POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

IMEN891K 데이터 엔지니어링 Final Presentation

소비자를 위한
개인 맞춤형 '리콜 정보 알리미'

Team 2
김유진, 남도은, 민선아, 봉재우, 심민준

2025. 12. 09

목차

1. 아이디어 및 시연

2. 데이터 파이프라인 개요

2-1. 데이터 파이프라인 개요

3-2. 데이터 모델링 (ERD)

3. 데이터 파이프라인 구현

3-1. 회원정보 등록 및 로그인

3-2. Service 1

3-3. Service 2

4. 기대효과 및 한계점

1-1. 아이디어 요약

소비자를 위한 개인 맞춤형 '리콜 정보 알리미'

문제점

- 식품, 의약품 등의 리콜이 증가하고 있음.
  - '23년 64건 → '24년 119건

    [출처: 식품안전나라, 2025.08.06]
- 소비자가 정보를 직접 찾아야 하는 정보 비대칭 문제 발생함.


엘빈즈 이유식 "한우듬뿍시금치아기밥" 세균수 초과 '리콜'

이용석 기자 | 승인 2025.09.18 17:53

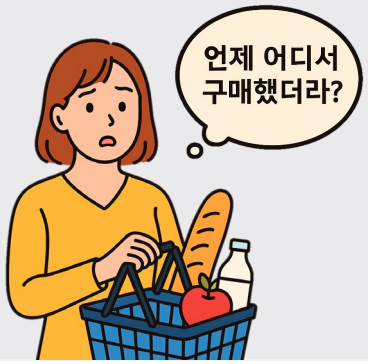
영·유아 이유식 브랜드 '엘빈즈' 제품이 세균수 초과로 회수 조치됐다.

출처=식품의약품안전처

[출처: 컨슈머치, 2025.09.18]

현재 상황

- '리콜 알림 서비스'를 통해 리콜 상품에 대해 알림 서비스를 제공함(소비자24).
- 하지만, 특정 상품을 개별로 '관심 상품'으로 등록해야 안내 받을 수 있음.
- 상투적인 안내로 인하여, 소비자가 상품 반품 시 절차가 오래 걸리고 복잡함.



해결방안

- **생성형 AI**를 통해 공식 리콜 정보, 개인 구매 내역, 최신 뉴스를 종합하여 신뢰도 높은 **맞춤형 알림 메시지**를 발송함.
- 정확한 구매처와 상세 반품 방법을 안내함으로써, 소비자의 즉각적인 행동을 지원함.



1-1. 아이디어 요약

- **영수증 스캔을 통해 구매한 상품 정보를 한번에 등록한 후 리콜이 발생했을 때 소비자에게 바로 리콜 알림 및 제품 환불 방법, 관련 뉴스 등의 정보를 맞춤형으로 제공하는 서비스를 구현하고자 함.**
- **(차별점)** 사용자의 수고를 요구하는 '정보 검색(Pull)'의 기존 방식에서 사용자의 구매 기록에 기반하여 필요한 정보만 선별하여 알려주는 '자동 알림(Push)' 방식으로 개선하고, 생성형 AI를 통해 개인의 구매 이력과 최신 뉴스를 반영한 맥락적이고 설득력 있는 알림 메시지를 자동 생성하여, 소비자의 이해도와 신뢰도 극대화

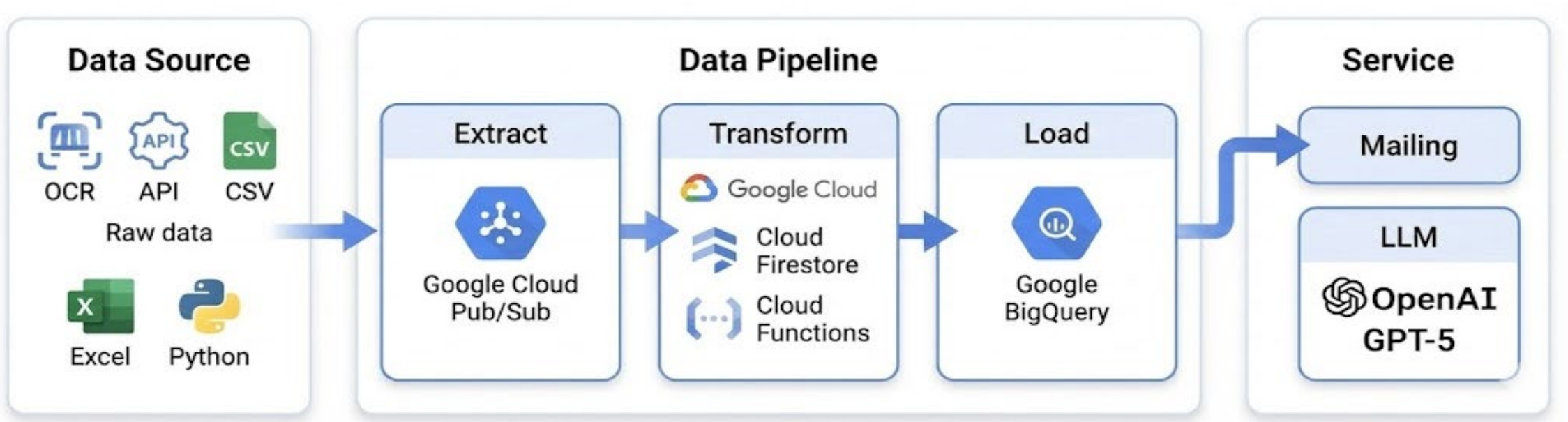


1-2. 시연영상


Deploy
메뉴
페이지 선택
1 데이터 수집 및 적재
2 리콜 데이터 매칭
3 맞춤형 알림 서비스
1 영수증 데이터 수집 → 파이프라인 적재
영수증 이미지 파일을 선택하세요
Drag and drop files here
Limit 200MB per file • JPG, JPEG, PNG, HEIC, PDF
Browse files
영수증 이미지를 업로드하고 OCR 실행 버튼을 누르세요.
➡ 다음 (리콜 매칭으로 이동)

2-1. 데이터 파이프라인 개요

- **이벤트 기반(Event-Driven) 서버리스(Serverless) 구조**
  - 데이터가 발생하면 전체 시스템을 연쇄적으로 실행하는 구조로, Google Cloud Platform의 서비스를 조립하여, 인프라 관리 부담을 최소화하고, 실시간 데이터 처리에 집중함.
  - **주요 강점:** 실시간성, 운영 부담 최소화, 분리되고 확장 가능한 구조

