

신발 매장 API 가이드 문서

작성일: 2025-01-XX

작성자: 김택권

목적: FastAPI 백엔드 API 사용 가이드

기본 URL: <http://127.0.0.1:8000>

목차

1. API 개요
 2. 기본 설정
 3. 인증 및 보안
 4. 공통 응답 형식
 5. 기본 CRUD API
 6. 인증 API
 7. JOIN API
 8. 관리자 API (Admin API)
 9. 고객용 Plus API
 10. 에러 처리
 11. 사용 예시
-

API 개요

서버 정보

- 프레임워크: FastAPI
- 데이터베이스: MySQL
- 데이터 형식: JSON (Form 데이터 방식)
- 문서: Swagger UI (<http://127.0.0.1:8000/docs>)
- 소셜 로그인 지원: Google, Kakao 등

API 구조

- 기본 **CRUD API**: 11개 테이블에 대한 CRUD 작업
 - branches, users, user_auth_identities, staff, makers
 - kind_categories, color_categories, size_categories, gender_categories, refund_reason_categories
 - products, purchase_items, pickups, refunds, chatting
 - 인증 **API**: 소셜 로그인 및 회원가입
 - JOIN **API**: 복잡한 조인 쿼리를 위한 2개 API 그룹 (Product Join, Refund Join)
 - 관리자 **API (Admin API)**: 관리자 페이지용 전용 API (구매 내역, 수령, 반품)
 - 고객용 **Plus API**: 고객용 주문/수령/반품 내역 조회 API (검색 및 정렬 기능 포함)
 - 총 엔드포인트: 약 100개 이상
-

기본 설정

서버 실행

방법 1: main.py 직접 실행

```
cd fastapi
python app/main.py
```

방법 2: uvicorn으로 실행 (권장)

```
cd fastapi
uvicorn app.main:app --host 127.0.0.1 --port 8000 --reload
```

참고:

- `main.py`는 `fastapi/app/main.py`에 위치합니다
- 실행은 반드시 `fastapi` 폴더에서 해야 합니다 (Python이 `app` 모듈을 찾을 수 있도록)

헬스 체크

```
GET /health
```

응답 예시:

```
{
  "status": "healthy",
  "database": "connected"
}
```

루트 엔드포인트

```
GET /
```

응답 예시:

```
{
  "message": "Shoes Store API - 새로운 ERD 구조",
  "status": "running",
  "endpoints": {
    "branches": "/api/branches",
  }
}
```

```

    "users": "/api/users",
    "user_auth_identities": "/api/user_auth_identities",
    "staff": "/api/staff",
    "makers": "/api/makers",
    "kind_categories": "/api/kind_categories",
    "color_categories": "/api/color_categories",
    "size_categories": "/api/size_categories",
    "gender_categories": "/api/gender_categories",
    "refund_reason_categories": "/api/refund_reason_categories",
    "products": "/api/products",
    "purchase_items": "/api/purchase_items",
    "pickups": "/api/pickups",
    "refunds": "/api/refunds",
    "chatting": "/api/chatting"
  },
  "join_endpoints": {
    "products_join": "/api/products/with_categories",
    "refunds_join": "/api/refunds/{id}/with_details,
    /api/refunds/{id}/full_detail"
  }
}

```

인증 및 보안

현재 버전에서는 소셜 로그인(Google, Kakao)을 지원하며, 로컬 로그인은 `user_auth_identities` 테이블을 통해 관리됩니다.

- 소셜 로그인: `/api/auth/social/login`
 - 로컬 로그인: `user_auth_identities` 테이블의 `password` 필드 사용
 - 향후 JWT 토큰 기반 인증이 추가될 예정입니다.
-

공통 응답 형식

성공 응답

목록 조회:

```

{
  "results": [
    {
      "id": 1,
      "name": "값"
    }
  ]
}

```

단일 조회:

```
{
  "result": {
    "id": 1,
    "name": "값"
  }
}
```

생성/수정/삭제:

```
{
  "result": "OK",
  "id": 1 // 생성 시에만 포함
}
```

에러 응답

```
{
  "result": "Error",
  "errorMsg": "에러 메시지",
  "message": "상세 메시지" // 선택적
}
```

기본 CRUD API

1. 지점 (Branch)

기본 경로: `/api/branches`

메서드	엔드포인트	설명
GET	<code>/api/branches</code>	전체 지점 조회
GET	<code>/api/branches/{branch_seq}</code>	지점 상세 조회
POST	<code>/api/branches</code>	지점 추가
POST	<code>/api/branches/{branch_seq}</code>	지점 수정
DELETE	<code>/api/branches/{branch_seq}</code>	지점 삭제

데이터 모델:

```
{
  "br_seq": 1,
  "br_name": "강남점",
  "br_phone": "02-1234-5678",
}
```

```
    "br_address": "서울시 강남구 테헤란로 123",
    "br_lat": 37.5010,
    "br_lng": 127.0260
  }
```

2. 고객 (User)

기본 경로: /api/users

△ 중요 변경사항:

- u_id, u_password 필드 제거
- u_email 필드 추가 (필수, UNIQUE)
- u_phone 필드는 nullable

메서드	엔드포인트	설명
GET	/api/users	전체 고객 조회
GET	/api/users/{user_seq}	고객 상세 조회
POST	/api/users	고객 추가 (이미지 필수)
POST	/api/users/{user_seq}	고객 수정 (이미지 제외)
POST	/api/users/{user_seq}/with_image	고객 수정 (이미지 포함)
GET	/api/users/{user_seq}/profile_image	프로필 이미지 조회
DELETE	/api/users/{user_seq}/profile_image	프로필 이미지 삭제
DELETE	/api/users/{user_seq}	고객 삭제

데이터 모델:

```
{
  "u_seq": 1,
  "u_email": "user@example.com",
  "u_name": "홍길동",
  "u_phone": "010-1111-1111",
  "u_address": "서울시 강남구",
  "created_at": "2025-01-15T10:30:00",
  "u_quit_date": null
}
```

고객 추가 예시 (Form 데이터):

```
curl -X POST "http://127.0.0.1:8000/api/users" \
  -F "u_email=user@example.com" \
  -F "u_name=홍길동" \
```

```
-F "u_phone=010-1111-1111" \
-F "u_address=서울시 강남구" \
-F "file=@profile.jpg"
```

3. 사용자 인증 정보 (User Auth Identities)

기본 경로: `/api/user_auth_identities`

설명: 소셜 로그인 지원을 위한 인증 정보 관리 테이블. 하나의 사용자가 여러 인증 수단(로컬, Google, Kakao)을 가질 수 있습니다.

메서드	엔드포인트	설명
GET	<code>/api/user_auth_identities/user/{user_seq}</code>	사용자별 인증 정보 조회
GET	<code>/api/user_auth_identities/provider/{provider}</code>	제공자별 인증 정보 조회
POST	<code>/api/user_auth_identities</code>	인증 정보 추가
POST	<code>/api/user_auth_identities/{auth_seq}</code>	인증 정보 수정
POST	<code>/api/user_auth_identities/{auth_seq}/update_login_time</code>	마지막 로그인 시간 업데이트
DELETE	<code>/api/user_auth_identities/{auth_seq}</code>	인증 정보 삭제

데이터 모델:

```
{
  "auth_seq": 1,
  "u_seq": 1,
  "provider": "local",
  "provider_subject": "user@example.com",
  "provider_issuer": null,
  "email_at_provider": null,
  "password": "hashed_password",
  "created_at": "2025-01-15T10:30:00",
  "last_login_at": "2025-01-20T14:30:00"
}
```

필드 설명:

- `provider`: 'local', 'google', 'kakao' 등
- `provider_subject`: 제공자별 고유 식별자 (로컬: 이메일, Google: sub, Kakao: id)
- `provider_issuer`: 소셜 제공자 발급자 (iss) - 소셜 로그인만 사용
- `email_at_provider`: 소셜 제공자에서 받은 이메일 - 소셜 로그인만 사용
- `password`: 로컬 로그인용 비밀번호 (해시 필요) - 로컬 로그인만 사용

인증 정보 추가 예시 (로컬 로그인):

```
curl -X POST "http://127.0.0.1:8000/api/user_auth_identities" \
-F "u_seq=1" \
-F "provider=local" \
-F "provider_subject=user@example.com" \
-F "password=plain_password"
```

4. 직원 (Staff)

기본 경로: /api/staff

메서드	엔드포인트	설명
GET	/api/staff	전체 직원 조회
GET	/api/staff/{staff_seq}	직원 상세 조회 (s_seq로 조회)
GET	/api/staff/by_id/{staff_id}	직원 조회 (s_id로 조회, 로그인용)
GET	/api/staff/by_branch/{branch_seq}	지점별 직원 조회
POST	/api/staff	직원 추가 (이미지 필수)
POST	/api/staff/{staff_seq}	직원 수정 (이미지 제외)
POST	/api/staff/{staff_seq}/with_image	직원 수정 (이미지 포함)
GET	/api/staff/{staff_seq}/profile_image	프로필 이미지 조회
DELETE	/api/staff/{staff_seq}/profile_image	프로필 이미지 삭제
DELETE	/api/staff/{staff_seq}	직원 삭제

데이터 모델:

```
{
  "s_seq": 1,
  "s_id": "staff001",
  "br_seq": 1,
  "s_password": "hashed_password",
  "s_name": "김점장",
  "s_phone": "010-1001-1001",
  "s_rank": "점장",
  "s_superseq": null,
  "created_at": "2025-01-15T10:30:00",
  "s_quit_date": null
}
```

직원 추가 예시:

```
curl -X POST "http://127.0.0.1:8000/api/staff" \
  -F "s_id=staff001" \
  -F "br_seq=1" \
  -F "s_password=pass1234" \
  -F "s_name=김점장" \
  -F "s_phone=010-1001-1001" \
  -F "s_rank=점장" \
  -F "file=@profile.jpg"
```

s_id로 직원 조회 예시 (로그인용):

```
curl "http://127.0.0.1:8000/api/staff/by_id/staff001"
```

