BESTSELLER®

Coffee store- Backend API

We need a **backend** for our new online coffee place startup business, where users can place drinks with toppings orders and admins can create/update/delete drinks/toppings and have access to the most used toppings. Also, there will be a discount program for the clients that order more than average.

Functional Requirements

- Using **Java & Spring boot** is encouraged, but not mandatory. You can use another language or framework (Kotlin, Quarkus... **JVM** languages are **our preference**) to build the assignment.
- Develop an API that will be used to **order drinks** with any of the topping combinations.
- Visitor journeys should be transparent; the **current amount of the cart** and the products should be communicated back to the caller of the API.
- When finalizing the order, the original **amount and the discounted amount** should be

Drinks examples:

- Black Coffee - 4€
- Latte - 5€
- Mocha - 6€
- Tea - 3€

Toppings/sides examples:

- Milk - 2€
- Hazelnut syrup - 3€
- Chocolate sauce - 5€
- Lemon - 2€

Discount logic

We would like to recompensate clients that are ordering more than average, so we will offer them a discount system:

1. If the total cost of the cart is more than 12 euros, there should be a 25% discount.
2. If there are 3 or more drinks in the cart, the one with the lowest amount (including toppings) should be free.
3. If the cart is eligible for both promotions, the promotion with the lowest cart amount should be used and the other one should be ignored.

Admin API

- Should be able to create/update/delete products and toppings.
- Most used toppings for drinks.

Note

What we would like to see is your structure and how you approach this problem, we are not interested in the boilerplate code. If there is a functionality that is taking too long or is too complex, just write some notes explaining how you would approach it, and have it prepared for the technical interview for any follow-up questions. You can time-box this assignment for 4 hours.

There is no requirement to implement the authentication using any complex solution, nor a login process, you can make some assumptions and use a simple approach for the admin functionality.

Same applies for any frontend functionality, you only need to focus on the **API** and the **backend** part.

It is up to you to the level of detail you will use to achieve the functionality described above, it is broad on purpose, you can build it simple, or you can take a more detailed approach. All decisions you take will be evaluated and you should be able and feel confident to explain these decisions.

What we will look at in the assignment

Project structure

- Are the packages well structured?
- Is it easy to run the application?
- Is it dockerised?
- Does it have good README?
- Is it deployable to any cloud provider as it is?

Endpoints

- Are best practices followed? Usage of correct HTTP verbs?
- Is it documented?

Testing

- Do you have enough coverage?
- Are you testing the right things? And how are you testing them?

Logging

- Does it contain meaningful logs?

Should you have any questions, please do not hesitate to contact our recruitment team or any of the people assigned in the email.