LAB: Product Details

NEXT: LAB: ADMIN FEATURES

To create the details component, we need to add another route to show product details. It's a dynamic route which accepts an ID as parameter and uses this ID to fetch a single product from the server:

JS

```
// You can add this to your existing
// array of routes
{
  path: '/details/:id',
  name: 'Details',
  component: Details
}
```

Then we need to create the component for this route as well:

JS

```
// ./src/pages/Details
<template>
  <div>
    <product-details :product="product" :isAdding="true" ></product-details>
  </div>

</template>

<script>
  import ProductDetails from '../components/product/ProductDetails'
  export default {
    created () {
      if (!this.product.name) {
        this.$store.dispatch('productById', this.$route.params['id'])
      }
    },
    computed: {
      product () {
        return this.$store.getters.productById(this.$route.params['id'])
      }
    },
    components: {
      'product-details': ProductDetails
    }
  }
</script>
```

First, we check if the product already exists before dispatching an action to fetch the product on the server. This is done immediately the component is loaded in the `created` lifecycle.

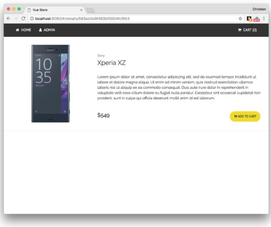
As usual, the `product` computed method holds the retrieved value which is received from `productById` getter. The `product` is not displayed immediately, but passed down to `ProductDetails` component. This abstract exists because we could use the same `ProductDetails` in the Cart component.

JS

```
<template>
  <div class="product-details">
    <div class="container">
      <div class="row">
        <div class="col-lg-4 col-md-4 col-sm-4 col-xs-12 product-details__image">
          <img
            class="img-responsive" :src="product.image" alt="">
        </div>
        <div class="col-lg-8 col-md-8 col-sm-4 col-xs-12 product-details__info">
          <div class="product-details__description">
            <small>{{product.manufacturer && product.manufacturer.name}}</small>
            <h3>{{product.name}}</h3>
            <p>
              {{product.description}}
            </p>
          </div>
          <div class="product-details__price-cart">
            <p>${{product.price}}</p>
            <product-button :product="product" ></product-button>
          </div>
        </div>
      </div>
    </div>
  </div>
</template>

<script>
  import ProductButton from './ProductButton'
  export default {
    props: ['product'],
    components: {
      'product-button': ProductButton
    }
  }
</script>
```

Just a UI component that receives a `product` from its parent component via `prope` and displays it. It also uses the `ProductButton` to toggle the item in the cart.



Cart Page

Speaking of cart, let's create a page/component for that:

JS

```
// ./src/pages/Cart
<template>
  <div v-if="cart.length > 0">
    <div class="title">
      <h1><i class="fa fa-superpowers"></i> Your Cart</h1>
    </div>
    <template v-for="product in cart">
      <product-details :product="product" :key="product.id" ></product-details>
    </template>
  </div>
  <div v-else class="title"><h1><i class="fa fa-superpowers"></i> Your Cart is Empty</h1></div>
</template>

<script>
  import ProductDetails from '../components/product/ProductDetails'
  export default {
    data () {
      return {
        cart: this.$store.state.cart
      }
    },
    components: {
      productDetails: ProductDetails
    }
  }
</script>
```

It's basically a variant of `ProductDetails` component; in fact, it re uses the `ProductDetails` component.