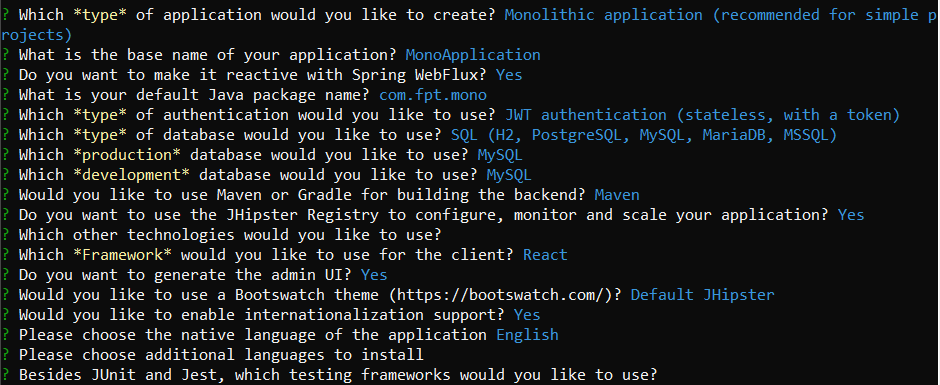
**Monolithic structure**

**Link GitHub**: [git@github.com:hafbn/Monolithique.git](mailto:git@github.com:hafbn/Monolithique.git)

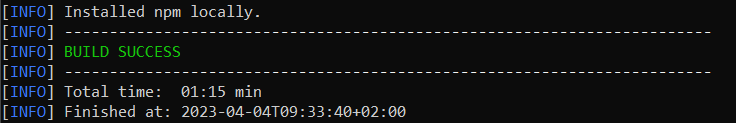
Generate a monolithic application using JHipster:

Java version: jdk-11.0.16.1 (error when using jdk-19)

* Install JHipster
* Run command: ***jhipster*** to generate an application with these options:



Build success:



The application is up on ***localhost:9000***, using command***: npm start*** for real-time loading

**Structure:**

* **Backend**: based on Spring Boot. The code is located in ***src/main/java***

The main application class is MonoApplicationApp.java which contains the main method to start the application.

The application uses Spring Security for authentication and authorization, and JPA (Java Persistence API) for database access.

* **Frontend**: interfaces built using ReactJS

The frontend code is located in ***src/main/webapp***

The ReactJS application uses Redux for state management and Axios for HTTP communication with the backend.

* **Database**

The database is based on MySQL. Database config is located in src/main/resources/config.

The application uses JPA for database access.

**Frontend modules**

All the code for rendering views, handling users’ interactions, managing states and communicating with the backend.

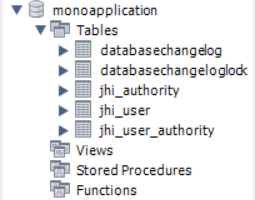
In the application, we have 2 main entities: **Customers** and **Products**.

The interface of the application allows us to view a table of customers/ create new customers/ edit or delete existing customers and to view also a list of products in another page. In the product page, we can select products and place orders, and then see a recap of the order with the total of payment.

**Database**: MySQL (SQL Workbench)

First jdbc url needs to be changed in ***application-dev.yml***, change to port mySQL 3306 and add name of schema: ***jdbc:mysql://127.0.0.1:3306/monoapplication?user=root***

Run ***mvnw*** to build app and generate automatically database in MySQL:

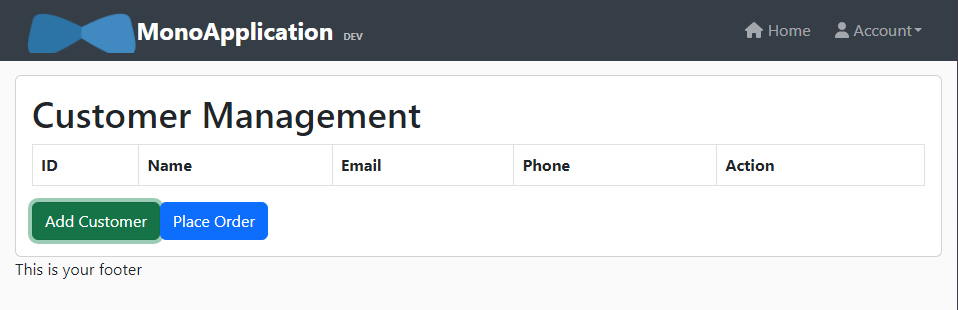


This allows to store login information. Now we can sign in as admin or user.

