
fAlshion

AI 기반 체형 맞춤 가상 피팅 플랫폼

팀장 : 박세원

팀원 : 권택준, 유부미, 이현호

1. 프로젝트 개요
2. 프로젝트 팀 구성 및 역할
3. 프로젝트 수행 절차 및 방법
4. 프로젝트 시연 및 기능소개
5. 비즈니스 및 마케팅 전략
6. 자체 평가 및 개인 의견
7. Q&A

산업 / 생활경제

"화면과 다른데 왜 환불 안해줘?"...'비대면 소비' 늘자 분쟁도 폭증

업데이트 2020.04.06 오전 07:50 ▾

'상세설명과도 다른데'...온라인·모바일거래 상담 지난해 대비 폭증
온라인 쇼핑 특성상 하자여부·책임소재 판단 힘들어



비대면 상품 구매 피해자 증가

AI 기반 체형 맞춤 가상 피팅 플랫폼

현대 사회에서 패스트 패션의 확산과 온라인 쇼핑의 활성화는 의류 소비를 급격히 증가시켰다.

그러나 이 과정에서 사이즈 불일치로 인한 반품과 빠른 유행 변화로 인한 의류 폐기가 심각한 문제로 대두되고 있다.

특히 온라인 쇼핑은 직접 입어볼 수 없다는 특성상 불필요한 구매와 잦은 반품이 발생하며, 이로 인해 소비자는 만족스럽지 못한 쇼핑 경험을 하고 기업과 사회는 자원 낭비와 환경 부담을 떠안게 된다.

프로젝트 문제정의 및 목표

프로젝트 개요

문제 정의

- 온라인 의류 쇼핑에서 사이즈 불일치와 유행 변화로 인한 반품·폐기가 증가하고 있음
- 이러한 반품 과정은 탄소 배출과 자원 낭비로 이어져 환경 오염을 유발함
- 소비자는 만족스럽지 못한 쇼핑 경험을 하고, 기업은 비용 부담이 가중됨
- 보다 정확하고 개인화된 쇼핑 경험을 제공할 수 있는 기술적 해결책이 필요함

목표

- AI 기반 가상 피팅 기술을 통해 소비자가 의류를 착용한 듯한 경험 제공
- 사이즈 불일치로 인한 반품률 감소 및 고객 구매 만족도 향상
- 개인의 체형·스타일·날씨 등을 고려한 개인 맞춤형 추천 시스템 제공
- 불필요한 반품과 폐기 감소를 통해 지속 가능한 소비 환경 조성

라이프 스타일

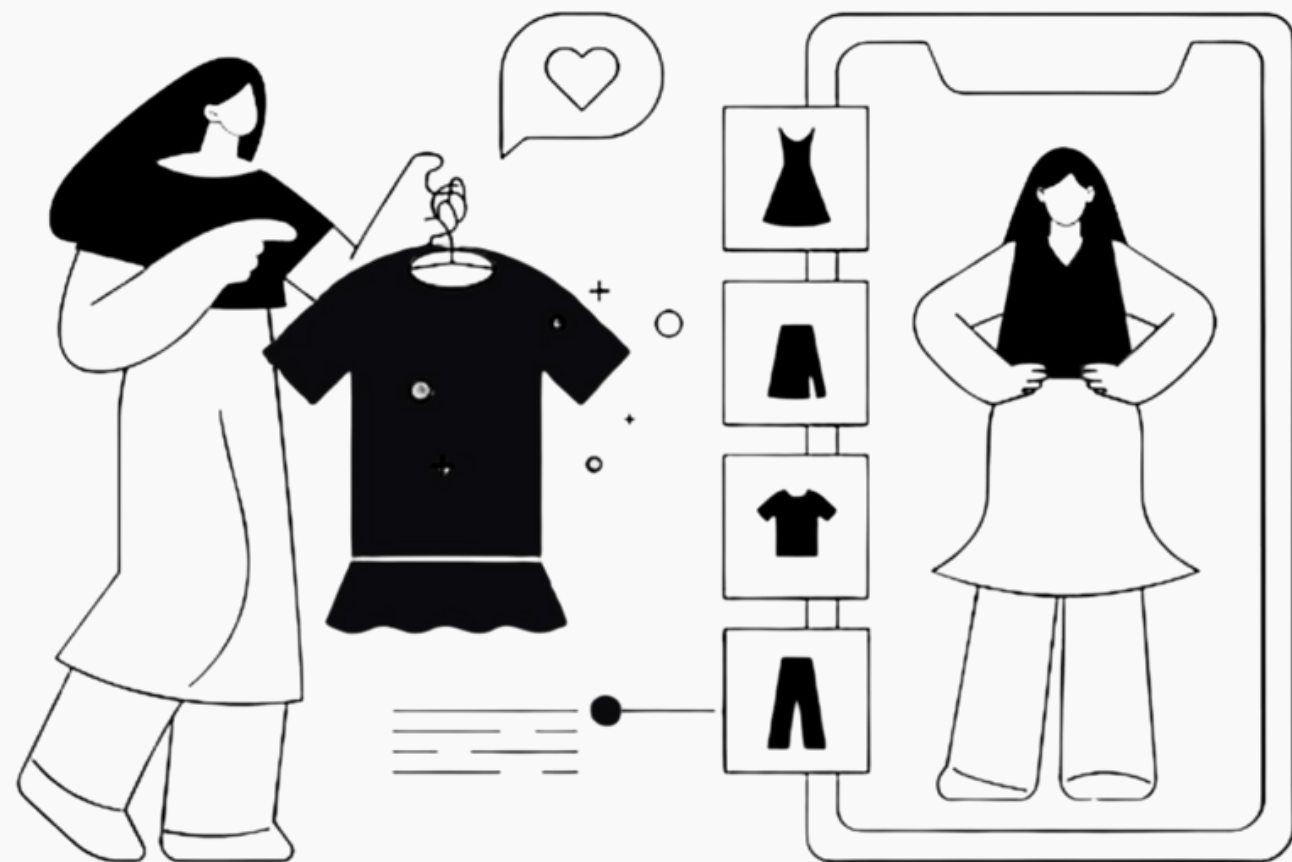
- 평일엔 회사 출근, 주말에는 카페나 전시회 등 외출을 즐김
- 유행에 민감하고, SNS(인스타그램, 유튜브)에서 스타일 정보를 자주 탐색
- 옷은 주로 무신사, 29CM, W Concept, 쿠팡 스타일 등에서 온라인 구매

