1. Class Diagram

텍스트, 도표, 평면도, 개략도이(가) 표시된 사진

자동 생성된 설명

<Class Diagram>

텍스트, 도표, 평면도, 평행이(가) 표시된 사진

자동 생성된 설명

<Class Diagram - Left>

텍스트, 도표, 평면도, 개략도이(가) 표시된 사진

자동 생성된 설명

<Class Diagram – Right>

* 1. User: 일반 사용자(actor)를 나타내는 클래스로, Reviewer와 Admin을 자식 클래스로 둔 부모 클래스이다. 사용자 계정 생성, 로그인, 로그아웃, 이메일 인증, 프로필 변경 등을 수행한다.
  2. Reviewer: User의 하위 클래스로, 리뷰어(actor)를 나타내며 리뷰에 관련된 전반적인 기능을 담당한다. 리뷰 생성(작성), 리뷰 수정, 리뷰 삭제, Reviewer의 즐겨찾기 목록 반환 등을 수행한다.
  3. Admin: User의 하위 클래스로, 관리자(actor)를 나타낸다. 리뷰 신고 관리, 경고 메세지 전송, 식당 정보 관리(생성, 수정, 삭제), 정보 수정 제안 관리(승인, 거절) 등을 수행한다.
  4. Restaurant: 식당 정보를 저장하는 클래스로, 식당의 전반적인 세부 정보와 여러 Reviewer가 남긴 리뷰를 저장하고 관리한다. 식당 세부 정보 제공, 리뷰 제공, 메뉴 정보 제공 등의 기능을 수행한다.
     1. Meal: 단일 메뉴 정보를 저장하기 위한 클래스로, Restaurant의 부분 클래스이다. 식사 이름, 설명, 가격, 이미지 정보 등을 저장한다.
     2. BusinessDay: 식당 영업 정보를 저장하기 위한 Restaurant의 부분 클래스이다. 영업 날짜, 오픈 시간, 마감 시간을 저장한다.
  5. Review: Reviewer가 남긴 리뷰 정보를 저장하고 관리하는 클래스이다. reviewId, restaurantId, userId 등의 Id 정보와, 별점, comment, 작성 날짜, 이미지, 태그, 추천/비추천 수 등의 review 정보를 저장하고 관리한다. 또한 reportReview()를 통해 해당 리뷰를 신고할 수 있으며, 리뷰를 신고할 경우 그에 해당하는 Report 객체가 생성된다.
  6. Report: 신고 정보를 저장하는 클래스로, Review 클래스의 reportReview() 메소드를 통해 생성된다. 신고 Id, 해당 reviewId, 신고된 유저 Id, 신고한 유저 Id, 신고 내용(Category를 통해 선택) 등의 정보를 저장하고 관리한다.
  7. NaverMapAPI: Naver Map API라는 외부 API를 통해 User에게 지도 정보를 제공하는 클래스이다. 기본적인 지도 제공 외에도 User 실시간 위치, Marker 기능(지도에 Marker 표시, 제거, Click event) 등의 기능을 제공한다.
     1. Coordinate: NaverMapAPI와 Restaurant의 부분 클래스로, 위도, 경도 좌표를 저장하는 클래스이다.
     2. Marker: NaverMapAPI의 부분 클래스로, 지도에 마커를 표시하기 위해 사용하는 클래스이다. Coordinate를 부분 클래스로 두어 위치 정보를, label로 마커 이름을 저장한다.
  8. SearchRestaurant: 식당 검색을 위한 클래스로, 검색어를 통한 식당 검색, category를 통한 식당 검색, locationFilter를 통한 식당 검색 등의 식당 검색 기능과 검색된 식당 리스트 filter에 맞게 정렬, 지도에 마커로 표시 등의 기능을 제공한다.
  9. EditSuggestion: 식당 정보 수정 제안을 위한 클래스로, 기존 식당의 정보를 수정하거나 새로운 식당의 정보를 추가하고자 할 때 사용된다. 기존 식당 정보 가져오기, 제안 상태 설정(승인, 거절, 대기) 등의 기능을 수행한다.
  10. Favorites: User의 즐겨찾기 목록을 저장하는 클래스이다. 즐겨찾기 목록 반환, 식당 추가, 제거, 해당 식당 정보 제공 등의 기능을 수행한다.

1. Sequence Diagram
   1. Use case 1 : 검색어 기반 식당 검색

텍스트, 도표, 평행, 라인이(가) 표시된 사진

자동 생성된 설명

Use case 진행 과정:

