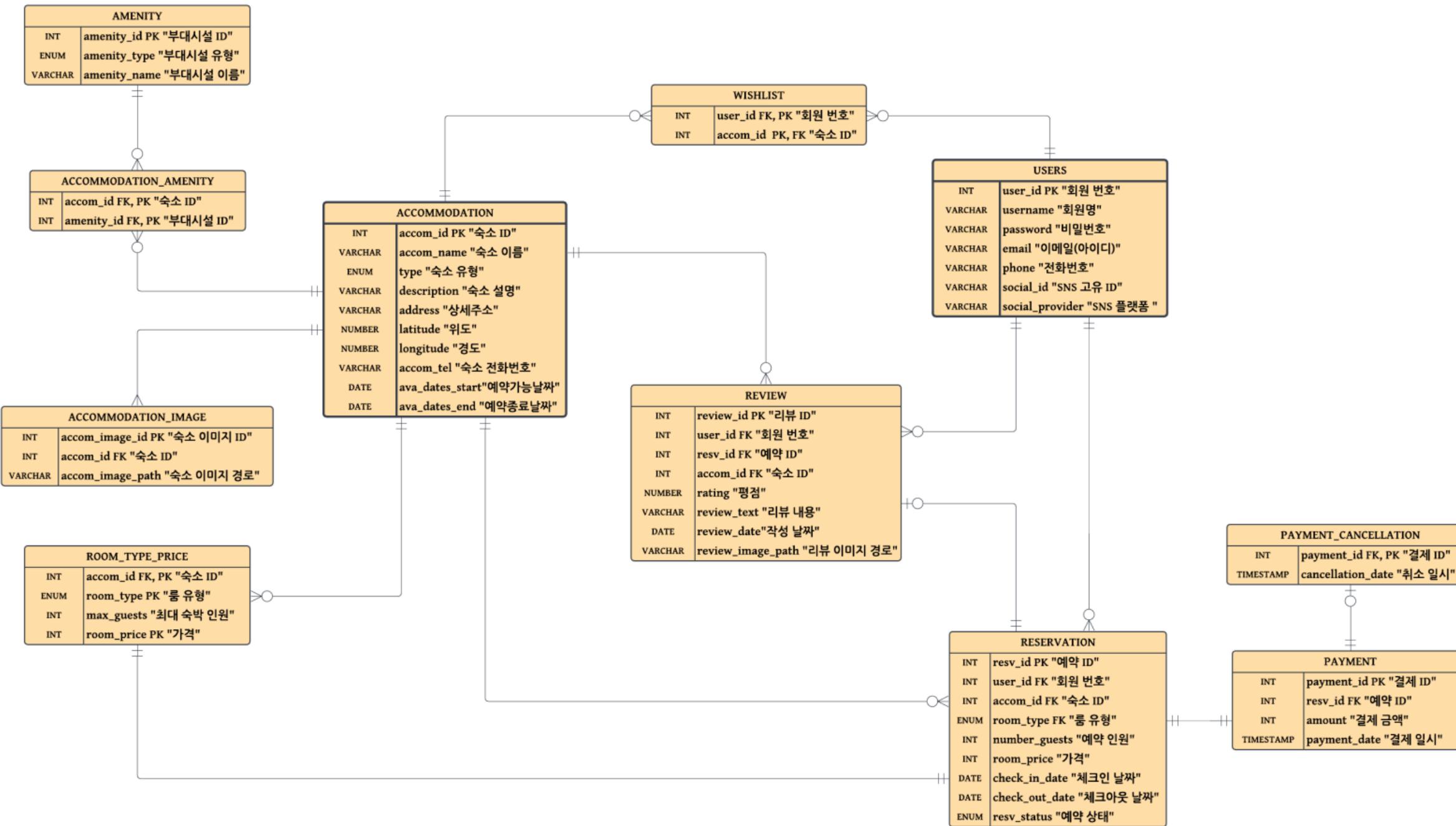


화면 디자인, 설계 총괄 및 PPT 제작 (팀장)