문제점

- 쇼핑몰마다 사이즈 기준이 달라 매번 헛갈림
- 반품 시 번거롭고, 환경에 좋지 않다는 죄책감도 느낌
- 자신에게 잘 맞는 옷을 찾기까지 시간이 오래 걸림

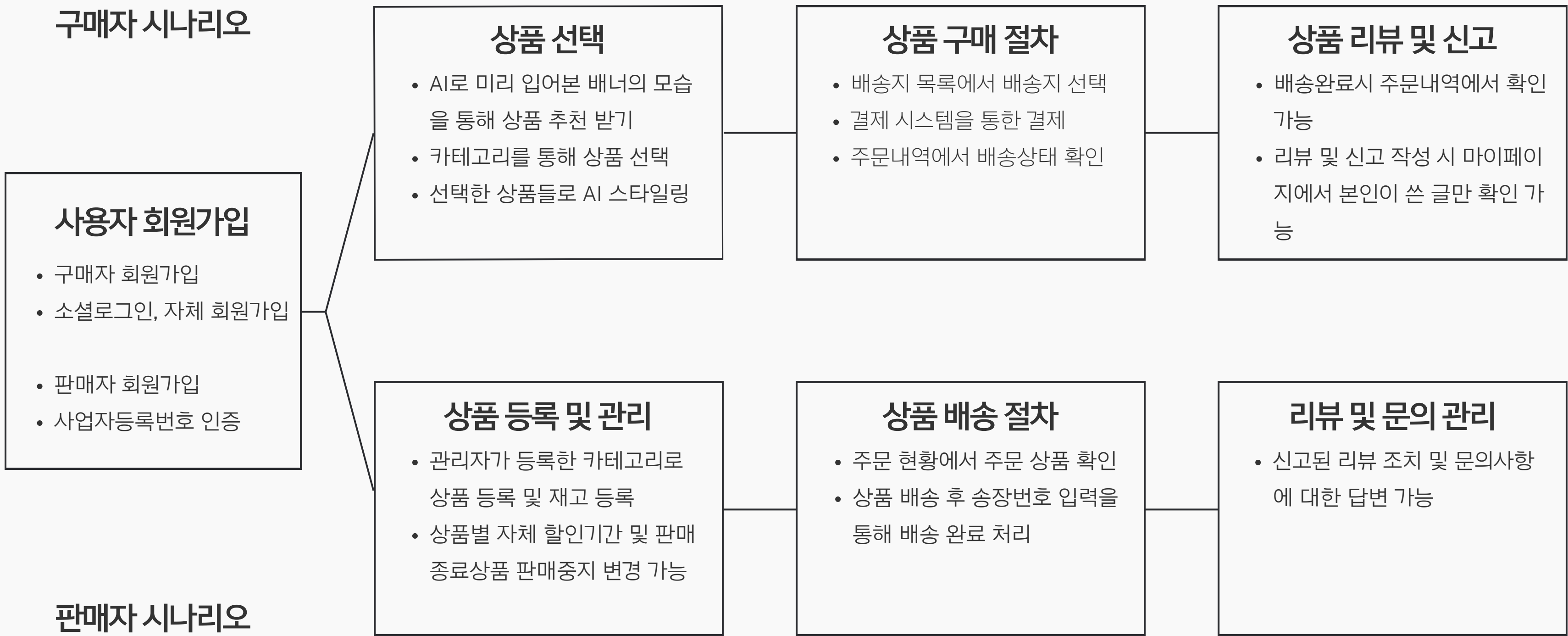
필요와 목표

- 구매 전 가상 피팅을 통해 핏을 미리 확인하고 싶음
- 실패 없는 쇼핑 경험을 원함
- 환경에 부담을 덜 주는 소비를 하고 싶음



사용자 시나리오

프로젝트 개요



프로젝트 팀 구성 및 역할

프로젝트 팀 구성 및 역할

팀장 / 기능, 디자인 담당
박세원

- 전체 일정 및 디자인 틀 구성
- 마이페이지 구현
- 장바구니 페이지 구현
- 결제 기능 구현
 - 토스페이먼츠API
- 주문서, 배송 내역 구현
- QnA·공지사항 게시판

운영 및 관리 기능 담당
권택준

- 자체 배송 시스템 구현
- 재고 관리 기능 구현
- 판매자, 관리자 페이지 구현
- AI 배너 상품추천 구현
 - Gemini 이미지생성 API
- DB 테이블 설계 및 관리

로그인 및 보안 담당
유부미

- 회원가입 및 로그인 구현
 - 소셜로그인(네이버) 구현
- 사업자 등록 확인 기능 구현
 - 상태조회 서비스API
- 스프링 시큐리티를 통한 권한 제어 (판매자, 구매자, 운영자)

AI 스타일링 / 기능, 디자인 담당
이현호

- AI 활용한 간접 피팅 기능 구현
 - Gemini 이미지생성 API
- 카테고리 별 상품 목록 조회
- 상품 상세보기 페이지 구현
 - 리뷰, 신고, 문의
- 마이페이지 상세기능 구현
 - 배송지관리, 우편번호 API
 - 사진등록 및 개인