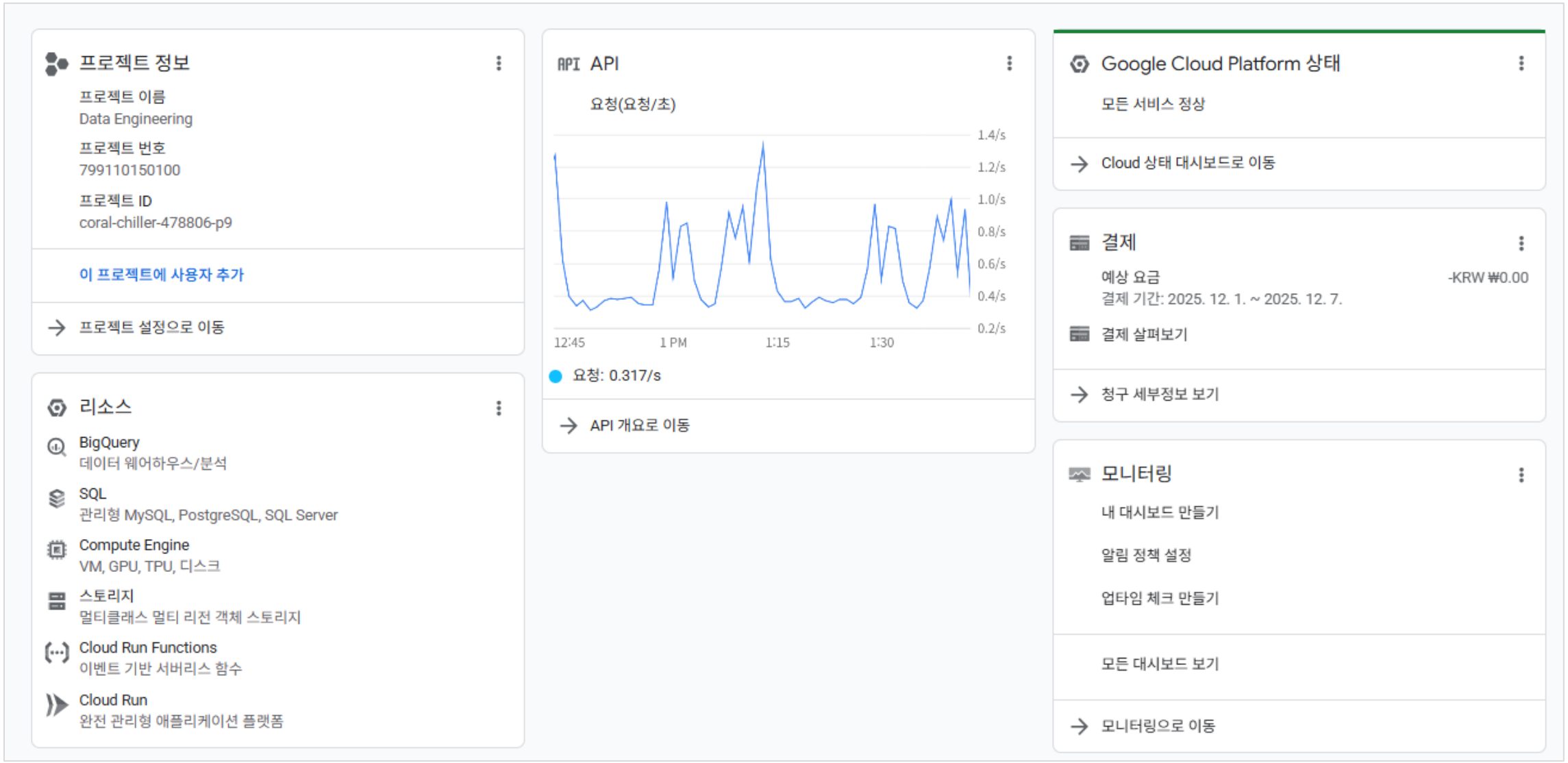


2-1. 데이터 파이프라인 개요



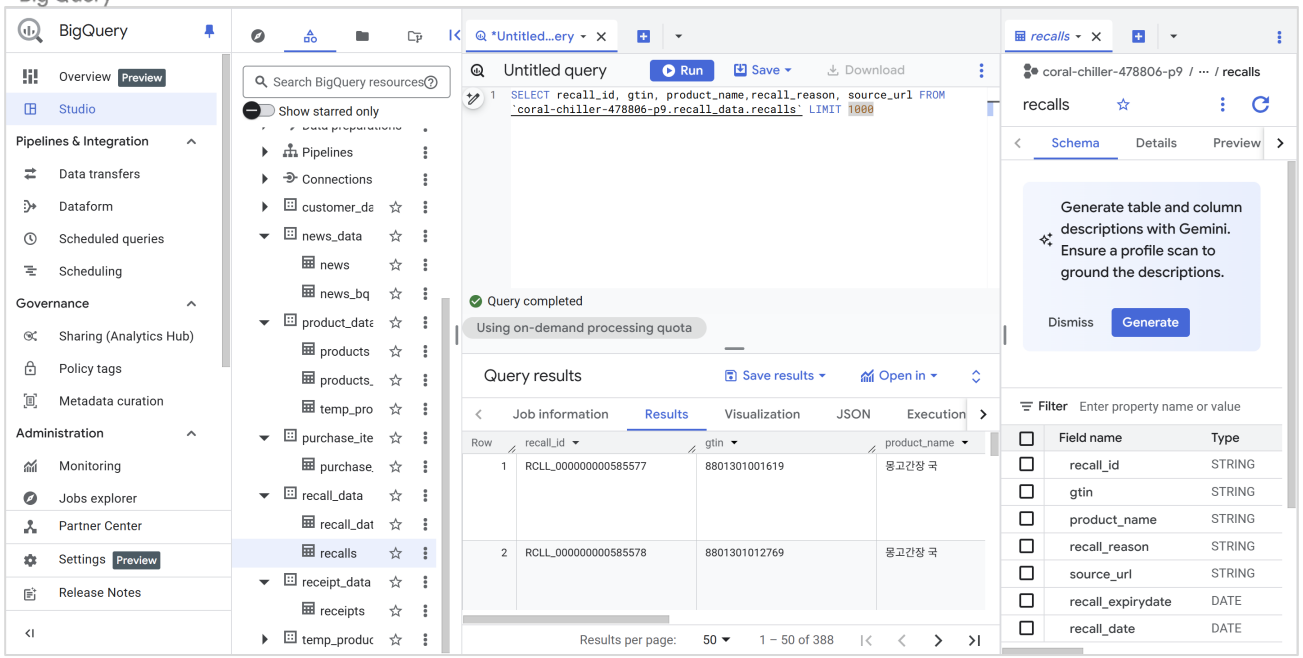
분석		서버리스 운영	데이터베이스
Google Big Query	Google Cloud Pub/Sub	CLOUD FUNCTIONS	Cloud Firestore
대규모 데이터 저장 및 비교 조회 가능한 SQL 기반 데이터 웨어하우스	각 단계를 자동으로 실행될 수 있도록 이벤트 메시지로 연결	이벤트 발생 시 task를 자동으로 서버리스 환경에서 실행	Task 결과 및 이력을 실시간으로 저장하는 서버리스 문서형 데이터베이스
• 상품, 리콜 정보, 고객 등의 데이터 저장 • GTIN 기반 상품 매칭 결과 분석 • 사용자별 리콜 상품 매칭 조회	• OCR 결과, 전처리 결과, 리콜 매칭 결과 등을 이벤트 메시지로 연결 • LLM 메일링 서비스와 연결하여 자동으로 메일 발송	• 영수증 OCR 및 데이터 전처리 트리거 • 누적된 고객의 상품 정보에서 리콜 상품 정보 매칭 수행 • 리콜 매칭 결과 이메일 작성 및 발송	• 각 사용자의 영수증 업로드 기록, 구매 상품, 리콜 매칭 결과, 알림 발송 이력 등 실시간 저장 • 소비자가 이력 확인 가능

2-1. 데이터 파이프라인 개요: 실제 화면 예시

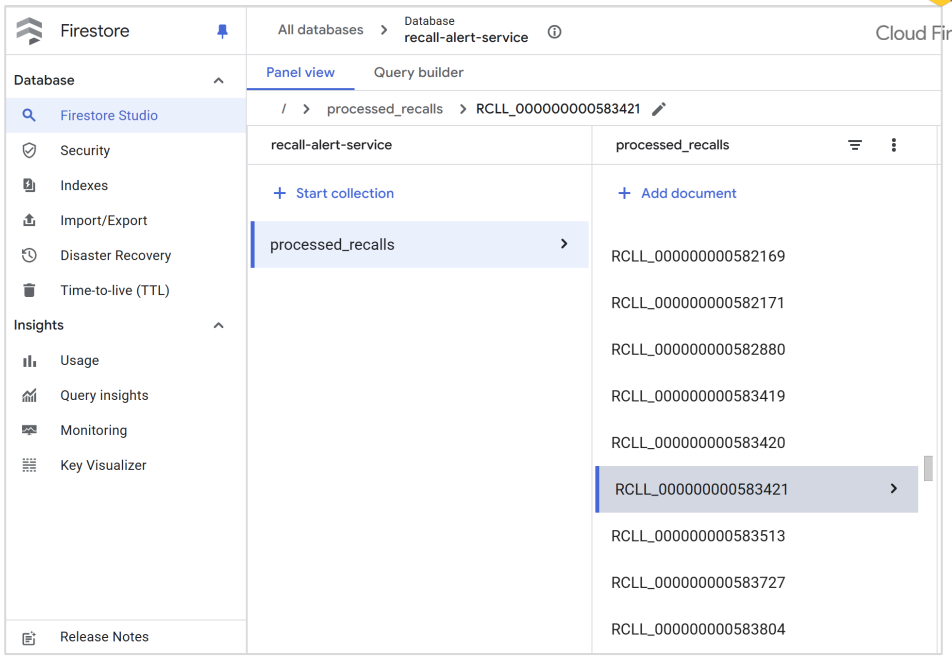


2-1. 데이터 파이프라인 개요: 실제 화면 예시

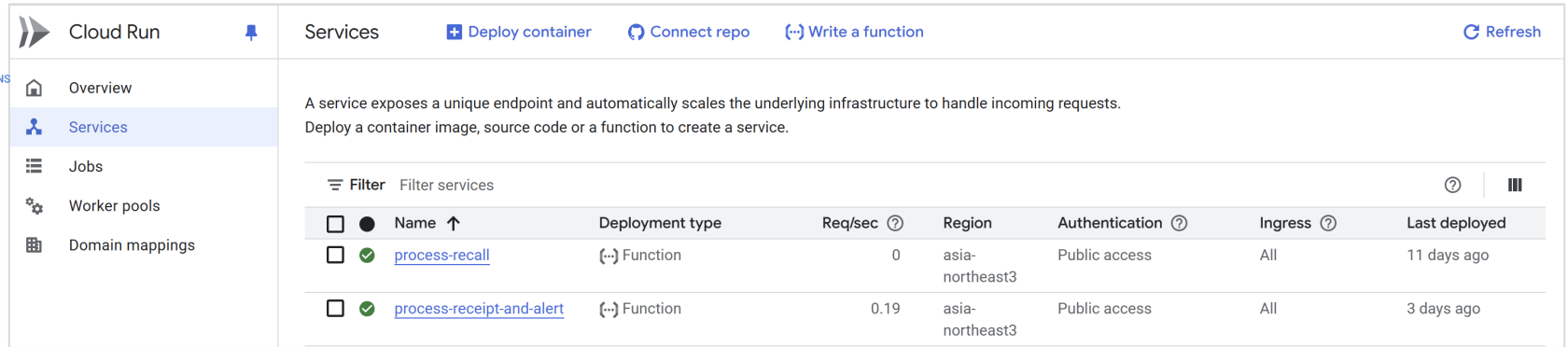
Google Big Query



Cloud Firestore



CLOUD FUNCTIONS



2-2. 데이터 모델링 (ERD)