04 | 플로우 차트

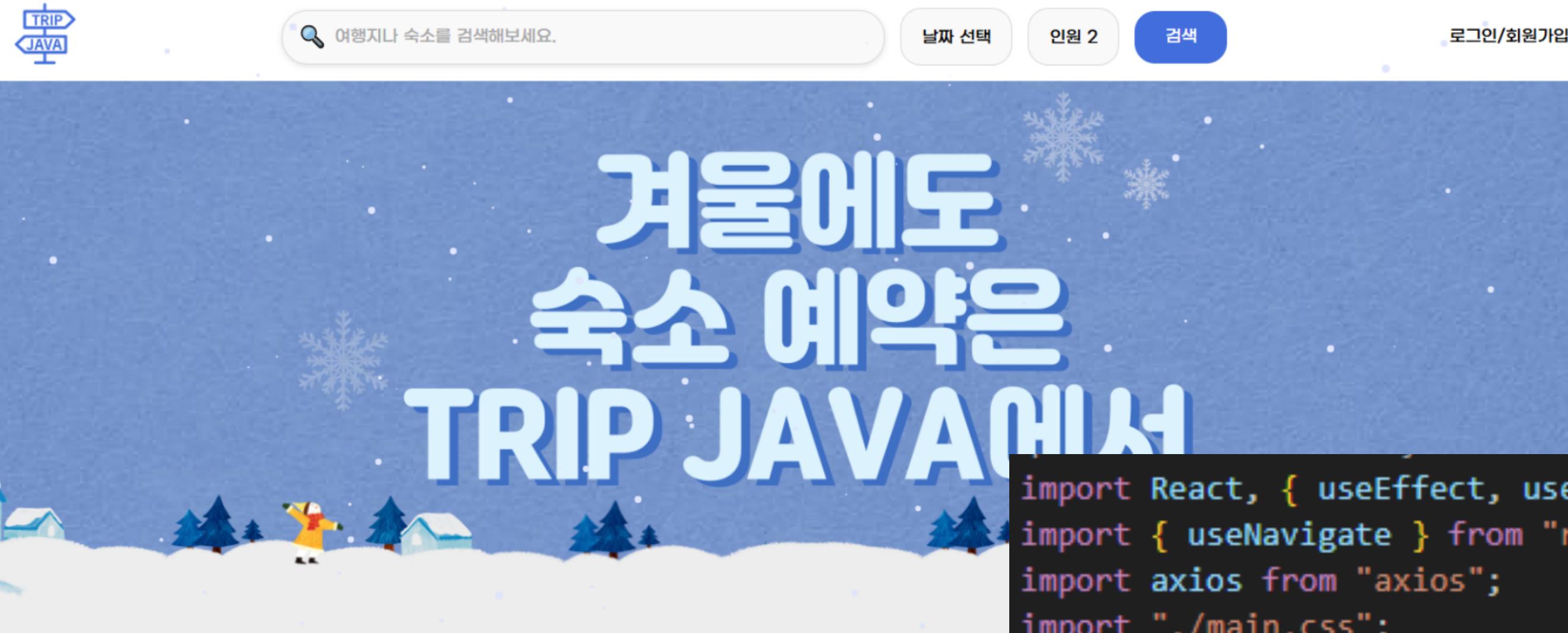


05 | ERD



06 | 시연

07 | 코드 리뷰_한서진님



The screenshot shows the homepage of the TRIP JAVA website. The header includes the logo 'TRIP JAVA' with a blue arrow icon, a search bar with the placeholder '여행지나 숙소를 검색해보세요.', and buttons for '날짜 선택', '인원 2', '검색', and '로그인/회원가입'. The main banner has a blue background with white snowflakes and features the text '겨울에도 숙소 예약은 TRIP JAVA에서'.

```
import React, { useEffect, useState } from "react";
import { useNavigate } from "react-router-dom";
import axios from "axios";
import "./main.css";
import { images } from "../../img.js";
import { Link } from 'react-router-dom';
import Snowfall from 'react-snowfall';
```

A code snippet is displayed in a dark box at the bottom right of the page, showing React imports and components used for the winter-themed landing page.