기술 스택

프로젝트 수행 절차 및 방법

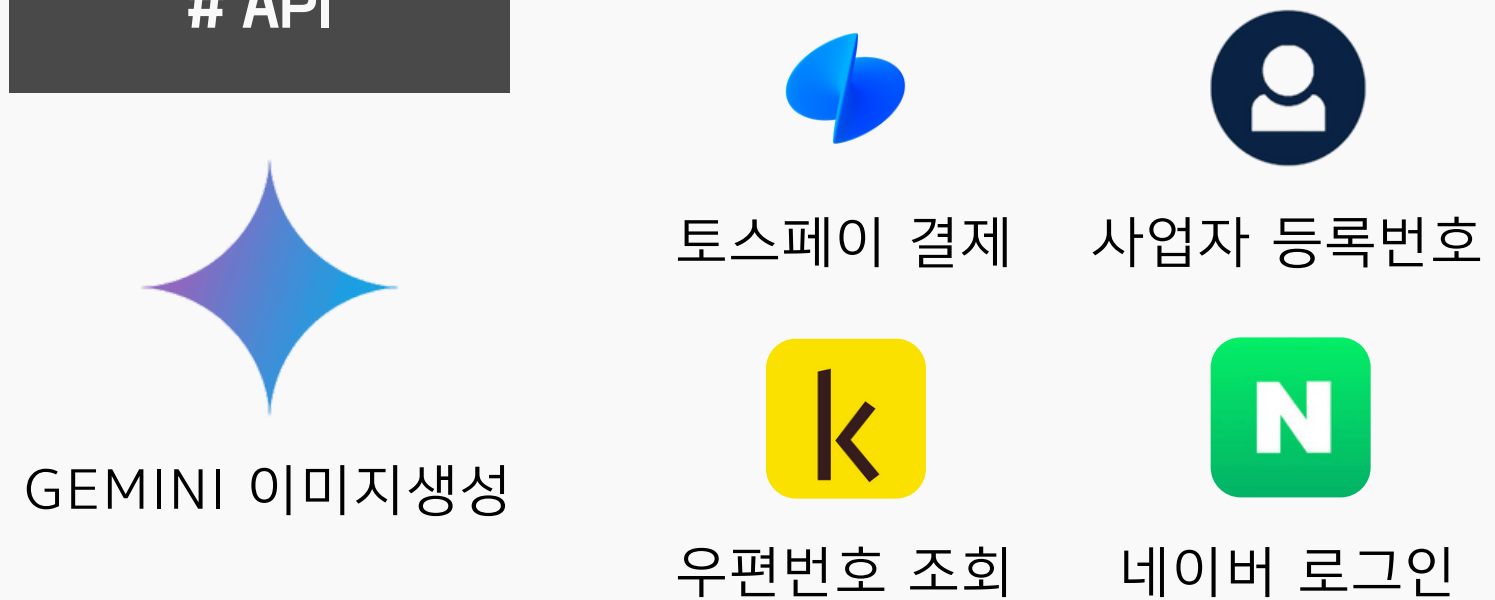
FRONT



BACK



API



형상관리 및 배포



기능 소개(영상)

프로젝트 시연 및 기능소개

중복검사

사용 가능한 아이디입니다.

사용 가능한 비밀번호입니다.

비밀번호 확인

비밀번호 확인을 입력해주세요.

이메일 아이디

@

naver.com

중복검사

회원가입

정규화

아이디 중복여부 확인

- 이미 등록된 아이디 사용불가

비밀번호 정규화

- 비밀번호 8자이상(문자, 숫자, 특수문자 포함)

- 비밀번호 일치여부 확인

이메일 중복여부 확인

- 이미 등록된 이메일 사용불가

전화번호 정규화

- 형식에 맞지 않는 전화번호 불가

(주) fAlshion

AI 기반 버주얼 트라이온 멀티 벤더 쇼핑몰 fAlshion 입니다. 편리하고 즐거운 쇼핑 경험을 제공하겠습니다.

☎ 1644-0000 평일 10:00~17:00 (점심 12:40~13:50)

✉ help@faishion.co.kr

📍 서울특별시 관악구 봉천로 227, 5F



빠른메뉴

공지사항

Q&A

주문/배송조회

교환/반품 안내

판매자 입점문의

입금계좌 안내

국민 123456-01-000000 (☎fAlshion)