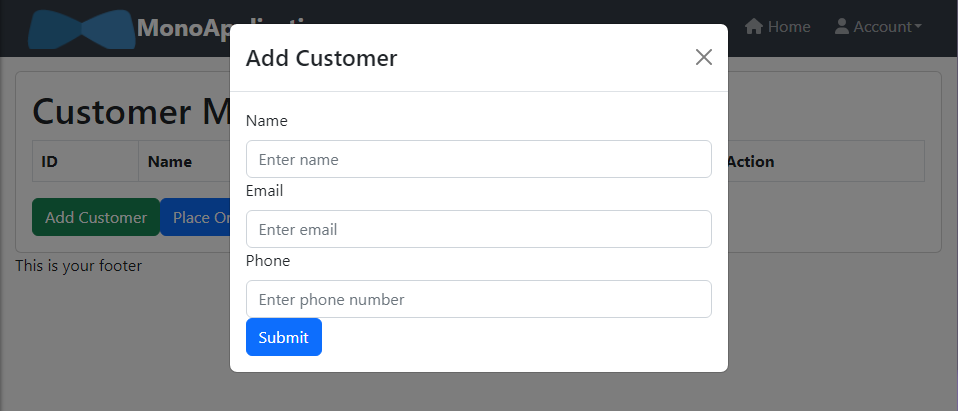
App is up on ***localhost:8080***.

**Demo**

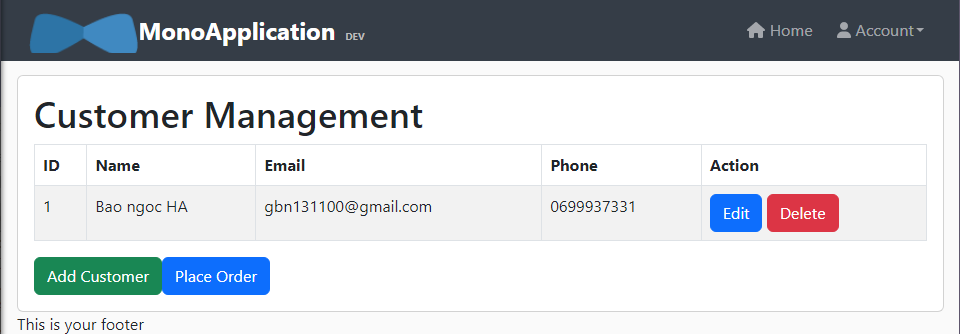
Customer Management: ***http://localhost:9000/customer***



The Add Customer buttons shows an Add customer modal, which allows users to fill in name, email and phone number of a customer. Name and email are unique.