응답 예시:

```
{
  "result": {
    "s_seq": 1,
    "s_id": "staff001",
    "br_seq": 1,
    "s_password": "pass1234",
    "s_name": "김점장",
    "s_rank": "점장",
    "s_phone": "010-1001-1001",
    "s_superseq": null,
    "created_at": "2025-01-15T10:30:00",
    "s_quit_date": null
  }
}
```

참고:

- `/by_id/{staff_id}` 엔드포인트는 탈퇴하지 않은 직원만 조회합니다 (`s_quit_date IS NULL OR s_quit_date > NOW()`)
- 로그인 시 이 엔드포인트를 사용하여 s_id로 직원을 조회하고 비밀번호를 검증합니다

5. 제조사 (Maker)

기본 경로: `/api/makers`

메서드	엔드포인트	설명
GET	<code>/api/makers</code>	전체 제조사 조회
GET	<code>/api/makers/{maker_seq}</code>	제조사 상세 조회

메서드	엔드포인트	설명
POST	<code>/api/makers</code>	제조사 추가
POST	<code>/api/makers/{maker_seq}</code>	제조사 수정
DELETE	<code>/api/makers/{maker_seq}</code>	제조사 삭제

6. 카테고리 (Categories)

6.1 종류 카테고리 (Kind Category)

기본 경로: `/api/kind_categories`

메서드	엔드포인트	설명
GET	<code>/api/kind_categories</code>	전체 조회
GET	<code>/api/kind_categories/{kind_category_seq}</code>	상세 조회
POST	<code>/api/kind_categories</code>	추가
POST	<code>/api/kind_categories/{kind_category_seq}</code>	수정
DELETE	<code>/api/kind_categories/{kind_category_seq}</code>	삭제

6.2 색상 카테고리 (Color Category)

기본 경로: `/api/color_categories`

6.3 사이즈 카테고리 (Size Category)

기본 경로: `/api/size_categories`

6.4 성별 카테고리 (Gender Category)

기본 경로: `/api/gender_categories`

6.5 환불 사유 카테고리 (Refund Reason Category)

기본 경로: `/api/refund_reason_categories`

7. 제품 (Product)

기본 경로: `/api/products`

메서드	엔드포인트	설명
GET	<code>/api/products/group_by_name</code>	제품명 그룹화 조회 (같은 이름의 제품을 그룹화)
GET	<code>/api/products/id/{product_seq}</code>	제품 ID로 상세 조회

메서드	엔드포인트	설명
GET	<code>/api/products/getBySeqs</code>	제조사별 제품 조회 (쿼리 파라미터: m_seq)
POST	<code>/api/products</code>	제품 추가
POST	<code>/api/products/{product_seq}</code>	제품 수정
POST	<code>/api/products/{product_seq}/stock</code>	제품 재고 수정
POST	<code>/api/products/{product_seq}/upload_file</code>	제품 이미지 업로드
DELETE	<code>/api/products/{product_seq}</code>	제품 삭제

데이터 모델:

```
{
  "p_seq": 1,
  "kc_seq": 1,
  "cc_seq": 1,
  "sc_seq": 1,
  "gc_seq": 1,
  "m_seq": 1,
  "p_name": "에어맥스 90",
  "p_price": 150000,
  "p_stock": 50,
  "p_image": "/images/product_1.jpg",
  "p_description": "나이키 에어맥스 90 클래식",
  "created_at": "2025-01-15T10:30:00"
}
```

8. 구매 내역 (Purchase Item)

기본 경로: `/api/purchase_items`

메서드	엔드포인트	설명
GET	<code>/api/purchase_items</code>	전체 구매 내역 조회
GET	<code>/api/purchase_items/{purchase_item_seq}</code>	구매 내역 상세 조회
GET	<code>/api/purchase_items/by_user/{user_seq}</code>	고객별 구매 내역 조회
GET	<code>/api/purchase_items/by_datetime</code>	분 단위 그룹화된 주문 조회
POST	<code>/api/purchase_items</code>	구매 내역 추가
POST	<code>/api/purchase_items/{purchase_item_seq}</code>	구매 내역 수정
DELETE	<code>/api/purchase_items/{purchase_item_seq}</code>	구매 내역 삭제

데이터 모델:

```
{
  "b_seq": 1,
  "br_seq": 1,
  "u_seq": 1,
  "p_seq": 1,
  "b_price": 150000,
  "b_quantity": 2,
  "b_date": "2025-01-15T14:30:00",
  "b_status": "0",
  "b_tnum": "TRANS001"
}
```

b_status 값 설명:

- "0": 준비중
- "1": 준비완료
- "2": 수령완료
- "3": 반품완료
- null: 상태미정

9. 수령 (Pickup)

기본 경로: /api/pickups

메서드	엔드포인트	설명
GET	/api/pickups	전체 수령 내역 조회
GET	/api/pickups/{pickup_seq}	수령 내역 상세 조회
GET	/api/pickups/by_bseq/{purchase_item_seq}	구매 내역별 수령 조회
POST	/api/pickups	수령 내역 추가
POST	/api/pickups/{pickup_seq}	수령 내역 수정
POST	/api/pickups/{pickup_seq}/complete	수령 완료 처리
DELETE	/api/pickups/{pickup_seq}	수령 내역 삭제

