

# StockSense: AI Inventory Management

Taehan Kim, Chaewon Mun, Donghwan Seo, Namgyu Youn, Jaehoon Lee, Jiwoo Jang

## Introduction

최근 유통업계는 대형마트 및 소매점의 점포 감소, 무인점포의 경제적 한계, 소상공인의 폐업 증가 등 구조적 변화를 겪고 있다.

메트로신문에 따르면, 대형마트는 소비 트렌드 변화와 온라인 쇼핑 확산으로 최근 7년간 37개 점포를 폐점하며 운영 효율성 강화를 추진하고 있다. 한국경제는 무인점포의 낮은 수익성이 지속 가능성을 위협한다고 보도했으며, 서울경제는 경기 침체와 소비 위축으로 자영업자의 경영 환경이 코로나19 팬데믹보다 악화되었다고 분석했다.

이러한 변화는 비효율적인 재고 관리, 수동 데이터 분석 한계, 소비 트렌드 및 리콜 정보 부족 등의 문제를 초래하며, 기업과 소상공인의 부담을 가중시키고 있다. 이를 해결하기 위해 AI 기반 실시간 데이터 분석 및 자동화된 재고 관리 시스템이 필요하다.

## Proposed Solution

본 연구에서는 AI 기반 실시간 재고 관리 시스템을 통해 유통업의 비효율적인 재고 운영 문제를 해결하고자 한다.

**AI 기반 실시간 재고 관리.** 판매량을 예측하고, 재고 부족 시 관리자에게 즉각 알림을 제공한다.

**통합 대시보드 제공.** 재고 및 판매 데이터를 시각화하여 효율적인 의사결정을 지원한다.

**맞춤형 매장 관리 솔루션.** 트렌드 상품 추천 및 리콜 분석을 통해 최적의 재고 운영을 가능하게 한다.

## Team and Responsibilities

김태한	- Webshop 가장 환경 세팅 - 주문 Agent(ReAct + Reflexion) 개발 - n8n 기반 Workflow 자동화
문채원	- n8n 기반 Workflow 자동화 - 웹 프론트엔드(UI/UX) 개발 - 챗봇 프롬프트 엔지니어링
서동환	- 프론트엔드(UI/UX), 백엔드 구현 - 챗봇 API 연결 및 프롬프트 튜닝 - 기간 별 매출 및 인기 상품 데이터 조회 구현
윤남규	- DB(PostgreSQL - Supabase) setup, migration - n8n recall workflow 구현 - Airflow 기반의 Category Searching 구현
이재훈	- Dashboard 백엔드 및 프론트엔드 구현 - 판매량 예측 모델 개발
장지우	- Inventory 백엔드 및 프론트엔드 구현 - 자동 주문과 Dashboard 실시간 통합 아키텍처 설계

Table 1. 구성원 및 역할 소개

## System Overview

본 프로젝트에서는 AI 기반 재고 관리 시스템의 구성과 주요 기능을 설명한다.

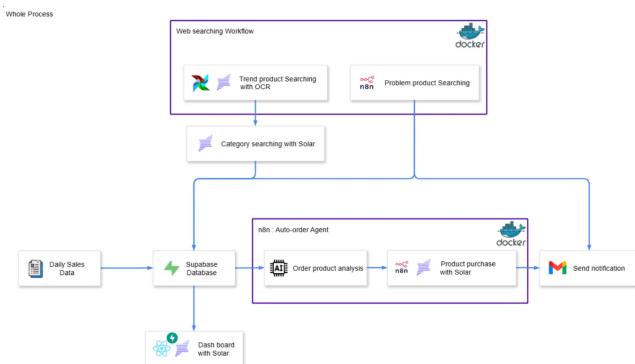


Fig. 1. Whole Workflow

## Database and Data Sources.

- **Database:** Supabase 기반 PostgreSQL을 사용하여 원격 네트워크 접속 및 다중 사용자 동시 접근을 지원했다.
- **Datasets:** Dacon의 온라인 채널 제품 판매량 예측 AI 온라인 해커톤 시계열 학습 데이터 및 Amazon Categories 9,500개 taxonomy를 활용하여 서비스 완성도를 향상시켰다.