* + 1. User(actor)가 GUI에게 검색 요청을 한다.
    2. GUI는 SearchRestaurant의 인스턴스의 searchByKeyword(keyword) 메소드를 호출해 키워드에 맞는 식당 리스트를 반환할 것을 요청한다.
    3. SearchRestaurant의 인스턴스(이하 SearchRestaurant)는 Restaurant 인스턴스 모임의 getDetails() 메서드를 호출하고, 각 인스턴스에게 restaurant(Restaurant의 인스턴스)를 반환 받는다.
    4. SearchRestaurant는 GUI에게 restaurantList(List<Restaurant)를 반환한다.
    5. GUI는 SearchRestaurant의 displayRestaurantsOnMap(restaurantList, APIKey) 메소드를 호출해 식당들을 마커로 표시할 것을 요청한다.
    6. SearchRestaurant는 NaverMapAPI의 인스턴스 MapAPI1의clearMarkers()(기존 마커 초기화), addMarkers(restaurantMarkers)(새로운 식당 마커 추가) 메소드를 호출한다.
    7. 그 후 filter(정문, 쪽문, 후문 중 선택할 수 있는 List<Enum>)가 NULL인지 여부에 따라, NULL일 경우 저장된 마커를 모두 표시하는 displayRestaurantMarkers() 메소드를, NULL이 아닐 경우 저장된 마커 중 filter에 부합하는 마커만 표시하는 displayRestaurantMarkers(filter) 메소드를 호출한다.
    8. 마지막으로, 지도의 중심 위치를 표시한 마커들의 중간으로 맞추는 setMapCenter(centerLocation) 메소드를 호출한다.
    9. GUI는 User에게 결과를 보여주고 UseCase를 마친다.
  1. Use case 2 : 리뷰 열람 및 상호작용

텍스트, 도표, 평행, 평면도이(가) 표시된 사진

자동 생성된 설명

Use case 진행 과정:

* + 1. User가 GUI에게 특정 식당(aRestaurant)의 리뷰 목록을 요청한다.
    2. GUI는 NaverMapAPI의 인스턴스 MapAPI1에게 해당 식당의 마커를 클릭하는 clickMarker(marker) 메소드를 호출한다. MapAPI1은 반환값으로 해당 식당을 나타내는 인스턴스 restaurant를 반환한다.
    3. GUI는 aRestaurant의 getReviews() 메소드를 호출한다.
    4. aRestaurant는 자신이 소유한 Review 인스턴스 모임에게 getReview() 메소드를 호출한다. Review 인스턴스 모임은 review 인스턴스를 반환한다.
    5. aRestaurant는 getReviews()의 반환값으로 자신이 소유한 Review 인스턴스들의 모임 reviewList(List<Review>)를 반환한다.
    6. GUI는 User에게 해당 식당의 리뷰 목록을 제시한다.
    7. User는 리뷰 목록 중 특정 리뷰에 ‘싫어요’를 누른다.
    8. GUI는 review1(해당 리뷰)의 nonRecommendReview(userId) 메소드를 호출한다.
    9. review1의 nonRecommendCount가 1 증가하고, GUI는 이를 사용자의 화면에 피드백한다.
    10. User는 해당 리뷰를 신고한다. 이때 신고 내용 역시 포함된다(e.g., 욕설).
    11. GUI는 review1의 reportReview(reportingUserId, reportDetails) 메소드를 호출한다.
    12. review1은 해당 리뷰의 신고 인스턴스인 report1ofReview1 인스턴스(Report 클래스의 인스턴스)를 생성한다.
    13. GUI는 사용자에게 완료 메시지를 표시한다.
  1. Use case 3 : 다른 사용자의 즐겨찾기 목록 열람

텍스트, 도표, 평행, 라인이(가) 표시된 사진

자동 생성된 설명

Use case 진행 과정:

* + 1. User는 GUI에게 특정 리뷰어의 즐겨찾기 목록에 접근할 것을 요청한다.
    2. GUI는 review1(해당 리뷰어의 리뷰)의 getUserId() 메소를 호출해 userId를 반환 받는다.
    3. GUI는 해당 userId를 갖는 reviewer1(Reviewer의 인스턴스)의 getFavoritesByReviewer(userId) 메소드를 호출해 해당 리뷰어의 즐겨찾기 목록 favorites1(Favorites의 인스턴스)를 반환 받는다.
    4. GUI는 요청 리뷰어의 즐겨찾기 목록을 User에게 제공한다.
    5. User는 GUI에게 즐겨찾기 목록 중 특정 식당의 정보를 요청한다.
    6. GUI는 favorites1의 accessRestaurantByRestaurantId(restaurantId) 메소드를 호출해 해당 식당 정보를 요청한다.
    7. favorites1은 aRestaurant(해당 식당 인스턴스)의 getDetails() 메소드를 호출해 식당 정보 restaurant를 반환 받는다.
    8. favorites1은 GUI에게 restaurant를 반환한다.
    9. GUI는 User에게 요청 식당 정보를 제공한다.

수정요청사항:

1. UC001 🡪 검색어 기반 식당 검색: ‘또는 위치를 기준으로’ 삭제할 수 있나요. 이 기능은 하자고 안 한 것 같아서..
2. 유스케이스 명세 🡪 정보 수정 제안하기: 신규 식당 정보 제안 내용이 빠졌어요. 그리고 관리자가 정보 수정을 관리하기 편하게 승인, 거절, 대기 상태를 만들어서 관리하는 것은 어떨까요? 승인을 누르면 식당 데이터베이스에 추가되고, 거절을 누르면 말 그대로 거절되는 거고, 대기는 아직 처리되지 않은 상태에요.