- **서비스 구현을 위해 총 6개의 데이터 테이블을 구성하였으며, GTIN(상품 바코드)을 primary key로 활용하여 필요한 경우 다른 테이블을 조회하여 데이터를 활용할 수 있도록 함.**

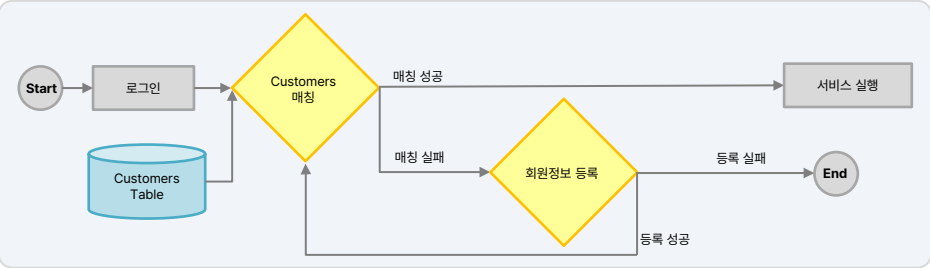


테이블명	설명	수집방법
Customers	고객의 ID, 이름 등 개인정보	회원등록
Recalls	식품 리콜 정보	소비자24 API 연결
Receipts	영수증 정보	OCR 수집
Purchase_items	구매상품 누적 정보	Receipts 데이터 전처리
Products	상품의 기초 상세정보	대한상공회의소 제공(csv)
News	리콜 관련 외부 뉴스 정보	GPT API 활용

3. 데이터 파이프라인 구현

- **소비자가 구매한 상품에 대해 리콜 정보를 제공하기 위해서는 개인정보 등록 및 로그인이 필수적임.**

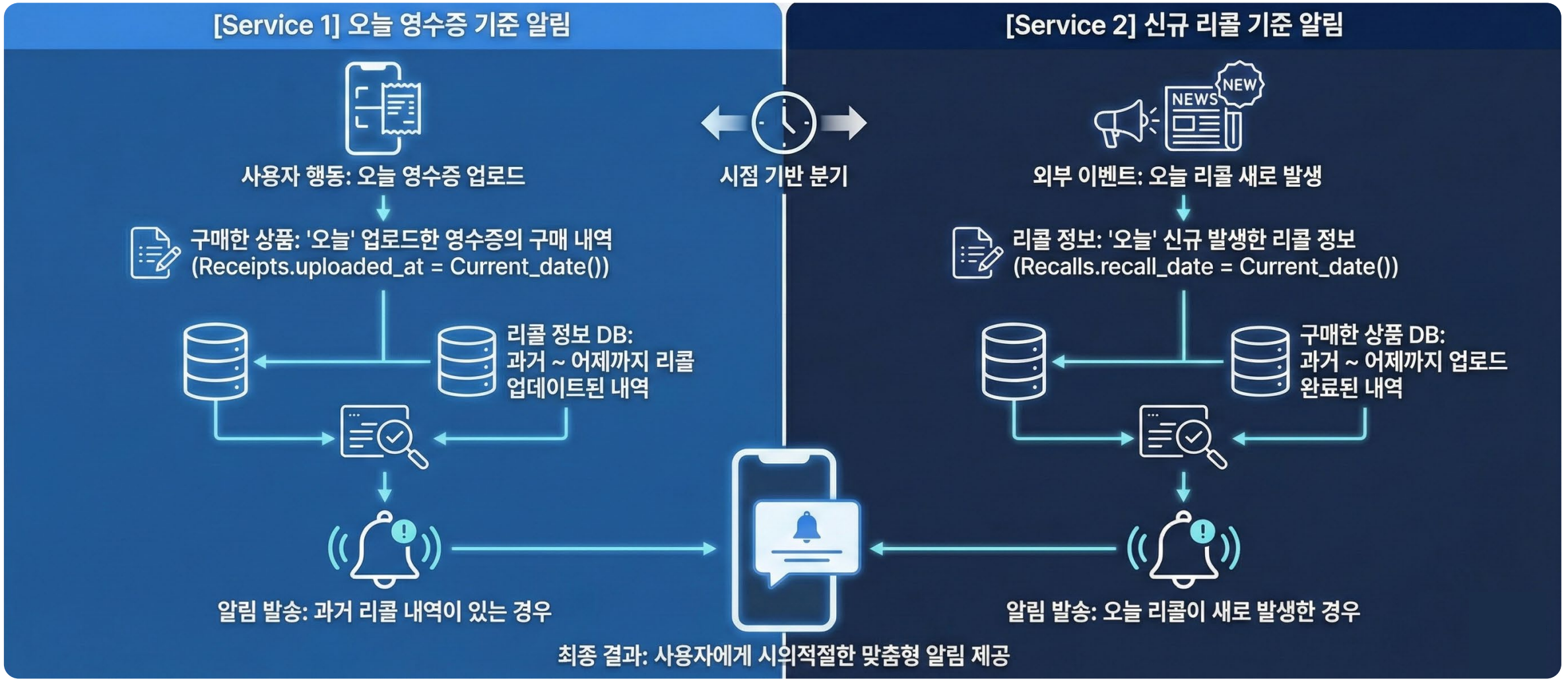
Log-in



- **서비스 시행에 있어서 다수의 데이터를 활용하기 때문에, 효율적인 운영 및 관리를 위해 서비스를 2가지로 구분함.**
  → 각 데이터의 조건을 바탕으로 다르게 수행됨.
  - [Service 1] 오늘 업로드한 영수증에서 과거 리콜 내역이 있는 경우 알림
  - [Service 2] 과거에 업로드했던 영수증에서 오늘 리콜이 새로 발생한 경우 알림

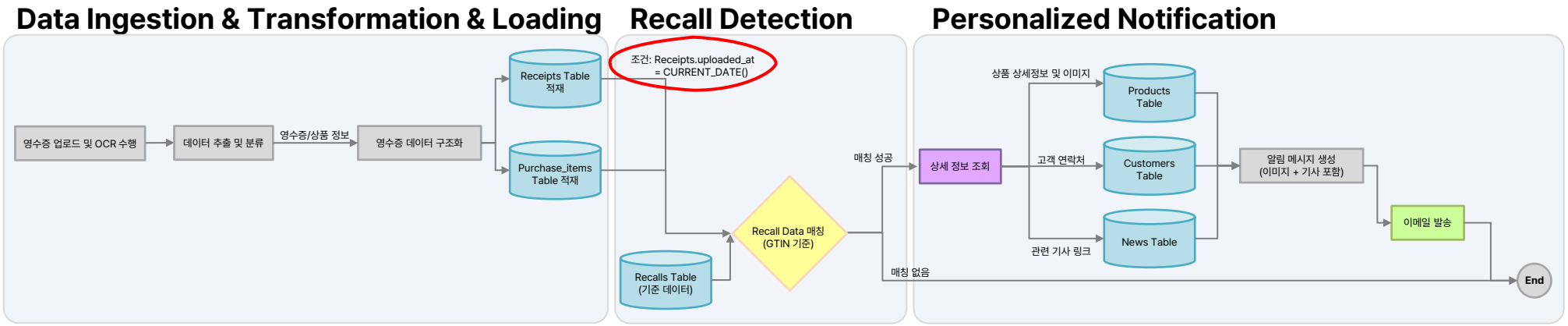
DATA	구매한 상품	리콜 정보
Service 1	"오늘" 업로드한 영수증의 구매 내역 (Receipts.uploaded_at = Current_date())	과거 ~ 어제까지 리콜 업데이트된 내역
Service 2	과거 ~ 어제까지 업로드 완료된 내역	"오늘" 신규 발생한 리콜 정보 (Recalls.recall_date = Current_date())

