Analytics Engineer Take-Home Test

Background

You are working with a mobile game studio. The studio wants to track player behaviour starting with basic events: `install` and `purchase`. Your task is to build the skeleton of a system to support this.

Deliverables

Submit a small Python project with:

1. A simple Python SDK for game clients to:
   - Send an `install` event
   - Send a `purchase` event
   - Communicate with a backend via HTTP
2. A RESTful API (also Python) that:
   - Receives these events via POST requests
   - Sends the event data to AWS Kinesis Firehose
   - Is structured to eventually store events in a Snowflake table
3. Documentation (README or inline comments) describing:
   - How to use the SDK
   - Your design decisions
   - Assumptions made
   - What you would do next to make this production-ready

Requirements

- You do not need to create a fully working system. Use pseudo code, mock clients and placeholders where appropriate.
- Demonstrate your intentions with code structure, documentation, and abstractions.
- Use any Python web framework (e.g., FastAPI, Flask).
- Use `boto3` for interacting with AWS (mock if needed).
- Snowflake interaction should be described, not implemented.
  - We use multiple currencies, how would you design this in a Snowflake table?
- Please structure your code as a Python package (installable SDK folder and separate API folder).

What We're Looking For

- Clear modular code and package structure
- Well-documented SDK and REST API design
- Understanding of event-driven architectures
- Basic knowledge of Kinesis Firehose & Snowflake workflows
- Snowflake database design
- Awareness of real-world considerations (e.g., retries, schema validation, future extensibility)

Project Structure Example

```
game_event_tracking/
├── sdk/
│   ├── __init__.py
```

```
4  │   ├── events.py
5  │   └── client.py
6  ├── api/
7  │   ├── main.py
8  │   ├── models.py
9  │   └── firehose_client.py
10 ├── tests/
11 │   ├── test_sdk.py
12 │   └── test_api.py
13 └── README.md
```

Time Expectation

Spend no more than 3–5 hours on this. If something is unclear or you want to make assumptions, state them in the README.