## Trend Workflow.

1. Naver DataLab Shopping Insight 크롤링
2. Document OCR API를 활용한 트렌드 상품 추출
3. Solar Chat API를 통한 연관 카테고리 서치

## Recall Workflow.

1. 제품안전정보센터 크롤링
2. 매칭된 물품을 주문 DB에 반영

## Dashboard.

- Solar Chat API를 활용한 챗봇 구현
- 매출 데이터 시각화 대시보드 제공
- 재고 데이터 시각화 대시보드 제공

## Automated Ordering Agent.

1. 최적 재고량 계산
2. Solar Chat API를 활용한 Agent 구축
3. 자동주문된 재고를 DB에 반영

**Trend Workflow.** Trend Workflow는 Naver Datalab에서 실시간 트렌드 상품 정보를 수집해 자동 주문에 필요한 상품을 파악하는 과정이다. 이 워크플로우의 주요 내용은 다음과 같다.

- **Airflow DAG**를 사용하여 일정 주기마다 Selenium을 통해 Naver Datalab Shopping Insights 페이지를 크롤링하고, 웹 페이지에서 스크린샷을 자동으로 생성한다.
- 생성된 스크린샷에서 **Upstage Document OCR API**를 활용해 각 카테고리별 트렌드 상품을 인식하고, 해당 상품 이름과 순위를 데이터베이스에 저장한다.
- 수집된 정보를 요약한 리포트를 Gmail을 통해 개인 사업자에게 전송해 주 단위로 트렌드 상품 정보를 확인할 수 있도록 한다.
- 웹 기반 챗봇을 통해 예측된 트렌드 분석 및 다음 달 트렌드 상품 예측 정보를 제공하여, 개인 사업자가 마케팅 의사결정에 실질적으로 활용할 수 있도록 지원한다.

이를 통해 개인 사업자는 실시간 트렌드 상품 정보를 손쉽게 파악하고, 수집된 데이터를 기반으로 효과적인 마케팅 전략을 수립할 수 있다.

**Category Search.** Category Search는 트렌드 상품이 실제로 판매되는 카테고리를 효율적으로 탐색하기 위한 계층적 분류 기법이다. 이 기법은 Hierarchy 기반 검색 제한, Parent-Child Context 활용, Few-Shot Learning 기반 분류의 세 가지 핵심 요소로 구성된다.

- **Hierarchy 기반 검색 제한** - 계층적 구조 (main > sub1 > sub2 > sub3)를 적용하여 검색 범위를 점진적으로 좁힌다.
- **Parent-Child Context 활용** - 상위 카테고리 검색 결과를 하위 카테고리 탐색의 Context로 활용한다. 예: "Potato chip"이 main category "food"에 속하는 경우, sub1 탐색 시 "Potato chip is food"와 같은 추가 Context를 적용한다.
- **Few-Shot Learning 기반 분류 기법 적용** - 각 카테고리별 대표 예제를 Prompt로 활용하여 모델의 분류 성능 향상시켰다.

이러한 Category Search 기법을 통해 트렌드 상품의 정확한 판매 카테고리를 빠르게 탐색할 수 있으며, 이는 자동 주문 시스템의 검색 정확도를 개선하는 데 기여한다.

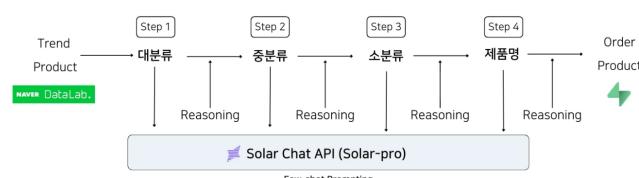


Fig. 2. Category Search Workflow

**Recall Workflow.** safetykorea(제품안전정보센터)의 공시 정보를 통해서 리콜 상품 파악 Jina API로 주기적으로 웹화면 정보를 가져와 Database에 저장한다.

만약 조회된 리콜 상품이 판매중이라면, 사업자에게 알림 제공 및 자동으로 재고에서 제거하고 인벤토리 대시보드에서 반영결과를 확인함으로써 사후 문제를 예방할 수 있다.

## Dashboard

### Chatbot

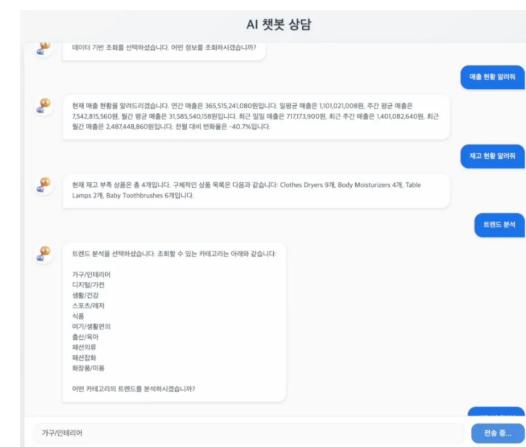


Fig. 3. 챗봇을 통해 매장 재고 분석 결과와 트렌드 상품 관련 인사이트를 제공한다.

## Dashboard

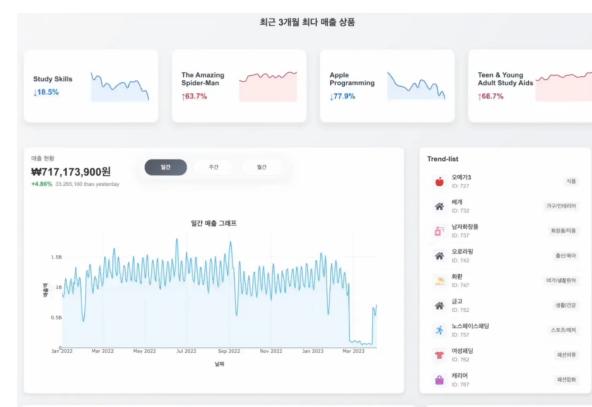


Fig. 4. 일별, 주별, 월별, 연간 매출 지표와 변화 추이를 제시하고, 최다 매출 상품과 트렌드 상품을 강조한다.

