**YMV 비즈니스 시스템 개발 현황 보고서 (최종 통합판)**

**🎯 개발 방향 및 목표**

**시스템 개요**

* **대상: 베트남 소재 한국 기업 (직원 20명)**
* **목적: 영업, 총무, 인사 통합 관리**
* **기술 스택: Python/Streamlit + PostgreSQL(Supabase)**
* **배포: Streamlit Cloud + GitHub**

**핵심 비즈니스 정책**

* **통화 정책: 사내 운영(VND), 해외 거래(USD)**
* **문서 번호: YMV-Qyymmdd-count (견적서), YMV-Pyymmdd-count (발주서)**
* **직원 ID: YYMMDD-Count 형식**
* **권한 관리: Master 계정 + 메뉴별 권한**

**제품 코드 체계 (신규 추가)**

**A-001-002-003-004-005-006**

**│ │ │ │ │ │ │**

**│ │ │ │ │ │ └── Level 6 (3자리)**

**│ │ │ │ │ └────── Level 5 (3자리)**

**│ │ │ │ └────────── Level 4 (3자리)**

**│ │ │ └────────────── Level 3 (3자리)**

**│ │ └────────────────── Level 2 (3자리)**

**│ └────────────────────── Level 1 (3자리)**

**└───────────────────────── 대분류 (A~I)**

**구매 시스템 분류**

**1. 일반 구매 (총무 관리)**

* **사무용품, 현장용품, 운영비 등**
* **단순 비용 처리, 현금 흐름에만 기록**

**2. 판매용 제품 구매 (영업 관리)**

* **재판매 목적의 제품 구매**
* **원가 관리 및 마진 분석 필요**
* **전체 판매 프로세스와 연동**

**📋 비즈니스 프로세스 플로우**

**판매용 제품 관리 프로세스**

**1. 6단계 카테고리 체계 구축 (A~I, Level 1~6)**

**↓**

**2. 제품 등록 (카테고리 코드 + 원가, 물류비, 마진율, 표준 판매가격)**

**↓**

**3. 제품 구매 (제품 코드 기반 발주)**

**↓**

**4. 견적서 작성 (등록된 제품 코드 기반)**

**↓**

**5. 고객 주문 확정**

**↓**

**6. 세금계산서 발행**

**↓**

**7. 금액 회수 (현금 흐름 자동 연동)**

**↓**

**8. 수익성 분석 (원가 대비 실제 판매가)**

**📊 모듈별 개발 현황**

**1. 인증 시스템 (완료 100%)**

**파일: app/modules/auth/login.py**

* **상태: 구현 완료, 테스트 통과**
* **기능: Master 계정 인증, JWT 토큰, bcrypt 암호화, 권한 확인**

**2. 사용자 관리 (완료 100%)**

**파일: app/modules/system/users.py**

* **상태: 구현 완료, 편집/삭제 포함**
* **기능: 직원 ID 자동 생성(YYMMDD-Count), 베트남 기업 기준 부서/직급, CRUD, 권한 관리**

**3. 데이터베이스 시스템 (완료 100%)**

**파일: app/shared/database.py, database/init\_db.sql**

* **상태: 17개 테이블, Supabase 연결 확인**

**4. 6단계 카테고리 시스템 (신규 구현 완료 100%)**

**파일: app/modules/system/categories.py**

* **상태: 방금 구현 완료**
* **기능:**
  + **A6 계층 구조**
  + **부모-자식 관계 코드 체계**
  + **카테고리 트리 시각화**
  + **자동 코드 생성기**
  + **CRUD 기능**

**5. 총무 관리 - 일반 구매 (진행 80%)**

**파일: app/modules/admin/general\_affairs.py**

* **상태: 기본 구매 관리 구현, 메뉴 구조 정리 필요**
* **기능: 구매 등록/조회/수정/삭제, 현금 흐름 관리**

**6. 제품 관리 - 판매용 제품 (진행 30%)**

**파일: app/modules/system/products.py**

* **현재: 기본 틀만 구현**
* **필요 확장:**
  + **6단계 카테고리 연동**
  + **원가 관리 (구매가, 물류비, 기타 비용)**
  + **마진율 설정 및 표준 판매가격 자동 계산**
  + **제품별 수익성 분석**
  + **재고 관리 연동**

**7. 영업 관리 - 판매용 제품 구매 (미구현 0%)**

**파일: 미생성**

* **필요 기능:**
  + **제품 코드 기반 발주 시스템**
  + **공급업체별 제품 관리**
  + **구매가 변동 히스토리 추적**
  + **원가 구성 요소별 세분화 관리**

**8. 견적서 관리 (미구현 0%)**

**파일: 미생성**

* **필요 기능:**
  + **등록된 제품 코드 기반 견적서 작성**
  + **수량별 가격 계산 (마진율 자동 적용)**
  + **PDF 생성 및 YMV-Qyymmdd-count 문서 번호 자동 생성**
  + **견적서 → 주문서 전환 워크플로우**