3. 데이터 파이프라인 구현

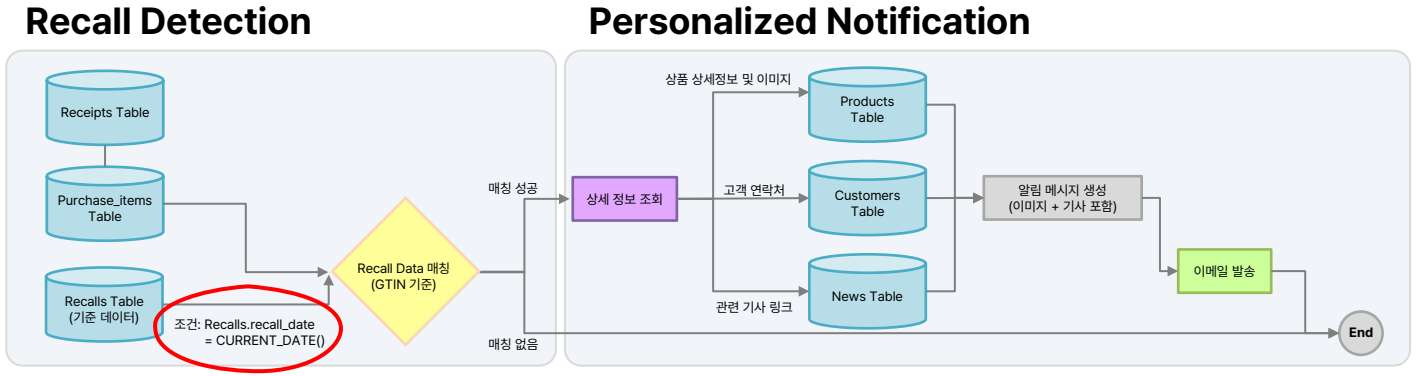


3. 데이터 파이프라인 구현

[Service 1] 오늘 업로드한 영수증에서 과거 리콜 내역이 있는 경우 알림



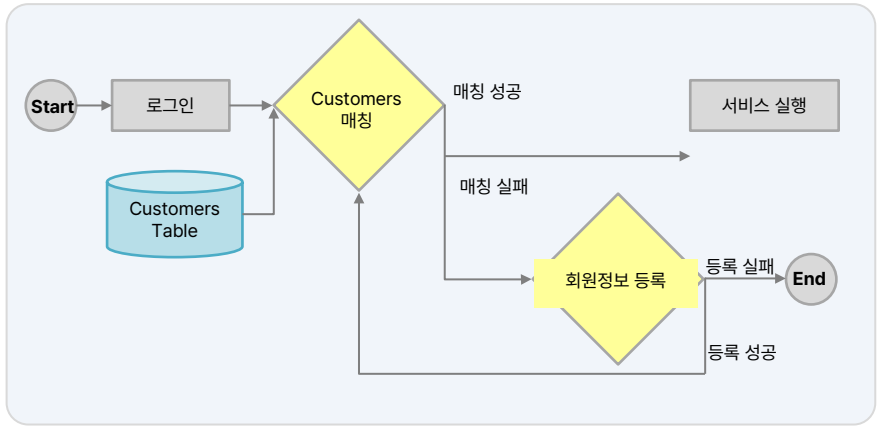
[Service 2] 과거에 업로드했던 영수증에서 오늘 리콜이 새로 발생한 경우 알림

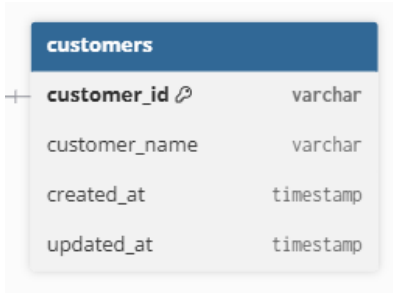


3-1. 회원정보 등록 및 로그인

[Customers Table]

- **로그인 과정을 위해 Big Query를 통해 고객의 인적사항을 저장하는 고객 정보 데이터 테이블을 구성하고, 매칭시 없으면 회원정보 등록을 유도함.**





<빅쿼리, Customers Table>

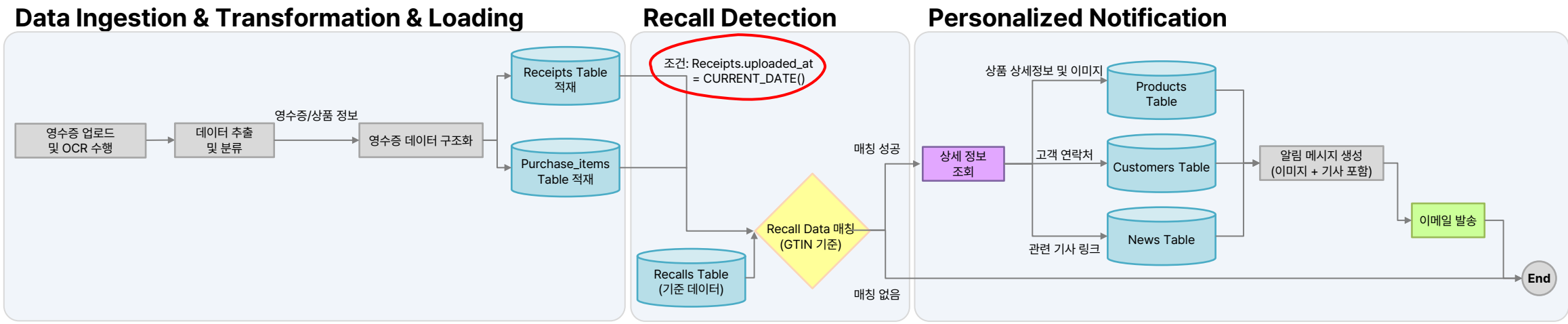
Field name	Type	Mode	Description
customer_id	STRING	REQUIRED	Primary Key
customer_name	STRING	NULLABLE	-
created_at	TIMESTAMP	NULLABLE	-
updated_at	TIMESTAMP	NULLABLE	-

```
INSERT INTO `coral-chiller-478806-p9.customer_data.customers` (customer_id, customer_name)
VALUES
 ('mosw7@postech.ac.kr'), '모상우');
```

Row	customer_id	customer_name	created_at	updated_at
1	mosw7@postech.ac.kr	모상우	2025-12-03 06:44:30.037564 UTC	2025-12-04 14:26:32.377404 UTC
2	abcbjw@postech.ac.kr	봉재우	2025-12-04 13:37:36.556127 UTC	2025-12-04 14:16:41.418843 UTC
3	sunaslight1508@gmail.com	민선아	2025-12-01 16:25:37.988103 UTC	2025-12-04 14:15:47.011565 UTC
4	yujin24@postech.ac.kr	김유진	2025-12-04 13:30:58.304940 UTC	2025-12-04 13:30:58.304940 UTC

3-2. Service 1

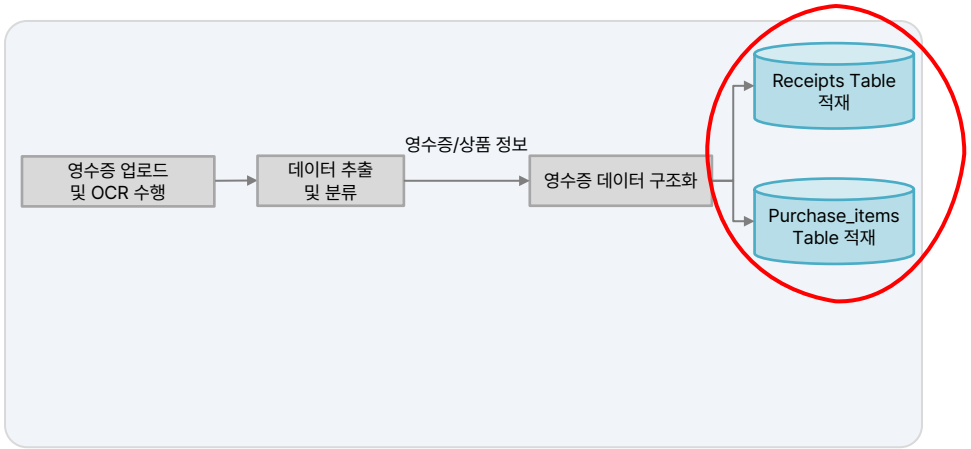
- OCR을 통해 영수증 이미지로부터 구매한 상품에 대한 데이터를 수집함.
- 현재 구매한 상품 중에 리콜 상품이 있는지 확인함.
- 리콜 상품 매칭에 성공하면, 상품 기본 정보, 구매한 고객 정보, 리콜 관련 외부 뉴스 정보를 조회하여 메일을 발송함.

