需要改進的地方

- **Contextual Relevancy (上下文相關性) 偏低 (0.43 - 0.76)**：
    - 雖然抓到了正確的文檔（Recall=1），但也抓進來了很多不相關的雜訊。
    - 例如第 5 題（牛奶水），分數只有 0.4375。這代表餵給 LLM 的參考資料中，有一半以上都是跟問題無關的廢話。雖然 LLM 很聰明排除了干擾，但這會浪費 Token 費用並增加延遲。
- **Answer Relevancy (答案相關性) 波動 (第 1 題僅 0.625)**：
    - 第 1 題分數較低，可能代表生成的回答雖然正確，但沒有「直接命中」使用者的痛點，或者回答太過冗長、包含了不需要的資訊。
- **Contextual Precision (上下文精確度) 不穩定 (第 2 題僅 0.5)**：
    - 第 2 題（游泳池味道）Precision 只有 0.5，代表正確的答案雖然被召回了，但在候選名單（Top-K）中的**排名不夠靠前**（可能排在第 2 或第 3，而不是第 1）。這顯示 Reranker（重排模型）在該題還有進步空間。

針對性優化策略

根據上述數據，分二個階段進行優化：

第一階段：優化檢索品質 (針對 Contextual Relevancy 低的問題)

目標：減少餵給 LLM 的雜訊，提高純度。

1. **調整 Chunk Size (切分大小)**：
    - **問題推測**：目前的切分可能包含太多不相關的上下文。例如一個 Chunk 有 500 字，但真正回答問題的只有 50 字，剩下 450 字都會拉低 Relevancy 分數。
    - **解法**：嘗試縮小 chunk_size（例如從 500 改為 250），並適度增加 chunk_overlap（重疊）。
2. **優化 Reranker 的 Top-K 策略**：
    - **問題推測**：你可能在 Rerank 後選了 Top-4 給 LLM。如果第 4 名的相關性分數其實很低，它就會成為雜訊。
    - **解法**：不要固定取 Top-4。改用 **「閾值截斷 (Thresholding)」**。例如：只取 Rerank 分數 > 0.5 的文檔。如果只有 1 篇超過 0.5，就只給 LLM 這一篇，不要硬湊 4 篇。

第二階段：優化生成品質 (針對 Answer Relevancy 低的問題)

目標：讓回答更精簡、更像人類。

1. **Prompt Engineering (提示詞工程)**：
    - 針對第 1 題（簡訊帳單），使用者問「要去哪裡登記？」。
    - **目前回答**：「可至本公司網路 e 櫃台/E 帳單申辦...」
    - **優化方向**：在 Prompt 中加入規則：「回答必須直接、簡潔。如果使用者問地點，優先回答地點或連結。避免過多的官腔。」
2. **Few-Shot Prompting**：
    - 在 Prompt 裡給 LLM 看一兩個好回答的範例，讓它模仿那種風格。