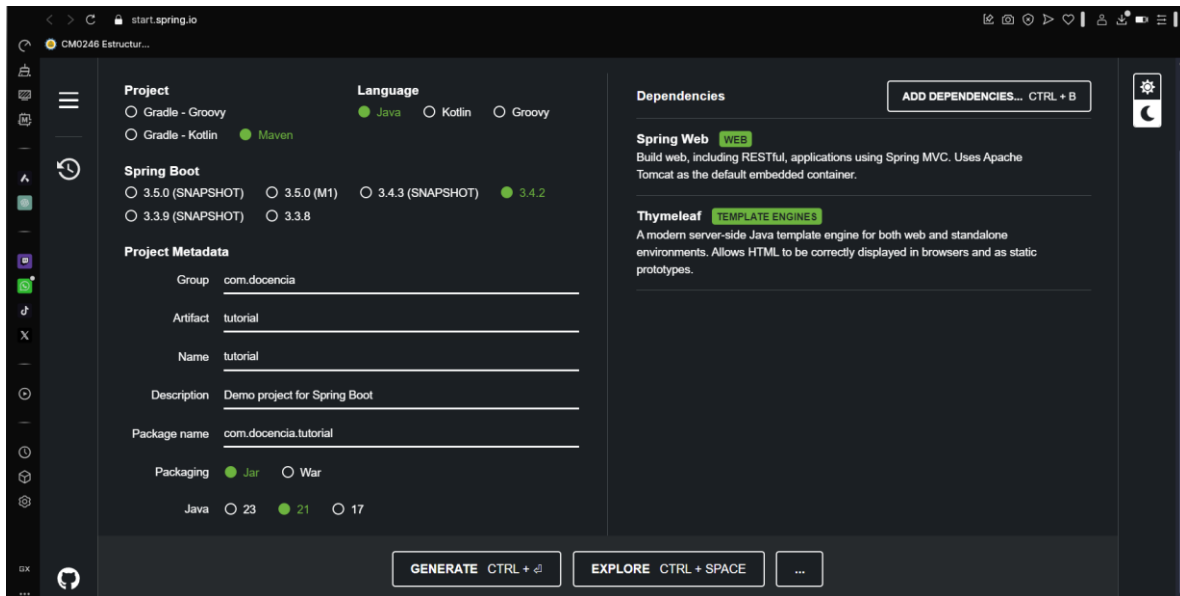


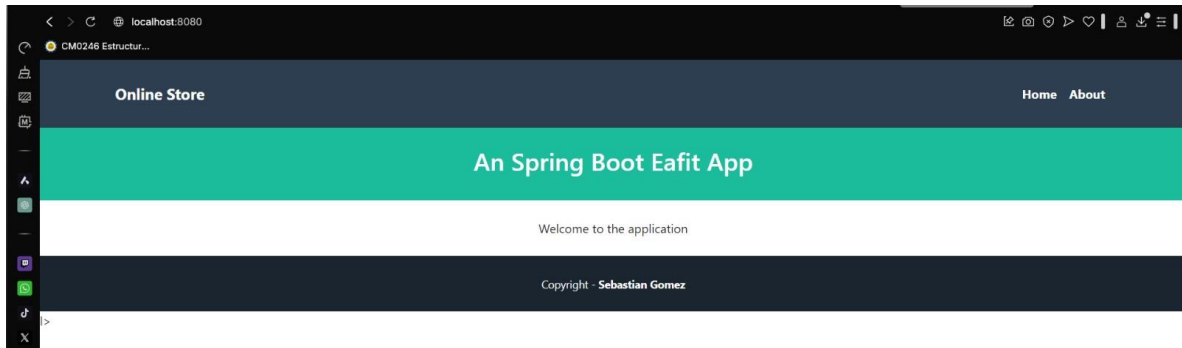
Tutorial 01 to do in class – Remember to upload the repo link to Teams