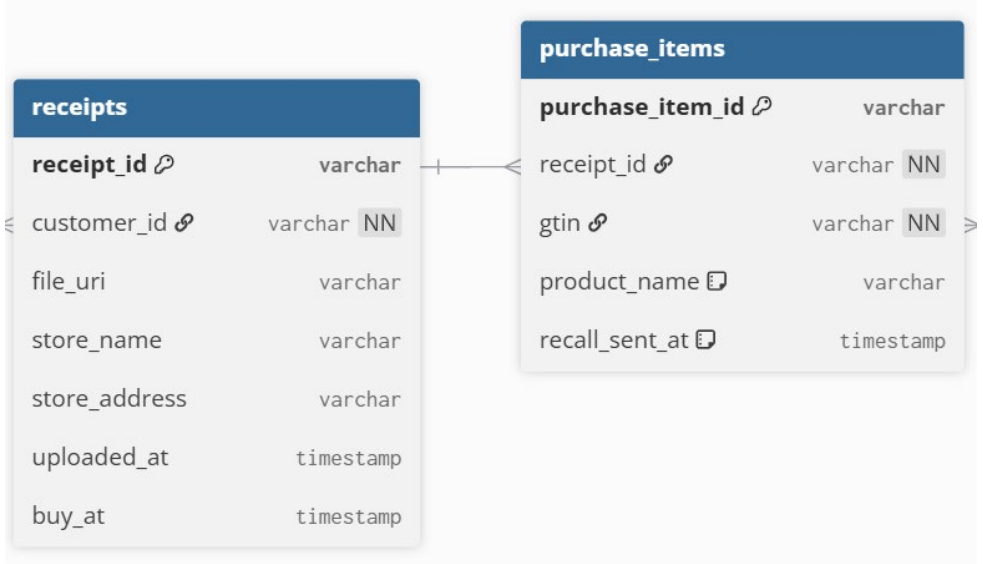


3-2. Service 1: Data Ingestion & Transformation & Loading

[Receipts Table]

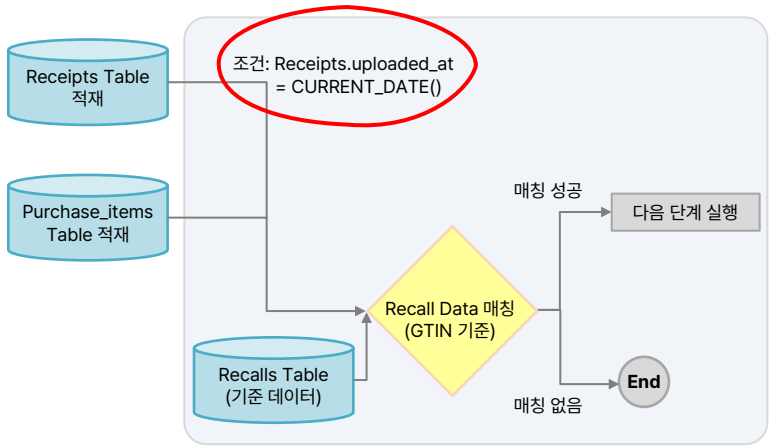
- **고객의 상품 구매 내역을 수집하기 위해 GPT-4o API를 사용한 OCR을 통해 영수증 이미지의 텍스트를 추출함.**
  - 상품명, GTIN, 구매처, 구매주소, 구매일 데이터 추출
- **추출한 데이터를 빅쿼리의 Receipts 테이블과 Purchase_items 테이블에 각각 적재함.**
  - Receipts 테이블: 영수증만 관리
  - Purchase_items 테이블: 실제 구매 내역을 모두 누적하여 저장

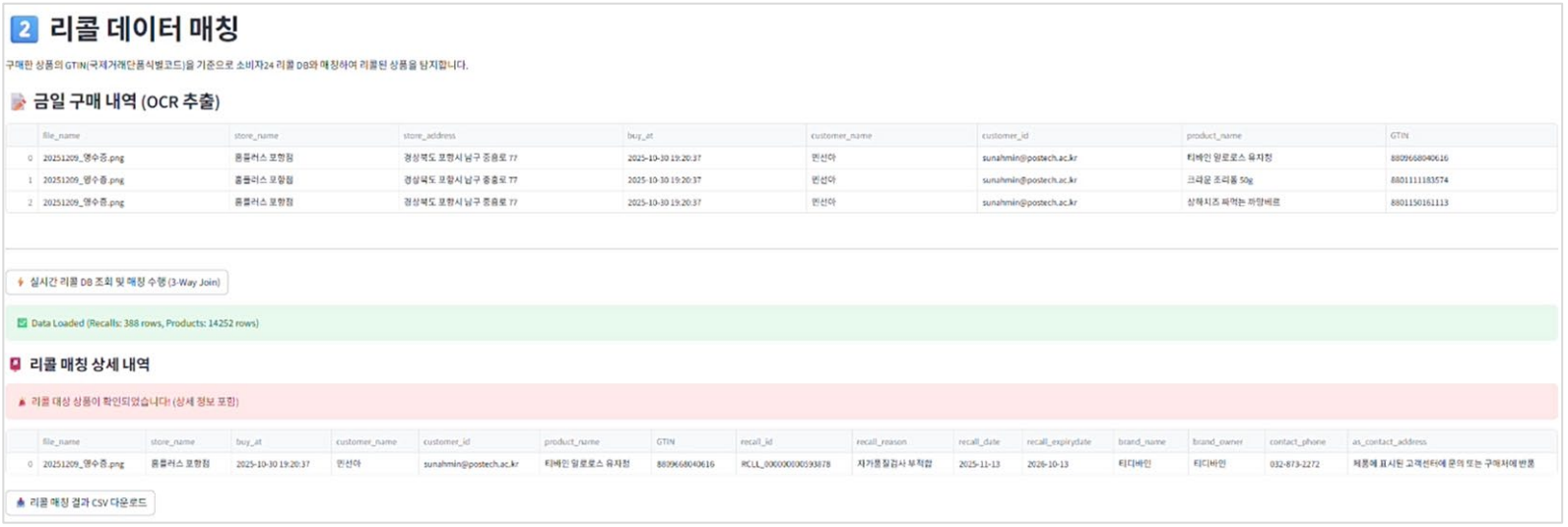




3-2. Service 1: Recall Detection

- **오늘 등록된 영수증의 구매 내역을 기준으로 전체 Recalls 테이블과 join하여 오늘 구매한 상품 중 리콜 상품이 있는지 확인함.**
  - 조건: Receipts.uploaded_at = CURRENT_DATE() ← 업로드가 오늘 등록된 것인지 확인하기 위함.
- **이 중 리콜 상품이 있다면, 리콜 관련 정보를 발송하는 다음 단계로 넘어가고, 리콜이 없다면 별도의 알림 없이 종료함.**

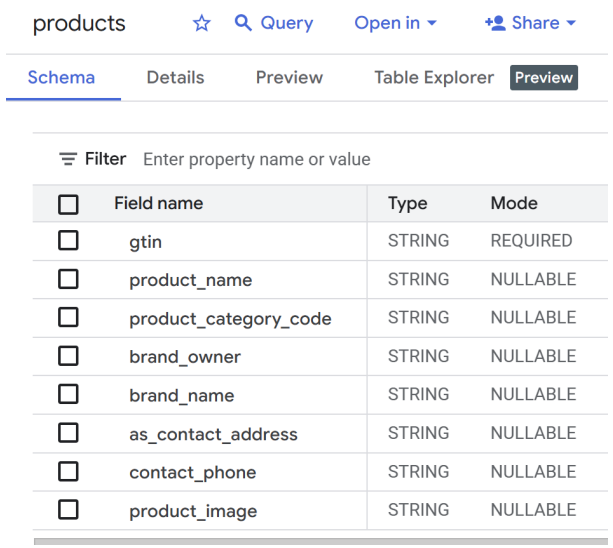


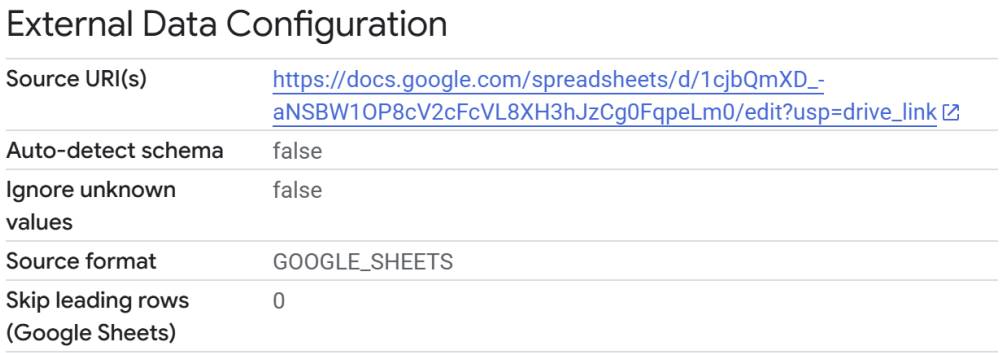


3-2. Service 1: Personalized Notification

[Products Table]

- **Big Query를 통해 엑셀파일로 수집된 Products(상품 정보) 테이블을 구성함.**
  - 대한상공회의소에서는 API 연결을 비허용하며, 주기적으로 엑셀파일로 공유받아야 하기 때문에 1차적으로 구글 스프레드 시트로 데이터를 관리해야함.
  - 그에 따라 빅쿼리에서도 바로 반영될 수 있도록 외부 데이터 참조 가능한 형식으로 설정하였음.