신한 110-000-000000 (☎fAlshion)

농협 302-0000-0000-00 (☎fAlshion)

안전거래를 위해 현금 결제 시 구매안전(에스크로) 서비스를 이용하실 수 있습니다. [에스크로 확인](#)

회사 정보

대표자명 : 권택준 · 박세원 · 이현호

사업자등록번호 : 211-88-00000

[사업자정보확인](#)

통신판매업신고 : 제20241010호

개인정보보호책임자: 유부미

[이용약관](#) | [개인정보처리방침](#) | [이용안내](#)

Spring Security + JWT 토큰 기반 사용자 인증

프로젝트 시연 및 기능소개

기능 소개

JWT 액세스 토큰

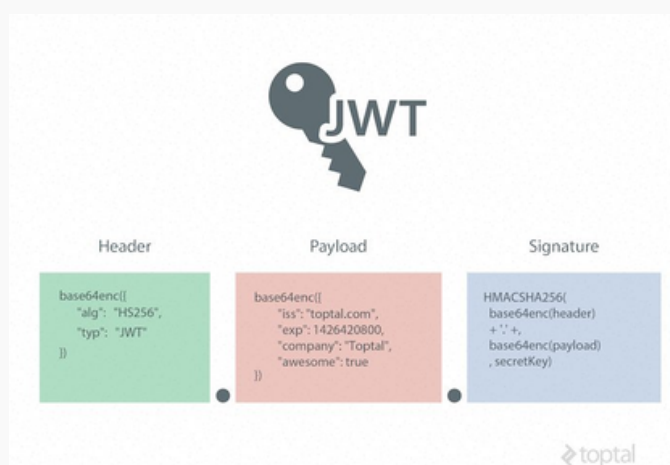
- 실제 API 접근에 사용되는 액세스 토큰(ACCESS TOKEN)은 짧은 만료 시간을 설정하여, 혹시라도 탈취당하더라도 공격 가능 시간을 최소화

JWT 리프레시 토큰

- 장기적인 로그인 유지를 위한 리프레시 토큰(REFRESH TOKEN)을 클라이언트 JAVASCRIPT가 접근 불가능한 HTTPONLY 쿠키에 저장

필터 활용 권한 통제

- SPRING SECURITY FILTER CHAIN을 활용하여, 모든 요청이 핵심 로직에 도달하기 직전에 커스텀 필터가 액세스 토큰의 유효성과 권한을 가장 먼저 검증



