

신발 매장 API 가이드 문서

버전: 2.0

최종 업데이트: 2025-01-XX

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. 특수 기능 API
9. 에러 처리
10. 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:** 15개 테이블에 대한 CRUD 작업
 - branches, users, user_auth_identities, staffs, makers
 - kind_categories, color_categories, size_categories, gender_categories, refund_reason_categories
 - products, purchase_items, pickups, refunds, receives, requests
- 인증 API:** 소셜 로그인 및 회원가입
- JOIN API:** 복잡한 조인 쿼리를 위한 6개 API 그룹
- 총 엔드포인트:** 약 110개 이상

기본 설정

서버 실행

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

또는

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

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",  
        ...  
    }  
}
```

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	/api/branches	전체 지점 조회
GET	/api/branches/{br_seq}	지점 상세 조회
POST	/api/branches	지점 추가
POST	/api/branches/{br_seq}	지점 수정
DELETE	/api/branches/{br_seq}	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/{u_seq}	고객 상세 조회
POST	/api/users	고객 추가 (이미지 필수)

메서드	엔드포인트	설명
POST	/api/users/{u_seq}	고객 수정 (이미지 제외)
POST	/api/users/{u_seq}/with_image	고객 수정 (이미지 포함)
GET	/api/users/{u_seq}/profile_image	프로필 이미지 조회
DELETE	/api/users/{u_seq}/profile_image	프로필 이미지 삭제
DELETE	/api/users/{u_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	/api/user_auth_identities	전체 인증 정보 조회
GET	/api/user_auth_identities/{auth_seq}	인증 정보 상세 조회
GET	/api/user_auth_identities/user/{u_seq}	사용자별 인증 정보 조회

메서드	엔드포인트	설명
GET	/api/user_auth_identities/provider/{provider}	제공자별 인증 정보 조회
POST	/api/user_auth_identities	인증 정보 추가
POST	/api/user_auth_identities/{auth_seq}	인증 정보 수정
POST	/api/user_auth_identities/{auth_seq}/update_login_time	마지막 로그인 시간 업데이트
DELETE	/api/user_auth_identities/{auth_seq}	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/staffs

메서드	엔드포인트	설명
GET	/api/staffs	전체 직원 조회
GET	/api/staffs/{s_seq}	직원 상세 조회
GET	/api/staffs/by_branch/{br_seq}	지점별 직원 조회
POST	/api/staffs	직원 추가 (이미지 필수)
POST	/api/staffs/{s_seq}	직원 수정
POST	/api/staffs/{s_seq}/with_image	직원 수정 (이미지 포함)
GET	/api/staffs/{s_seq}/profile_image	프로필 이미지 조회
DELETE	/api/staffs/{s_seq}/profile_image	프로필 이미지 삭제
DELETE	/api/staffs/{s_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_superset": 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/staffs" \
-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"
```

5. 제조사 (Maker)

기본 경로: /api/makers

메서드	엔드포인트	설명
GET	/api/makers	전체 제조사 조회
GET	/api/makers/{m_seq}	제조사 상세 조회
POST	/api/makers	제조사 추가
POST	/api/makers/{m_seq}	제조사 수정
DELETE	/api/makers/{m_seq}	제조사 삭제

6. 카테고리 (Categories)

6.1 종류 카테고리 (Kind Category)

기본 경로: /api/kind_categories

메서드	엔드포인트	설명
GET	/api/kind_categories	전체 조회
GET	/api/kind_categories/{kc_seq}	상세 조회
POST	/api/kind_categories	추가
POST	/api/kind_categories/{kc_seq}	수정
DELETE	/api/kind_categories/{kc_seq}	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
-----	-------	----