## Inventory

제품명	최근 판매량	최소 재고 기준		상태	우편
		최소 재고량	재고량		
Body Moisturizers	1.3	0.1	1.0	상태	X
Baby Bath Products	1.6	0.5	1.0	재고 부족	X
Diapers	1.6	0.1	1.0	상태	X
Baby Solder Inner Trousers & Tops	20.4	8.7	12.0	재고 부족	X
Medical Supply Goods	1.6	0.1	1.0	상태	X
Children's Cotton Handkerchiefs	14.6	3.8	10.0	재고 부족	X
Leather Handbags	2.6	1.0	1.0	상태	X
Footwear	2.1	0.7	1.0	상태	X
Endocrine System Diseases	28.9	9.0	19.0	재고 부족	X
Aut Gardening	7.4	0.2	7.0	상태	X
Sage Garden Care	1.5	0.0	0.0	상태	X
Children's Knit & Amphibian Books	39.4	1.1	37.0	재고 부족	X
Commercial Acne Care	2.9	0.0	1.0	상태	X
Women's Parts	74.5	2.5	70.0	재고 부족	X
Cargo Liners	104.5	51.4	100.0	재고 부족	X
Household & Household Cleaning	21.2	4.7	20.0	상태	X
Socks/Diving	16.8	1.0	20.0	재고 부족	X
Ranchers	109.9	34.7	75.0	재고 부족	X
Space Clean & Space Mini Accessories	1.5	0.5	1.0	상태	X
Car Care & Repair & Sticks	1.5	0.0	1.0	상태	X
Deck Supply Holders & Organizers	2.8	0.1	2.0	재고 부족	X
Dating	20.1	16.0	20.0	재고 부족	X
Hair Conditioner	78.5	2.0	75.0	재고 부족	X

Fig. 5. 현재 상품 재고 현황과 예측 판매량 기반의 최소 재고 기준, 주문 상태를 종합적으로 모니터링한다.

## 자동 주문 Agent

**Stacked LSTM 기반 판매량 예측.** 본 프로젝트에서는 시계열 데이터 분석을 위해 Stacked LSTM 모델을 사용하였다.

Stacked LSTM은 다층 구조로 이루어져 있어 단기와 장기의 준성을 잘 학습할 수 있다는 장점이 있다. LSTM과 GRU 모델도 고려하였으나, LSTM은 구조가 단순하여 본 데이터셋에 대한 성능 한계를 보였고, GRU는 계산 비용이 낮아 inference 속도가 빠르지만 성능이 다소 낮았다. 따라서 최종적으로 Stacked LSTM을 선택하였다.

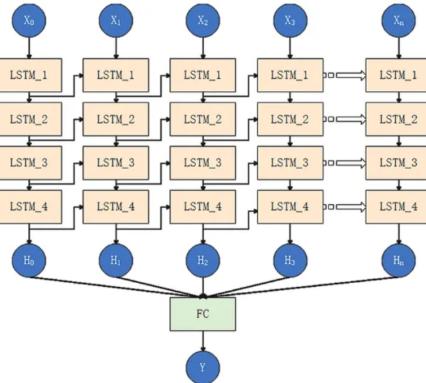


Fig. 6. Stacked LSTM 구조

모델	PSFA Score
LSTM	0.48
GRU	0.50
Stacked LSTM	0.55

Table 2. 모델별 PSFA 성능 비교

최종 선택된 Stacked LSTM 모델의 PSFA 성능은 약 0.55로, 다른 모델에 비해 우수한 성능을 보였다.

**자동 주문 알고리즘.** 효과적인 재고 관리를 위해 ReOrder Point(ROP) 개념을 도입하였다. ROP은 (평균 일일 수요 × 리드 타임) + 안전재고로 계산되며, 평균 일일 수요는 Stacked LSTM으로 예측한 판매량을 사용한다. 재고 부족 상품의 ROP와 트렌드 상품을 종합하여 주문 에이전트에 전달하는 방식으로 자동 주문 프로세스를 구현하였다. 주문 에이전트는 Webshop을 환경으로 하여 구축되었으며, solar-pro(Chat API)를 모델로 사용한다. 에이전트의 추론 및 계획 수립을 위해 ReAct와 one-shot prompting 기법을 적용하였고, 에이전트의 기억 능력 향상을 위해 Reflexion 기법을 활용하였다.

## Webshop

Webshop은 웹 기반 시뮬레이션 이커머스 환경으로, 에이전트의 이커머스 문제 해결 능력을 평가하기 위한 벤치마크로 활용된다. Amazon.com에서 스크랩된 1,000개의 상품과 12,087개의 instructions로 구성되어 있다. 주문 에이전트는 Webshop을 환경으로 하여 구축되었으며, solar-pro(Chat API)를 모델로 사용한다. 에이전트의 추론 및 계획 수립을 위해 ReAct와 one-shot prompting 기법을 적용하였고, 에이전트의 기억 능력 향상을 위해 Reflexion 기법을 활용하였다. Webshop을 통해 에이전트가 관측 정보(Observation)를 얻는 과정은 다음과 같다. 먼저 에이전트의 행동(Action)을

URL 형태로 변환한 후, request GET을 통해 해당 HTML을 읽어온다. 그 다음 BeautifulSoup 라이브러리를 활용하여 HTML 파싱을 진행하고, 필요한 부분(Observation)만 텍스트로 추출한다.

