

E-COMMERCE CODE SAMPLE

Author: Diana Sormani

Introduction

This document provides an overview of an e-commerce system created as a code sample project. It's code is available at my GitHub, both [backend](#) and [frontend](#).

This project implements a shopping cart. It provides features for:

- ❑ An administrator to create, update and delete products to be sold and upload it's information.
- ❑ A (buyer) user to navigates through those products.
- ❑ A (buyer) user to add and remove products to the shopping cart.
- ❑ A (buyer) user to analyse his/hers shopping cart.
- ❑ A (buyer) user to login and logout using his/her google account or create a new one.

Design and Archtecture overview

The system was implemented with a frontend and a backend over a microservices architecture.

The backend provides the APIs required from the frontend to implement the e-commerce features. It also implements the data access layer to manipulate the information.

The frontend implements the forms, components, pages and styles necessary for the user interaction.

The system also provides **auth0** authorization, to allow the users to properly login and out of the system.

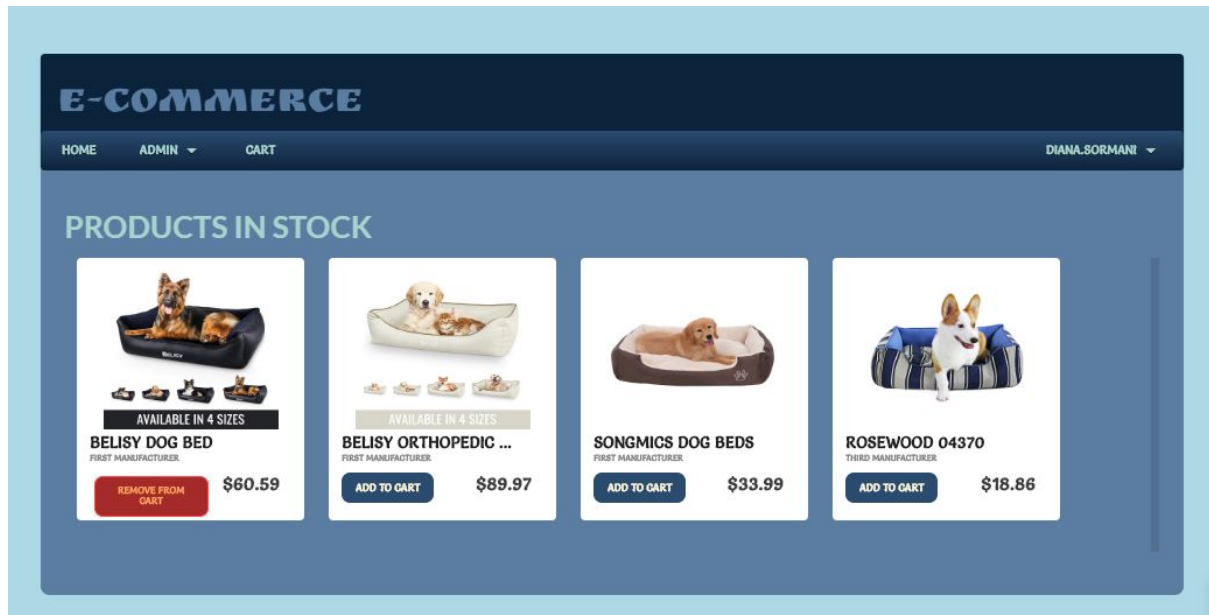
Frontend

The frontend was implemented with the following technologies and programming languages:

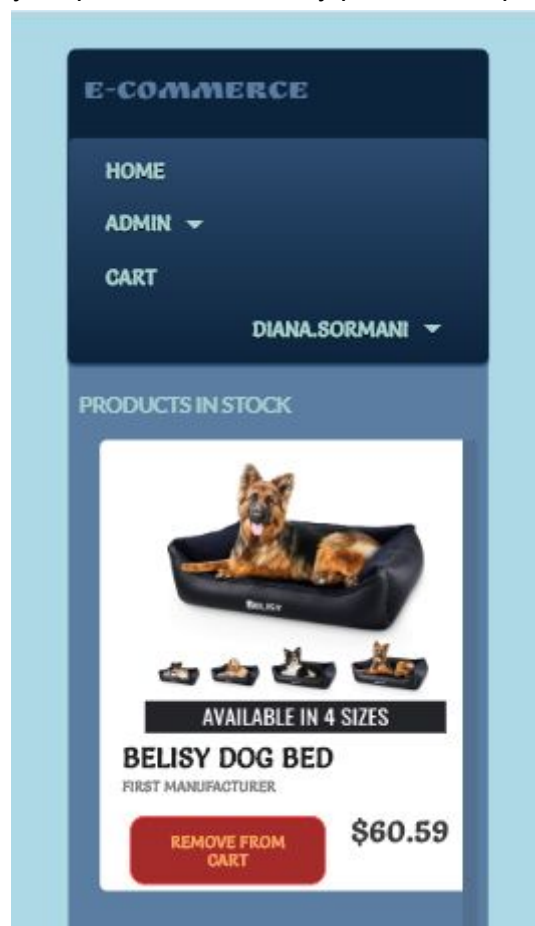
- ➔ **Vue.js**
- ➔ **Vuex**
- ➔ **CSS / HTML**
- ➔ **Auth0**
- ➔ **Git**
- ➔ **npm**

The frontend system requires user authentication, is responsive and has field to field validation. Here is a depiction of the different pages offered:

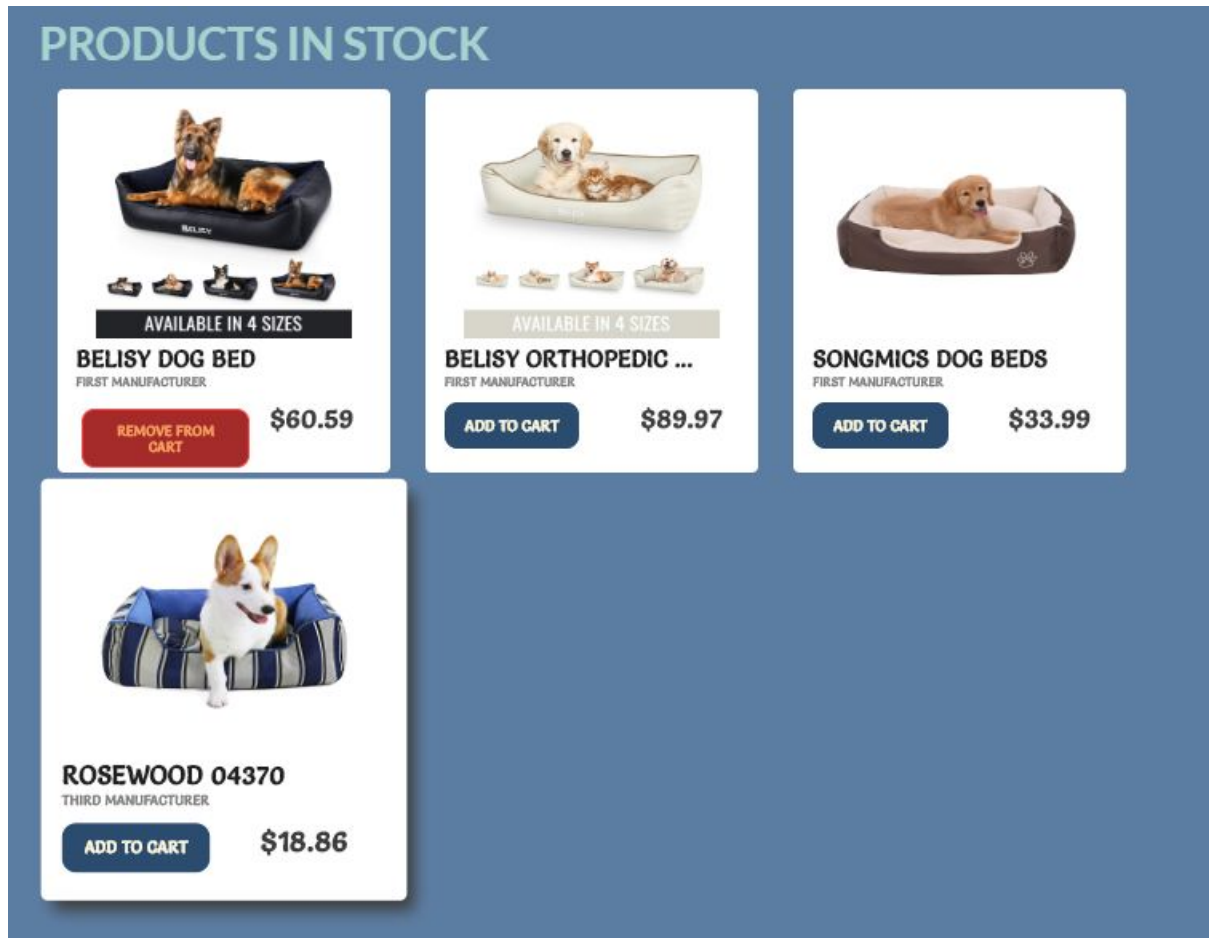
- ❑ The home page displays a list of available products. The products



