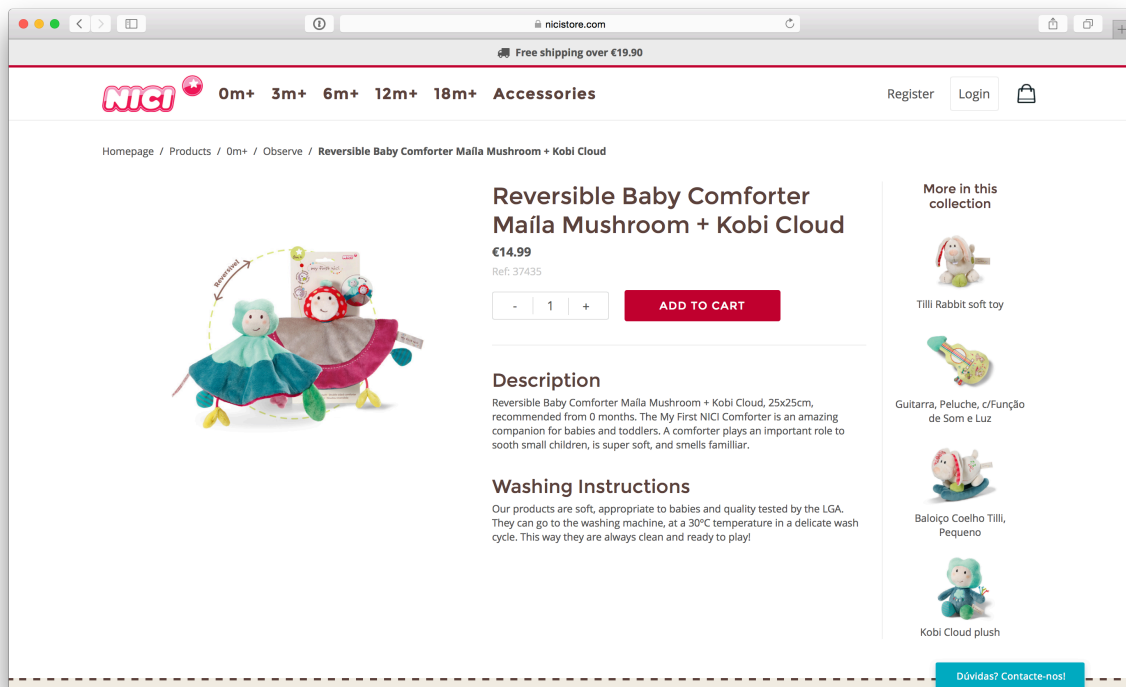

Welcome

This project is the NICI Store (discontinued e-commerce store) incarnation of an Isomorphic (first render on the server, then Single-page App on the client) React Application for E-Commerce Storefronts.



Currently, there is no vanilla version of it.

Since this is a storefront application, it requires an API Backend to operate and have some use. Even though it was written with Atlas in mind, if you wish, you can run it against any other API you like just as long as you either: - Ensure that API has the same endpoints and request/response as Atlas - Re-write the storefront API Plugin that is used for wrapping Atlas and making requests

Isomorphic?

The Single-Page Apps (SPA) paradigm has suffered from some bad rep (in certain applications like... e-commerce) mainly because of:

- First page load performance
- SEO un-friendliness

What if we can have the best of both worlds? Well, Isomorphic Apps give us precisely that. With a single code base and without relying on tools like headless browsers on the server we can have the first render be done on the server and, as soon as the page is rendered on the client, it starts working as a SPA with all it's benefits.

Why?

Why build plugins when you can build your own e-commerce platform?

Currently, all major open-source e-commerce offerings have several shortcomings due to the fact that they try to be easily installable and point-and-click-manageable and, at the same time, have all the features ranging from catalog management to order fulfillment processes (with everything in between).

Instead of a huge bloated monolith, Yoonic proposes a set of modular applications. The Storefront is solely responsible for: - Catalog - Cart - Checkout

By focusing on a specific set of features, isolating them from the whole picture and not having a complex base needed to support plugins and whatnot, you can easily start hacking away from this base and branch out your own custom storefront/platform.

Then, adding that spanking new feature, use-case or special page template will be a breeze!

Batteries Included

- Mobile-first responsiveness (try resizing the browser window)
- i18n translation
- Crisp.im for live chat
- Mailchimp for newsletters
- Mailgun for transactional emails
- Switch Payments for payments

Requirements

- Node.js + NPM (v4.x LTS)
- Atlas E-Commerce Backend API (optional)

Installation

In order to setup, run and start hacking the app locally you just have to:

-
1. Clone this repository
 2. `npm install`
 3. Open the configuration file for the development environment `config/client/development.js` and replace the value of `api.atlas.baseUrl` by `http://nicistore.com/api/v1`
 4. `npm run dev`
 5. Open a browser and navigate to `http://localhost:3000`

At this point, you should be seeing your local deploy of nicistore/storefront using the data provided by our live API.

Local Atlas Install

TL;DR

Update your `api.atlas.baseUrl` setting to `http://localhost:8000/v1`

By default, when running the Atlas development environment, the API base URL is available in `http://localhost:8000/v1` and this is the value you should set `api.atlas.baseUrl` to. However, if you change the port (e.g. “9000”) or add a route prefix (e.g. “/api”) you should put the respective value as the base URL (e.g. “http://localhost:9000/api/v1”).

Admin Backoffice

In order to perform administrative tasks like adding and editing products or checking out created orders, there’s a backoffice in `http://localhost:3000/en/adm`. In order to access it you need an Administrator account and, if you’re running your own Atlas, you can check its README in how-to create such account.