MODULE 2: BUILDING THE PRODUCT STORE

LESSON 4: Creating Product Pages

Learning Objective

By the end of this lesson, you will be able to create product cards and display them on your homepage.

4.1 Planning Our Product Display

In any e-commerce site, products are the star. We need to decide:

1. What information to show per product
    - Product name
    - Price
    - Image
    - Short description
    - "Add to Cart" button

2. How to organize this information
    - Create a reusable Product Card component
    - Display multiple products in a grid

4.2 Creating the Product Card Component

Step 1: Create `app/components/ProductCard.js`

javascript

```
function ProductCard({ product }) {
 return (
  <div style={{
   border: '1px solid #e2e8f0',
   borderRadius: '8px',
   padding: '1rem',
   backgroundColor: 'white',
   boxShadow: '0 2px 4px rgba(0,0,0,0.1)',
   transition: 'transform 0.2s',
   cursor: 'pointer'
  }}
  onMouseEnter={e => e.currentTarget.style.transform = 'scale(1.02)'}
  onMouseLeave={e => e.currentTarget.style.transform = 'scale(1)'}
  >
   {/* Product Image Placeholder */}
   <div style={{
    width: '100%',
    height: '200px',
    backgroundColor: '#f1f5f9',
    borderRadius: '4px',
    marginBottom: '1rem',
    display: 'flex',
    alignItems: 'center',
    justifyContent: 'center',
```

```
  color: '#64748b'
}}>
  🖼️ {product.name} Image
</div>

{/* Product Details */}
<h3 style={{ margin: '0 0 0.5rem 0', fontSize: '1.2rem' }}>
  {product.name}
</h3>

<p style={{ color: '#64748b', margin: '0 0 1rem 0', fontSize: '0.9rem' }}>
  {product.description}
</p>

<div style={{
  display: 'flex',
  justifyContent: 'space-between',
  alignItems: 'center'
}}>
  <span style={{
    fontSize: '1.3rem',
    fontWeight: 'bold',
    color: '#1e40af'
  }}>
    Rs. {product.price.toLocaleString()}
  </span>

  <button style={{
    backgroundColor: '#1e40af',
    color: 'white',
    border: 'none',
    padding: '0.5rem 1rem',
    borderRadius: '4px',
    cursor: 'pointer',
    fontSize: '0.9rem'
  }}>
    Add to Cart
```

```
      </button>
    </div>
  </div>
 );
}

export default ProductCard;
```

New Concept: Props

Notice `{ product }` in the function parameters. This is called a prop (short for property). It allows us to pass data into components.

Think of props like arguments you pass to a function:

```javascript
// Function with argument
function greet(name) {
 return `Hello ${name}`;
}

// Component with prop
function ProductCard({ product }) {
 return <div>{product.name}</div>;
}
```

4.3 Creating Sample Product Data

Before we display products, we need product data. Let's create a file to store our products.

Step 1: Create `app/data/products.js`

javascript

```javascript
const products = [
  {
    id: 1,
    name: 'Ceylon Tea (Premium)',
    price: 850,
    description: 'High-grown orthodox tea from Nuwara Eliya. 100g pack.',
    category: 'Food & Beverages'
  },
  {
    id: 2,
    name: 'Handloom Sarong',
    price: 2450,
    description: 'Traditional Sri Lankan handloom sarong. 100% cotton.',
    category: 'Clothing'
  },
  {
    id: 3,
    name: 'Spice Collection',
    price: 1200,
    description: 'Set of 5 authentic Ceylon spices: cinnamon, cardamom, cloves, pepper,
nutmeg.',
    category: 'Food & Beverages'
  },
  {
    id: 4,
    name: 'Elephant Figurine',
    price: 1650,
    description: 'Hand-carved wooden elephant, traditional Sri Lankan craft.',
    category: 'Home Decor'
  },
  {
    id: 5,
    name: 'King Coconut',
    price: 150,
    description: 'Fresh Thambili (King Coconut), direct from local farms.',
    category: 'Food & Beverages'
```

```
  },
  {
    id: 6,
    name: 'Batik Sarong',
    price: 3950,
    description: 'Handmade batik sarong with traditional patterns.',
    category: 'Clothing'
  }
];

export default products;
```

4.4 Displaying Products on Homepage

Now let's update our homepage to show all products in a grid.