07 | 코드 리뷰_한서진님

```

function AccommodationList() {
  const location = useLocation();
  const { accomName, accomAddress, checkIn, checkOut, guests } =
    location.state || {};
  const [accommodations, setAccommodations] = useState([]);
  const [sortOption, setSortOption] = useState("주천순");
  const [isLoading, setIsLoading] = useState(true);
  const [error, setError] = useState(null);
  const [isExpanded, setIsExpanded] = useState(false);

  const [filterType, setFilterType] = useState("전체");
  const [priceRange, setPriceRange] = useState([0, 200000]);

  const handlePriceRangeChange = (values) => {
    setPriceRange(values); // 가격 범위 상태 업데이트
  };

  const handleSearch = async (searchParams) => {
    try {
      setIsLoading(true);
      console.log("검색 조건:", { ...searchParams, filterType, priceRange });

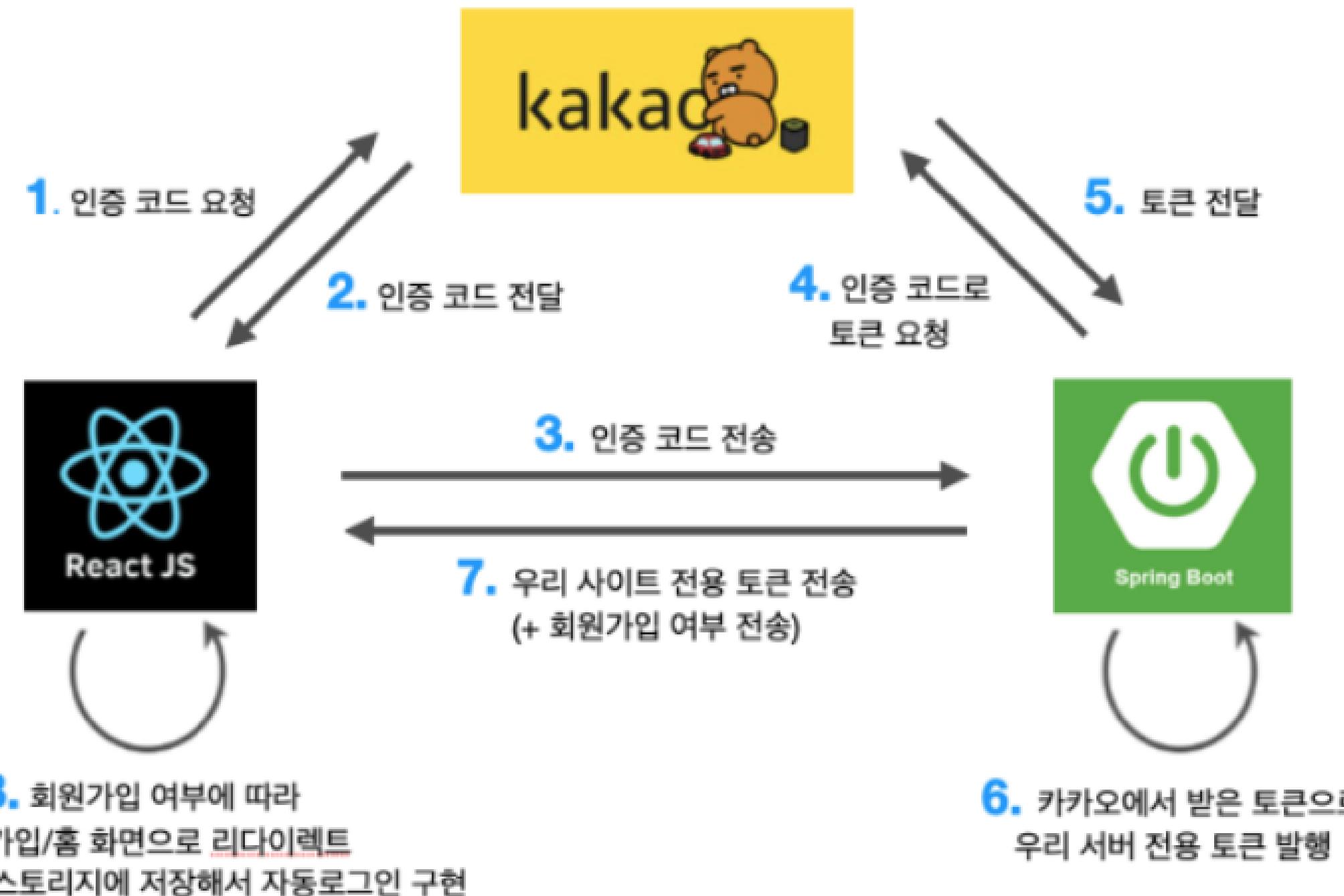
      const type =
        filterType !== "전체"
          ? {
              모텔: "MOTEL",
              호텔: "HOTEL",
              리조트: "RESORT",
              펜션: "PENSION",
              게스트하우스: "GUESTHOUSE",
            }[filterType]
          : null;

      const response = await axios.post(
        "http://localhost:9090/api/accommodations/search",
        {
          ...searchParams,
          type,
          minPrice: priceRange[0],
          maxPrice: priceRange[1],
        }
      );
      const data = response.data
        .filter(item => item.accomName.includes("제주") || item.address.includes("제주"))
        .map(item => {
          id: item.accomId,
          name: item.accomName,
          type: item.type,
          address: item.address,
          price: item.cheapestPrice,
          imageUrl: item.representativeImage
            ? item.representativeImage.split(",")[0]
            : null,
          rating: item.averageRating || "평점 없음",
        });
      setAccommodations(data);
      setIsExpanded(false);
      setIsLoading(false);
    } catch (err) {
      console.error("API 요청 실패:", err);
      setError("숙소 정보를 불러오는 데 실패했습니다.");
      setIsLoading(false);
    }
  };
}

const handleSortChange = (option) => {
  setSortOption(option);
  const sortedData = [...accommodations];
  switch (option) {
    case "평점높은순":
      sortedData.sort((a, b) => b.rating - a.rating);
      break;
    case "낮은가격순":
      sortedData.sort((a, b) => a.price - b.price);
      break;
    case "높은가격순":
      sortedData.sort((a, b) => b.price - a.price);
      break;
    default:
      break;
  }
  setAccommodations(sortedData);
};