3-2. Service 1: Personalized Notification

[News Table]

- **Google HTML Web Scraping 방식을 통해 Recalls 테이블에 저장된 상품명을 기준으로 뉴스 url 및 텍스트 크롤링하여 데이터를 수집함.**
  - 소비자24에서 리콜이 발생하여 Recalls 테이블에 신규 리콜 정보가 추가되면, Cloud Run을 통해 자동으로 뉴스 데이터를 수집함.
  - 리콜 기사에서 언급되는 상품명과 회수 조치, 리콜 발생 원인 등의 키워드를 통해 핵심 내용을 추출할 수 있도록 시작 및 종료 지점을 설정함.
    - 시작: 상품명, 식품의약품안전처(식약처), 보건복지부, 검출, 기준 초과, 리콜, 회수 조치 등
    - 종료: 저작권/댓글/푸터 키워드

- **추출한 텍스트를 바탕으로 gpt-4o 를 활용하여 텍스트 요약문을 생성하여 News table에 업데이트함.**

```
def summarize_news_article(article_text: str, max_lines: int = 5):
    # 프롬프트(시스템+유저 메시지) 구성
        system_prompt = (
          "당신은 한국어 뉴스를 간결하고 정확하게 요약하는 전문가입니다.
          중립적인 tone을 유지하고, 핵심 사실과 숫자, 인물, 날짜 중심으로 요약하세요."
        )

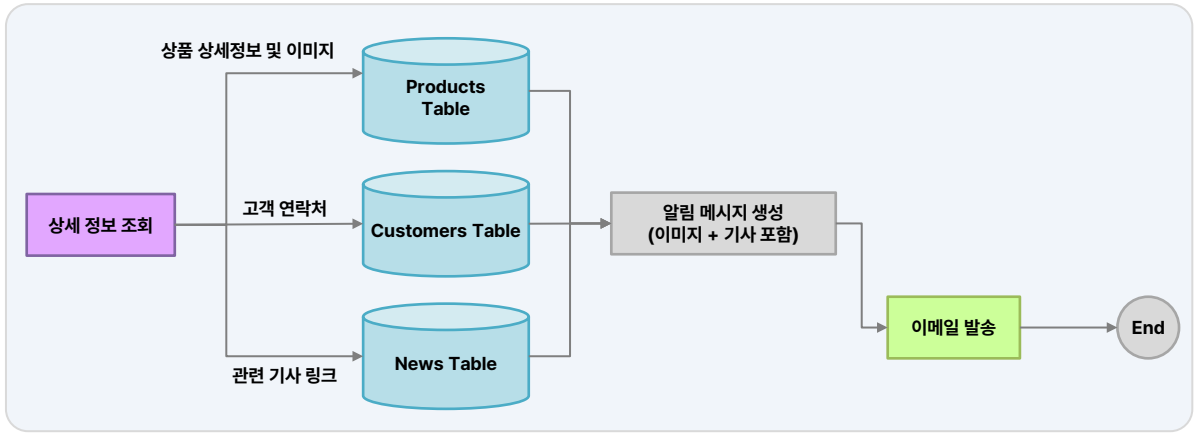
        user_prompt = f"""
            다음 뉴스 기사를 {min_lines}줄 이상, {max_lines}줄 이하로 요약해줘.
            - 각 줄은 한 문장으로 작성해줘.
            - 최대한 핵심 내용만 담아줘.
            - 불필요한 수식어는 줄이고, 사실 위주로 작성해줘.
            """
```

3-2. Service 1: Personalized Notification

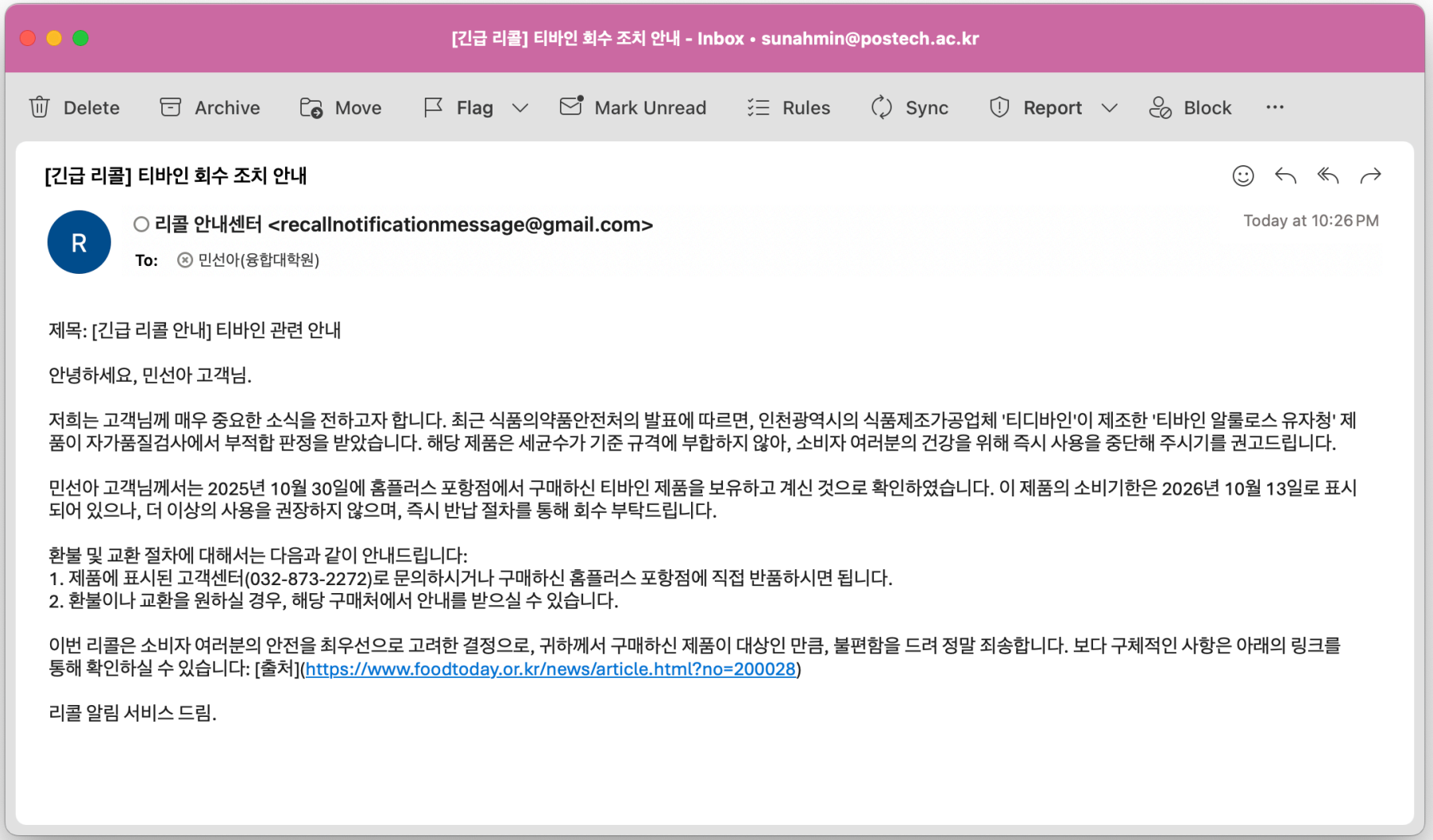
- 오늘 구매한 상품에 리콜 상품이 있어 안내를 발송하기 위해서, 리콜 상품의 구체적인 기본 정보, 고객의 개인정보, 리콜 관련 발간된 외부 뉴스 데이터를 모두 조회함.
- 조회된 내용을 바탕으로 LLM을 활용하여 사용 중단 안내, 환불/교환 절차, 의료기관 상담 권고, 참고 뉴스 요약 등과 함께 소비자에게 발송할 리콜 안내 메일을 작성함.
- 완성된 이메일 제목과 본문은 SMTP 서버를 통해 고객 이메일 주소로 실시간으로 발송함.

[이메일 작성 요구사항]

1. 제목은 "[리콜 안내] {product_info['product_name'] or product_name} 관련 안내".
2. 첫 문장은 "안녕하세요, {buyer_name} 고객님."으로 시작.
3. '리콜 알림 서비스'에서 발송된 서비스 메일임을 설명.
4. 섭취/사용 중단 및 환불/교환 절차를 구체적으로 안내.
5. 이미 섭취/사용한 경우 의료기관 상담을 권유.
6. '참고 뉴스' 섹션으로 위 [관련 뉴스] 내용을 요약해서 정리.