10. 반품 (Refund)

기본 경로: /api/refunds

메서드	엔드포인트	설명
GET	/api/refunds	전체 반품 내역 조회
GET	/api/refunds/{refund_seq}	반품 내역 상세 조회
GET	/api/refunds/by_user/{user_seq}	고객별 반품 내역 조회

메서드	엔드포인트	설명
POST	/api/refunds	반품 내역 추가
POST	/api/refunds/{refund_seq}	반품 내역 수정
POST	/api/refunds/{refund_seq}/process	반품 처리
DELETE	/api/refunds/{refund_seq}	반품 내역 삭제

11. 채팅 (Chatting)

기본 경로: /api/chatting

설명: 고객과 직원 간의 채팅 세션을 관리하는 API입니다. Firebase Firestore와 연동하여 실시간 채팅을 지원합니다.

메서드	엔드포인트	설명
GET	/api/chatting	전체 채팅 세션 조회
GET	/api/chatting/by_user_seq	고객별 채팅 세션 조회 (쿼리 파라미터: u_seq, is_closed)
GET	/api/chatting/{chatting_seq}	채팅 세션 상세 조회
POST	/api/chatting	채팅 세션 추가
POST	/api/chatting/{chatting_seq}	채팅 세션 수정
DELETE	/api/chatting/{chatting_seq}	채팅 세션 삭제

데이터 모델:

```
{
  "chatting_seq": 1,
  "u_seq": 1,
  "u_name": "홍길동",
  "fb_doc_id": "firebase_document_id",
  "s_seq": 1,
  "s_name": "김점장",
  "created_at": "2025-01-15T10:30:00",
  "is_closed": false
}
```

필드 설명:

- **chatting_seq**: 채팅 세션 고유 ID (PK)
- **u_seq**: 고객 번호 (FK → user.u_seq)
- **u_name**: 고객 이름 (JOIN으로 조회)
- **fb_doc_id**: Firebase Firestore 문서 ID (선택 사항)
- **s_seq**: 담당 직원 번호 (FK → staff.s_seq, 선택 사항)
- **s_name**: 직원 이름 (JOIN으로 조회)
- **created_at**: 채팅 세션 생성 일시

- **is_closed**: 채팅 종료 여부 (boolean)

고객별 채팅 조회 예시:

```
curl "http://127.0.0.1:8000/api/chatting/by_user_seq?u_seq=1&is_closed=false"
```

is_closed 파라미터 동작:

- **is_closed=false**: 열린 채팅방(**is_closed=0**)을 먼저 찾고, 없으면 닫힌 채팅방(**is_closed=1**)을 찾아서 자동으로 다시 열어서 반환합니다. 둘 다 없으면 **{"result": "Error"}** 반환
- **is_closed=true**: 닫힌 채팅방만 조회합니다

채팅 세션 추가 예시:

```
curl -X POST "http://127.0.0.1:8000/api/chatting" \
-F "u_seq=1" \
-F "fb_doc_id=firebase_document_id" \
-F "s_seq=1" \
-F "is_closed=false"
```

응답 예시 (전체 조회):

```
{
  "results": [
    {
      "chatting_seq": 1,
      "u_seq": 1,
      "fb_doc_id": "firebase_document_id",
      "s_seq": 1,
      "created_at": "2025-01-15T10:30:00",
      "is_closed": false,
      "u_name": "홍길동",
      "s_name": "김점장"
    }
  ]
}
```

참고:

- 채팅 메시지는 Firebase Firestore에 저장되며, **fb_doc_id**로 연결됩니다.
- **s_seq**가 NULL인 경우 아직 담당 직원이 배정되지 않은 상태입니다.
- **is_closed**가 **true**인 경우 종료된 채팅 세션입니다.
- 같은 유저 ID로 **is_closed=false** 요청 시, 닫힌 채팅방이 있으면 자동으로 다시 열어서 재사용합니다.

소셜 로그인 및 회원가입

기본 경로: `/api/auth`

1. 소셜 로그인 (1단계)

POST `/api/auth/social/login`

설명: 소셜 로그인 후 사용자 생성 또는 조회

- 기존 사용자면 조회하여 반환
- 신규 사용자면 기본 정보만 저장하고 미완료 상태로 반환

요청 파라미터 (Form):

- **provider** (필수): 'google', 'kakao' 등
- **provider_subject** (필수): 소셜 제공자의 고유 ID (Google: sub, Kakao: id)
- **email** (선택): 이메일 주소
- **name** (선택): 이름
- **provider_issuer** (선택): 소셜 제공자 발급자 (iss)

요청 예시:

```
curl -X POST "http://127.0.0.1:8000/api/auth/social/login" \
-F "provider=google" \
-F "provider_subject=123456789" \
-F "email=user@gmail.com" \
-F "name=홍길동" \
-F "provider_issuer=https://accounts.google.com"
```

