**📋 카테고리 시스템 전체 구조 분석 및 개선 방향**

**🎯 현재 시스템 문제점**

1. **코드 입력 불편: 상위 레벨 코드를 매번 수동 입력**
2. **용도 혼재: 카테고리 관리 ≠ 제품 관리가 섞여 있음**
3. **워크플로우 분리 필요: 코드 등록 → 제품 관리로 연결되어야 함**

**🔄 개선된 워크플로우**

**📝 카테고리 관리 (categories.py)**

**├── 1단계: 6단계 카테고리 계층 생성**

**├── 2단계: 코드 등록 (A-001-002-003-004-005-006)**

**└── 3단계: 등록된 코드 목록 관리**

**⬇️ 자동 연결**

**📦 제품 관리 (products.py)**

**├── 1단계: 등록된 카테고리 코드 목록 표시**

**├── 2단계: 코드 선택 → 제품 정보 입력**

**└── 3단계: 실제 제품으로 등록 (원가, 마진 등)**

**🛠️ 필요한 개선 사항**

**A. 카테고리 관리 페이지 개선**

1. **드롭다운 기반 코드 입력**
   * **Level 1: A, B, C... 선택**
   * **Level 2: A-001, A-002... 선택 (Level 1 기반)**
   * **Level 3: A-001-001, A-001-002... 선택 (Level 2 기반)**
2. **코드 등록 테이블**

**카테고리 | 완성 코드 | 카테고리명 | 설명 | [코드등록] |**

**A | A-001-002-003-004 | XXXX | XXX | [등록] |**

1. **등록 완료 처리**
   * **성공 메시지 + 입력창 리셋**
   * **product\_categories 테이블에 저장**

**B. 제품 관리 페이지 연동**

1. **등록된 코드 목록 표시**

**등록된 카테고리 코드 | 제품명 | 제품설명 | 단위 | 원가 | [제품등록] |**

**A-001-002-003-004 | [입력] | [입력] | EA | 0 | [등록] |**

1. **제품 등록 처리**
   * **제품 코드: A-001-002-003-004-PRD**
   * **products 테이블에 저장**
   * **원가, 마진율 등 비즈니스 정보 추가**

**🔧 수정할 파일들**

**1. categories.py (전체 개선 필요)**

* **드롭다운 기반 계층 입력**
* **코드 등록 기능**
* **제품 관리 연동 제거**

**2. products.py (신규 개발 필요)**

* **카테고리 코드 기반 제품 목록**
* **제품 등록 기능**
* **원가/마진 관리**

**📝 개발 순서**

**Step 1: categories.py 전체 새로 작성 (코드 등록 전용) Step 2: products.py 신규 개발 (제품 관리 전용)  
Step 3: 두 모듈 간 데이터 연동 테스트**

# 카테고리 시스템 코딩 방향 최종 보고서

## 1. 코드 수정 및 삭제 로직 설계

### 1.1 삭제 로직 (계층형 연쇄 삭제)

계층 구조에서 특정 카테고리 코드(A-001-002 등)를 삭제할 경우, 해당 코드뿐 아니라 모든 하위 계층 코드들도 함께 삭제되어야 합니다.

이를 위해 다음 단계가 필요합니다.

* 재귀적 자식 검색: 삭제하려는 코드의 모든 하위 계층을 재귀적으로 탐색합니다.
* 제품 등록 확인: 하위 계층 중 제품으로 등록된 코드가 있는지 확인합니다.
* 안전 삭제: 하위 또는 해당 코드에 제품이 등록된 경우 삭제를 차단하고, 그렇지 않은 경우 전체 계층을 삭제합니다.

예시: A-001-002 삭제 시, 하위 코드(A-001-002-003, A-001-002-003-004 등) 모두 자동으로 삭제됩니다.

### 1.2 수정 로직 (계층형 연쇄 수정)

카테고리 코드(A-001 등)를 수정하는 경우, 변경된 코드가 반영된 형태로 모든 하위 계층 코드들도 일괄적으로 수정되어야 합니다.