**ReAct.** ReAct는 Reasoning과 Acting을 결합하여 에이전트가 사고(think)와 행동(search, click)을 적절히 수행하도록 하는 과정이다. ReAct의 진행 과정은 다음과 같다.

1. 입력: one-shot 예제 + Instruction  
(a) 출력: LLM 모델이 가장 적합한 Action을 도출
2. 입력: 1번 입력 + 1번 출력 + 1번 Action으로 나온 Observation  
(a) 출력: LLM 모델이 가장 적합한 Action을 도출

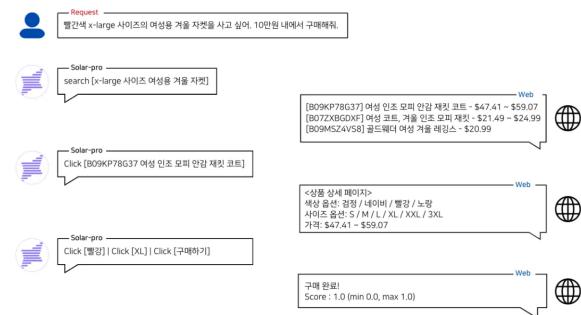


Fig. 7. ReAct 과정

이러한 ReAct 과정을 통해 에이전트는 주어진 상황에 대해 적절한 추론을 수행하고, 최적의 행동을 선택할 수 있게 된다. 이는 자동 주문 시스템의 효율성과 정확성을 크게 향상시키는 데 기여한다.

**Reflexion.** Reflexion은 AI의 자기 피드백(Self-Reflection) 기법으로, 문제와 이전에 생성한 답, 정답 여부(실패)를 주고 왜 실패했는지, 어떻게 하면 다음 시도에 성공할 수 있는지를 묻고 그 결과를 메모리에 저장하는 방식이다. ReAct로 Webshop Task를 풀다가 실패한 경우, few-shot과 함께 자기 피드백을 생성한다.

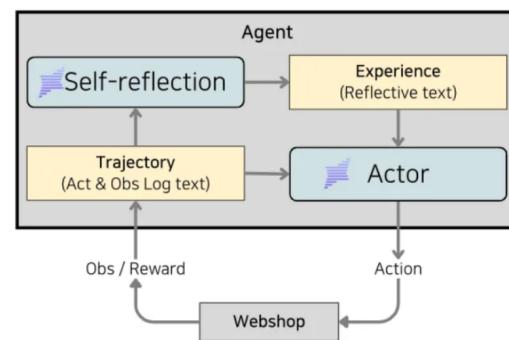


Fig. 8. Reflexion: Self-Improving AI Agent

**Reflexion Process.** Reflexion의 주요 진행 과정은 다음과 같다.

1. 초기 시도 및 실패 감지 - Webshop 내 "A" Instruction 수행 중 첫 번째 시도에서 문제 해결 실패 - 기본 ReAct Agent와 동일한 방식으로 수행

## 2. 자기 피드백(Reflexion) 및 메모리 저장

- 입력: Few-shot 예제 + "A" Instruction 실행 로그 (Instruction, Action, Observation)
- 출력 실패 원인 분석 + 다음 시도에서 변경할 사항 도출

## 3. "A" Instruction 재시도

- 입력: One-shot 예제 + 메모리(자기 피드백 내용) + "A" Instruction
- 출력: 최적의 Action 도출

## 4. 반복적인 Reflexion 적용 - 실패 발생 시 추가 자기 피드백 수행 후 재시도 - 최대 8회 반복 (총 8 Trial까지 허용)

Reflexion을 통해 에이전트는 실패 원인을 학습하고 최적의 해결 방안을 도출하는 과정을 반복하면서 문제 해결 능력을 점진적으로 향상시킨다.

**Integration with Automated Ordering Agent.** Reflexion 기법은 자동 주문 시스템과 결합하여 최적의 구매 프로세스를 지원한다. 해당 과정은 다음과 같다.

1. 주문 상품 데이터 조회 - 주문 상품 테이블에서 **sub3** 값을 읽어온다.
2. Webshop Instruction 입력 생성
  - "**i am looking for sub3.**" 형태로 Instruction 변환 후 에이전트를 실행한다.
3. 실패 시 Reflexion 적용 여부 결정
  - 1차 실행 (Trial-1)에서 실패 발생 시 분석
  - 대부분의 실패는 Webshop에서 sub3 검색 결과가 없기 때문에 이 경우 Reflexion 적용이 불가능하므로 추가 시도 횟수를 제한하여 불필요한 연산을 방지한다.
4. 성공 조건 확인 - 모델이 '**click[Buy Now]**'를 출력하면 정상 구매 완료로 간주

Reflexion 기반 자동 주문 시스템은 문제 해결 과정의 피드백을 통해 반복적으로 최적화되며, 이를 통해 에이전트의 학습 및 문제 해결 능력을 극대화할 수 있다.