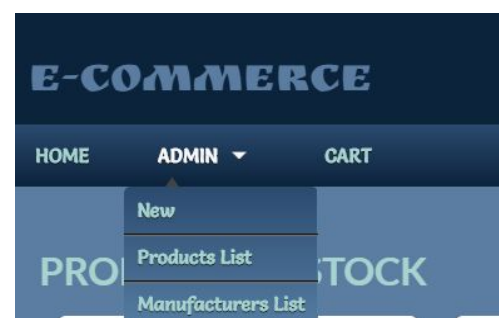
The page is completely responsive: from every product component to its menus.



- ❑ This page also presents a little effect (CSS transition) to highlight the selected product:



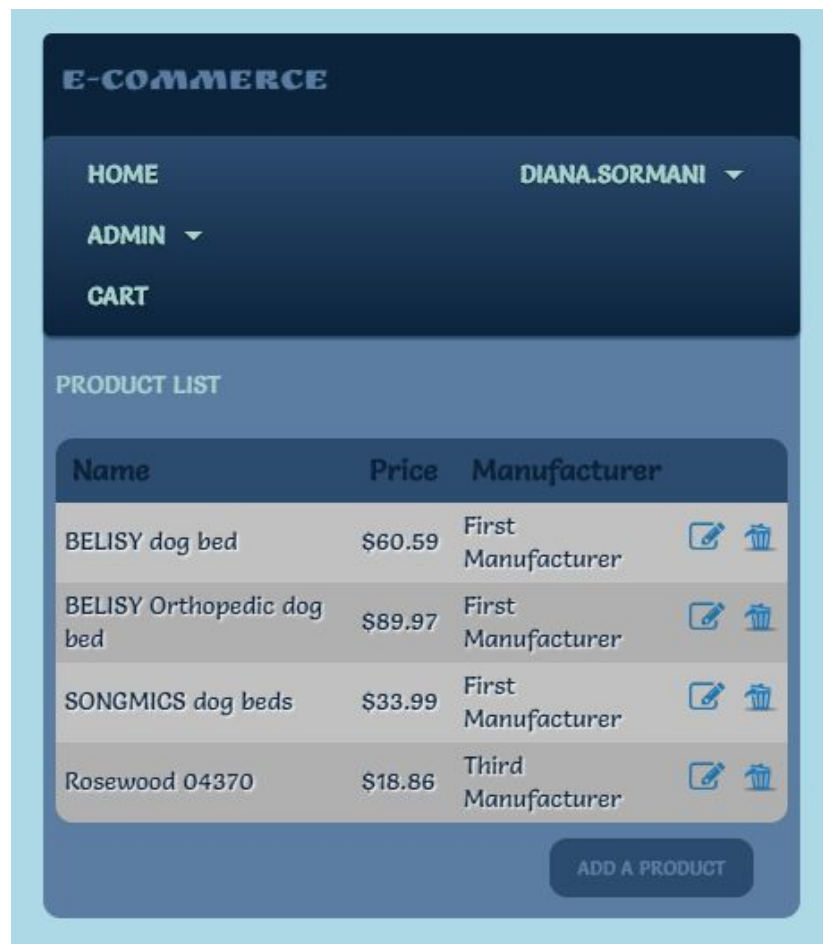
- ❑ The system provides a menu for the administrator. On this menu the administrator can access the following options:
 - New: allows the administrator to create a new product.
 - Product List: allows the administrator to manage his product list (create, update and delete products).
 - List Manufacturers: allows the administrator to manage the manufacturers' list (create, update and delete manufacturers)