```

07 | 코드 리뷰_반선우님



07 | 코드 리뷰_반선우님

```

@RestController
@RequiredArgsConstructor
public class SocialLoginController {

    private final KakaoService kakaoService;
    private final UserService userService;

    @GetMapping("/api/auth/kakao")
    public void kakaoLoginRedirect(HttpServletRequest response) throws IOException {
        String clientId = "6f14e0deef1d4d349512266f3dd47fc";
        String redirectUri = "http://localhost:9090/auth/kakao/callback";
        String kakaoAuthUrl = "https://kauth.kakao.com/oauth/authorize"
            + "?response_type=code"
            + "&client_id=" + clientId
            + "&redirect_uri=" + redirectUri;

        response.sendRedirect(kakaoAuthUrl);
    }

    @GetMapping("/auth/kakao/callback")
    public void kakaoLoginCallback(@RequestParam("code") String code, HttpSession session,
        HttpServletResponse response) throws IOException {
        // 카카오 서버로부터 받은 인가 코드로 토큰 요청
        String token = kakaoService.getAccessToken(code);

        // 사용자 정보 요청 및 세션 생성
        User kakaoUser = kakaoService.getUserInfo(token);

        // 사용자 정보 처리 및 DB 저장
        User savedUser = userService.processKakaouser(kakaoUser);
        session.setAttribute(SessionConst.LOGIN_MEMBER, savedUser);
        session.setAttribute("kakaoAccessToken", token); // 액세스 토큰

        // 리액트 페이지로 리다이렉트
        if (userService.isNewUser(savedUser.getUserId())) {
            response.sendRedirect("http://localhost:5173/login/regitPhone"); // 신규 사용자
        } else {
            response.sendRedirect("http://localhost:5173/"); // 기존 사용자
        }
    }
}

    /**
     * 2. Access Token을 이용해 사용자 정보 요청
     */
    public User getUserInfo(String token) {
        HttpHeaders headers = new HttpHeaders();
        headers.add("Authorization", "Bearer " + token);

        HttpEntity<String> entity = new HttpEntity<>(headers);
        try {
            ResponseEntity<Map<String, Object>> response = restTemplate.exchange(USER_INFO_URL, HttpMethod.GET, entity, Map.class);
            Map<String, Object> userInfo = response.getBody();
            Long socialId = (Long) userInfo.get("id"); // 고유 사용자 ID (socialId)

            // 사용자 정보 파싱
            Map<String, Object> kakaoAccount = (Map<String, Object>) userInfo.get("kakao_account");
            Map<String, Object> profile = (Map<String, Object>) kakaoAccount.get("profile");

            String email = (String) kakaoAccount.get("email");
            String username = (String) profile.get("nickname");

            if (socialId == null || email == null || username == null) {
                throw new IllegalArgumentException("카카오 사용자 정보가 올바르지 않습니다. (email: " + email + ", nickname: " + username + ")");
            }

            User kakaoUser = new User();
            kakaoUser.setSocialId(String.valueOf(socialId)); // socialId는 문자열로 변환
            kakaoUser.setEmail(email);
            kakaoUser.setUsername(username);

            return kakaoUser;
        } catch (Exception e) {
            throw new RuntimeException("카카오 사용자 정보 요청 실패: " + e.getMessage(), e);
        }
    }
}

