5st Latam Full Stack Dev Bootcamp Dev Test

Challenges

The Problem

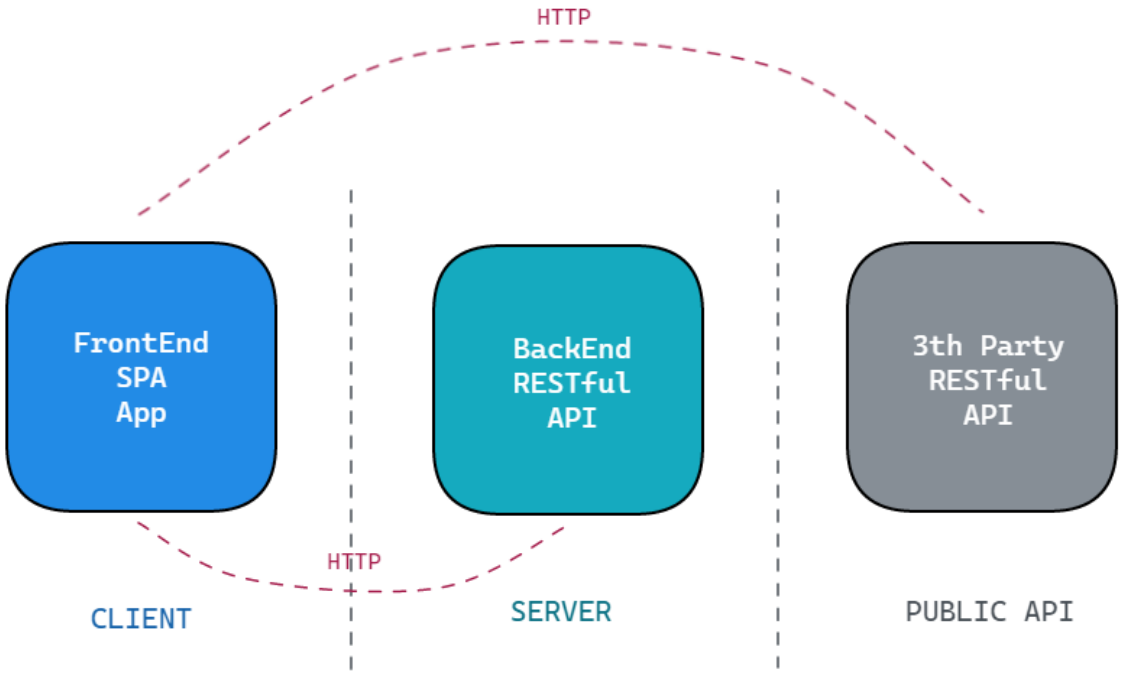
Nowadays, many people have stayed at home so much time that the way they relate to other persons and their habits have changed. This is the main reason to develop an easy and intuitive ecommerce web application that will help persons to search, and buy products online. Also, give the possibility to have a cart functionality to buy a set of products as in real life.

The application will be a place to search products, displaying the main information about the products, such as the first name, description, price, etc., or any other information that can be useful for the user.

The user is gonna be able to select products and add them to the cart in order to make a cart order, and get the historic information of my purchases.

Cart order is a set of products that the client bought.