Step 1: Update `app/page.js`

```javascript
import Header from './components/Header';
import Footer from './components/Footer';
import ProductCard from './components/ProductCard';
import products from './data/products';

export default function Home() {
  return (
    <div>
      <Header />

      {/* Hero Section */}
      <section style={{
        backgroundColor: '#f8fafc',
        padding: '3rem 2rem',
```

```
        textAlign: 'center',
        borderBottom: '1px solid #e2e8f0'
      }}>
        <h1 style={{ fontSize: '2.5rem', marginBottom: '1rem' }}>
          Welcome to CeylonMart 🇱🇰
        </h1>
        <p style={{ fontSize: '1.2rem', color: '#475569', maxWidth: '600px', margin: '0 auto' }}>
          Discover authentic Sri Lankan products, delivered to your doorstep.
        </p>
      </section>

      {/* Products Grid */}
      <main style={{ padding: '2rem' }}>
        <h2 style={{ fontSize: '2rem', marginBottom: '2rem' }}>
          Our Products
        </h2>

        <div style={{
          display: 'grid',
          gridTemplateColumns: 'repeat(auto-fill, minmax(300px, 1fr))',
          gap: '2rem'
        }}>
          {products.map(product => (
            <ProductCard key={product.id} product={product} />
          ))}
        </div>
      </main>

      <Footer />
    </div>
  );
}
```

New Concept: map()

`products.map()` loops through each product and creates a ProductCard for it. The `key` prop helps React keep track of each item.

4.5 Understanding the Grid Layout

The grid we used is responsive:

```css
gridTemplateColumns: 'repeat(auto-fill, minmax(300px, 1fr))'
```

This means:

- Each column is at least `300px` wide
- If there's space, columns expand equally (`1fr` means "1 fraction of remaining space")
- `auto-fill` creates as many columns as will fit

On different screens:

- Mobile: 1 column (300px each)
- Tablet: 2-3 columns
- Desktop: 3-4 columns

4.6 Adding Categories Filter (Optional Enhancement)

Let's add simple category buttons to filter products.

Step 1: Update `app/page.js` to include state for filtering

```javascript
"use client";

import { useState } from 'react';
```

```jsx
import Header from './components/Header';
import Footer from './components/Footer';
import ProductCard from './components/ProductCard';
import products from './data/products';

export default function Home() {
 const [selectedCategory, setSelectedCategory] = useState('All');

 // Get unique categories
 const categories = ['All', ...new Set(products.map(p => p.category))];

 // Filter products based on selected category
 const filteredProducts = selectedCategory === 'All'
  ? products
  : products.filter(p => p.category === selectedCategory);

 return (
  <div>
   <Header />

   <section style={{
    backgroundColor: '#f8fafc',
    padding: '3rem 2rem',
    textAlign: 'center',
    borderBottom: '1px solid #e2e8f0'
   }}>
    <h1 style={{ fontSize: '2.5rem', marginBottom: '1rem' }}>
     Welcome to CeylonMart 🇱🇰
    </h1>
    <p style={{ fontSize: '1.2rem', color: '#475569', maxWidth: '600px', margin: '0 auto' }}>
     Discover authentic Sri Lankan products, delivered to your doorstep.
    </p>
   </section>

   <main style={{ padding: '2rem' }}>
    <div style={{
     display: 'flex',
```

```
 justifyContent: 'space-between',
 alignItems: 'center',
 marginBottom: '2rem',
 flexWrap: 'wrap'
}}>
 <h2 style={{ fontSize: '2rem', margin: '0' }}>
  Our Products
 </h2>

 {/* Category Filter Buttons */}
 <div style={{ display: 'flex', gap: '0.5rem' }}>
  {categories.map(category => (
   <button
    key={category}
    onClick={() => setSelectedCategory(category)}
    style={{
     padding: '0.5rem 1rem',
     border: 'none',
     borderRadius: '4px',
     backgroundColor: selectedCategory === category ? '#1e40af' : '#e2e8f0',
     color: selectedCategory === category ? 'white' : '#1e293b',
     cursor: 'pointer',
     fontWeight: selectedCategory === category ? 'bold' : 'normal'
    }}
   >
    {category}
   </button>
  ))}
 </div>
</div>

<div style={{
 display: 'grid',
 gridTemplateColumns: 'repeat(auto-fill, minmax(300px, 1fr))',
 gap: '2rem'
}}>
 {filteredProducts.map(product => (
```

```
          <ProductCard key={product.id} product={product} />
        ))}
      </div>
    </main>

    <Footer />
  </div>
 );
}
```

Important: Notice `"use client";` at the top. This tells Next.js this component needs to run in the browser (because it uses `useState`).

Lesson 4 Summary

- Components can receive data through props
- `map()` is used to render lists of items
- CSS Grid creates responsive layouts
- `useState` lets components have interactive state
- Category filtering improves user experience

Practice Exercises

1. Add a new product to the products array with your own Sri Lankan item
2. Style the Product Card differently (change colors, add border radius, etc.)
3. Add a search bar that filters products by name
4. Challenge: Add a "Sort by Price" button that sorts products low to high