```

07 | 코드 리뷰_반선우님

```
// 카카오 사용자 정보를 처리. 기존 회원 검사 후 저장하거나 기존 회원 반환.
@Transactional
public User processKakaoUser(User kakaoUser) {
    if (kakaoUser.getEmail() == null || kakaoUser.getUsername() == null) {
        throw new IllegalArgumentException("사용자 정보가 누락되었습니다. (email: " +
            kakaoUser.getEmail() + ", username: " +
            kakaoUser.getUsername() + ")");
    }

    // 1. 기존 회원 검사
    User existingUser = userRepository.findByLoginId(kakaoUser.getEmail());
    if (existingUser != null) {
        return existingUser;
    }

    // 2. 신규 회원 생성 및 저장
    User newUser = new User();
    newUser.setSocialProvider("kakao");
    newUser.setSocialId(kakaoUser.getSocialId());
    newUser.setUsername(kakaoUser.getUsername());
    newUser.setEmail(kakaoUser.getEmail());
    userRepository.save(newUser);
    return newUser;
}

// 소셜로그인 진행시, 신규 유저인지 확인
public boolean isNewUser(Integer userId) {
    System.out.println("userId : " + userId);
    User user = userRepository.findOne(userId);
    if (user.getPhone() == null) {
        return true;
    }
    return false;
}
```

06 | 코드 리뷰_박선우님

```

public List<SearchResultDTO> searchAccommodationsWithImage(SearchAccommodationRequestDTO request) {
    QAccommodation accommodation = QAccommodation.accommodation;
    QRoomTypePrice roomTypePrice = QRoomTypePrice.roomTypePrice;
    QAccommodationImage accommodationImage = QAccommodationImage.accommodationImage;
    QReview review = QReview.review;

    return queryFactory
        .select(Projections.constructor(SearchResultDTO.class,
            accommodation.accomId.as("accomId"),
            accommodation.accomName.as("accomName"),
            accommodation.type.as("type"),
            accommodation.address.as("address"),
            accommodation.avaDatesStart.as("avaDatesStart"),
            accommodation.avaDatesEnd.as("avaDatesEnd"),
            roomTypePrice.roomPrice.min().as("cheapestPrice"),
            accommodationImage.accomImagePath.min().as("representativeImage"),
            review.rating.avg().as("averageRating") // 리뷰 평점 평균 추가
        ))
        .from(accommodation)
        .leftJoin(accommodation.roomTypePrices, roomTypePrice)
        .leftJoin(accommodation.images, accommodationImage)
        .leftJoin(accommodation.reviews, review) // 숙소와 리뷰 조인
        .where(
            locationContains(request.getAccomName(), request.getAccomAddress()),
            typeMatches(accommodation, request.getType()),
            guestsLteq(roomTypePrice, request.getGuests()),
            availableDates(accommodation, request.getCheckIn(), request.getCheckOut()),
            priceRangeMatches(roomTypePrice, request.getMinPrice(), request.getMaxPrice())
        )
        .groupBy(
            accommodation.accomId,
            accommodation.accomName,
            accommodation.type,
            accommodation.address,
            accommodation.avaDatesStart,
            accommodation.avaDatesEnd,
            review.accommodation.accomId // 리뷰의 숙소 ID 추가
        )
        .fetch();
}