Architecture



Main Guidelines

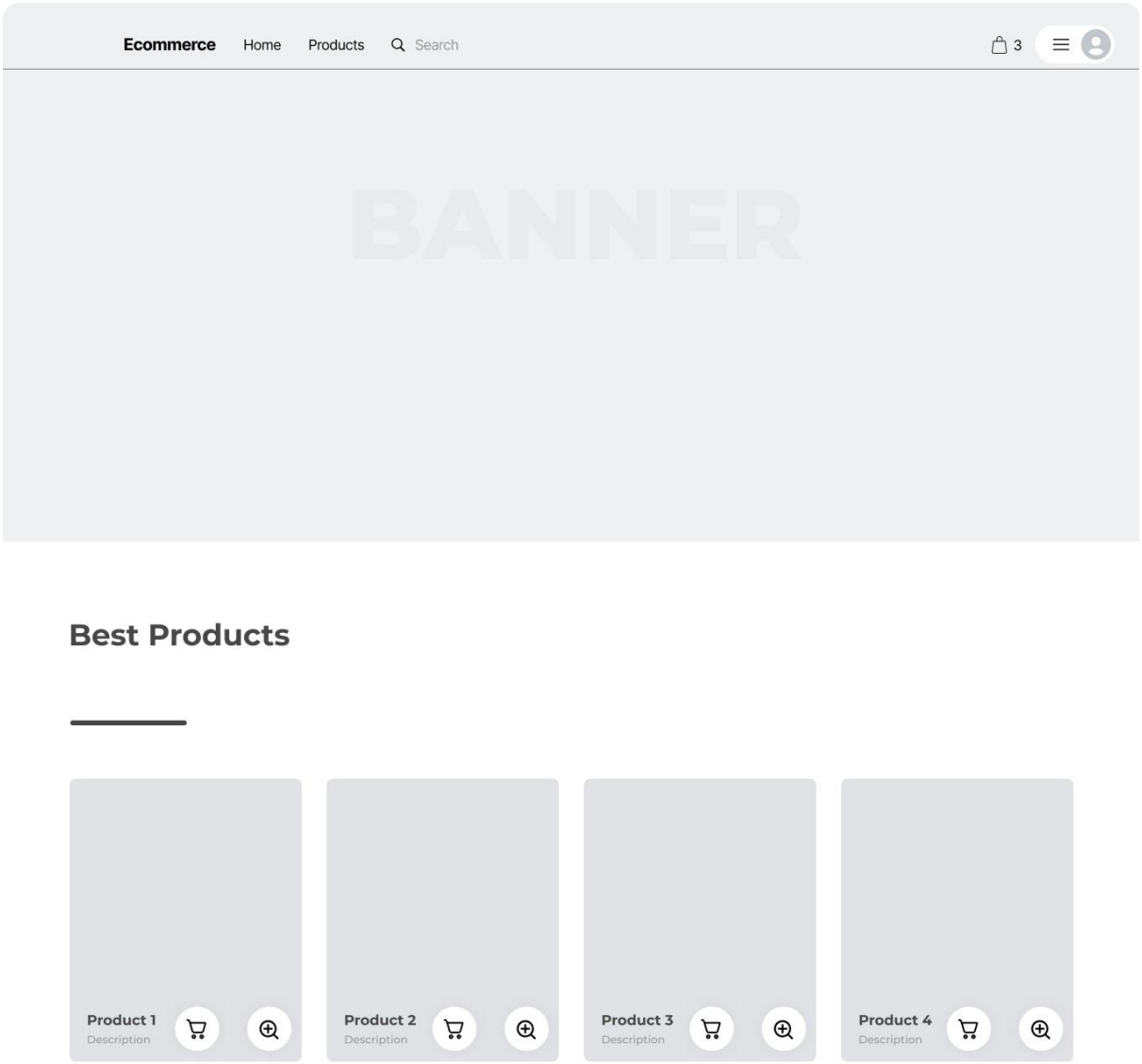
- You are allowed to consult any source you want (Google, Books, StackOverflow, etc).
- You are allowed to ask questions to Jalasoft Coaches on any question you may have.
- Try to do your best in terms of code best practices, styling, correct use of tools, clean code, comments, etc.

Note for the third party API

- Documentation (https://storerestapi.com/)
- In order to consume the data from the third party API, it is important to implement the authentication flow. Please read the third partyAPI documentation.

FrontEnd

Create a SPA (Single Page Application) to consume the third party API for Random Products (https://api.storerestapi.com/products - more details of how to consume the API in its official documentation). In order to show a list of products, with the following reference layout:



Figma wireframe link

Guidelines

- Create a SPA (Single Page Application), you are free to choose the Libraries/Framework to develop this app:
    - React
    - Angular
    - Vue
    - Svelte
    - Vanilla Javascript
    - etc
- You are allowed to use any npm library to work in the requirements (Axios, Fetch, Lodash, etc)

- If the app is responsive is a plus (render correctly on a variety of devices or screen sizes), you are allowed to use any CSS Framework to accomplish this task (Material UI, Theme UI, Bootstrap, Tailwind, etc).

Challenges

1. Display a list of products. (It is important to add the “Add to cart” button) (Note: The image could be a default image)
2. Implement the Tab/Button to view Cart items in a new page/modal.
3. Implement the Cart order functionality (Add the “Add order” button).
4. Once a product is bought you must display this information in the list of products
5. Implement the search functionality to search products at the top of the products list.

Backend Requirements

In order to have an easy, reliable, fast, and scalable way to manage the contacts information, we want to build a REST API where the users will be able to perform different operations.

Tech:

- C#
- Java
- Python
- NodeJS
- Database (optional – participant choice)

Considerations

- You can use only an in-memory storage instead of a database, try not to invest too much time in this option.
- Contact Information entities can be filled with mocked data when starting the REST API if you are only focused on the backend requirements.
- Make sure that relationships on the diagram are implemented on the backend side project.

Challenges

1. Endpoint to add a cart order.
2. Endpoint to determine if a specific product was already bought. (You can determine if a product was bought when a product was part of a order)
3. Endpoint to get products using a search word or letters as a criteria to search. For example ´milk´, ‘mouse’, and others. (Note if I send the word ´use´ it should return all the products filtering this word)
4. Endpoint to get orders between two dates (end and start dates).

Endpoint design

This is an example of endpoint design that you can choose to follow for the implementation of your REST API based on naming, http verbs and http status code for the response:

Name	**Verb**	**Resource Name**	**Status Code**
Create Order	POST	/api/v1/order	201
Get Product Bought	GET	/api/v1/productbought	200
Get products of all orders	GET	/api/v1/order?searchProduct='milk'	200
Get products between two dates	GET	/api/v1/order?startDate='06/04/2022'&endDate='06/24/2022'	200

Back-End & Front-End (Bonus)

Integration of the Frontend app with the REST API.

Challenges

1. Option to buy many items of the same product (stock)
2. Implement the product's details.
3. Implement the home page.
4. Create a page to display how many items you bought for an specific product
5. Create a page to generate an historic report of all the items that you bought (all)