Use Case:

Retail Customer Support Chatbot with Purchase Insights

You are building an **AI-powered retail customer support chatbot**. This chatbot:

- Answers customer queries about their previous purchases, order status, and recommends products.
- Maintains conversational context (memory) for a better user experience.
- Extracts structured information from customer messages to perform specific actions (e.g., fetch order details, recommend products).
- Uses robust schema validation to ensure extracted data is reliable.

Assignment Questions

Scenario:

You work for "QuickShop", an e-commerce retailer. You are tasked to implement the backend logic for their new LLM-powered chatbot. Your system should:

- Use **LangChain Chains** to process customer queries.
- Extract structured data (like order ID, product name, issue type) using **OutputParsers**.
- Validate all structured data using **Pydantic** models.
- Maintain conversation context with **ChatHistoryMemory** so the agent can reference earlier user intents and responses.

Part 1: Schema Design & Output Parsing

Q1.
Design a **Pydantic model** called `OrderQuery` to capture structured information from a customer's message, containing:

- `order_id` (int, optional)
- `product_name` (str, optional)
- `issue_type` (str, e.g., "return", "status", "recommendation", etc.)
- `details` (str, optional)

Then, using **LangChain's StructuredOutputParser**, define a schema for extracting these fields from an LLM's response.

Part 2: Chain Construction

Q2.
Build a **LangChain Chain** (using LLMChain or similar) that:

- Takes a user query as input (e.g., "Where is my order #12345?" or "Suggest me a phone under ₹20,000").
- Uses your output parser from Q1 to extract structured information.

Part 3: Data Validation

Q3.
After extracting structured data, use your **Pydantic model** to validate and instantiate an `OrderQuery` object.

- What happens if the LLM output is missing a required field or gives a wrong data type?
- Show code handling such cases with clear error messages.

Part 4: Memory Integration

Q4.
Demonstrate use of **ChatHistoryMemory**:

- Maintain context for a multi-turn conversation where the customer first asks for order status, then wants to initiate a return for the same order—without repeating the order ID.
- Show code snippets and a sample chat transcript.