```

```

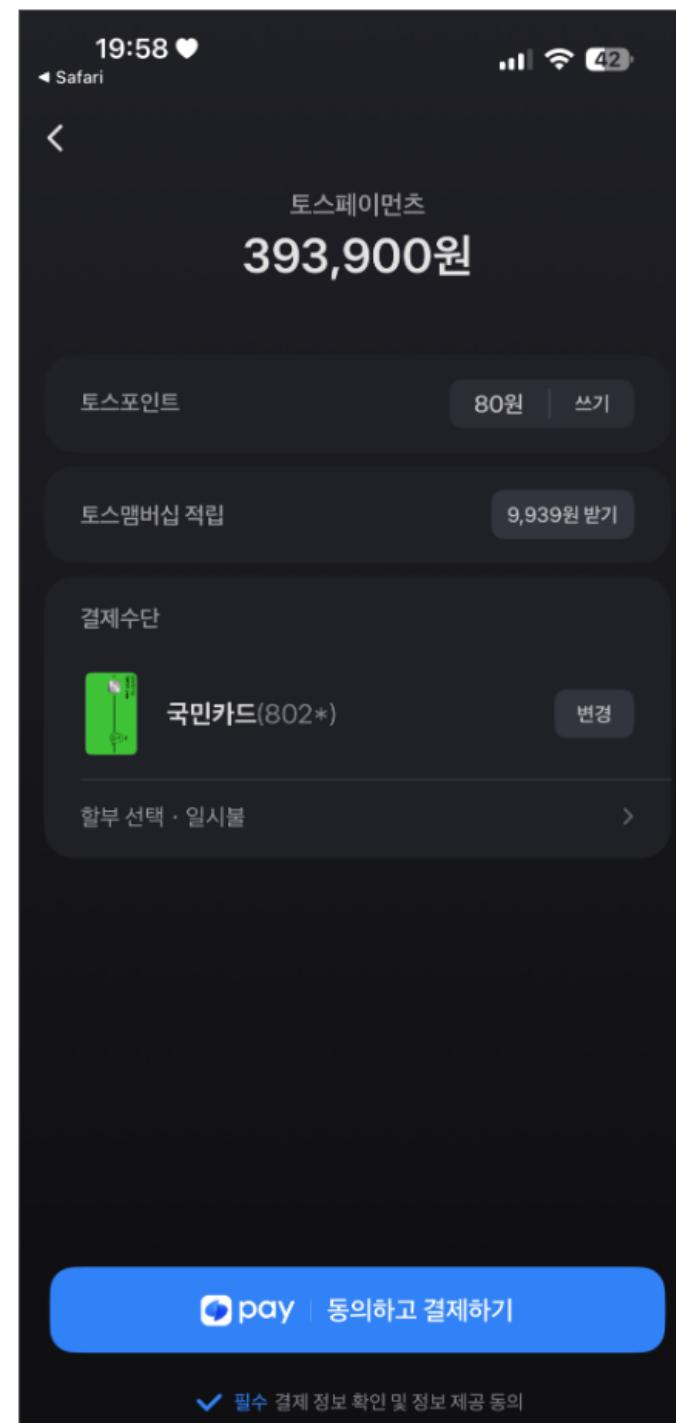
const handleFavoriteClick = async () => {
    if (!userId) { // userId 확인
        alert("로그인이 필요합니다.");
        navigate("/login"); // 로그인 페이지로 이동
        return;
    }
    console.log("userId:", userId, "accomId:", accomId);
    try {
        if (isFavorited) {
            await axios.delete(
                `http://localhost:9090/api/wishlist/${userId}/${accomId}` , // userId 사용
                { withCredentials: true }
            );
        } else {
            await axios.post(
                `http://localhost:9090/api/wishlist` ,
                { accomId },
                { withCredentials: true }
            );
        }
        setIsFavorited(!isFavorited);
    } catch (error) {
        console.error("찜 상태 변경 실패:", error);
    }
};

