tracers.py 檔案所撰寫的**功能規格說明（Functional Specification）**：

**功能規格說明：追蹤日誌記錄模組（tracers.py）**

**一、目的**

本模組實作一個自定義的追蹤處理器 **LogTracer**，用於記錄應用程式中追蹤（trace）和區段（span）的啟動與結束事件，並寫入日誌資料庫以利後續查詢與除錯。

**二、功能描述**

**1. 建立追蹤 ID**

* 函數名稱：make\_trace\_id(tag: str) -> str
* 功能：
  + 生成一個唯一的追蹤 ID，格式為 trace\_<tag><隨機字串>。
  + 隨機字串長度依據 tag 長度動態計算，確保總長度為 38 字元（含 "trace\_"）。

**2. 自定義追蹤處理器：LogTracer**

* 繼承自 TracingProcessor，實作以下功能：

**2.1 名稱擷取**

* 方法：get\_name(trace\_or\_span)
* 功能：從 trace\_id 中萃取名稱（即 tag），用以識別追蹤或區段所屬邏輯。

**2.2 追蹤事件**

* on\_trace\_start(trace)：在追蹤開始時記錄 "Started: <trace.name>"。
* on\_trace\_end(trace)：在追蹤結束時記錄 "Ended: <trace.name>"。

**2.3 區段事件**

* on\_span\_start(span)：
  + 在區段開始時記錄 "Started" 訊息。
  + 若 span\_data 包含 type, name, server 或錯誤訊息，則一併加入日誌。
* on\_span\_end(span)：
  + 在區段結束時記錄 "Ended" 訊息。
  + 同樣支援 type, name, server, error 等屬性紀錄。

**2.4 強制刷新與關閉**

* force\_flush() 和 shutdown()：目前為空實作（保留擴充用）。

**三、相依模組與元件**

* agents.TracingProcessor, agents.Trace, agents.Span：核心追蹤資料結構。
* database.write\_log：寫入日誌的函數介面。
* secrets, string：用於隨機 ID 的產生。

**四、輸出格式**

每筆日誌紀錄以如下格式寫入（透過 write\_log 函式）：

write\_log(name: str, type: str, message: str)

* name：由 trace\_id 中萃取出的邏輯名稱。
* type：例如 "trace"、"span"、或自定義型別。
* message：如 "Started X"、"Ended Y"，可能包含錯誤描述。