How long did you spend on the test? What would you add if you had more time?

Response: I spent approximately 5 horas on the test. If I had more time, I would:
Enhance Error Handling: Improve the robustness of error handling in the API to manage edge cases and unexpected input more gracefully.
Extend Test Coverage: Add more unit tests and integration tests to cover additional scenarios and ensure comprehensive validation of the functionality.
Optimize Performance: Review and optimize the performance of the code, particularly in scenarios involving large datasets or complex promotions.
Improve Documentation: Create more detailed documentation for the API, including usage examples and detailed explanations of the business logic.

What was the most useful feature that was added to the latest version of your chosen language? Please include a snippet of code that shows how you've used it.

Response: One of the most useful features added to the latest version of Java (Java 21) is the Record Classes, which provide a compact syntax for creating data-carrying classes. This feature helps to reduce boilerplate code and enhances readability.
Example

```
public record Item(String id, String name, int priceInCents, int quantity) {
    // The record automatically provides getters, equals(), hashCode(), and
toString() methods.
}
```

Explanation: Using record classes simplifies the creation of immutable data objects, making the code more concise and easier to understand.

What did you find most difficult?

Response: The most challenging aspect of the project was implementing the promotion logic effectively. Handling different types of promotions, ensuring correct application of discounts, and integrating this logic with the cart system required careful consideration and testing. Additionally, managing edge cases where promotions overlap or multiple promotions apply simultaneously added complexity.

What mechanism did you put in place to track down issues in production on this code?
If you didn’t put anything, write down what you could do.

Response: To track down issues in production, I would implement the following mechanisms:
Logging: Integrate a comprehensive logging framework (e.g., Logback or SLF4J) to capture detailed logs of API requests, responses, and errors. This

will help in diagnosing issues and understanding the behavior of the system in production.
Monitoring: Set up monitoring tools (e.g., Prometheus, Grafana) to track system metrics, performance, and errors. Alerts can be configured to notify the team of any anomalies or issues in real-time.
Error Tracking: Use an error tracking service (e.g., Sentry or New Relic) to capture and report exceptions, providing insights into the frequency and nature of errors encountered by users.
Unit and Integration Tests: Ensure thorough testing in development to minimize the risk of issues reaching production.

The WireMock represents one source of information. We should be prepared to integrate with more sources. List the steps that we would need to take to add more sources of items with different formats and promotions
.
Response: To integrate additional sources of items with different formats and promotions, follow these steps:

Identify Requirements: Determine the requirements and formats for each new data source. Understand the structure and type of data each source provides.

Create Data Adapters: Implement data adapters or parsers for each new source to convert data into a uniform format compatible with the existing system. This may involve creating new classes or methods to handle specific formats.

Update API Integration: Modify the API integration logic to accommodate additional data sources. Ensure that the application can fetch and process data from multiple sources seamlessly.

Configure Source Management: Implement configuration management to handle different sources. This might include setting up environment-specific configurations or source-specific settings.

Testing: Thoroughly test the integration with new sources to ensure that data is correctly fetched, processed, and applied. Verify that promotions and pricing logic works as expected with the new data formats.

Documentation: Update documentation to reflect the integration of new sources, including details on how to configure and use the new data sources.