메서드	엔드포인트	설명
GET	/api/products	전체 제품 조회
GET	/api/products/{p_seq}	제품 상세 조회
GET	/api/products/by_maker/{m_seq}	제조사별 제품 조회
POST	/api/products	제품 추가
POST	/api/products/{p_seq}	제품 수정
POST	/api/products/{p_seq}/stock	제품 재고 수정
DELETE	/api/products/{p_seq}	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	/api/purchase_items	전체 구매 내역 조회
GET	/api/purchase_items/{b_seq}	구매 내역 상세 조회
GET	/api/purchase_items/by_user/{u_seq}	고객별 구매 내역 조회
GET	/api/purchase_items/by_datetime	분 단위 그룹화된 주문 조회
POST	/api/purchase_items	구매 내역 추가
POST	/api/purchase_items/{b_seq}	구매 내역 수정
DELETE	/api/purchase_items/{b_seq}	구매 내역 삭제

9. 수령 (Pickup)

기본 경로: [/api/pickups](#)

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

10. 반품 (Refund)

기본 경로: [/api/refunds](#)

메서드	엔드포인트	설명
GET	/api/refunds	전체 반품 내역 조회
GET	/api/refunds/{ref_seq}	반품 내역 상세 조회
GET	/api/refunds/by_user/{u_seq}	고객별 반품 내역 조회
POST	/api/refunds	반품 내역 추가
POST	/api/refunds/{ref_seq}	반품 내역 수정
POST	/api/refunds/{ref_seq}/process	반품 처리
DELETE	/api/refunds/{ref_seq}	반품 내역 삭제

11. 입고 (Receive)

기본 경로: [/api/receives](#)

메서드	엔드포인트	설명
GET	/api/receives	전체 입고 내역 조회
GET	/api/receives/{rec_seq}	입고 내역 상세 조회
GET	/api/receives/by_product/{p_seq}	제품별 입고 내역 조회
POST	/api/receives	입고 내역 추가
POST	/api/receives/{rec_seq}	입고 내역 수정

메서드	엔드포인트	설명
POST	/api/receives/{rec_seq}/process	입고 처리
DELETE	/api/receives/{rec_seq}	입고 내역 삭제

12. 발주 (Request)

기본 경로: /api/requests

메서드	엔드포인트	설명
GET	/api/requests	전체 발주 내역 조회
GET	/api/requests/{req_seq}	발주 내역 상세 조회
POST	/api/requests	발주 내역 추가
POST	/api/requests/{req_seq}	발주 내역 수정
POST	/api/requests/{req_seq}/approve_manager	팀장 결재 처리
POST	/api/requests/{req_seq}/approve_director	이사 결재 처리
DELETE	/api/requests/{req_seq}	발주 내역 삭제

인증 API

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."
}
```

2. 회원가입 완료 (2단계)

POST /api/users/{user_seq}/complete_registration

설명: 소셜 로그인 사용자의 회원가입 완료 처리

- 필수 필드: `u_phone`
- 선택 필드: `u_name` (수정), `u_address`

요청 파라미터 (Form):

- `u_name` (선택): 이름 (수정 가능)
- `u_phone` (필수): 전화번호
- `u_address` (선택): 주소

요청 예시:

```
curl -X POST "http://127.0.0.1:8000/api/users/2/complete_registration" \
-F "u_name=홍길동" \
-F "u_phone=010-1111-1111" \
-F "u_address=서울시 강남구"
```

응답 예시:

```
{
  "result": "OK",
  "message": "회원가입이 완료되었습니다"
}
```

3. 회원가입 상태 확인

GET /api/users/{user_seq}/registration_status

설명: 사용자의 회원가입 완료 상태 확인

- 미완료인 경우 누락된 필드 목록 반환

응답 예시 (완료):

```
{
  "registration_completed": true,
  "message": "회원가입이 완료되었습니다"
}
```

응답 예시 (미완료):

```
{  
    "registration_completed": false,  
    "missing_fields": ["u_phone"],  
    "message": "추가 정보 입력이 필요합니다"  
}
```

JOIN API

1. 제품 JOIN API

기본 경로: /api/products

1.1 제품 전체 상세 조회

```
GET /api/products/{p_seq}/full_detail
```

설명: 제품 + 모든 카테고리 + 제조사 정보 (6테이블 JOIN)

응답 예시:

```
{  
    "result": {  
        "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",  
        "maker_phone": "02-1111-1111",  
        "maker_address": "서울시 강남구"  
    }  
}
```

1.2 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"
```

2. 구매 내역 JOIN API

기본 경로: `/api/purchase_items`

2.1 구매 내역 상세 조회

```
GET /api/purchase_items/{b_seq}/with_details
```

설명: 구매 내역 + 고객 + 제품 + 지점 정보 (4테이블 JOIN)

2.2 구매 내역 전체 상세 조회

```
GET /api/purchase_items/{b_seq}/full_detail
```

설명: 구매 내역 + 고객 + 제품 + 지점 + 모든 카테고리 + 제조사 (9테이블 JOIN)

2.3 분 단위 그룹화된 주문 조회

```
GET /api/purchase_items/by_datetime/with_details
```

파라미터:

- `user_seq` (필수): 고객 번호
- `order_datetime` (필수): 주문 일시 (YYYY-MM-DD HH:MM 형식)
- `branch_seq` (필수): 지점 번호

3. 수령 JOIN API

기본 경로: [/api/pickups](#)

3.1 수령 상세 조회

```
GET /api/pickups/{pic_seq}/with_details
```

설명: 수령 + 구매 내역 + 고객 + 제품 + 지점 정보 (5테이블 JOIN)

3.2 수령 전체 상세 조회

```
GET /api/pickups/{pic_seq}/full_detail
```

설명: 수령 + 구매 내역 + 고객 + 제품 + 지점 + 모든 카테고리 + 제조사 (10테이블 JOIN)

4. 반품 JOIN API

기본 경로: [/api/refunds](#)

4.1 반품 상세 조회

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

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

4.2 반품 전체 상세 조회

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

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

5. 입고 JOIN API

기본 경로: [/api/receives](#)

5.1 입고 상세 조회

```
GET /api/receives/{rec_seq}/with_details
```

설명: 입고 + 직원 + 제품 + 제조사 정보 (4테이블 JOIN)

5.2 입고 전체 상세 조회

```
GET /api/receives/{rec_seq}/full_detail
```

설명: 입고 + 직원 + 제품 + 제조사 + 모든 카테고리 정보 (9테이블 JOIN)

6. 발주 JOIN API

기본 경로: /api/requests

6.1 발주 상세 조회

```
GET /api/requests/{req_seq}/with_details
```

설명: 발주 + 직원 + 제품 + 제조사 정보 (4테이블 JOIN)

6.2 발주 전체 상세 조회

```
GET /api/requests/{req_seq}/full_detail
```

설명: 발주 + 직원 + 제품 + 제조사 + 모든 카테고리 정보 (9테이블 JOIN)

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."}

# 2. 회원가입 완료 (2단계: 추가 정보 입력)
curl -X POST "http://127.0.0.1:8000/api/users/2/complete_registration" \
-F "u_phone=010-1111-1111" \
-F "u_address=서울시 강남구"

# 응답: {"result": "OK", "message": "회원가입이 완료되었습니다."}
```

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/1/full_detail"  
  
# 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=주문완료"
```

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

- 이미지 업로드: 고객/직원 추가 시 이미지는 필수입니다.
- 외래 키 제약: 참조하는 테이블의 데이터가 먼저 존재해야 합니다.
- UNIQUE 제약: u_email, s_id, u_phone, s_phone 등은 중복될 수 없습니다.
- 주문 그룹화: 같은 분에 구매한 항목만 그룹화되므로, 정확한 시간 설정이 중요합니다.
- 소프트 삭제: u_quit_date, s_quit_date를 설정하여 탈퇴 처리를 할 수 있습니다.
- 로컬 회원가입: user 테이블과 user_auth_identities 테이블을 별도 API로 생성해야 합니다.
- 소셜 로그인: user 테이블과 user_auth_identities 테이블이 하나의 API에서 함께 생성됩니다.

추가 리소스

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

문서 버전: 2.0

최종 업데이트: 2025-01-XX