## Experimental Results

Webshop 벤치마크 내 200개의 instructions을 사용하여 ReAct only 모델과 Reflexion + ReAct 모델을 비교하였다.

ReAct only 모델의 경우, Temperature를 시도마다 0.2씩 증가시키며 진행하였으나 큰 성능 변화가 없어 Trial 5에서 종료되었다. Reflexion + ReAct 모델은 6회 시도 이후 반복적인 Reflexion이 발생하며 추가적인 성능 향상이 없었고, Trial 8에서 종료되었다.

Webshop 벤치마크에서는 200개 task에 대한 성공 확률과 스코어 점수를 평가하였으며, 이를 데이터 생성에 참여한 13명의 평균 성능 및 그중 상위 7명의 전문가 성능과 비교하였다.

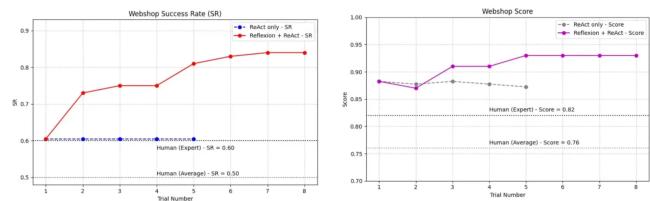


Fig. 9. Webshop Task scores and Success Rate

ReAct only 모델은 인간 전문가 수준과 유사한 성능을 보였으며, Reflexion 프레임워크를 추가한 경우 더욱 높은 성능 향상을 달성하였다. 이는 Reflexion 기법이 에이전트의 문제 해결 능력을 강화하는 데 효과적임을 시사한다.

제안된 자동 주문 시스템은 실제 e-commerce 환경에서도 인간 수준 이상의 성능을 발휘할 가능성이 높으며, 복잡한 주문 문제를 효과적으로 해결하고 최적의 주문 의사결정을 내리는 데 기여할 것으로 기대된다.

## Conclusion

본 프로젝트에서는 AI 기반의 자동화 시스템을 도입하여 기존 수동 재고 관리 방식의 운영 효율성을 극대화하였다. 최신 소비 트렌드 분석과 리콜 상품 감지를 통한 신속한 대응으로 매출 증대와 소비자 신뢰 확보가 가능해졌다. 머신러닝 기반의 판매량 예측과 소비 트렌드 상품 파악을 활용한 지능형 자동 주문 시스템은 비용 절감 및 재고 최적화에 기여하였다. 또한, 통합 대시보드와 챗봇 기능을 통해 관리자의 편의성을 향상시키고 실시간 데이터 분석 기반의 의사 결정을 지원하였다.

이러한 시스템은 중소 규모 자영업자 및 무인 매장 운영자의 운영 부담을 감소시키고 경쟁력을 강화하는 데 도움을 줄 수 있다. 소비자들에게는 안전하고 트렌드에 맞는 상품을 제공함으로써 만족도를 높일 수 있으며, 리콜 상품의 신속한 감지 및 제거를 통해 안전한 유통망 구축에 기여할 수 있다. 본 프로젝트에서 개발된 AI 및 데이터 기반 자동화 솔루션은 지속 가능한 스마트 리테일 환경 조성에 핵심적인 역할을 할 것으로 기대된다.

## Further Work

향후 개발 계획으로는 카테고리 검색 알고리즘의 고도화, 판매량 예측 모델의 성능 개선, 자동 주문 에이전트의 정밀도 향상이 포함된다. 트렌드 상품이 매출에 미치는 영향을 분석하고, 유사 상품의 판매 여부를 평가하는 데이터셋을 구축하여 검색 알고리즘을 최적화할 수 있다. 또한, Imputation 및 데이터 전처리 기법을 적용해 판매량 예측 모델의 정확도를 높이고, 임베딩 기반 유사도 분석과 Few-Shot Learning을 활용해 자동 주문 에이전트의 성능을 향상시킬 수 있다.

아울러, Webshop에서의 성능이 실제 e-commerce 환경에서도 유효하다는 선행 연구를 바탕으로, 네이버 쇼핑, 쿠팡 등 국내 e-commerce 플랫폼에 HTML Parsing을 적용해 시스템을 확장할 계획이다.

## 개인회고(김태한)

### 이번 프로젝트에서 어떤 기술적 도전과 성장이 있었나요?

- 자동 주문 시스템을 구축하면서 AI Agent 개념을 이해하고 프로젝트에 적용해 보았습니다. ReAct, Reflexion 등의 개념을 논문을 통해 익히고, 이를 실제 코드에 적용하면서 이론이 코드의 어떤 부분에 활용되는지와 그 동작 방식을 이해하게 되었습니다.