**9. 고객 관리 (미구현 0%)**

**파일: 미생성**

* **필요 기능: 고객 정보 관리, 거래 히스토리, 신용 관리**

**10. 주문 및 세금계산서 관리 (미구현 0%)**

**파일: 미생성**

* **필요 기능: 주문서 관리, 세금계산서 발행, 배송 관리**

**11. 현금 흐름 연동 시스템 (진행 60%)**

**파일: app/modules/admin/general\_affairs.py**

* **현재: 기본 현금 흐름 관리**
* **필요 확장: 판매 수금 자동 연동, 제품 구매와 연결**

**12. 수익성 분석 시스템 (미구현 0%)**

**파일: 미생성**

* **필요 기능:**
  + **제품별 원가 대비 실제 수익률 분석**
  + **고객별 수익성 분석**
  + **월별/분기별 수익 트렌드 분석**
  + **마진율 최적화 제안**

**⚠️ 주요 문제점 및 해결 과제**

**1. 구매 시스템 이원화 필요**

* **문제: 현재 단일 구매 시스템만 존재**
* **해결: 총무용 일반 구매 vs 영업용 제품 구매 완전 분리**

**2. 제품-카테고리 연동 시스템**

* **문제: 카테고리 시스템과 제품 관리가 분리됨**
* **해결: 6단계 카테고리 기반 제품 코드 자동 생성 연동**

**3. 전체 판매 프로세스 통합**

* **문제: 각 모듈이 독립적 작동**
* **해결: 카테고리 → 제품 → 구매 → 견적 → 판매 → 수금 전체 연동**

**4. 원가 및 수익성 분석 부재**

* **문제: 단순 제품 정보만 관리, 실제 수익률 파악 불가**
* **해결: 실시간 원가 추적 및 수익성 분석 대시보드**

**🚀 개발 우선순위 (수정)**

**Phase 1: 핵심 인프라 완성**

1. **제품 관리 확장 - 6단계 카테고리 연동, 원가 관리**
2. **영업용 제품 구매 시스템 - 제품 코드 기반 발주**
3. **메뉴 구조 정리 - 일반 구매와 제품 구매 분리**

**Phase 2: 판매 프로세스 구축**

1. **견적서 관리 - 제품 코드 기반 견적서 작성**
2. **고객 관리 - 기본 CRUD 및 거래 히스토리**
3. **주문 및 세금계산서 - 판매 완료 프로세스**

**Phase 3: 통합 분석 시스템**

1. **현금 흐름 자동 연동 - 판매 수금과 실시간 연결**
2. **수익성 분석 - 제품별, 고객별, 기간별 분석**
3. **전체 워크플로우 최적화**

**📈 전체 진행률**

**완료 (40%)**

* **인증 시스템 (100%)**
* **사용자 관리 (100%)**
* **데이터베이스 (100%)**
* **카테고리 시스템 (100%)**

**진행 중 (25%)**

* **총무 관리 (80%)**
* **제품 관리 (30%)**
* **현금 흐름 (60%)**

**미구현 (35%)**

* **영업용 제품 구매 (0%)**
* **견적서 관리 (0%)**
* **고객 관리 (0%)**
* **수익성 분석 (0%)**

**현재 시스템은 기본 관리 기능과 카테고리 체계는 완성되었으나, 핵심 비즈니스인 제품 중심의 통합 판매 프로세스 구축이 시급한 상태입니다.**

**Step 6: 6단계 카테고리 시스템 구현**

**완료**

**정리 내용에 추가 해줘. 코드 개발할때는 전체코드를 니가 제작 해야할지 아니면 수정 코드를 진행 할지 문의 하고 나서 그 답에 따라서 진행 하고 , 모든 업무는 한단계씩 진행 함**

**YMV 비즈니스 시스템 개발 현황 보고서 (업데이트)**

**🎯 개발 방향 및 목표**

**시스템 개요**

* **대상: 베트남 소재 한국 기업 (직원 20명)**
* **목적: 영업, 총무, 인사 통합 관리**
* **기술 스택: Python/Streamlit + PostgreSQL(Supabase)**
* **배포: Streamlit Cloud + GitHub**

**핵심 비즈니스 정책**

* **통화 정책: 사내 운영(VND), 해외 거래(USD)**
* **문서 번호: YMV-Qyymmdd-count (견적서), YMV-Pyymmdd-count (발주서)**
* **직원 ID: YYMMDD-Count 형식**
* **권한 관리: Master 계정 + 메뉴별 권한**

**제품 코드 체계**

**A-001-002-003-004-005-006**

**│ │ │ │ │ │ │**

**│ │ │ │ │ │ └── Level 6 (3자리)**

**│ │ │ │ │ └────── Level 5 (3자리)**

**│ │ │ │ └────────── Level 4 (3자리)**

**│ │ │ └────────────── Level 3 (3자리)**

**│ │ └────────────────── Level 2 (3자리)**

**│ └────────────────────── Level 1 (3자리)**

**└───────────────────────── 대분류 (A~I)**

**개발 진행 규칙 (신규 추가)**

1. **코드 개발 전 확인: 전체 코드 제작 vs 수정 코드 방식 문의 후 진행**
2. **단계별 진행: 모든 업무를 한 단계씩 순차 진행**
3. **완료 확인: 각 단계 완료 후 다음 단계 진행**