FIRST COMPLETE THE ENTIRE TUTORIAL (IGNORE THE ACTIVITES) – AT THE END, COMPLETE THE PROPOSED ACTIVITIES

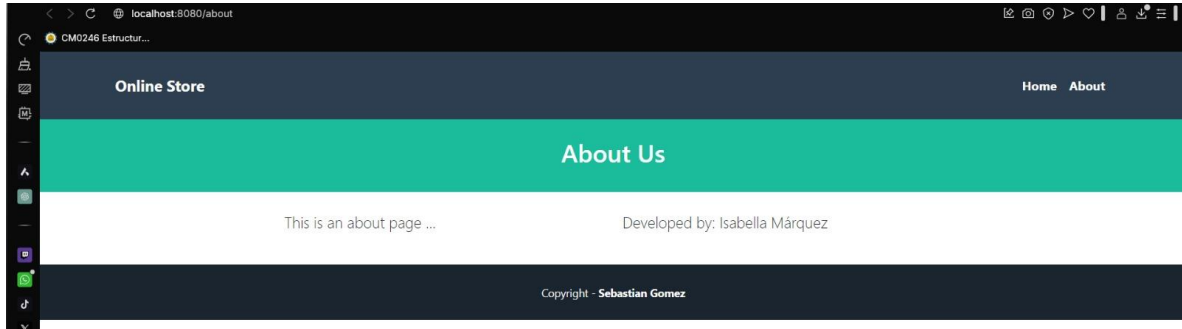
Paso 1: Creación del Proyecto Spring Boot



A. A “clean” hello world version



B. Una página "About"



Activity 1

- **Take a look of the previous routes. Do you have any comment? We will discuss it the next class.**

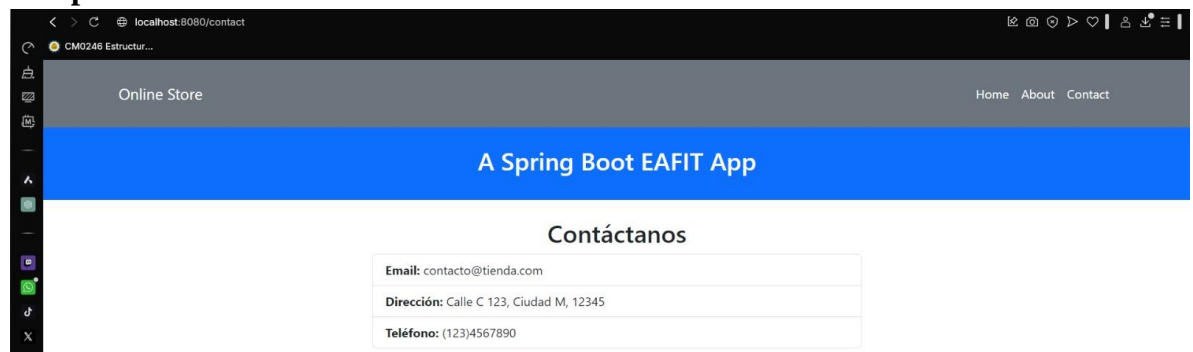
La ruta /about tiene una estructura apropiada con título, subtítulo, descripción y detalles del autor.

Sería bueno asegurarse de que las rutas y los datos proporcionados sean flexibles, permitiendo actualizaciones o modificaciones fáciles.

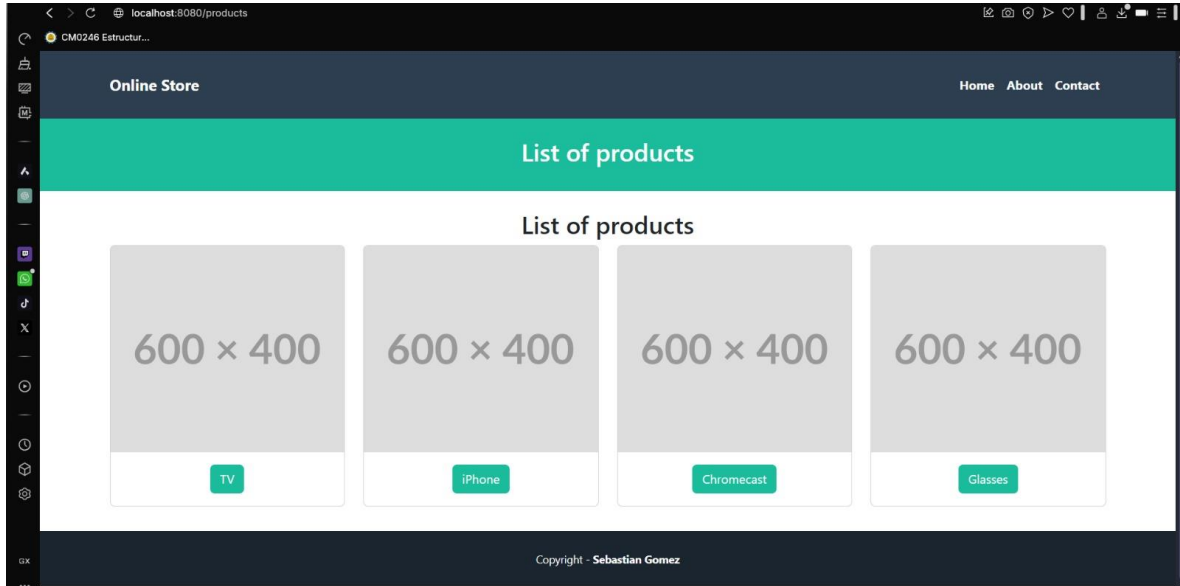
Se podrían agregar más rutas o información adicional para hacer el sitio web más interactivo y completo, como una ruta /services o /products.