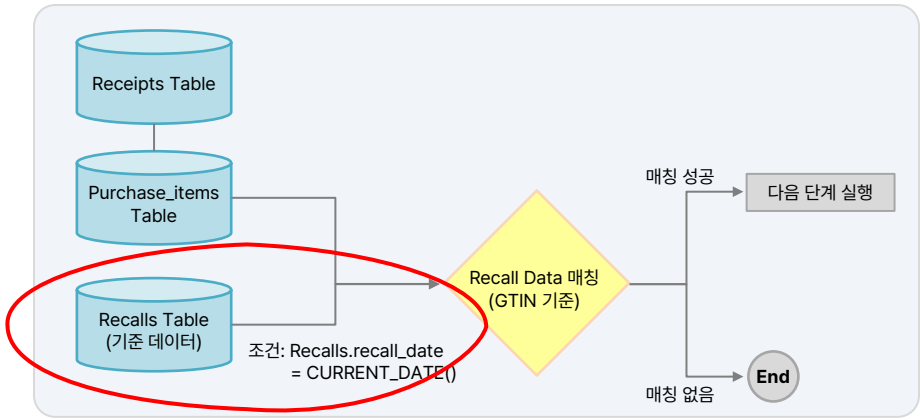


3-2. Service 1: Personalized Notification



3-3. Service 2: Recall Detection

- **소비자가 과거에 업로드한 구매 누적 정보(Purchase_items)를 바탕으로, 새롭게 발생한 리콜 상품과 매칭하여 리콜 정보가 존재하면 자동으로 안내 메일을 발송함.**
  - API 연동을 통해 소비자24의 식품 리콜 정보를 매일 조회하여, 신규 리콜이 발생하면 전처리하여 Recalls 테이블에 누적하여 저장함.
  - Pub/Sub 트리거를 활용하여, 신규 리콜이 발생했을 때에만 Cloud Function이 실행될 수 있도록 설정함.
    - 조건: Recalls.recall_date = CURRENT_DATE() → 리콜 정보가 오늘 업데이트된 것으로 한정하기 위함.

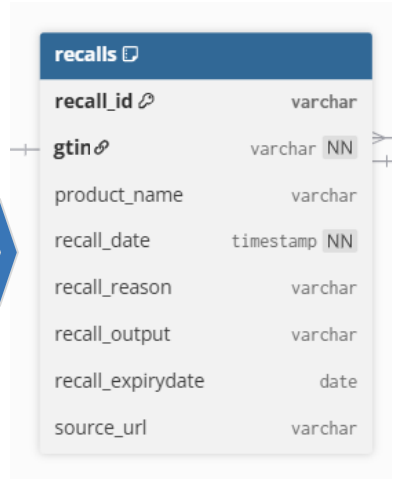



```python
# ------------------------------
# Pub/Sub 트리거 함수
# ------------------------------
@functions_framework.cloud_event
def recall_update(cloud_event):
    """
    Pub/Sub 트리거 (2세대 CloudEvents 방식)
    - 엔트리포인트 이름: recall_update
    """
    try:
        if cloud_event.data and "message" in cloud_event.data:
            pubsub_message = base64.b64decode(
                cloud_event.data["message"]["data"]
            ).decode("utf-8")
            print(f"🔔 Pub/Sub 트리거 수신: {pubsub_message}")

        result = process_and_load()
        return result

    except Exception as e:
        print(f"🚨 에러 발생: {e}")
        raise e
```





3-3. Service 2: Recall Detection

- **(예시) "롯O 아몬드 빼O로"**

2025.11.11
모O우님
롯O 아몬드 빼O로
구매

→

2025.12.09
(소비자 24)
이물질 혼합 이슈로
해당 상품 리콜 발생

→

(리콜 알리미)
리콜 및 뉴스 테이블
업데이트

→

(리콜 알리미)
소비자에게 리콜 정보
메일 발송

전체구매누적 (BigQuery: purchase_items table)

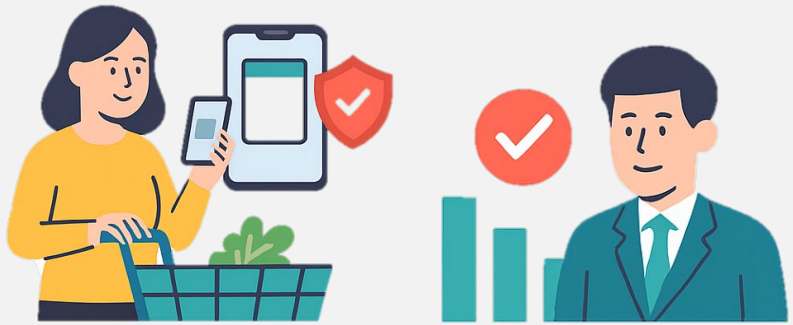
	purchase_item_id	receipt_id	gtin	product_name	recall_sent_at
0	074cb593-b76d-4892-9ac9-8145fc101121	2019ba59-3200-4294-8b14-38edc2c51696	8801019509391	탱크보이 120ml	None
1	e74cb3ce-29ed-4abe-a6fb-d5def7d77415	2019ba59-3200-4294-8b14-38edc2c51696	8801094793104	미닛메이드 스파클링 사과 345ml	None
2	8369d68f-00a7-47cd-b68e-571e3c7e5514	2019ba59-3200-4294-8b14-38edc2c51696	8801111923361	크라운 미니쉘 고소한 아몬드 24g	None
3	3ba5a6f1-b0ef-45ba-a069-d7e1c31614b1	2019ba59-3200-4294-8b14-38edc2c51696	8801062899890	LOTTE 아몬드빼빼로	None
4	4f31ca33-7aa6-42b4-8187-96d5dc326597	2019ba59-3200-4294-8b14-38edc2c51696	8804973304330	스니커즈 아몬드 500g	None
5	478845a0-14f4-4bf9-9a34-007da957f0d6	2019ba59-3200-4294-8b14-38edc2c51696	8801204204407	초코파이찰떡 215g	None
6	a78c678e-7eff-4ad3-9084-d36fdbf60c7d	2019ba59-3200-4294-8b14-38edc2c51696	8801062412730	나뚜루 바닐라&아몬드바 90㎖	None
7	c295e7b7-60a4-494e-93ab-fcd88ae2b121	2019ba59-3200-4294-8b14-38edc2c51696	8801117685201	오리온 포카칩 어니언 126g	None
8	64958a167d0e4ecdb536670d36820c04	27c4aeff1b2f48fda993e875290bb10b	8801150161113	상하치즈 짜먹는 까망베르	None
9	895badc40ae346a8ba63520153f8585a	27c4aeff1b2f48fda993e875290bb10b	8801111183574	크라운 조리퐁 50g	None

조건: Receipts.uploaded_at
= CURRENT_DATE()

4. 기대효과 및 한계점

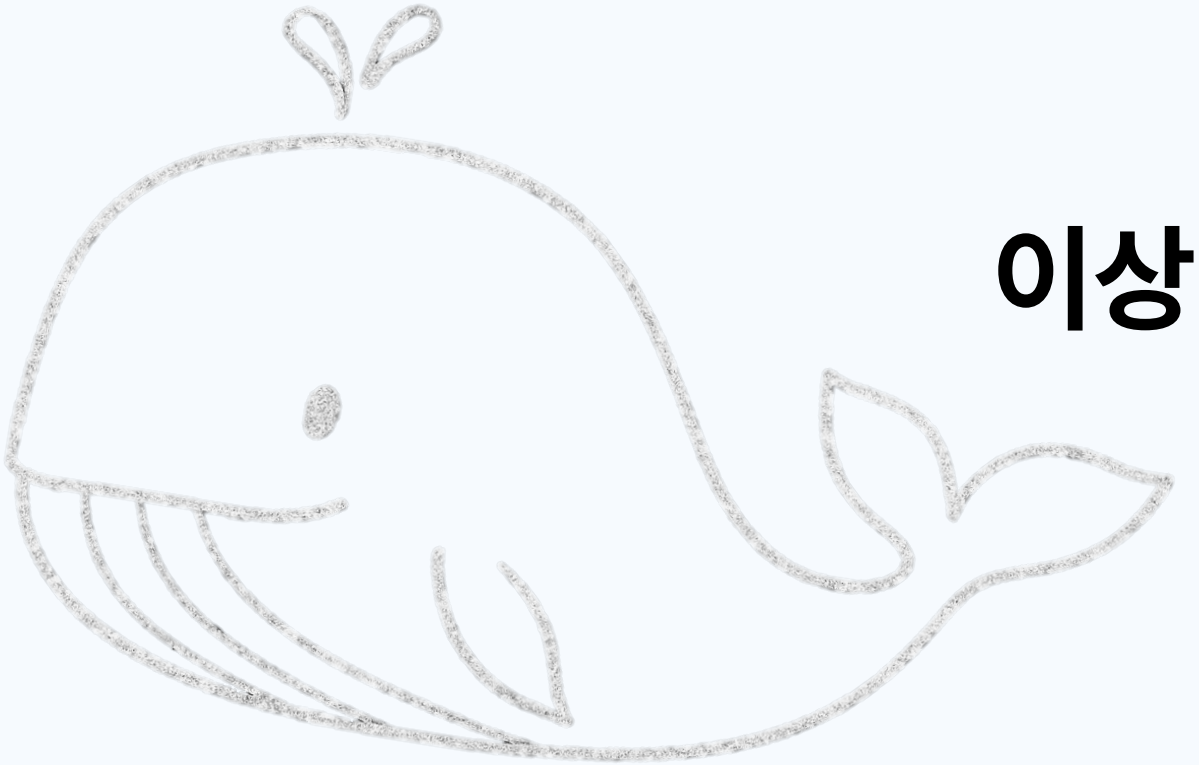
[기대 효과]

- **(소비자) 즉각적·실질적 안전 보장**
  - 제품 구매 이력에 대한 리콜 정보를 정확하고 신속하게 안내 받음
- **(사회/경제) 신뢰성·지속가능성 제고**
  - 리콜에 대한 대응 체계 구축을 통한 식품 안전 신뢰도 향상



[한계점]

- **가계부 앱 데이터는 소비 패턴, 계좌 정보, 카드 정보 등 고위험 개인정보로, 실시간으로 이를 받아서 활용하는 것이 불가능하였음.**
  - 개인정보보호법(2025.10.02): 제17조(개인정보의 제공), 제18조(개인정보의 목적 외 이용·제공 제한)
  - 전자금융거래법(2024.09.15): 전자금융업자의 이용자 정보 보호 의무, 금융거래 정보의 외부 제공 금지
- **온라인 구매내역의 경우 GTIN이 공개되지 않아 연결이 불가능하였음.**
  - 현재 GTIN을 공개하는 영수증만 분석이 가능하며, 추후 상품명을 바탕으로 GTIN을 매칭해서 분석에 활용하는 방향으로 확장 가능함.
- **실제 운영하는 어플/웹사이트 형태가 아니기에 이메일을 통해서만 알림 발송이 가능하였음.**



이상 발표를 마치겠습니다.