응답 예시 (기존 사용자):

```
{
  "result": {
    "u_seq": 1,
    "u_name": "홍길동",
    "u_email": "user@gmail.com",
    "u_phone": "010-1111-1111",
    "u_address": "서울시 강남구",
    "u_quit_date": null,
    "created_at": "2025-01-15T10:30:00",
    "auth_seq": 1,
    "provider": "google",
    "provider_subject": "123456789",
    "last_login_at": "2025-01-20T14:30:00"
  },
}
```

```
{
  "message": "기존 사용자 로그인 성공"
}
```

응답 예시 (신규 사용자):

```
{
  "result": {
    "u_seq": 2,
    "u_name": "홍길동",
    "u_email": "user@gmail.com",
    "u_phone": null,
    "u_address": null,
    "u_quit_date": null,
    "created_at": "2025-01-20T14:30:00",
    "auth_seq": 2,
    "provider": "google",
    "provider_subject": "123456789",
    "last_login_at": null
  },
  "message": "소셜 로그인 성공. 추가 정보 입력이 필요합니다."
}
```

JOIN API

1. 제품 JOIN API

기본 경로: `/api/products`

1.1 제품 목록 + 카테고리 조회

```
GET /api/products/with_categories
```

설명: 모든 제품과 카테고리 정보를 함께 조회 (필터링 가능)

쿼리 파라미터:

- `maker_seq` (선택): 제조사 ID
- `kind_seq` (선택): 종류 카테고리 ID
- `color_seq` (선택): 색상 카테고리 ID
- `size_seq` (선택): 사이즈 카테고리 ID
- `gender_seq` (선택): 성별 카테고리 ID

예시:

```
# 전체 제품 조회
curl "http://127.0.0.1:8000/api/products/with_categories"

# 필터링: 나이키 제품 중 남성용
curl "http://127.0.0.1:8000/api/products/with_categories?
maker_seq=1&gender_seq=1"
```

응답 예시:

```
{
  "results": [
    {
      "p_seq": 1,
      "p_name": "에어맥스 90",
      "p_price": 150000,
      "p_stock": 50,
      "p_image": "/images/product_1.jpg",
      "kind_name": "러닝화",
      "color_name": "블랙",
      "size_name": "260",
      "gender_name": "남성",
      "maker_name": "나이키"
    }
  ]
}
```

2. 반품 JOIN API

기본 경로: `/api/refunds`

2.1 반품 상세 정보 조회

```
GET /api/refunds/refunds/{refund_seq}/with_details
```

설명: 반품 + 고객 + 직원 + 수령 + 구매 내역 + 제품 + 지점 정보 (7테이블 JOIN)

실제 경로: `/api/refunds/refunds/{refund_seq}/with_details` (prefix `/api/refunds` + 라우터 경로 `/refunds/{refund_seq}/with_details`)

2.2 반품 전체 상세 정보 조회

```
GET /api/refunds/refunds/{refund_seq}/full_detail
```


설명: 반품 + 고객 + 직원 + 수령 + 구매 내역 + 제품 + 지점 + 모든 카테고리 + 제조사 (12테이블 JOIN)

실제 경로: `/api/refunds/refunds/{refund_seq}/full_detail`

2.3 고객별 반품 상세 정보 조회

```
GET /api/refunds/refunds/by_user/{user_seq}/with_details
```

설명: 특정 고객의 모든 반품 내역을 상세 정보와 함께 조회

2.4 직원별 반품 상세 정보 조회

```
GET /api/refunds/refunds/by_staff/{staff_seq}/with_details
```

설명: 특정 직원이 처리한 모든 반품 내역을 상세 정보와 함께 조회

관리자 API (Admin API)

관리자 페이지에서 사용하는 전용 API입니다. 전체 목록 조회 및 상세 정보 조회를 제공합니다.

1. 구매 내역 관리자 API

기본 경로: `/api/purchase_items/admin`

1.1 전체 구매 내역 조회

```
GET /api/purchase_items/admin/all
```

설명: 관리자용 전체 구매 내역 조회 (검색 기능 포함)

쿼리 파라미터:

- **search** (선택): 검색어 (구매 내역 번호 또는 고객 이름)

응답 예시:

```
{
  "results": [
    {
      "b_seq": 1,
      "b_price": 150000,
      "b_quantity": 2,
      "b_date": "2025-01-15 14:30",
      "b_status": "0",
    }
  ]
}
```

```
    "u_seq": 1,
    "u_email": "user@example.com",
    "u_name": "홍길동",
    "u_phone": "010-1111-1111",
    "p_seq": 1,
    "p_name": "에어맥스 90",
    "p_price": 150000,
    "p_stock": 50,
    "p_image": "/images/product_1.jpg",
    "kc_name": "러닝화",
    "cc_name": "블랙",
    "sc_name": "260",
    "gc_name": "남성",
    "m_name": "나이키",
    "br_seq": 1,
    "br_name": "강남점",
    "br_address": "서울시 강남구",
    "br_phone": "02-1234-5678"
  }
}
```

1.2 구매 내역 상세 조회

```
GET /api/purchase_items/admin/{purchase_item_seq}/full_detail
```

