Coding Task for Software Engineer Candidates

The following task will allow you to demonstrate your ability to deliver readable, maintainable and testable code. As an agile organization, we are applying equal emphasis on assessing both your problem solving and testing abilities.

Be prepared to present your solution over conversation that will lead to questions around design decisions, rationale and future expansion.

The exercise should take around 60-90 minutes; please don't spend much longer than this.

- Please complete the following task in Java or Kotlin
- Use git to version control the code and once complete, send us a link to the Github (or similar) repository via the recruitment agent

Instructions: Complete the steps in order. Don't read ahead. At each step build the simplest possible solution which meets the requirement. Tag and git commit after each step so that your approach is clear.

Step 1: Build an Orders Service

- Build a service that's able to receive simple orders of shopping goods from the command line
- Apples cost 60 cents and oranges cost 25 cents
- The service should be able to calculate that:
    - [Apple, Apple, Orange, Apple] => $2.05
- Make reasonable assumptions about the inputs to your solution; for example, candidates may take a list of strings as input
- Add unit tests that validate your code

Step 2: Simple offer

- The shop decides to introduce two new offers
    - buy one get one free on Apples
    - 3 for the price of 2 on Oranges
- Update your functions & unit tests accordingly

Step 3: Build a Customer Notification Service

- Customers complained that they don't know if their orders made it through or not as there is no notification of success
- Build a service that listens for when orders are complete and sends a notification to the customer regarding its status and estimated delivery time
- The Mail service subscribes to events from the Orders service and publishes an appropriate event that the customer (you) is able to read from the terminal

Step 4: Limited Stock

- Stock can now run out, this means that customers need to be notified that their order failed

Optional Step 5: Events via Kafka (or another appropriate message queue, ie. Rabbit MQ)

- When the customer submits an order, this data is published via Kafka as an 'OrderSubmitted' event over an 'order-submitted' topic
- Add additional appropriate topics that enable the conversion of existing events to events submitted over Kafka
- https://kafka.apache.org/quickstart
- https://docs.confluent.io/current/clients/java.html