A. 영수증 OCR

```python
# ------------------------
# JSON 스키마 & 프롬프트
# ------------------------
def build_schema() -> Dict[str, Any]:
    return {
        "type": "object",
        "properties": {
            "merchant_name": {"type": "string"},
            "merchant_address": {"type": "string"},
            "purchase_date": {"type": "string"},
            "customer_name": {"type": "string"},
            "customer_email": {"type": "string"},
            "line_items": {
                "type": "array",
                "items": {
                    "type": "object",
                    "properties": {
                        "product_name": {"type": "string"},
                        "gtin": {"type": "string"},
                    },
                    "required": ["product_name", "gtin"],
                },
            },
        },
        "required": [
            "merchant_name",
            "merchant_address",
            "purchase_date",
            "line_items",
        ],
    }


SYSTEM_PROMPT = (
    "You are an expert OCR+IE system for Korean retail receipts. "
    "Extract merchant_name, merchant_address, purchase_date, customer_name, customer_email and line_items. "
    "customer_name MUST be the person explicitly labelled with words like "
    "'고객', '고객명', '고객님', '주문자', '고객 정보' next to the name. "
    "Store/merchant names such as 이마트, 롯데마트, CU, ㈜oo must NEVER be put in customer_name. "
    "If there is no obvious personal name near those labels, leave customer_name empty. "
    "If an email address appears near such labels, put only the email in customer_email. "
)

USER_PROMPT = (
    "이 영수증 이미지에서 다음 정보를 JSON으로 추출해줘.\n"
    "1. 구매처 (merchant_name): 매장 이름 (예: 홈플러스 포항점)\n"
    "2. 구매처 주소 (merchant_address): 매장 주소 (예: 경상북도 포항시...)\n"
    "3. 구매일시 (purchase_date): 결제 시점의 날짜와 시간.\n"
    "   - 형식: 'YYYY-MM-DD HH:MM'\n"
    "   - 예시1: '2025/10/30 19:20:37[목]' -> '2025-10-30 19:20:37' (요일 제거)\n"
    "   - 예시2: '[구 매]2025-07-05 15:59' -> '2025-07-05 15:59' ('[구 매]' 제거)\n"
    "4. 고객 이름 (customer_name): '고객님', '주문자' 옆에 있는 이름 (없으면 빈값)\n"
    "5. 고객 이메일 (customer_email): 이메일 주소 (없으면 빈값)\n"
    "6. 구매 품목 (line_items): 각 상품명과 GTIN(880으로 시작하는 13자리 숫자). GTIN이 없으면 빈값.\n"
)
```

```python
# ------------------------
# BigQuery: 3개 테이블 생성 및 저장
# ------------------------
def build_three_tables(parsed_json_list: List[Dict[str, Any]], file_names: List[str], items_df: pd.DataFrame):
    now = datetime.now(timezone.utc)

    receipt_rows = []
    customer_rows = []
    file_to_receipt_id = {}

    for data, fname in zip(parsed_json_list, file_names):
        rec_id = uuid.uuid4().hex

        c_email = data.get("customer_email", "").strip()
        cust_id = c_email if c_email else uuid.uuid4().hex
        c_name = data.get("customer_name", "").strip()

        # Receipts Data
        receipt_rows.append({
            "receipt_id": rec_id,
            "customer_id": cust_id,
            "file_uri": fname,
            "store_name": data.get("merchant_name", ""),
            "store_address": data.get("merchant_address", ""),
            "uploaded_at": now,
            "buy_at": data.get("purchase_date", "")
        })

        # Customers Data
        customer_rows.append({
            "customer_id": cust_id,
            "customer_name": c_name if c_name else "Unknown",
            "created_at": now,
            "updated_at": now
        })

        file_to_receipt_id[fname] = rec_id
```

B. 리콜 API 호출

```python
# ----------------------------
# 설정 (전역 변수)
# ----------------------------
API_URL = "https://www.consumer.go.kr/openapi/recall/contents/index.do"
SERVICE_KEY = "PQZ2TNPZ33"  # 실제 배포시에는 환경변수 권장
CNT_PER_PAGE = 100
CATEGORY = "0201"   # 식품

PROJECT_ID = "coral-chiller-478806-p9"
DATASET_ID = "recall_data"
TABLE_ID = "recall_data_bong"   # ✅ 네가 만든 테이블 이름

FIRESTORE_COLLECTION = "processed_recalls"
FIRESTORE_DB_ID = "recall-alert-service"
```

```python
def fetch_page(page_no: int):
    """API에서 특정 페이지의 데이터를 가져와 파싱"""
    params = {
        "serviceKey": SERVICE_KEY,
        "pageNo": page_no,
        "cntPerPage": CNT_PER_PAGE,
        "cntntsId": CATEGORY,
    }
    try:
        r = requests.get(API_URL, params=params, timeout=10)
        r.raise_for_status()
        root = ET.fromstring(r.content)

        items = []
        for item_elem in root.findall(".//content"):
            item = {}
            for child in item_elem:
                item[child.tag] = (child.text or "").strip()
            items.append(item)
        return items
    except Exception as e:
        print(f"API 호출/파싱 에러 (page {page_no}): {e}")
        return []
```

```python
def process_and_load():
    """최신 데이터를 가져와 Firestore 중복 체크 후 BigQuery에 저장"""
    page = 1
    new_rows_for_bq = []
    stop_crawling = False

    print("🚀 [시작] 리콜 데이터 수집 및 동기화 시작")

    while not stop_crawling:
        print(f"... {page} 페이지 수집 중")
        items = fetch_page(page)

        if not items:
            print("더 이상 API 데이터가 없습니다.")
            break

        for item in items:
            row = map_item_to_row(item)
            if row is None:
                continue

            recall_sn = row["recallSn"]

            # Firestore 중복 체크 (recallSn 기반)
            doc_ref = db.collection(FIRESTORE_COLLECTION).document(recall_sn)
            if doc_ref.get().exists:
                print(f"✋ 중복 데이터 발견 ({recall_sn}). 크롤링 중단.")
                stop_crawling = True
                break
```

C. 리콜 알림 메일 생성 프롬프트 전문

당신은 고객에게 보내는 리콜 안내 전문 이메일 생성기입니다.
아래 제공되는 데이터를 기반으로 고객 이름, 제품 정보, 리콜 사유, 뉴스자료 등을 반영해
정중하고 신뢰감 있는 한국어 이메일 메시지를 작성하세요.

중요한 컨텍스트:
- 이 메일은 구매처가 아닌 '리콜 알림 서비스'에서 발송하는 안내입니다.

[고객 정보]
- 고객 이름: {buyer_name}
- 고객 이메일: {buyer_email}

[구매 정보]
- 구매처(매장명): {store_name}
- 구매일: {purchase_date}

[제품/리콜 정보]
- 제품명: {product_info['product_name'] or product_name}
- 브랜드/제조사: {product_info['brand_owner']}
- GTIN 바코드: {gtin_code}
- 리콜 사유: {recall_info['recall_reason']}
- 소비자 문의처 주소: {product_info['contact_address']}
- 소비자 문의처 전화번호: {product_info['contact_phone']}
- 제품 이미지 URL: {product_info['image_url']}

[관련 뉴스]
{news_block}

이메일 작성 요구사항:
1) 제목은 "[리콜 안내] {product_info['product_name'] or product_name} 관련 안내".
2) 첫 문장은 "안녕하세요, {buyer_name} 고객님."으로 시작.
3) '리콜 알림 서비스'에서 발송된 서비스 메일임을 설명.
4) 섭취/사용 중단 및 환불/교환 절차를 구체적으로 안내.
5) 이미 섭취/사용한 경우 의료기관 상담을 권유.
6) '참고 뉴스' 섹션으로 위 [관련 뉴스] 내용을 요약해서 정리.
7) 마지막 서명은 아래와 같이 마무리:

감사합니다.

리콜 알림 서비스 드림