설명: 관리자용 구매 내역 전체 상세 정보 (9테이블 JOIN)

2. 수령 관리자 API

기본 경로: `/api/pickups/admin`

2.1 전체 수령 내역 조회

```
GET /api/pickups/admin/all
```

설명: 관리자용 전체 수령 내역 조회 (검색 기능 포함)

쿼리 파라미터:

- `search` (선택): 검색어 (수령 번호 또는 고객 이름)

2.2 수령 내역 상세 조회

```
GET /api/pickups/admin/{pickup_seq}/full_detail
```

설명: 관리자용 수령 내역 전체 상세 정보 (10테이블 JOIN)

3. 반품 관리자 API

기본 경로: `/api/refunds/admin`

3.1 전체 반품 내역 조회

```
GET /api/refunds/admin/all
```

설명: 관리자용 전체 반품 내역 조회 (검색 기능 포함)

쿼리 파라미터:

- `search` (선택): 검색어 (반품 번호 또는 고객 이름)

3.2 반품 내역 상세 조회

```
GET /api/refunds/admin/{refund_seq}/full_detail
```

설명: 관리자용 반품 내역 전체 상세 정보 (12테이블 JOIN)

고객용 Plus API

고객이 자신의 주문, 수령, 반품 내역을 조회할 때 사용하는 API입니다. 검색 및 정렬 기능을 제공합니다.

1. 구매 내역 Plus API

기본 경로: `/api/purchase_items`

1.1 고객별 주문 그룹화 조회

```
GET /api/purchase_items/by_user/{user_seq}/user_bundle
```

설명: 고객별 주문 내역을 분 단위로 그룹화하여 조회 (주문 일자별로 묶음)

쿼리 파라미터:

- `keyword` (선택): 검색어 (제품명 또는 제조사명)
- `order` (선택): 정렬 방식 (최신순, 오래된 순, 가격 높은순, 가격 낮은순), 기본값: 최신순

응답 예시:

```
{
  "results": [
    {
      "b_date": "2025-01-15 14:30",
      "order_date": "2025-01-15",
      "order_time": "14:30",
      "item_count": 3,
      "total_amount": 450000,
      "items": [
        {
          "b_seq": 1,
          "b_price": 150000,
          "b_quantity": 2,
          "b_status": "0",
          "p_name": "에어맥스 90",
          "m_name": "나이키",
          ...
        }
      ]
    }
  ]
}
```

참고: **b_status**가 0(준비중) 또는 1(준비완료)인 항목만 조회됩니다.

2. 수령 Plus API

기본 경로: **/api/pickups**

2.1 고객별 수령 내역 조회

```
GET /api/pickups/by_user/{user_seq}/all
```

설명: 고객별 수령 내역 전체 조회 (검색 및 정렬 기능 포함)

쿼리 파라미터:

- **keyword** (선택): 검색어 (제품명 또는 제조사명)
- **order** (선택): 정렬 방식 (최신순, 오래된 순, 가격 높은순, 가격 낮은순), 기본값: 최신순

참고: **b_status**가 2(수령완료)인 항목만 조회됩니다.

3. 반품 Plus API

기본 경로: **/api/refunds**

3.1 고객별 반품 내역 조회

GET /api/refunds/refund/by_user/{user_seq}/all

설명: 고객별 반품 내역 전체 조회 (검색 및 정렬 기능 포함)

쿼리 파라미터:

- **keyword** (선택): 검색어 (제품명 또는 제조사명)
- **order** (선택): 정렬 방식 (최신순, 오래된 순, 가격 높은순, 가격 낮은순), 기본값: 최신순

응답 예시:

```
{
  "results": [
    {
      "ref_seq": 1,
      "ref_date": "2025-01-20 10:30",
      "ref_re_content": "사이즈가 맞지 않음",
      "ref_re_seq": 1,
      "ref_re_name": "사이즈 불일치",
      "u_seq": 1,
      "u_name": "홍길동",
      "s_seq": 1,
      "s_name": "김점장",
      "s_rank": "점장",
      "pic_seq": 1,
      "pic_created_at": "2025-01-18 14:30",
      "b_seq": 1,
      "b_price": 150000,
      "p_name": "에어맥스 90",
      "m_name": "나이키",
      ...
    }
  ]
}
```

특수 기능 API

주문 그룹화

구매 내역은 **b_date** 필드를 기준으로 분 단위(YYYY-MM-DD HH:MM)로 그룹화됩니다.

