Technical Evaluation Amenitiz

Problem to Solve

> You are the developer in charge of building a cash register. This app will be able to add products to a cart and display the total price.

Objective

Build an application prototype responding to these needs.

By prototype, we mean:

- It is usable while remaining as simple as possible,
- We place little emphasis on the visual,
- We do not expect complexity that does not meet the primary functional need of the app.

Technical requirements

- A web interface (even minimalist),
- Built-in Ruby on Rails,
- Covered by tests.

Bonus

- Using React,
- Following TDD methodology.

Assumptions

Products Registered

Product Code	Name	Price
GR1	Green Tea	3.11 €
SR1	Strawberries	5.00 €
CF1	Coffe	11.23 €

Special conditions

- The CEO is a big fan of buy-one-get-one-free offers and green tea. He wants us to add a
  rule to do this.
- The COO, though, likes low prices and wants people buying strawberries to get a price
  discount for bulk purchases. If you buy 3 or more strawberries, the price should drop to 4.50€.
- The VP of Engineering is a coffee addict. If you buy 3 or more coffees, the price of all coffees should drop to 2/3 of the original price.

Our check-out can scan items in any order, and because the CEO and COO change their minds
often, it needs to be flexible regarding our pricing rules.

Test data

Basket	Total price expected
GR1,SR1,GR1,GR1,CF1	22.45€

Basket	Total price expected
GR1,GR1	3.11€
SR1,SR1,GR1,SR1	16.61€
GR1,CF1,SR1,CF1,CF1	30.57€

Deliverable

- The codebase in a public git repository,
- The app: online on Heroku or on a custom server.

Things we are going to look into or ask about

- Best practices
- Commit history
- Code structure and flow
- To make some changes to the code.