Technical Assignment

(Backend Engineer Intern - March 2025)

Business Context

Our analytics platform ingests raw event data from multiple sources and needs to transform it into a standardized format before analysis. Your task is to build a simple data transformation pipeline that demonstrates your logical thinking and data processing skills.

Assignment Overview

Create a service that:

1. Ingests data from different public APIs
2. Transforms and normalizes the data
3. Applies business rules to enrich the data
4. Provides an API to query the processed data

Technical Requirements

1. Data Sources (Public APIs)

Source 1: E-commerce Product Data

- API: `https://fakestoreapi.com/products`
- Documentation: https://fakestoreapi.com/docs
- This represents your product catalog

Source 2: User Data

- API: `https://randomuser.me/api/?results=20`
- Documentation: https://randomuser.me/documentation
- This represents your user profiles

Source 3: Dummy Transactions/Orders

- API: `https://my.api.mockaroo.com/orders.json?key=e49e6840`
- This represents purchase transactions

2. Transformation Requirements

Convert data from all three sources into this unified format:

```
{
  "entity_id": "generated-unique-id",
```

```
  "entity_type": "product|user|transaction",
  "timestamp": "ISO-8601-format",
  "data": {
    // Normalized fields from the source
  },
  "metadata": {
    "source": "which API",
    "processed_at": "processing timestamp"
  }
}
```

3. Business Logic Requirements

Implement these data enrichment rules:

- Join transaction data with product data to include product details in transactions
- Join transaction data with user data to create complete purchase records
- Calculate total spending per user
- Identify the most popular product categories
- Calculate average transaction value

4. API Endpoints

Create these endpoints:

- `GET /data/{entity_type}` - Retrieve processed data by type
- `GET /insights/users` - User spending patterns and statistics
- `GET /insights/products` - Product popularity metrics

Implementation Guidelines

- Use any programming language/framework
- In-memory storage is acceptable
- Include comments explaining your logical approach
- Focus on data transformation and analysis logic

Evaluation Criteria

- Data fetching and handling approach

- Transformation logic and data normalization
- Implementation of business logic and data joins
- Error handling for API requests
- Code organization and readability

Deliverables

1. Source code in a GitHub repository
2. README with:
    - Setup instructions
    - Explanation of your data pipeline architecture
    - API documentation with example requests/responses
3. Brief summary of how you approached the problem

Time Frame

Date of submission: March 9, 2025

Getting Started Tips

1. Start by setting up functions to fetch data from each API
2. Test API responses and make sure you understand the data structure
3. Implement transformation functions for each data type
4. Apply business rules to combine and enrich the data
5. Create API endpoints to expose your processed data

Good luck! We're looking for your approach to handling multiple data sources and applying logical transformations. If you are having any doubt, reach out to chetan@opinium.ai