그룹화 규칙:

- 같은 분에 구매한 항목들이 하나의 주문으로 묶임
- 같은 고객(**u_seq**)과 같은 지점(**br_seq**)에서 구매한 항목만 그룹화
- 예: 2025-01-15 14:30에 구매한 모든 항목이 하나의 주문

에러 처리

공통 에러 코드

HTTP 상태 코드	의미	설명
200	OK	요청 성공
400	Bad Request	잘못된 요청
404	Not Found	리소스를 찾을 수 없음
500	Internal Server Error	서버 오류

에러 응답 형식

```
{
  "result": "Error",
  "errorMsg": "에러 메시지",
  "message": "상세 메시지" // 선택적
}
```

주요 에러 케이스

1. 리소스를 찾을 수 없음

```
{
  "result": "Error",
  "message": "User not found"
}
```

2. 중복 데이터

```
{
  "result": "Error",
  "errorMsg": "(1062, \"Duplicate entry 'user@example.com' for key 'user.idx_user_email'\")"
}
```

3. 외래 키 제약 조건 위반

```
{
  "result": "Error",
  "errorMsg": "(1452, \"Cannot add or update a child row: a foreign key constraint fails\")"
}
```

사용 예시

예시 1: 로컬 회원가입 및 로그인

```
# 1. 고객 가입 (user 테이블 생성)
curl -X POST "http://127.0.0.1:8000/api/users" \
  -F "u_email=user@example.com" \
  -F "u_name=홍길동" \
  -F "u_phone=010-1111-1111" \
  -F "u_address=서울시 강남구" \
  -F "file=@profile.jpg"

# 응답: {"result": "OK", "u_seq": 1}

# 2. 인증 정보 추가 (user_auth_identities 테이블 생성)
curl -X POST "http://127.0.0.1:8000/api/user_auth_identities" \
  -F "u_seq=1" \
  -F "provider=local" \
  -F "provider_subject=user@example.com" \
  -F "password=plain_password"

# 응답: {"result": "OK", "auth_seq": 1}
```

예시 2: 소셜 로그인 회원가입

```
# 1. 소셜 로그인 (1단계: 사용자 생성)
curl -X POST "http://127.0.0.1:8000/api/auth/social/login" \
  -F "provider=google" \
  -F "provider_subject=123456789" \
  -F "email=user@gmail.com" \
  -F "name=홍길동"

# 응답: {"result": {...}, "message": "소셜 로그인 성공. 추가 정보 입력이 필요합니다."}
```

예시 3: 고객 가입 및 주문

```
# 1. 고객 가입 (로컬)
curl -X POST "http://127.0.0.1:8000/api/users" \
  -F "u_email=user@example.com" \
  -F "u_name=홍길동" \
  -F "u_phone=010-1111-1111" \
  -F "u_address=서울시 강남구" \
  -F "file=@profile.jpg"

# 2. 제품 조회
```

```
curl "http://127.0.0.1:8000/api/products/with_categories"

# 3. 구매 내역 추가
curl -X POST "http://127.0.0.1:8000/api/purchase_items" \
  -F "br_seq=1" \
  -F "u_seq=1" \
  -F "p_seq=1" \
  -F "b_price=150000" \
  -F "b_quantity=2" \
  -F "b_date=2025-01-15T14:30:00" \
  -F "b_status=0" \
  -F "b_tnum=TRANS001"
```

데이터 타입 및 형식

날짜/시간 형식

- 데이터베이스: **DATETIME** (YYYY-MM-DD HH:MM:SS)
- **API 요청**: ISO 8601 형식 (2025-01-15T14:30:00) 또는 YYYY-MM-DD HH:MM
- **API 응답**: ISO 8601 형식 (2025-01-15T14:30:00.000000)

이미지 업로드

- 형식: Form 데이터 (**multipart/form-data**)
- 필드명: **file**
- 지원 형식: JPEG, PNG 등
- 저장 방식: **MEDIUMBLOB** (데이터베이스에 직접 저장)

주문 그룹화 날짜 형식

- 형식: YYYY-MM-DD HH:MM
- 예시: 2025-01-15 14:30
- URL 인코딩: 2025-01-15%2014:30

주의사항

1. 이미지 업로드: 고객/직원 추가 시 이미지는 필수입니다.
2. 외래 키 제약: 참조하는 테이블의 데이터가 먼저 존재해야 합니다.
3. **UNIQUE** 제약: **u_email**, **s_id**, **u_phone**, **s_phone** 등은 중복될 수 없습니다.
4. 주문 그룹화: 같은 분에 구매한 항목만 그룹화되므로, 정확한 시간 설정이 중요합니다.
5. 소프트 삭제: **u_quit_date**, **s_quit_date**를 설정하여 탈퇴 처리를 할 수 있습니다.
6. 로컬 회원가입: **user** 테이블과 **user_auth_identities** 테이블을 별도 API로 생성해야 합니다.
7. 소셜 로그인: **user** 테이블과 **user_auth_identities** 테이블이 하나의 API에서 함께 생성됩니다.
8. 채팅 API: **is_closed=false**로 요청 시 닫힌 채팅방이 있으면 자동으로 다시 열어서 재사용합니다.

