Orinoco Test Plan

# Introduction

Orinoco is an online shop for vintage cameras. The first page displays a list with all available cameras. From there users can click on a camera to see the details. On the detail page you can add an item to the shopping cart. From the shopping cart items can be removed one by one, or all together. A from needs to be filled out to complete the order. When that is correctly done, a confirmation page is displayed with an order number and the total amount.

## Index

**Test case 1:**

On opening the index page all available items are correctly displayed.

Inputs:

* On loading, the available products are called from the back-end.
* The available products are put in the display function

Expected results:

* An object for a camera that contains:
  + lenses: array
  + \_id: string
  + name: string
  + price: number
  + description: string
  + imageUrl: string
* Displayed product cards

**Test case 2:**

The product Id is displayed in the URL of the product page.

Inputs:

* User navigates from the main page by clicking on a product, and a product id is passed as a parameter.

Expected results:

* The product parameter is present in the URL.

## Product

**Test case 3:**

On clicking on the detail button of a product, the products page opens, and the right product is displayed.

Inputs:

* The product Id from the URL

Expected results:

* Product card with the right product

**Test case 4:**

When the add to cart button is clicked, the product id is pushed to the array in local storage.

Inputs:

* The current local storage is retrieved
* The current product is added

Expected results:

* The product id is added to the array in local storage

## Cart

**Test case 5:**

The products in the local storage array are displayed in cart, with a total price.

Inputs:

* The products ids from local storage are filtered against the total collection

Expected results:

* An object with all selected cameras
* Product cards that display the selected cameras with the details
* A total price

**Test case 6:**

When clicking on remove item, one item gets removed, and the total price gets recalculated.

Inputs:

* The products ids from local storage are filtered against the total collection

Expected results:

* The items not equal to the current id are return and remain displayed

**Test case 7:**

On click the local storage is emptied and no more items are displayed.

Inputs:

* The array cameras in local storage

Expected results:

* Empty local storage

**Test case 8:**

On order, a contact object is created and send to the server together with the array of product ids.

Inputs:

* The form’s input fields

Expected results:

* An object with the client’s details:
  + First name
  + Last name
  + Address
  + City
  + Email

**Test case 9:**

The server returns the contact object, the product array and an order ID.

Inputs:

* Contact object
* Product array

Expected results:

* The order id

# Features not to Test

The following features will not be tested because their result is visible on the pages:

* The header functionality 🡪 The header becomes transparent on scrolling
* The price and order id display on the confirmation page