주요 코드

```
private final JwtTokenProvider jwt;
private final UserRepository userRepository;
private final SellerRepository sellerRepository;
private final AdminRepository adminRepository;

@Override @SuppressWarnings("rawtypes")
protected void doFilterInternal(HttpServletRequest req, HttpServletResponse res, FilterChain chain)
    throws ServletException, IOException {
    String header = req.getHeader("Authorization");
    if (header != null && header.startsWith("Bearer ")) {
        String accessToken = header.substring(7);
        try {
            authenticateUser(jwt.parse(accessToken));
        } catch (ExpiredJwtException e) {
            // 액세스 토큰 만료 시 리프레시 토큰으로 재발급 시도
            handleRefreshToken(req, res);
        } catch (Exception e) {
            // 기타 토큰 관련 예외 처리
            SecurityContextHolder.clearContext();
        }
    }
}

private void authenticateUser(Jws<Claims> jws) {
    Claims claims = jws.getBody();
    String subject = claims.getSubject();
    /unchecked/
    List<SimpleGrantedAuthority> authorities = ((List<String>)claims.get("roles")).stream().map(String::trim).map(role -> new SimpleGrantedAuthority("ROLE_" + role)).collect(toList());
    UserDetails userDetails = new User(subject, "", authorities);
    UsernamePasswordAuthenticationToken auth = new UsernamePasswordAuthenticationToken(userDetails, null, authorities);
    SecurityContextHolder.getContext().setAuthentication(auth);
}

private void handleRefreshToken(HttpServletRequest req, HttpServletResponse res) {
    String refreshToken = extractRefreshTokenFromCookie(req);
    if (refreshToken != null) {
        try {
            // 리프레시 토큰 유효성 검증
            Jws<Claims> refreshJws = jwt.parse(refreshToken);
            Claims claims = refreshJws.getBody();
            String subject = claims.getSubject();
        }
    }
}
```

Gemini API를 활용한 사용자 맞춤 AI 스타일링

프로젝트 시연 및 기능소개

기능 소개

이미지 생성

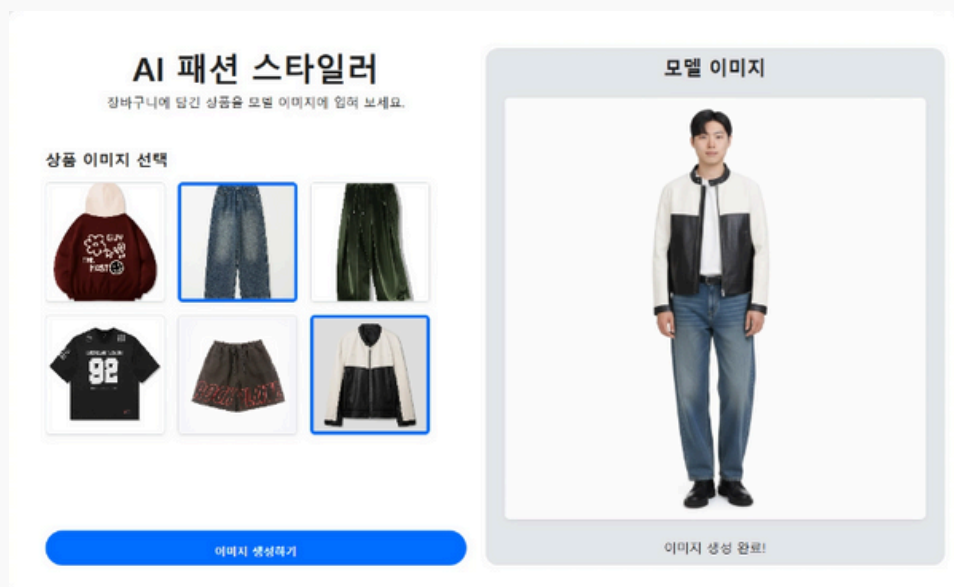
- 사용자의 사진등록 및 키, 몸무게 입력을 통해 AI스타일링에 적합한 이미지 생성