* 코드 중복 확인: 새로운 코드가 이미 존재하는지 검증합니다.
* 자식 코드 검색: 수정하려는 코드의 모든 하위 계층을 탐색합니다.
* 일괄 업데이트: 부모 코드 변경 후 모든 자식 코드를 자동으로 업데이트합니다.

예시: A-001 → A-999로 수정 시, 하위 코드(A-001-002 등)는 A-999-002 등으로 자동 변경됩니다.

## 2. 데이터베이스 구조 최적화

### 2.1 핵심 테이블 설계

데이터베이스에는 product\_categories 테이블을 중심으로 계층형 카테고리 구조를 관리합니다. 주요 컬럼 및 특성은 다음과 같습니다.

* category\_id (Primary Key)
* category\_code (Unique, 계층형 코드)
* category\_name (카테고리명)
* parent\_category\_id (부모 ID 참조)
* category\_level (1~6 레벨)
* full\_path (전체 경로 저장, 성능 최적화용)
* is\_active (활성화 상태)

참조 무결성을 위해 외래키 제약조건을 두며, ON DELETE CASCADE, ON UPDATE CASCADE 옵션을 사용하여 계층 구조 내 데이터의 연속성을 보장합니다.

## 3. 코딩 구현 방향

### 3.1 카테고리 관리 (categories.py)

* 계층 생성 기능:
* 드롭다운 기반 상위 코드 선택
* 신규 코드 입력 및 검증
* 계층 구조 저장

코드 수정 기능:

수정할 코드 선택

중복 검증

하위 계층 자동 업데이트

제품 등록 여부 확인

코드 삭제 기능:

삭제할 코드 선택

하위 계층 검색

제품 등록 여부 확인

연쇄 삭제 실행

코드 목록 관리:

계층별 코드 목록 표시

수정/삭제 버튼 제공

실시간 상태 업데이트

### 3.2 제품 관리 (products.py) 연동 기능

* 등록된 카테고리 코드 목록 조회
* 코드 선택 후 제품 정보 입력
* 실제 제품 등록(원가, 마진 등)

## 4. UI/UX 설계

### 4.1 카테고리 관리 화면 구성

* 6단계 카테고리 관리
* 탭1: 코드 생성(드롭다운 기반)
* 탭2: 코드 목록(수정/삭제 기능)
* 탭3: 코드 검색(조회 기능)

### 4.2 코드 목록 테이블 예시

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 레벨 | 카테고리 코드 | 카테고리명 | 상태 | 액션 |
| 1 | A-001 | 전자제품 | 활성 | [수정][삭제] |
| 2 | A-001-002 | 컴퓨터 | 활성 | [수정][삭제] |
| 3 | A-001-002-003 | 데스크톱 | 제품등록 | [수정][삭제불가] |

## 5. 안전장치 구현

### 5.1 삭제 방지 조건

* 해당 코드가 제품으로 등록되어 있을 경우 삭제 방지
* 하위 코드 중 제품 등록된 것이 있는 경우 삭제 방지
* 삭제 전 경고 메시지 및 사용자 확인 절차 제공

### 5.2 수정 제한 조건

* 동일 레벨에서 중복 코드 생성 차단
* 제품 등록된 코드 수정 시 제품 정보도 함께 업데이트

## 6. 개발 우선순위

1. Phase 1: 기본 CRUD
2. 코드 생성(드롭다운 기반)
3. 코드 목록 조회
4. 기본 수정/삭제 기능
5. Phase 2: 고급 기능
6. 계층형 연쇄 수정/삭제
7. 제품 등록 연동 확인
8. 안전장치 구현
9. Phase 3: 최적화
10. 성능 최적화
11. UI/UX 개선
12. 에러 처리 강화

**카테고리 관리 시스템 전체 재설계안**

**시스템 아키텍처**

**데이터 구조**

카테고리 테이블 구조:

- category\_id (PK)

- category\_code (A-HR-ST-VV-20-MAE-00)

- category\_name (각 레벨별 이름)

- category\_level (1~6)

- parent\_category\_id (상위 카테고리 참조)

- is\_active (활성화 상태)