**📊 모듈별 개발 현황 (업데이트)**

**1. 인증 시스템 (완료 100%)**

**파일: app/modules/auth/login.py**

* **상태: 구현 완료, 테스트 통과**

**2. 사용자 관리 (완료 100%)**

**파일: app/modules/system/users.py**

* **상태: 구현 완료, 편집/삭제 포함**

**3. 데이터베이스 시스템 (완료 100%)**

**파일: app/shared/database.py, database/init\_db.sql**

* **상태: 17개 테이블, Supabase 연결 확인**

**4. 6단계 카테고리 시스템 (완료 100%) ★ 신규 완료**

**파일: app/modules/system/categories.py**

* **상태: 구현 완료**
* **기능:**
  + **A6 계층 구조**
  + **부모-자식 관계 코드 체계**
  + **카테고리 트리 시각화**
  + **자동 코드 생성기 (A-001-002-003-004-005-006 형식)**
  + **카테고리 CRUD 기능 (생성/조회/수정/삭제)**
  + **제품 코드 생성기 (Level 6 카테고리 기반)**

**5. 총무 관리 - 일반 구매 (진행 80%)**

**파일: app/modules/admin/general\_affairs.py**

* **상태: 메뉴 구조 정리 필요**

**6. 제품 관리 - 판매용 제품 (진행 30%)**

**파일: app/modules/system/products.py**

* **다음 작업: 6단계 카테고리 시스템과 연동 필요**

**다음 단계 계획**

**Step 7: 제품 관리와 카테고리 시스템 연동**

**제품 등록 시 6단계 카테고리 코드를 기반으로 제품 코드를 생성하는 시스템 구축**

**질문: products.py 파일의 제품 관리 모듈을 카테고리 시스템과 연동하도록 전체 코드를 새로 제작할까요, 아니면 기존 코드를 수정하는 방식으로 진행할까요?**

**카테고리 화면구성**

**Step 6-3: 카테고리 입력 인터페이스 개선**

**categories.py 파일을 수정할까요, 아니면 새로운 구조로 전체 코드를 다시 작성할까요?**

**현재 파일이 이미 복잡하므로 전체 코드를 새로 작성하는 것을 추천합니다.**

**어떤 방식으로 진행하시겠습니까?**

1. **전체 코드 새로 작성 (추천)**
2. **기존 코드 수정**

**선택해주시면 다음과 같은 구조로 만들겠습니다:**

**대분류: A 카테고리명: 전자제품**

**레벨1: 001 카테고리명: 컴퓨터 코드: A-001**

**레벨2: 002 카테고리명: 데스크톱 코드: A-001-002**

**레벨3: 003 카테고리명: 게이밍PC 코드: A-001-002-003**

**레벨4: 004 카테고리명: Intel기반 코드: A-001-002-003-004**

**레벨5: 005 카테고리명: i5시리즈 코드: A-001-002-003-004-005**

**레벨6: 006 카테고리명: 특정모델 코드: A-001-002-003-004-005-006**

**제품 코드 구현 기준  
제품 코드 생성 시스템 개발 현황 보고서**

**현재 상황 분석**

**문제점 발견**

**현재 카테고리 목록 화면이 너무 복잡하게 표시되고 있습니다. 6단계 모든 레벨이 개별적으로 나열되어 사용자 경험이 좋지 않습니다.**

**요구사항 정리**

1. **단순화된 표시: 최종 완성 코드 1줄만 표시**
2. **제품 등록 버튼: "최종 코드 등록" 버튼으로 영업용 제품 코드 등록**
3. **시스템 연동: 제품 관리와 자동 연동**

**개선된 제품 코드 생성 플로우**

**카테고리 계층 생성 → 최종 코드 확정 → 제품 등록 → 영업 문서 활용**

**1단계: 카테고리 계층 생성 (현재 완료)**

* **6단계 입력 인터페이스**
* **A-001-002-003-004-005-006 형식 생성**

**2단계: 최종 코드 표시 및 등록 (개선 필요)**

**현재 문제: 모든 레벨이 개별 표시됨 개선 방향:**

**최종 제품 코드: A-HR-ST-VV-20-MAE-00**

**제품명: [입력란]**

**설명: [입력란]**

**[최종 코드 등록] 버튼**

**3단계: 제품 관리 연동 (미구현)**

* **등록된 코드가 제품 관리로 자동 전송**
* **원가, 마진율 등 추가 정보 입력**
* **영업 문서에서 활용 가능한 상태**

**다음 개발 단계**

**Step 6-5: 카테고리 목록 화면 개선**

**수정 방향:**

* **Level 6 (최종) 카테고리만 표시**
* **제품 등록 기능 추가**
* **제품 관리와 연동**

**Step 6-6: 제품 관리 연동**

* **카테고리 코드 기반 제품 등록**
* **제품 기본 정보 입력 인터페이스**

**제품 중심 통합 워크플로우**

카테고리 코드 생성 → 제품 등록 → 원가 관리 → 견적서 작성 → 판매 → 수익 분석