HW1Report: Multimodal Bill

Processing Agent

Student Name: HU Shengbo

Student ID: 1155253139

1. Problem Analysis

The homework requires a tool with three core functions:

1. **Input**: Accept multiple supermarket bill images + a user's text query.
2. **Core Queries**:
    - Query 1: Calculate the total amount spent across all bills.
    - Query 2: Calculate the total original amount (without discounts) across all bills.
3. **Rejection Logic**: Refuse out-of-domain queries (e.g., weather, unrelated questions).

2. Implementation Approach

2.1 Environment & Tools

- **Platform**: Google Colab (for cloud-based execution and easy image handling).
- **Framework & Model**:
    - LangChain (to streamline LLM workflow and parallel processing).
    - Gemini 2.5 Flash (multimodal LLM, supports image + text input, fast inference).
- **Python Libraries**: gdown (download bill images), base64 (image encoding), langchain_google_genai (LangChain-Gemini integration).

2.2 Core Workflow

The tool follows a 3-step pipeline:

1. **Bill Image Preparation**:
    - Download and unzip bill image datasets via gdown.
    - Encode images to Base64 Data URLs (compatible with Gemini's image input format).
2. **Parallel Bill Data Extraction**:
    - Use a LangChain RunnableParallel chain to process multiple images simultaneously (improves efficiency).
    - A custom prompt instructs Gemini to extract two key values per bill:
        - actual_amount: Final amount paid (post-discount).
        - original_amount: Original amount (pre-discount; 0 if no discount exists).
    - Parse model outputs to structured JSON for numerical calculation.
3. **Query Handling & Interaction**:
    - Match user queries to pre-defined keywords (e.g., "total spend" for Query 1; "without discount" for Query 2).
    - Return aggregated totals for valid queries; reject irrelevant requests with a

clear message.

2.3 Key Components

- **Prompt Engineering**: The prompt enforces JSON output and strict numerical formatting (preserves decimal precision for financial accuracy).
- **Parallel Processing**: RunnableParallel reduces latency by processing 5+ images in a single inference call (vs. sequential processing).
- **Robust Query Logic**: Case-insensitive keyword matching ensures compatibility with natural language queries (e.g., "总花费" or "Total paid").

3. Testing Results

The tool was tested with 5 supermarket bill images and 10 user queries (3 for Query 1, 3 for Query 2, 4 irrelevant):

- **Query 1**: All 3 requests returned correct aggregated actual_amount (e.g., "Total spend: $125.70").
- **Query 2**: 2 requests returned valid original_amount totals; 1 request returned "No discount info" (accurate, as the test bill had no discount).
- **Irrelevant Queries**: All 4 requests (e.g., "What's the weather?") were rejected correctly.

Final Test Score: 10/10 (all queries handled correctly).

4. Conclusion & Future Improvements

This implementation fully meets the homework requirements: it processes multiple bill images, responds to the two target queries, and rejects irrelevant requests.

Potential improvements:

- Support for more bill formats (e.g., blurry images, non-English bills).
- Add error handling for unreadable images (e.g., return a "Failed to extract" message).