**핵심 비즈니스 로직**

**1. 카테고리명 표시 규칙**

python

def get\_display\_name(category\_code):

"""

A-HR-ST-VV-20-MAE-00 → "Hot Runner" (A의 Level 1 이름)

B-IT-HW-LA-15-DEL-01 → "Information Technology" (B의 Level 1 이름)

"""

main\_code = category\_code.split('-')[0] *# A, B, C 추출*

return get\_level1\_name\_by\_main\_code(main\_code)

**2. 삭제 처리 로직**

python

def immediate\_delete(category\_id):

"""

1. 클릭 즉시 실행 (확인 없음)

2. 하위 카테고리 자동 비활성화

3. 본인 비활성화

4. UI 즉시 업데이트

"""

**3. 편집 처리 로직**

python

def inline\_edit():

"""

1. 행 더블클릭으로 편집 모드

2. Enter로 저장, Esc로 취소

3. 하위 코드 자동 업데이트

"""

**UI 설계**

**Tab 구성**

Tab 1: 코드 생성 (기존 유지)

Tab 2: 코드 목록 (완전 재설계)

Tab 3: 통계 및 검색 (단순화)

**Tab 2: 코드 목록 UI**

┌────────────────────────────────────────────────────────────────┐

│ 필터: [전체▼] [활성▼] [검색창\_\_\_\_\_\_\_\_\_\_\_\_] [새로고침] │

├────────────────────────────────────────────────────────────────┤

│ 레벨 │ 카테고리 코드 │ 제품명 │ 자체명 │ 액션 │

├────────────────────────────────────────────────────────────────┤

│ L6 │ A-HR-ST-VV-20-MAE-00│ Hot Runner │ Revision │ ✏️ 🗑️ │

│ L6 │ A-HR-ST-VV-20-MVE-00│ Hot Runner │ NZ Dim │ ✏️ 🗑️ │

│ L5 │ A-HR-ST-VV-20 │ Hot Runner │ 20mm │ ✏️ 🗑️ │

│ L1 │ A-HR │ Hot Runner │ Hot Runner│ ✏️ 🗑️ │

└────────────────────────────────────────────────────────────────┘

**코드 구조**

**메인 함수 구조**

python

def category\_management\_page():

tab1, tab2, tab3 = st.tabs(["코드 생성", "코드 목록", "통계"])

with tab1:

create\_category\_interface()

with tab2:

category\_list\_interface() *# 완전 재작성*

with tab3:

statistics\_interface()

**핵심 헬퍼 함수들**

python

def get\_level1\_category\_name(category\_code):

"""Level 1 카테고리명 조회"""

def delete\_category\_immediate(category\_id):

"""즉시 삭제 처리"""

def update\_category\_inline(category\_id, new\_data):

"""인라인 업데이트"""

def cascade\_update\_children(old\_code, new\_code):

"""하위 코드 연쇄 업데이트"""

**성능 최적화**

**데이터 로딩 전략**

python

def load\_categories\_optimized():

"""

1. 모든 카테고리 한 번에 로드

2. Python에서 메모리 내 처리

3. Level 1 이름 매핑 테이블 생성

"""

**캐싱 전략**

python

@st.cache\_data

def get\_level1\_mapping():

"""Level 1 카테고리명 매핑 캐시"""

return {

'A': 'Hot Runner',

'B': 'Information Technology',

...

}

**에러 처리**

**삭제 안전장치**

python

def safe\_delete(category\_id):

try:

*# 제품 등록 여부 확인 (향후 확장)*

*# 하위 카테고리 처리*

*# 삭제 실행*

return True

except Exception as e:

st.error(f"삭제 실패: {e}")

return False

**수정 검증**

python

def validate\_category\_update(category\_id, new\_data):

*# 중복 코드 확인*

*# 형식 검증*

*# 비즈니스 룰 확인*

**사용자 경험**

**즉시 피드백**

* 삭제: 클릭 → 즉시 삭제 → "삭제됨" 토스트
* 수정: 입력 → Enter → "저장됨" 토스트
* 에러: 명확한 에러 메시지

**키보드 단축키**

* Enter: 저장
* Esc: 취소
* Delete: 삭제 (선택된 행)