상품 추천

- 배너를 통해 추천상품을 미리 입어보는 시각적 효과 제공

AI 스타일러

- 장바구니에서 선택한 상품을 가지고 미리 입어보는 서비스 제공
- 상품 상세보기에서 바로 이동 가능



주요 코드

```
// 모델 이미지 추가
JsonObject inlineModelData = new JsonObject();
inlineModelData.addProperty( property: "mimeType", value: "image/png");
inlineModelData.addProperty( property: "data", base64ModelImage);
JsonObject modelPart = new JsonObject();
modelPart.add( property: "inlineData", inlineModelData);
partsArray.add(modelPart);

// 프롬프트 텍스트 추가
JsonObject promptPart = new JsonObject();
promptPart.addProperty( property: "text", prompt);
partsArray.add(promptPart);

JsonObject contentObject = new JsonObject();
contentObject.add( property: "parts", partsArray);
JsonArray contentsArray = new JsonArray();
contentsArray.add(contentObject);

JsonObject generationConfig = new JsonObject();
JsonArray responseModalities = new JsonArray();
generationConfig.addProperty( property: "temperature", value: 0); // Ai 창의성 레벨 0~1
responseModalities.add("IMAGE");
generationConfig.add( property: "responseModalities", responseModalities);

JsonObject mainPayload = new JsonObject();
mainPayload.add( property: "contents", contentsArray);
mainPayload.add( property: "generationConfig", generationConfig);

String payloadString = mainPayload.toString();

Request request = new Request.Builder()
    .url("https://generativelanguage.googleapis.com/v1beta/models/gemini-2.5-flash-image-preview:generateImage")
    .post(okhttp3.RequestBody.create(payloadString, MediaType.get("application/json; charset=utf-8")))
    .build();

try (Response response = client.newCall(request).execute()) {
```


PG(Payment Gateway) API를 활용한 결제 처리

프로젝트 시연 및 기능소개

기능 소개

다양한 결제 수단 통합 제공

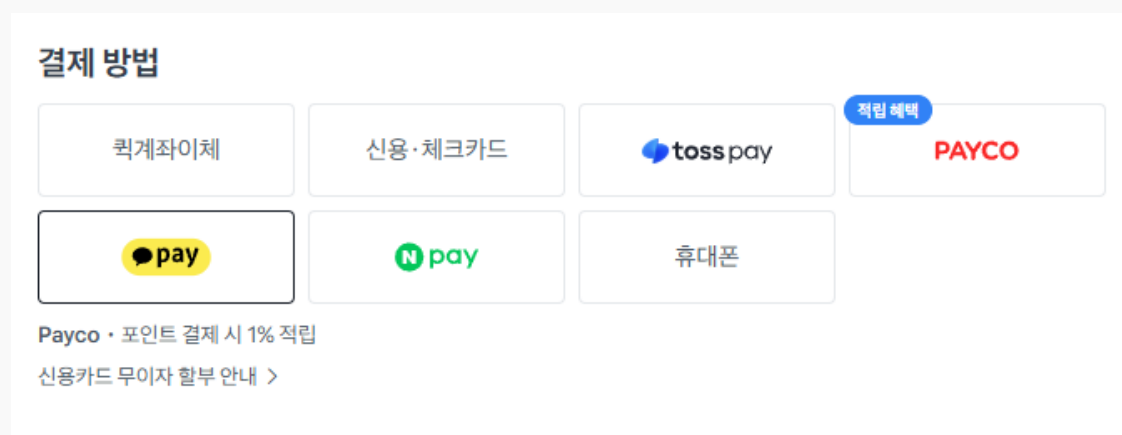
- 신용/체크카드, 계좌 이체, 가상 계좌, 휴대폰 결제, 그리고 카카오페이/네이버페이 등 주요 간편 결제까지 PG사가 제공하는 단일 API를 통해 모두 연동

최고 수준의 보안 및 법적 안정성 확보

- 고객의 민감한 결제 정보(카드번호 등)를 직접 처리하지 않아 보안 리스크를 PG사로 이관하고, 법적/기술적 안정성을 확보

주문-결제-정산 자동화

- 주문 시스템과 연동된 결제 완료 후 복잡한 정산 과정을 자동화하여 운영 효율을 극대화



주요 코드

```
// --- 2) Toss 승인 호출 ---
JSONObject tossReq = new JSONObject();
tossReq.put("orderId", orderId);
tossReq.put("amount", amount);
tossReq.put("paymentKey", paymentKey);

String widgetTestKey = "test_gsk_docs_0aPz8L5KdmQXkzRz3y47BMw6";
String auth = "Basic " + Base64.getEncoder()
    .encodeToString((widgetTestKey + ":").getBytes(StandardCharsets.UTF_8));

URL url = new URL( spec: "https://api.tosspayments.com/v1/payments/confirm");
URLConnection conn = (URLConnection) url.openConnection();
conn.setRequestProperty("Authorization", auth);
conn.setRequestProperty("Content-Type", "application/json");
conn.setRequestMethod("POST");
conn.setDoOutput(true);

try (OutputStream os = conn.getOutputStream()) {
    os.write(tossReq.toString().getBytes(StandardCharsets.UTF_8));
}

int code = conn.getResponseCode();
InputStream is = (code == 200) ? conn.getInputStream() : conn.getErrorStream();
JSONObject tossRes = (JSONObject) parser.parse(new InputStreamReader(is, StandardCharsets.UTF_8));
if (is != null) is.close();

if (code != 200) {
    // Toss 쪽 에러를 그대로 프론트로 전달
    log.warn("Toss confirm failed: {}", tossRes);
    return ResponseEntity.status(code).body(tossRes);
}

// tossRes에서 결제수단(method) 안전하게 추출
String paymentType = safeExtractPaymentMethod(tossRes);
```