```

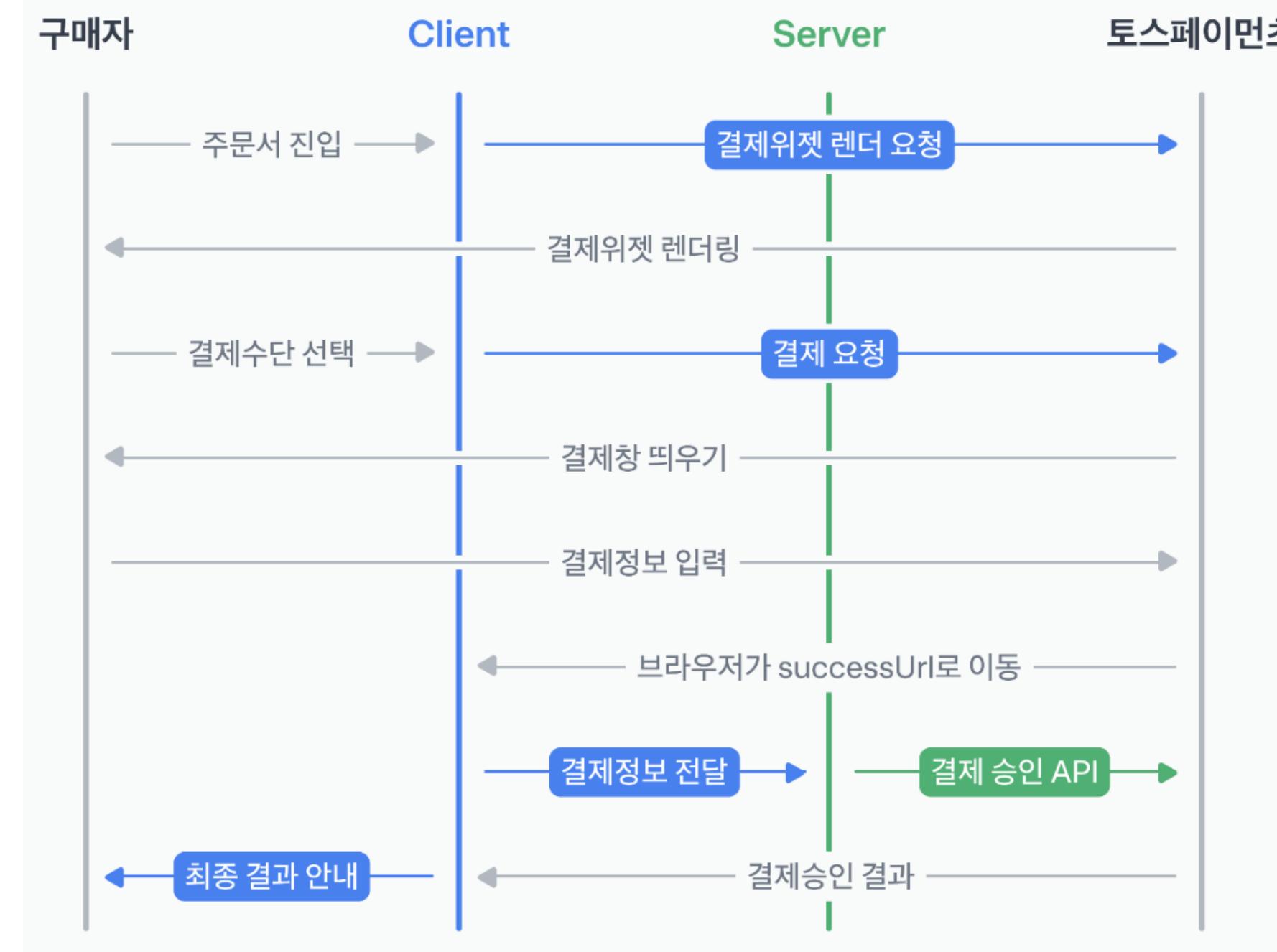
07 | 코드 리뷰_박보희님

```
0     // 리뷰 생성
1● @PostMapping
2 public ResponseEntity<?> createReview(@RequestBody ReviewCreateRequest request, HttpSession session) {
3     // 세션에서 로그인 사용자 정보 가져오기
4     User user = (User) session.getAttribute(SessionConst.LOGIN_MEMBER);
5     if (user == null) {
6         return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("로그인되지 않은 사용자입니다.");
7     }
8
9     // Review 객체 생성 및 데이터 설정
10    Review review = new Review();
11    review.setUser(user); // 세션에서 가져온 user 객체 사용
12    Accommodation accommodation = accommodationRepository.findById(request.getAccomId())
13        .orElseGet(() -> {
14        Accommodation newAccommodation = new Accommodation();
15        newAccommodation.setAccomId(request.getAccomId());
16        accommodationRepository.save(newAccommodation); // 신규 Accommodation 저장
17        return newAccommodation;
18    });
19    review.setAccommodation(accommodation); // Accommodation 객체를 Review에 설정
20    review.setRating(request.getRating());
21    review.setReviewText(request.getReviewText());
22    review.setReviewDate(new Date());
23
24    // 이미지가 제공되지 않으면 null 대신 빈 문자열로 설정
25    String reviewImagePath = request.getReviewImagePath();
26    if (reviewImagePath == null || reviewImagePath.trim().isEmpty() || reviewImagePath.equals("default-image.png"))
27        reviewImagePath = null; // null로 설정
28    review.setReviewImagePath(reviewImagePath);
29
30    // 예약 정보가 있다면, 예약 정보를 리뷰에 추가
31    if (request.getReservationId() != null) {
32        reservationRepository.findById((Integer) request.getReservationId()).ifPresent(reservation -> {
33            review.setReservation(reservation);
34        });
35    }
36
37    // 리뷰 저장
38    Review savedReview = reviewService.addReview(review);
39    return ResponseEntity.ok(new ReviewDTO(savedReview));
40 }
```