Activity 2

- **Create a “/contact” section in which you display the application email, address, and phone number. Use fake information.**



C. Mostrar Productos



Activity 3

- Add the (“/products”) route as a new menu option (in the header navbar).

```
<a class="nav-link active" th:href="@{/products}">Products</a>
```

Activity 4

- Add prices for each product and display the information in the product.show view.

```
<h6 class="card-subtitle mb-2 text-muted">
    Price: $<span th:text="${product.price}"></span>
```

Activity 5

- Modify the show method. If the product number entered by the URL is not valid, redirect the user to the home page (“home.index”) route.

```
@GetMapping("/products/{id}")
public String show(@PathVariable String id, Model model) {
    try {
        int productId = Integer.parseInt(id) - 1;
        if (productId < 0 || productId >= products.size()) {
            return "redirect:/";
        }

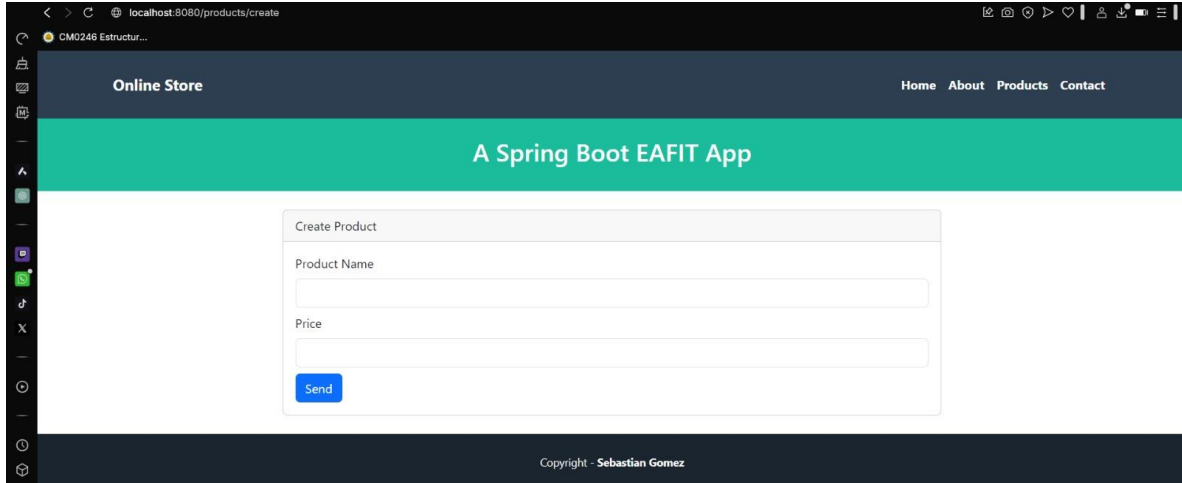
        Map<String, String> product = products.get(productId);
        model.addAttribute("title", product.get("name") + " - Online St
        model.addAttribute("subtitle", product.get("name") + " - Produc
        model.addAttribute("product", product);
        return "product/show";
    } catch (NumberFormatException e) {
        return "redirect:/";
    }
}
```

Activity 6

- Add a conditional in the “product.show” view. If the price of a product is greater than 100, display the product name in red

```
h5 class="card-title" th:style="${product.price != null && product.price > 100 ? 'color:red;':
```

D. Creación de Productos (Simulación)



The screenshot displays a web browser window with the address bar showing `localhost:8080/products/create`. The browser's title bar indicates the tab name is `CM0246 Estructur...`. The web application has a dark blue header with the text **Online Store** on the left and navigation links [Home](#), [About](#), [Products](#), and [Contact](#) on the right. Below the header is a teal banner with the text **A Spring Boot EAFIT App**. The main content area features a 'Create Product' form with a light gray border. The form contains two text input fields: 'Product Name' and 'Price'. A blue 'Send' button is positioned at the bottom left of the form. The footer of the application is a dark blue bar with the text `Copyright - Sebastian Gomez`.

localhost:8080/products/create

CM0246 Estructur...