**확장성 고려사항**

**향후 기능 연동**

python

def check\_product\_usage(category\_code):

"""제품 등록 여부 확인 (제품 관리 모듈 연동)"""

def export\_categories():

"""CSV 내보내기"""

def import\_categories():

"""CSV 가져오기"""

**구현 단계**

**Phase 1: 핵심 기능 (2시간)**

1. Level 1 이름 표시 로직
2. 즉시 삭제 기능
3. 기본 목록 UI

**Phase 2: 사용성 개선 (1시간)**

1. 인라인 편집
2. 필터링 개선
3. 에러 처리

**Phase 3: 최적화 (30분)**

1. 성능 최적화
2. 캐싱 적용
3. 최종 테스트

**카테고리 시스템 완전 재설계 보고서**

**현재 문제점 분석**

**문제 1:** Level 6 완성 코드에 제품 등록 버튼 없음 **문제 2:** 카테고리 관리와 제품 관리의 역할 분리 불명확 **문제 3:** 워크플로우가 중단된 상태

**전체 시스템 구조 재설계**

**Phase 1: 카테고리 코드 생성 (현재 완료)**

목적: 6단계 계층 구조 코드 생성

결과: A-HR-ST-VV-20-MAE-00 (완성된 코드)

위치: 카테고리 관리 모듈

**Phase 2: 제품 코드 등록 (누락된 부분)**

목적: 완성된 카테고리 코드를 실제 제품으로 등록

기능:

- 완성 코드 목록 표시

- 제품 기본 정보 입력 (제품명, 설명, 단위, 원가)

- 제품 코드 생성 (A-HR-ST-VV-20-MAE-00-PRD)

- products 테이블에 저장

**Phase 3: 제품 관리 (별도 모듈)**

목적: 등록된 제품의 상세 관리

기능:

- 원가 관리 (구매가, 물류비, 마진율)

- 판매가 설정

- 재고 관리 연동

- 견적서/발주서에서 사용

**구체적 구현 방안**

**1. 카테고리 관리 모듈 수정**

**Tab 2: 코드 목록**에 제품 등록 기능 추가

현재: 레벨 | 코드 | 제품명(L1) | 자체명 | 액션[편집][삭제]

수정: 레벨 | 코드 | 제품명(L1) | 자체명 | 등록상태 | 액션[편집][삭제][제품등록]

**제품 등록 버튼 조건:**

* Level 6 카테고리만 표시
* 이미 제품으로 등록된 경우 "등록완료" 표시
* 미등록된 경우 "제품등록" 버튼 표시

**2. 제품 등록 프로세스**

Step 1: Level 6 코드에서 [제품등록] 버튼 클릭

Step 2: 제품 정보 입력 모달창

- 제품명 (필수)

- 제품 설명

- 단위 (EA, Set, Pcs 등)

- 기본 원가

Step 3: 제품 코드 자동 생성 (카테고리코드-PRD)

Step 4: products 테이블에 저장

Step 5: 제품 관리 모듈에서 관리 가능

**3. 데이터베이스 연동**

카테고리 테이블: product\_categories (현재)

제품 테이블: products (기존 테이블 활용)

연결: category\_code 기반 매핑

**수정이 필요한 부분**

**1. 카테고리 목록 화면 개선**

* Level 6 완성 코드에 제품 등록 상태 표시
* 제품 등록 버튼 추가
* 등록 완료된 항목 시각적 구분

**2. 제품 등록 모달창 추가**

* 카테고리 코드 표시 (읽기 전용)
* 제품 정보 입력 폼
* 제품 코드 미리보기
* 등록 완료 후 자동 페이지 새로고침

**3. 제품 관리 모듈 연동**

* 등록된 제품 목록 표시
* 카테고리별 필터링
* 제품 상세 정보 관리

**우선 구현 순서**

**1단계 (즉시):** 카테고리 목록에 제품 등록 버튼 추가 **2단계:** 제품 등록 모달창 구현 **3단계:** products 테이블 연동 **4단계:** 제품 관리 모듈과 연결