비관적 락(Pessimistic Lock)을 통한 재고 관리

프로젝트 시연 및 기능소개

기능 소개

동시성 문제 차단

- 재고 차감 요청이 들어오는 순간, 해당 상품 데이터에 락을 걸어 다른 모든 요청을 대기시키며, 데이터 정합성을 최우선으로 지켜 신뢰도 높은 쇼핑 환경을 제공

SPRING 트랜잭션 롤백

- 재고 락을 건 트랜잭션 수행 중 예기치 않은 오류가 발생하거나 락을 획득하지 못해 경쟁이 발생했을 때 데이터의 일관성을 자동으로 복구하는 안전 장치를 제공

JPA 엔티티 비관적 락

- @LOCK(LOCKMODETYPE.PESSIMISTIC_WRITE) 어노테이션으로 로직을 적용하여 순수 SQL 대신 객체지향 방식으로 락킹 관리



주요 코드

```
// 주문 상품 생성 및 재고 차감
for (OrderCreateRequestDTO.OrderItemDTO itemDTO : request.getItems()) {
    Stock stock;
    try {
        stock = stockRepository.findByIdForUpdate(itemDTO.getStockId()) // 락이 걸린 메서드 사용
            .orElseThrow(() -> new IllegalArgumentException("재고를 찾을 수 없습니다: " + itemDTO.getStockId()));
    } catch (PessimisticLockingFailureException e) {
        // 락 획득 실패 시 (동시성 충돌 발생)
        throw new RuntimeException("동시 주문 충돌로 인해 주문 처리에 실패했습니다. 다시 시도해 주세요.");
    }
    // 재고 확인 및 차감 로직 (락이 걸린 상태에서 안전하게 처리)
    if (stock.getQuantity() < itemDTO.getQuantity()) {
        // 재고 부족 시 트랜잭션 롤백
        throw new IllegalArgumentException("상품 " + stock.getProduct().getName() + "의 재고가 부족합니다.");
    }
    // 재고 차감 (수정)
    stock.setQuantity(stock.getQuantity() - itemDTO.getQuantity());
    stockRepository.save(stock);
}

public interface StockRepository extends JpaRepository<Stock, Integer> {
    @Lock(LockModeType.PESSIMISTIC_WRITE) 1개 사용 위치 신규 *
    @QueryHints({
        @QueryHint(name = "jakarta.persistence.lock.timeout", value = "3000")
    })
    Optional<Stock> findByIdForUpdate(Long stockId);

    @Lock(LockModeType.PESSIMISTIC_WRITE) 1개 사용 위치 신규 *
    @QueryHints({
        @QueryHint(name = "jakarta.persistence.lock.timeout", value = "3000")
    })
    Optional<Stock> findByProductIdAndColorAndSizeForUpdate(Long productId, String color, String size);
}
```

기대 효과

쇼핑 경험 혁신

- “입어보는 듯한 경험” 제공
→ 온라인 쇼핑의 불편함 해소
- 개인 맞춤형 추천과 결합 시, 구매 전환율 상승

환경적 가치

- 불필요한 구매와 반품 감소
- 의류 폐기물 감소
- 지속 가능한 소비문화 형성 기여

사이즈 불일치 문제 해결

- 고객이 직접 가상 피팅을 통해 자신의 체형에 맞는 사이즈를 미리 확인 가능
 - 반품률 감소
- 물류비 절감 및 고객 만족도 향상

마케팅

체험형 캠페인

- “내가 직접 입어본 듯한 쇼핑”
무료 체험 이벤트 제공
- 인플루언서/패션 유튜버와 협업
→ 실제 체험 영상 제작

다양한 이벤트 제공

- 가상 피팅 후 SNS 공유 시
할인 쿠폰 제공
- “내 스타일 챌린지”