Online Store

Home About Products Contact

A Spring Boot EAFIT App

Create Product

Product Name

Price

Send

Copyright - Sebastian Gomez

Activity 7

- Try to understand the previous code. Add a new product but leave the name empty (and click send). Then, leave the price empty. Then, enter the two fields.

The screenshot shows a web application running on localhost:8080/products/save. The page has a dark blue header with 'Online Store' and navigation links: Home, About, Products, Contact. Below the header is a green banner with the text 'A Spring Boot EAFIT App'. The main content area features a 'Create Product' form. The form has a red error message box at the top stating: 'The product name is required' and 'The price is required'. Below the error box are two input fields: 'Product Name' and 'Price'. A blue 'Send' button is at the bottom of the form. Below the form is a dark blue footer with the text 'Copyright - Sebastian Gomez'. Below the footer, there is a list of products displayed as cards. Each card has a placeholder image with the text '600 x 400' and a green button at the bottom with the product name: TV, iPhone, Chromecast, Glasses, and Samsung Phone.

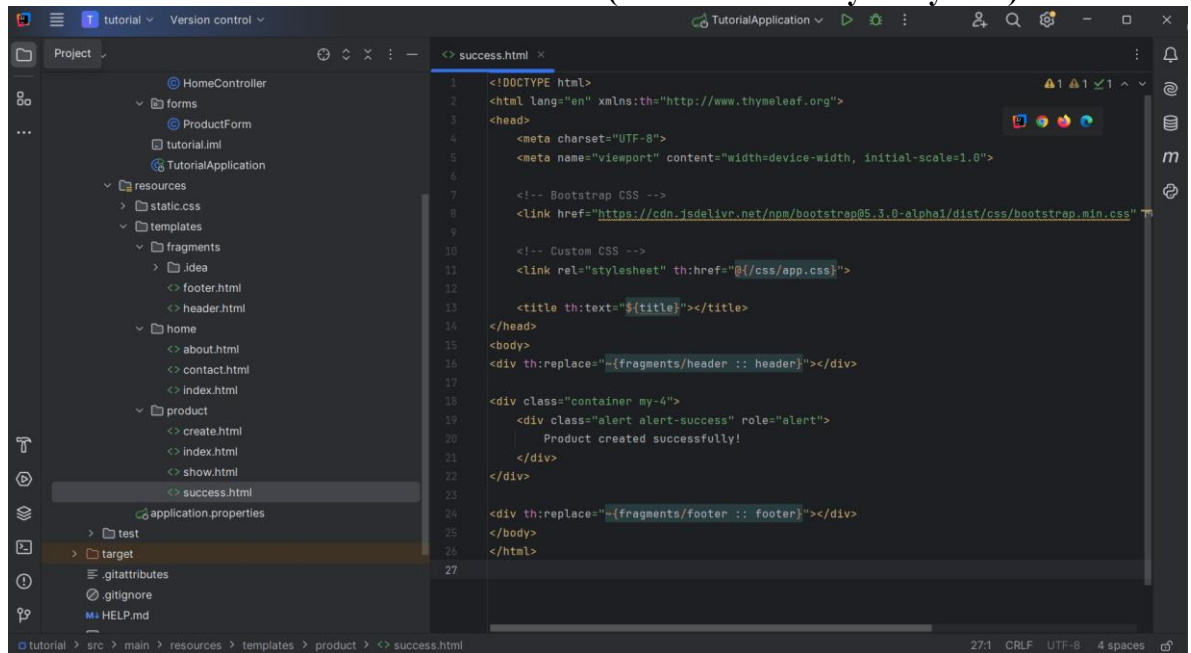
Activity 8

- Modify the previous code to only allow numbers greater than zero for the prices.

```
@NotNull(message = "The price is required") 2 usages
@Positive(message = "The price must be greater than zero")
private Double price;
```


Activity 9

- If the info entered by the form is valid. Then display a message saying, **“Product created”**. Note: create a new view (which uses the layout system).



Activity 10

- Add a new menu option in the header (app layout), that links to the **“/products/create”** page.