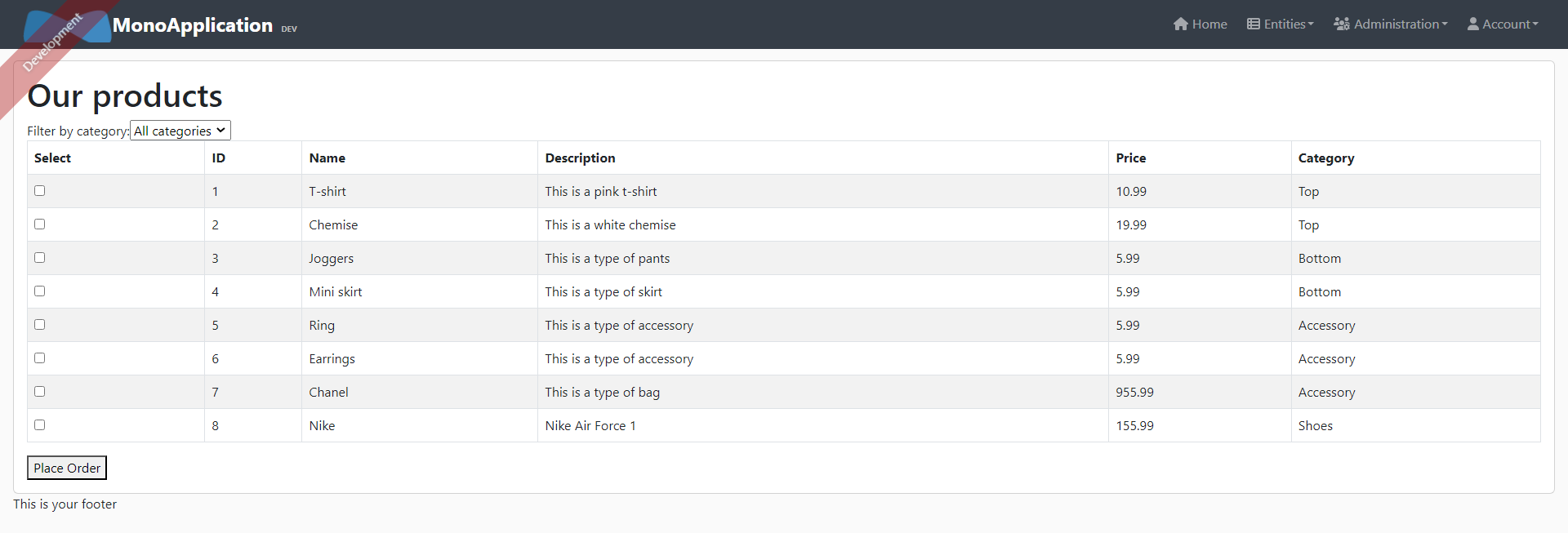
When click on Submit button, a new customer is added to the table. Here, we can edit or delete the information.



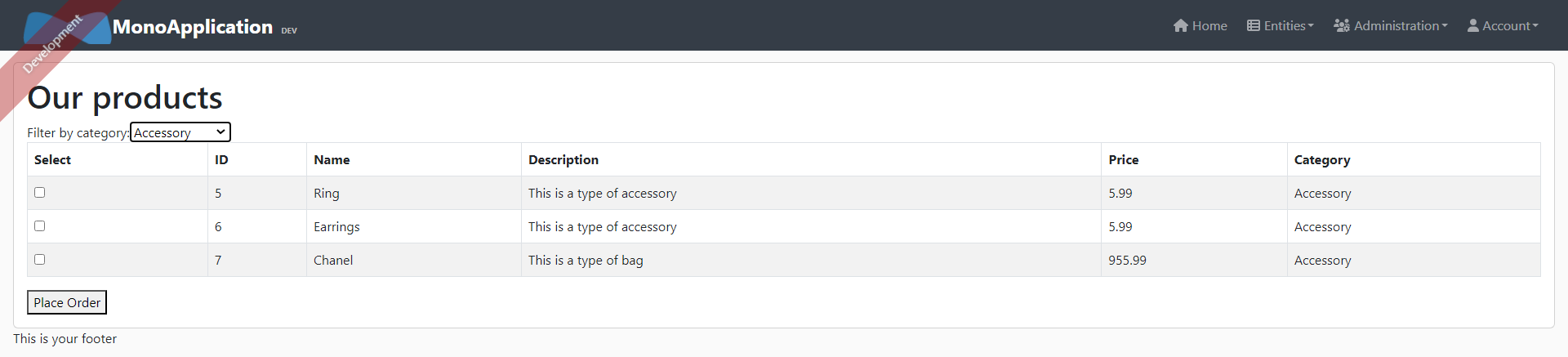
***Place Order*** button leads to product page, where users can choose products and place order.