❑ The Products List page is as follows: a pretty standard responsive grid.



The same page
in a smaller
device:



- ❑ The system also provides a form for product's CRUD functions:

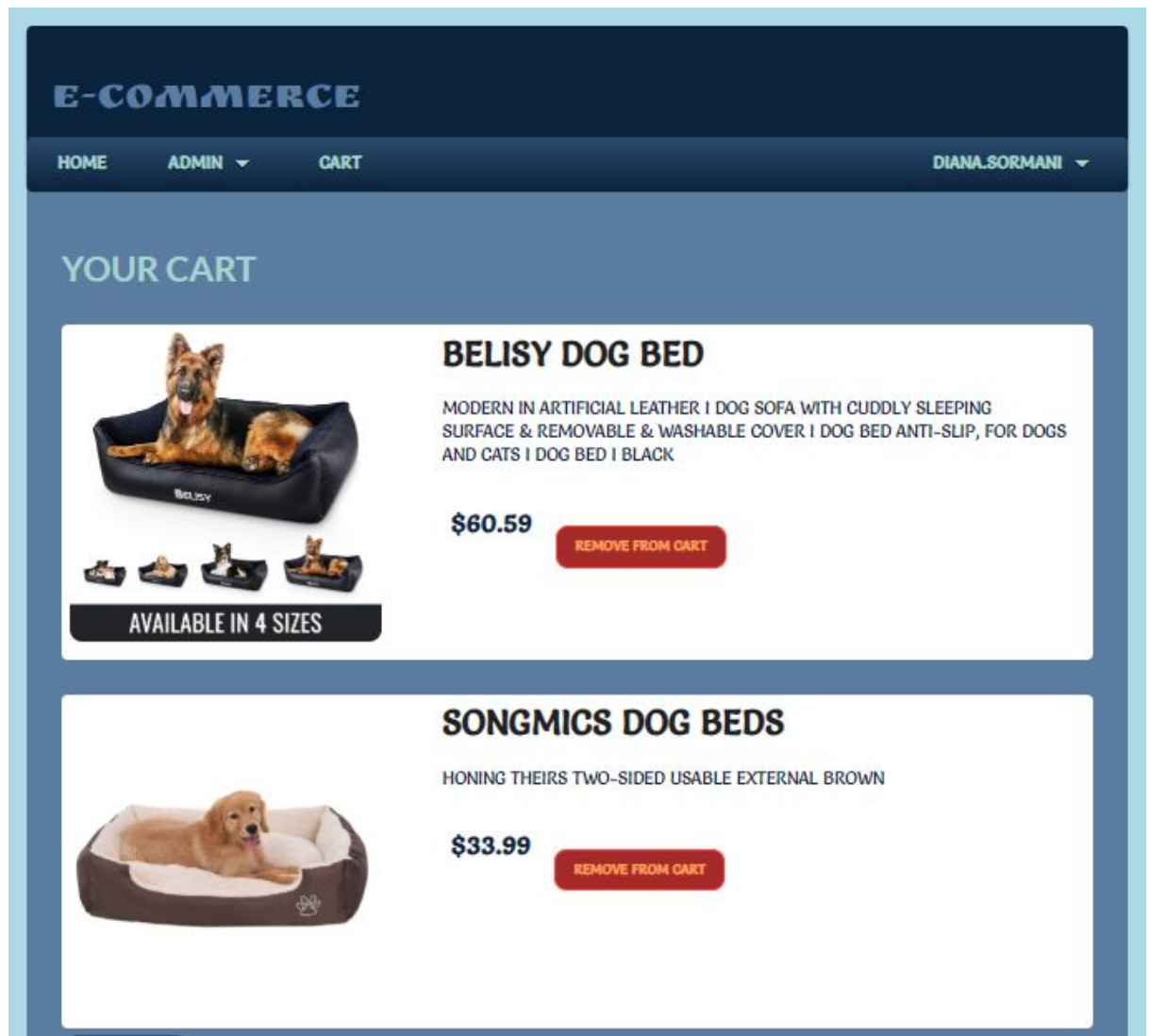
The screenshot shows a web interface for updating a product. At the top, there is a dark blue header with the text 'E-COMMERCE' in white. Below the header is a navigation bar with links for 'HOME', 'ADMIN' (with a dropdown arrow), and 'CART'. On the right side of the navigation bar, the user's name 'DIANA.SORMANI' is displayed with a dropdown arrow. The main content area is titled 'UPDATE A PRODUCT'. It contains several input fields: 'Name' with the value 'BELISY dog bed', 'Price' with the value '60.59', and 'Manufacturer' with a dropdown menu showing 'First Manufacturer'. There is also an 'Image' field with the value 'Belisy_1.jpg' and an 'Upload a file' button. A 'Description' field contains the text: 'Modern in artificial leather I dog sofa with cuddly sleeping surface & removable & washable cover I dog bed anti-slip, for dogs and cats I Dog Bed I black'. At the bottom right of the form are two buttons: 'CLOSE' and 'UPDATE PRODUCT'.