### 프로젝트 진행 중 만난 문제를 어떻게 해결했나요?

- 이번 프로젝트에서 가장 큰 문제점은 업스테이지의 Solar Chat API를 통한 구현으로 Fine Tuning이 불가능한 점이 있습니다. 이를 해결하기 위해 다양한 프롬프팅 기법을 활용한 LLM의 추론 능력 향상에 초점을 맞췄고 Reflexion이라는 언어 형태의 자기 피드백 기법을 도입하였습니다. 문제 해결에 실패할 경우 실패 원인과 개선 계획을 자연어 형태로 모델로 다시 입력을 함으로써 모델을 학습시키지 않고도 WebShop의 문제 해결률을 60%에서 84%까지 향상시켰습니다.

### 프로젝트에서 본인이 기여한 핵심적인 부분은 무엇인가요?

- 자동 주문 기능을 구현하기 위해 WebShop 환경을 세팅하고 DB에 저장된 물건 리스트를 받아와 자동 주문 Agent를 활용하여 가상 구매를 하는 부분을 엔드포인트로 구현하였습니다. 또한, 주문 결과를 n8n workflow 툴을 통해 DB에 업데이트하고 유저에게 메일을 발송하는 워크플로우를 만들어 자동화하였습니다.

### 다음 프로젝트에서 개선하고 싶은 점은 무엇인가요?

- 예제 데이터셋을 구축하고 임베딩을 통해 실제 요청과 가장 유사도가 높은 예제를 Few-Shot Learning으로 활용하여 Agent의 성능을 좀 더 향상시키고 싶습니다. 또한, WebShop이라는 정제된 환경이 아닌 쿠팡이나 네이버 쇼핑과 같은 실제 한국 이커머스 환경에 적용하여 고도화를 진행할 예정입니다.

## 개인회고(문채원)

### 이번 프로젝트에서 어떤 기술적 도전과 성장이 있었나요?

- n8n을 활용한 로우코드 자동화를 처음 접하며, 다양한 노드를 조합하여 워크플로우를 구성하는 경험을 하였습니다. 또한, FastAPI와 n8n Webhook 트리거를 연동하여 백엔드 API와의 데이터 흐름을 최적화하는 시도를 진행하였으며, 이를 통해 비동기 API 처리 방식과 효율적인 데이터 연동 구조를 구축하는 경험을 쌓았습니다.

### 프로젝트 진행 중 만난 문제를 어떻게 해결했나요?

- 월간 트렌드 상품 예측 챗봇에서 제품 카테고리와 소비자 행동, 조회 기간의 주기적 특성 간의 연관성을 충분히 반영하지 못하는 한계가 있었습니다. 이를 해결하기 위해 Zero-shot, Few-shot, Chain-of-Thought 등 다양한 프롬프팅 기법을 실험한 결과, 개별 기법만으로는 분석의 깊이나 논리적 일관성이 부족하다는 점을 확인하였고 Thought Process를 포함한 Few-shot Prompting을 구현하여 챗봇이 보다 신뢰도 높은 응답을 생성할 수 있도록 최적의 프롬프트를 구성하였습니다.

### 프로젝트에서 본인이 기여한 핵심적인 부분은 무엇인가요?

- 크롤링된 이미지를 기반으로 Solar OCR API를 활용해 일정한 패턴을 인식하고, 이를 Supabase에 저장하는 방안을 구현하였습니다. 또한, OCR 분석 결과를 활용하여 n8n을 통해 유저에게 트렌드 상품 정보 메일을 발송하는 워크플로우를 추가하여 데이터 전달을 자동화하였습니다. 이와 함께, DB에 저장된 트렌드 상품 데이터를 활용한 AI 기반 트렌드 해석 챗봇 엔드포인트를 구현하였으며, 웹 프론트엔드에서는 사용자가 주요 기능을 직관적으로 탐색할 수 있도록 사이드바 UI를 개발 및 개선하였습니다.

### 다음 프로젝트에서 개선하고 싶은 점은 무엇인가요?

- LangChain이나 LangGraph 같은 AI 워크플로우 프레임워크를 활용하여 AI 기반 멀티스텝 자동화를 설계하고, 더 유연한 환경에서 최적의 워크플로우를 구현하고 싶습니다. 또한, 현재 내부 DB 데이터에 의존하는 방식인 챗봇 기능을 확장하여 SERP API와 소비자 동향 리포트, 전자상거래 트렌드 데이터 등 다양한 외부 정보를 반영한 RAG 기법을 적용함으로써, 보다 신뢰도 높은 답변을 제공하고 실시간 트렌드를 효과적으로 반영할 수 있도록 고도화할 계획입니다.

## 개인회고(서동환)

### 이번 프로젝트에서 어떤 기술적 도전과 성장이 있었나요?.

- n8n이라는 자동화 툴을 사용해 코드 없이 전반적인 워크플로우를 구성하였습니다. 초기에는 정보나 오픈소스 자료가 상대적으로 부족해 어려움을 겪었으나, 조원들과 함께 강의를 듣고 소스코드를 참고하며 해결하기 위해 노력하였습니다. 또한, 강의를 통해 학습했던 FastAPI를 활용하여 라우팅 및 비동기 처리 과정을 깊이 이해할 수 있었으며, Docker를 통해 실행 환경을 효율적으로 관리할 수 있었습니다.