추가 리소스

- **Swagger UI:** <http://127.0.0.1:8000/docs>
- **ReDoc:** <http://127.0.0.1:8000/redoc>

변경 이력

2025-12-30 김택권

- 최초 작성: API 가이드 문서 작성

2026-01-01 김택권

- 실제 코드 반영: 경로 파라미터 이름 수정
 - `{u_seq}` → `{user_seq}`
 - `{b_seq}` → `{purchase_item_seq}`
 - `{pic_seq}` → `{pickup_seq}`
 - `{ref_seq}` → `{refund_seq}`
 - `{rec_seq}` → `{receive_seq}`
 - `{req_seq}` → `{request_seq}`
 - `{p_seq}` → `{product_seq}`
 - `{m_seq}` → `{maker_seq}`
- 실행 방법 경로 수정 (`fastapi` 폴더에서 실행)
- 제품 이미지 업로드/다운로드 API 추가

2026-01-02 김택권

- **Staff API 경로 수정:** `/api/staffs` → `/api/staff` (단수형으로 통일)
- **Staff API 엔드포인트 추가:**
 - `GET /api/staff/by_id/{staff_id}`: s_id로 직원 조회 (로그인용)
 - `GET /api/staff/{staff_seq}/profile_image`: 프로필 이미지 조회
 - `DELETE /api/staff/{staff_seq}/profile_image`: 프로필 이미지 삭제
- Staff API 경로 파라미터 수정: `{s_seq}` → `{staff_seq}`, `{id}` → `{staff_seq}`

2026-01-03 유다원

- **관리자 API (Admin API) 섹션 추가:**
 - `GET /api/purchase_items/admin/all`: 관리자용 전체 구매 내역 조회 (검색 기능 포함)
 - `GET /api/purchase_items/admin/{purchase_item_seq}/full_detail`: 관리자용 구매 내역 상세 조회
 - `GET /api/pickups/admin/all`: 관리자용 전체 수령 내역 조회 (검색 기능 포함)
 - `GET /api/pickups/admin/{pickup_seq}/full_detail`: 관리자용 수령 내역 상세 조회
 - `GET /api/refunds/admin/all`: 관리자용 전체 반품 내역 조회 (검색 기능 포함)
 - `GET /api/refunds/admin/{refund_seq}/full_detail`: 관리자용 반품 내역 상세 조회
- **고객용 Plus API 섹션 추가:**
 - `GET /api/purchase_items/by_user/{user_seq}/user_bundle`: 고객별 주문 그룹화 조회 (검색 및 정렬 기능 포함)
 - `GET /api/pickups/by_user/{user_seq}/all`: 고객별 수령 내역 조회 (검색 및 정렬 기능 포함)
 - `GET /api/refunds/refund/by_user/{user_seq}/all`: 고객별 반품 내역 조회 (검색 및 정렬 기능 포함)

- **API 개요 업데이트:** 관리자 API 및 고객용 Plus API 추가 반영
- **총 엔드포인트 수정:** 약 110개 → 약 130개 이상

2026-01-04 이광태

- **채팅 API (Chatting API) 섹션 추가:**
 - **GET /api/chatting:** 전체 채팅 세션 조회
 - **GET /api/chatting/by_user_seq:** 고객별 채팅 세션 조회 (쿼리 파라미터: u_seq, is_closed)
 - **GET /api/chatting/{chatting_seq}:** 채팅 세션 상세 조회
 - **POST /api/chatting:** 채팅 세션 추가
 - **POST /api/chatting/{id}:** 채팅 세션 수정
 - **DELETE /api/chatting/{chatting_seq}:** 채팅 세션 삭제
- **채팅 API 특징:**
 - Firebase Firestore와 연동하여 실시간 채팅 지원
 - 고객과 직원 간의 채팅 세션 관리
 - 채팅 메시지는 Firebase에 저장되며, fb_doc_id로 연결
- **API 개요 업데이트:** 기본 CRUD API 15개 → 16개 테이블로 증가 (chatting 추가)
- **총 엔드포인트 수정:** 약 130개 → 약 136개 이상

2026-01-05

- **삭제된 API 섹션 제거:**
 - Receives API (입고) 전체 삭제
 - Requests API (발주) 전체 삭제
 - Purchase Item Join API 전체 삭제
 - Pickup Join API 전체 삭제
 - Receive Join API 전체 삭제
 - Request Join API 전체 삭제
- **삭제된 엔드포인트 제거:**
 - Product API: **GET /api/products**, **GET /api/products/{product_seq}**, **GET /api/products/by_maker/{maker_seq}**, **GET /api/products/{product_seq}/file_info**, **GET /api/products/{product_seq}/file**
 - User Auth Identities API: **GET /api/user_auth_identities**, **GET /api/user_auth_identities/{auth_seq}**
 - Product Join API: **GET /api/products/{product_seq}/full_detail**, **GET /api/products/by_maker/{maker_seq}/with_categories**, **GET /api/products/by_category**
- **채팅 API 업데이트:**
 - **is_closed** 파라미터 동작 상세 설명 추가
 - **is_closed=false**일 때 닫힌 채팅방 자동 재사용 기능 설명 추가
- **Refund Join API 경로 수정:**
 - 실제 경로 명시: **/api/refunds/refunds/{refund_seq}/with_details** (prefix + 라우터 경로)
- **API 개요 업데이트:** 기본 CRUD API 16개 → 11개 테이블로 감소
- **총 엔드포인트 수정:** 약 136개 → 약 100개 이상

문서 버전: 3.0
최종 수정일: 2026-01-05
최종 수정자: 김택권