- ❑ Analogously, the system provides a grid for the manufacturer's information and a form to handle its CRUD functions.

- ❑ From the User's section of the menu, the system allows to logout. Afterwards, the system will require the user to login. The login is implemented with an auth0 component.

The screenshot shows a login page for an e-commerce system. At the top, there is a logo consisting of a red star inside a hexagon, followed by the text 'ecommerce'. Below the logo are two links: 'Log In' and 'Sign Up'. The main content area features a 'LOG IN WITH GOOGLE' button with a Google 'G' logo. Below this button is the text 'or'. There are two input fields: one for the email address 'diana.sormani@gmail.com' and one for the password, which is masked with dots. Below the password field is a link that says 'Don't remember your password?'. At the bottom of the page is a large orange button with the text 'LOG IN >'.

- ❑ Finally, the system offer a page for the user to check the selected items on his/her shopping cart:



Backend

The backend was implemented with the following technologies and programming languages:

- **Node.js**
- **Express**
- **MongoDB** (and mongoose)

It provides a layer for handling the APIs that provides the endpoints for the frontend. The api layer defines three controllers for providing the required services:

- **cart.js**
- **manufacturer.js**

→ product.js

It also provides a data access layer for managing the interaction with a NoSQL database (MongoDB in this case). That layer defines the models for three data collections and also provides the proper function for accessing them:

- Manufacturers
- Products
- Carts

Installation and Execution

As mentioned above, the system's code can be obtained from my GitHub:

- [Backend](#)
- [Frontend](#)

It is necessary to clone both applications locally and run:

- **npm install**

To run the backend, once cloned locally,:

- cd to the application's folder
- run: **npm start**

To run the frontend, once cloned locally,:

- cd to the application's folder
- run: **npm run dev**