### 프로젝트 진행 중 만난 문제를 어떻게 해결했나요?.

- Upstage에서 제공된 Solar Chat API를 활용해 챗봇 기능을 구현하였습니다. 초기에는 질문에 일관된 답변을 제공하지 못해 프롬프트 구성에 어려움을 겪었지만, instruction을 적절히 추가하고 프롬프트를 수정하여 답변의 퀄리티를 개선할 수 있었습니다. 또한, 데이터베이스에서 값을 직접 불러오는 방식이 시간이 오래 걸려, 미리 함수를 정의해 값을 계산하고 이를 전달하는 방식을 적용하여 활용성을 높였습니다.

### 프로젝트에서 본인이 기여한 핵심적인 부분은 무엇인가요?.

- 사용자가 이전 판매 기록과 트렌드 정보를 직관적으로 확인할 수 있도록 대시보드, 챗봇, 인벤토리 페이지의 프론트 엔드(UI/UX)를 설계 및 구현하였습니다. 또한, 데이터베이스의 정보를 기반으로 FastAPI를 활용하여 기간별 매출, 최다 매출 상품 등의 데이터를 제공하는 엔드포인트를 설계하고, 이를 처리하는 함수를 구현하여 데이터를 효과적으로 전달할 수 있도록 구성하였습니다.

### 다음 프로젝트에서 개선하고 싶은 점은 무엇인가요?.

- 다음 프로젝트에서는 웹 크롤링이나 Chat API를 단순히 연결하는 방식을 넘어, CV나 NLP 요소를 추가하여 모델링 과정까지 포함시켜 프로젝트의 완성도를 더욱 높여보고 싶습니다.

## 개인회고(윤남규)

### 이번 프로젝트에서 어떤 기술적 도전과 성장이 있었나요?.

- 이전부터 개인적으로 공부해왔던, 다양한 기술 스택을 실제로 적용해볼 수 있었습니다. 노코드 기반으로 빠른 기능 구현이 가능한 n8n, PostgreSQL & pgvector을 활용하는 협업용 데이터베이스인 Supabase, batch serving으로 유용한 도구인 Airflow, 완성도 높은 코드를 위한 Ruff까지 사용해봄으로써 사용할 수 있는 기술 스택의 범위가 넓어졌다는 느낌을 받았습니다.

### 프로젝트 진행 중 만난 문제를 어떻게 해결했나요?.

- 노코드 기반의 n8n을 활용해 빠르게 원하는 기능을 구현할 수 있었지만, 코드 기반의 커스텀 로직 구현이 제한적이었다는 점이 한계로 느껴졌습니다. 따라서 프로젝트 후반부에 코드 기반의 Apache Airflow로 기능 전환을 팀원들과 논의 후 결정했고, DAG 기반의 자유로운 로직을 도입함으로써 성능을 크게 개선할 수 있었습니다.

### 프로젝트에서 본인이 기여한 핵심적인 부분은 무엇인가요?.

- 프로젝트에 필요한 데이터베이스의 전처리 및 스키마 설계를 주도했습니다. 간편하게 사용할 수 있는 SQLite부터 시작해서, 확장성 및 유연성이 뛰어난 PostgreSQL, 그리고 최종적으로 동시 작업이 가능한 Supabase까지 데이터베이스를 옮기는 과정을 주도했습니다. 또한 OCR 및 크롤링을 활용해 불러온 트렌드 상품의 실제 유사 상품을 추천해주는 알고리즘을 개발함으로써 “트렌드 탐색 - 재고 관리 - 가상 주문” 까지 전 과정을 자동화하는 흐름을 완성했습니다.

### 다음 프로젝트에서 개선하고 싶은 점은 무엇인가요?.

- 다양한 오픈 소스의 장단점을 비교하는 능력이 부족한 것 같아서 아쉽습니다. 예를 들어서 이번 프로젝트는 batch serving이 주 기능이므로, Airflow가 유용하게 사용될 수 있었지만 이를 프로젝트 막바지에 떠올려 급하게 기능을 전환했던 점이 아쉽습니다. 점점 사용할 수 있는 기술 스택은 많아지면서, 필요한 상황에 어울리는 적당한 도구를 사용할 수 있도록 빠르게 판단하는 능력이 필요한 것 같습니다.

## 개인회고(이재훈)

### 이번 프로젝트에서 어떤 기술적 도전과 성장이 있었나요?

- n8n이라는 새로운 툴을 배우면서 사용해보고 Agent에 대한 개념과 개발 방법등을 알 수 있게되었고 이러한 기능을 사용자에서 보여주도록 FastAPI와 React를 이용해 백엔드와 프론트엔드를 개발하면서 웹에 대한 지식도 알게 되었습니다.