가상 피팅 록북 제공

- 본인 체형 데이터를 저장
→ 시즌마다 신상품을 자동으로 가상 피팅 록북으로 제공
 - 개인 맞춤 화보집 받는 느낌
- 신선한 경험 제공

사업 확장 방안 및 영업 이익

비즈니스 및 마케팅 전략

사업 확장

AI 맞춤형 추천 고도화

- 체형·취향·날씨를 반영한 **AI 코디 추천 서비스** 제공
- 사용자 피드백을 학습해 **추천 정확도와 만족도 향상**

브랜드 제휴 및 플랫폼 확장

- **브랜드 입점형 온라인 피팅룸** 구축으로 제휴 확대
- **AI 스타일 큐레이션**으로 브랜드 노출 및 구매 유도

글로벌 및 메타버스 연계

- **가상 아바타 피팅 서비스**로 해외 사용자 확보
- **메타버스 패션쇼·쇼룸** 등 체험형 콘텐츠 확장

영업이익

판매 수수료 및 광고 수익

- **입점 브랜드 판매 수수료 + 맞춤형 광고 수익 창출**
- AI 추천을 통한 **브랜드 노출 극대화 및 전환율 상승**

프리미엄 구독 모델

- 무광고·고해상도 피팅 등 **구독 전용 기능** 제공
- **개인 맞춤 코디 리포트 서비스**로 차별화된 경험 제공

데이터 기반 B2B 서비스

- 체형·스타일 데이터를 활용한 **기업 맞춤 분석 리포트** 제공
- **트렌드 예측·시장 인사이트** 판매로 부가 수익 확보

느낀점

자체평가 및 개인의견

박세원

이번 프로젝트는 Spring Boot와 React 기반으로 진행되었고, JPA를 활용하였습니다. 특히 Toss Payments API를 처음으로 사용해 보면서, 실제 결제와 동일한 흐름을 구현하는 과정을 경험할 수 있었고, 이를 통해 API 연동 방식과 동작 원리를 이해하는 데 큰 도움이 되었습니다.

또한 처음으로 팀장 역할을 맡아 프로젝트를 이끌어가는 경험을 했습니다. 단순히 개발에 참여하는 것을 넘어 팀원들과 역할을 분담하고 일정 관리를 하면서 협업의 중요성과 책임감을 크게 느낄 수 있었습니다. 또한 팀원들의 의견을 조율하고 방향을 제시하는 과정에서, 기술적인 성장 뿐만 아니라 리더십과 소통 능력에서도 많은 배움을 얻을 수 있었습니다.

권택준

초기에는 JPA 사용에 익숙하지 않아 어려움을 겪었지만, 점차 경험을 쌓으며 ORM의 효율성을 체감하게 되었고 이를 통해 개발 편의성과 생산성을 크게 높일 수 있었습니다. 또한 재고 관리 시스템에 트랜잭션 및 락킹 기반의 동시성 제어를 적용해 데이터 정합성을 성공적으로 확보했습니다. 이러한 과정을 통해 단순한 기능 구현을 넘어, 시스템의 안정성과 신뢰성을 확보하는 데 필수적인 기술들을 익히게 되었으며, 이는 향후 복잡하고 안정적인 시스템을 구축하는 데 중요한 자산이 될 것입니다.

유부미

이번 프로젝트는 프론트엔드와 백엔드가 분리된 구조 속에서 데이터를 주고받는 과정이 핵심이었습니다. 특히 JPA를 활용하여 데이터베이스를 관리하며, 이전보다 훨씬 구조적이고 체계적인 개발 방식을 경험할 수 있었습니다. 또한 AI, Spring, React 등 다양한 기술을 직접 통합하면서, 백엔드와 프론트엔드의 협업 방식과 서비스 전체 흐름을 명확히 이해하게 되었습니다. 제가 맡았던 로그인 및 보안 파트에서는 처음엔 복잡하게 느껴졌던 토큰 발급, 세션 관리, 예외 처리 흐름을 직접 구축하며 실제 서비스가 동작하는 구조를 이해할 수 있었던 값진 경험을 얻었습니다. 이번 프로젝트를 통해 어려움 앞에서도 차분히 해결책을 찾아가는 자신감을 가질 수 있었습니다.

이현호

React와 Spring를 연동하여 프로젝트를 진행하면서 그동안 해왔던 React만 사용하거나 JSP나 Thymeleaf를 사용하는 방식보다 더욱 편리하다고 느꼈습니다. JPA 사용에도 처음에는 거부감이 있었지만 사용하다 보니 왜 편리하다고 하는지 와닿았던 프로젝트였고, 그동안 해왔던 방식을 경험하고 지금의 프로젝트를 진행하면서 개발방식이 발전해 온것을 느낄 수 있었습니다. 다양한 API 를 사용해 볼 수 있어 좋았고 처음에 규모를 크게 잡은 만큼 연계해서 만들 수 있던 기능들도 많고 웹 사이트라면 만들 수 있어야 하는 기능들을 많이 구현해서 재미있던 프로젝트였습니다.

FAISHION

Q & A

발표자 : 이현호