07 | 코드 리뷰_김창범님



07 | 코드 리뷰_김창범님



07 | 코드 리뷰_김창범님

```
// 토스페이 키
const clientKey = "test_gck_docs_0vk5rk1EwkEbP0W43n07x1zm";
const customerKey = generateRandomString();

useEffect(() => {
  async function fetchPaymentWidgets() {
    try {
      const tossPayments = await loadTossPayments(clientKey);
      const widgets = tossPayments.widgets({ customerKey });
      setWidgets(widgets);
    } catch (error) {
      console.error("Error fetching payment widget:", error);
    }
    fetchPaymentWidgets();
  }, []);
}

useEffect(() => {
  async function renderPaymentWidgets() {
    if (widgets == null) {
      return;
    }
    await widgets.setAmount(amount);
    await Promise.all([
      widgets.renderPaymentMethods({
        selector: "#payment-method",
        variantKey: "DEFAULT",
      }),
    ]);
    setReady(true);
  }
  renderPaymentWidgets();
}, [widgets, amount]);

```

```
const handlePayment = async () => {
  try {
    await widgets.requestPayment({
      orderId: generateRandomString(),
      orderName: sessionStorage.getItem("accomName"),
      customerEmail: "example@example.com",
      customerName: userData.username,
    });
    const reservationData = {
      user_id: parseInt(sessionStorage.getItem("userId")),
      accom_id: parseInt(sessionStorage.getItem("accomId")),
      roomType: sessionStorage.getItem("roomType"),
      roomPrice: parseInt(sessionStorage.getItem("roomPrice")),
      number_guests: parseInt(sessionStorage.getItem("guests")),
      checkInDate: sessionStorage.getItem("checkIn"),
      checkOutDate: sessionStorage.getItem("checkOut"),
    };
    console.log("Reservation Data:", reservationData);
    await axios.post(
      "http://localhost:9090/api/reservation/create",
      reservationData
    );
    navigate("/payment/success");
  } catch (error) {
    navigate("/payment/fail");
  }
};
```

Q&A

질문 있으시면 편하게 해주세요.

08 | 소감

김창범님

DB설계를 처음 하다보니 우여곡절이 많았지만, 공들여서 기초를 탄탄하게 잡은 만큼 큰 변동 없이 진행할 수 있었다. 또한 예약 및 결제 부분에서는 간편 결제 api를 채택하여 개발자에게는 더 나은 개발 경험을, 소비자에게는 더 편리한 사용자 경험을 제공하기 위해 노력했다. 마지막으로 팀원분들 모두 적극적으로 참여해주셔서 프로젝트를 무탈하게 마무리할 수 있어 감사함을 전하고 싶다.

박보희님

이번 프로젝트를 통해 프론트엔드와 백엔드의 통합 작업을 하면서 데이터 흐름과 서버 간 상호작용을 깊이 이해할 수 있었습니다. 또한, 사용자 경험을 고려한 기능 설계가 얼마나 중요한지 체감했으며, 실제로 사용자가 더 편리하게 이용할 수 있도록 고민하게 되었습니다. 마지막으로, API 연동과 비동기 통신을 활용하며 데이터가 실시간으로 처리되는 방식에 대해 심화 이해할 수 있었습니다.

박선우님

이번 프로젝트를 통해 리액트와 스프링을 활용한 웹 개발의 흐름을 체감하며, 기능 구현뿐 아니라 사용자 관점에서 설계를 고민하는 경험을 했습니다. 특히, 숙소 검색과 찜하기 기능을 구현하며 효율적인 데이터 처리와 상태 관리의 중요성을 느꼈습니다.

08 | 소감

반선우님

이번 국비 지원 과정을 마무리하며 세미프로젝트를 통해서 mvc2패턴으로 웹개발을 진행하며 어렵게만 느껴졌던 프로그래밍 코드가 한층 더 쉽게 다가왔고, 팀원들과의 소통을 어떻게 하는지 깨달을 수 있었고, 마지막 프로젝트를 진행하면서는 자바언어와 자바스크립트로 사용할 수 있는 프레임워크와 api를 사용을 해봄으로써 실제 개발 프로젝트를 간접적으로 경험할 수 있었습니다.

한서진님

새로운 문법과 프로그램을 통해 코딩하면서 저의 부족한 부분에 대해 깨닫게 되는 계기였습니다. 이를 통해 어떤 부분을 열심히 공부해야할지 깨달았습니다. 또한, 팀원들과 같이 이 코드를 리뷰해나가는 과정에서 제가 잘하는 부분은 알려주고, 모르는 부분은 배우게 되는 시간을 통해 한층 더 성장할 수 있게 된 프로젝트였습니다.

Thank you!

감사합니다.

질문 있으시면 편하게 해주세요.

KH 정보 교육원

김태근 강사님

개발 일정

2021-11-04 ~ 2024-12-24

발표 일시

2024-12-24