### 프로젝트 진행 중 만난 문제를 어떻게 해결했나요?

- 기존의 Dash 라이브러리를 이용해 대시보드 작업을 진행할 계획이었지만 Dash의 한계를 느끼고 FastAPI + React로 변경하면서 백엔드의 많은 엔드포인트와 초반 UI를 설계하는데 어려움을 겪었습니다. 팀원들의 의견을 많이 물어보기도 하고 다양한 웹사이트에서 insight를 얻어가면서 진행했고 결과적으로 정상적인 웹 사이트 개발에 성공했습니다.

### 프로젝트에서 본인이 기여한 핵심적인 부분은 무엇인가요?

- 백엔드 및 프론트엔드, 판매량 예측 모델 부분입니다. 사용자에게 우리의 서비스를 보여주기 위해 FastAPI, React 개발 작업을 진행했고 자동 주문 Agent에 사용되는 주문량 계산을 위한 판매량 예측 모델을 개발했습니다. 시계열 데이터 분석 모델을 이용해 약 PFSA 스코어 기준 0.55의 성능을 기록하는 모델을 만들었습니다.

### 다음 프로젝트에서 개선하고 싶은 점은 무엇인가요?

- 이번 프로젝트에 CV적 요소가 없어 아쉬웠습니다. CV 요소도 포함하는 주제를 하고 싶고 팀원들을 통해서 여러 오픈소스나 기술들을 많이 알게 되었습니다. 다양한 툴을 경험하면서 다음 프로젝트를 진행하면 좋을 것 같습니다.

## 개인회고(장지우)

### 이번 프로젝트에서 어떤 기술적 도전과 성장이 있었나요?

- Agent를 구현하기 위해 팀원 전부 처음 접한 n8n 툴을 활용하는 과정에서, 다같이 강의를 듣고 응용해보며 결과물을 낼 수 있었습니다. 또한 Fastapi를 활용하여 비동기 처리 방식, 실시간 웹소켓 방식과 엔드포인트 구축에 대해 많이 배우고 사용해보았으며, React를 활용해 프론트엔드를 구현할 수 있었습니다. 이렇게 다양한 프레임워크와 Github, Jira, Notion이라는 다양한 협업툴을 사용해보며 많은 지식과 경험을 쌓을 수 있었습니다.

### 프로젝트 진행 중 만난 문제를 어떻게 해결했나요?

- n8n으로 구현된 자동 주문 agent를 활용해 주문 현황을 실시간으로 불러와 Fastapi로 엔드포인트를 구축하는 과정에서 실시간 정보 처리가 안되는 문제가 발생해 어려움을 겪었습니다. 이를 해결하기 위해 n8n에서 webhook으로 주문 현황 정보를 전달하고, 해당 정보를 헤더의 크기가 작아 효율적이고 실시간 네트워킹이 가능한 웹소켓을 이용해 구현할 수 있었습니다. 또한 인벤토리 관련 데이터를 Supabase에서 불러오는 과정에서, 서버 과부화 문제가 있었지만 이를 청크사이즈로 불러오는 방식으로 해결할 수 있었습니다.

### 프로젝트에서 본인이 기여한 핵심적인 부분은 무엇인가요?

- 사용자의 제품들, 재고 현황, 재고 부족 정보, 주문해야 할 수량, 주문 현황 등을 직관적으로 볼 수 있도록 인벤토리의 프론트엔드(UI/UX)를 설계 및 구현하였으며 Fastapi를 활용해 엔드포인트를 구축하고 웹소켓과 n8n의 Webhook을 활용해 실시간 정보를 불러오는 기능을 구현하였습니다. 또한 인벤토리 관련 데이터를 Supabase에서 효율적으로 불러올 수 있도록 구현하였습니다.

### 다음 프로젝트에서 개선하고 싶은 점은 무엇인가요?

- 부스트캠프에서 배웠던 CV적 요소를 활용할 수 있게끔 프로젝트를 진행하고 싶고, 데이터베이스를 처음부터 확실하게 설계, 구축해두고 개발을 진행할 것입니다. 또한 팀원들을 통해 알게된 다양한 툴을 사용해보고 싶습니다.

## References

- [1] Shunyu Yao (2022). "WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents". NeurIPS 2022.
- [2] Shunyu Yao (2023). "ReAct: Synergizing Reasoning and Acting in Language Models". ICLR 2023.
- [3] Noah Shinn (2023). "Reflexion: Language Agents with Verbal Reinforcement Learning". NeurIPS 2023.
- [4] Yukun Dong, Yu Zhang, Fubin Liu, Xiaotong Cheng (2021). "Reservoir Production Prediction Model Based on a Stacked LSTM Network and Transfer Learning."
- [5] 메트로신문 (2024.05.20). "소비 트렌드 변했다" 대형마트 점포 7년 새 37개 감소..."본업 집중 효율성 강화". <https://www.metroseoul.co.kr/article/20240520500503>
- [6] 한국경제 (2024.02.18). "월세가 160만원인데 매출이 60만원..." 무인점포 '의 눈물. <https://www.hankyung.com/article/202402168434g>
- [7] 서울경제 (2024.02.18). "1년도 못 버티고 폐업...코로나때보다 힘들어" 깊어지는 자영업자 한숨. <https://news.nate.com/view/20241105n21180>