***Notes: Data gestion (save, update, delete in database) coming soon. Now when we quit the page data will be disappeared.***

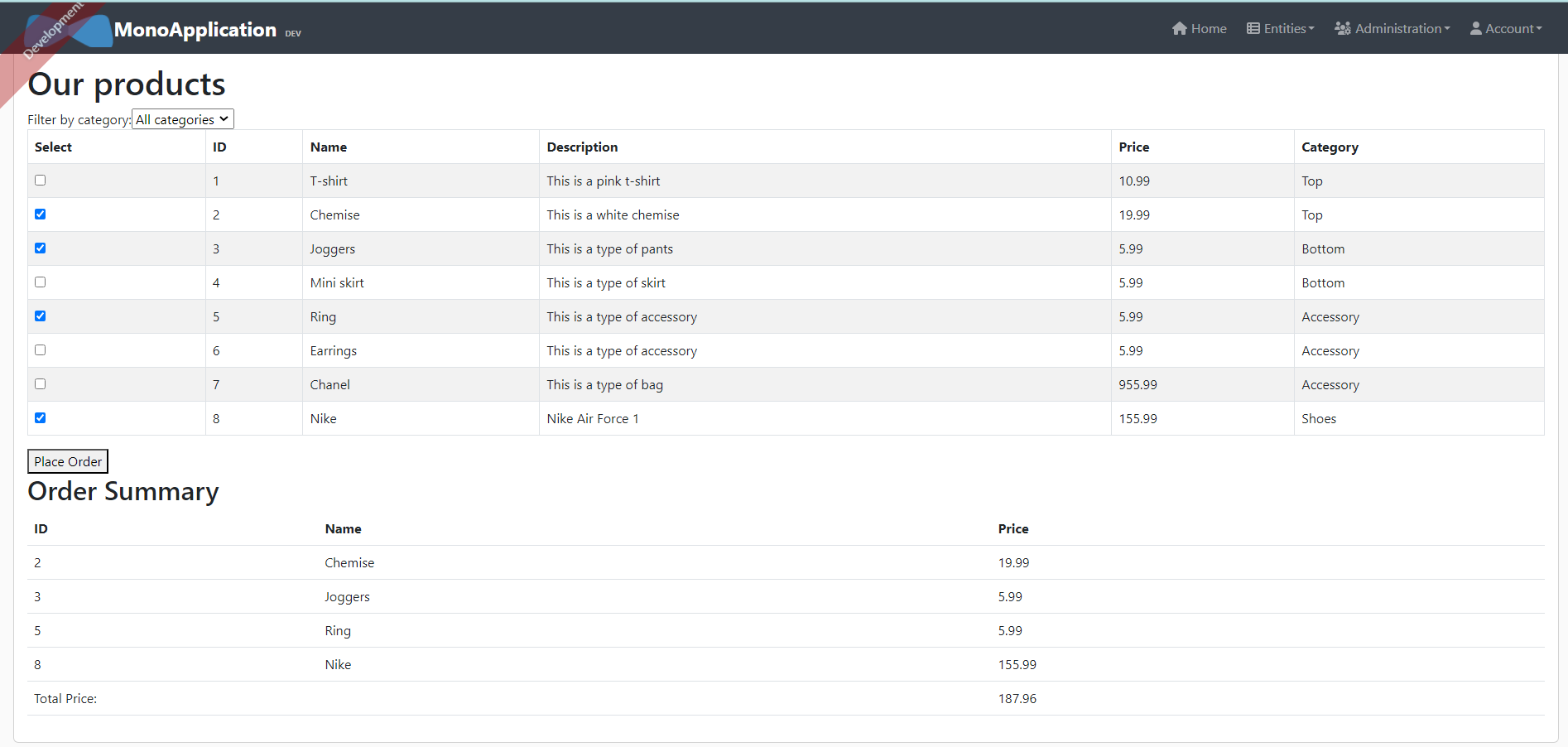
Product: [***http://localhost:9000/product***](http://localhost:9000/product)



Here, users can see a list of products, as well as filter the list according to categories:



The checkbox allows to choose a product. After selecting all products wanted, the button Place Order will return a recap of the order, with a total of payment:



***Notes: Design has not been done and list of products is created manually (needs to improve by implementing back-end, coming soon)***