MVP: technical and functional specification

BACKGROUND

Orinoco offers specific, themed apps which each sell one group of products. This document outlines the requirements for an MVP, a single app with one of the following themes: handmade teddy bears, vintage cameras, or oak furniture.

REQUIREMENTS

ARCHITECTURE

The front end will require 4 pages:

- **A list view page**, showing all items available for sale.
- **A single product page** (using URL query parameters), which will dynamically show the item selected by the user, display a description and price in dollars, and allow users to personalize the product and add it to their cart.
- **A cart page** (using the localStorage JavaScript functionality), showing a summary of products in the cart, the total price, and a form with which to submit an order.
- **An order confirmation page**, thanking the user for their order, showing the total price and the order ID returned by the server.

APIs

Although the MVP will be a single application, the back-end for three applications have already been set up. Choose from the following:

- Handmade teddy bear store - http://localhost:3000/api/teddies
- Vintage camera store - http://localhost:3000/api/cameras
- Oak furniture store - http://localhost:3000/api/furniture

Each API contains three endpoints:

Verb	Endpoint	Expected request body	Response
GET	/	-	Returns an array of all items
GET	/:_id	-	Returns item corresponding to given _id
POST	/order	JSON request containing a contact object and a products array	Returns contact object, products array and orderId (string)

DATA VALIDATION

For POST routes, the contact object sent to the backend must contain firstName, lastName, address, city and email fields (all required). The products array sent to the backend must be an array of product _id strings. The type and presence of these fields must be validated before being sent to the server.

For the MVP, product personalization will not be functional: the single item page will have a dropdown menu allowing the user to choose a personalization option, but this will not be sent to the server or reflected in the server response.

DATA TYPES

All products have the following attributes:

Field	**Type**
_id	String
name	String
price	Number
description	String
imageUrl	String

Each product type then has an array containing Strings corresponding to personalization options:

Product type	**Personalization array**
Cameras	lenses
Teddy bears	colors
Oak furniture	varnish

TEST PLAN

The tests should cover at least 80% of the front-end code base. The test plan should indicate which features to test and which to not test.

COMPATIBILITY

The Orinoco core team has a DevOps who will generate the final build for this website, so no build tools such as Gulp, Grunt or Webpack may be used. Only the original source code is to be provided in the Git repo so it can be code reviewed, linted, tested, and built by the DevOps for deployment.

REPOSITORY TO CLONE

The existing code is located in this repository. After cloning the repository, run npm install from within the project folder, and then run the server with node server (requires Node installed on your machine and run locally).