# 

# React Hands-on Project

Learn by doing with this practical project, progressing at your own pace.

You won’t find any step-by-step instructions for this task - instead, apply your problem-solving skills and knowledge of React concepts to overcome challenges and create your own solutions. You can always ask your trainer for assistance.

### Summary

In this project, you’ll **build the front end for a simple shop website** using React.

This project is composed of the following challenges:

* **Challenge 1**: Build out a static UI
* **Challenge 2**: Add simple interactivity
* **Challenge 3**: Use external data
* **Challenge 4**: Manage application state
* Extension challenges

### You should attempt each challenge after you have covered the concepts associated with it.

### Challenge 1 Build out a static UI

#### Task summary

Build a static website using React based on [these designs](https://www.figma.com/file/G4BvhqPVp8IUnYaXENMdZa/React-Shop-Project---Challenge-1?node-id=0%3A1&t=34Umbhu3XKqIHrCU-1) - don’t worry about matching the visual design perfectly. Resources have been provided to help speed up your work.

#### Concepts

* HTML/CSS/JS fundamentals
* React components, props, rendering lists
* React Router

#### Resources to help you

* Homepage image ([link](https://user-images.githubusercontent.com/118732445/203775030-f6d39588-6b73-42eb-a396-d50098078501.jpg))
* Items data ([link](https://seed-theory-api.netlify.app/data.json)) \*

#### Creating a React app

* **Option 1**: Develop using a web-based IDE ([link](https://stackblitz.com/edit/react-eoo2cr?file=src%2FApp.js)). Click “Fork” and log in using GitHub to save your work. The react-router-dom dependency has been added to the environment.
* **Option 2**: Develop locally using create-react-app ([guide](https://beta.reactjs.org/learn/start-a-new-react-project#getting-started-with-a-minimal-toolchain)). You will need to initialise a git project and push regularly to GitHub to save your work. Run npm install react-router-dom to use React Router features.

#### Tips

* Instead of viewing the designs as a whole, break down the problemby identifying the different elements that make up each page.
* Make sure to create a reusable component for the items on the shop page.
* The “Add to cart” buttons should not do anything at this stage.

\* Hard-code this data into your app by placing it in a file called itemsData.json within your project. To render this data on the shop page, you should be familiar with using Array.map to render lists of React elements.

### 

### Challenge 2 Add simple interactivity

#### Concepts

* Handling events
* React state (useState)
* Conditional rendering

#### Task summary

On the shop page, each item component has an “Add to cart” button that currently does nothing. Change this so that clicking the buttons toggles the text back and forth between “Add to cart” and “In cart”. Click [this link](https://react-simple-shop.pages.dev/items) for a live demo.

At this stage, there’s no need to store the cart data - simply change the appearance of the buttons.

#### Tips

* To change the visual appearance of the buttons, you can change which CSS classes are applied to the button element.

### 

### 

### Challenge 3 Use external data

#### Concepts

* useEffect hook
* Using external data
* State (useState)

#### Task summary

Currently, the items data is hard-coded into your app in a JSON file. Instead, change the implementation so that the app fetches the data from an external source when the site is first loaded.

The JSON data is hosted [here](https://seed-theory-api.netlify.app/data.json).

Alternatively, if you are developing locally, you can use [json-server](https://www.npmjs.com/package/json-server) to host the data on a fake API on your development machine.

#### Tips

* **Do not forget the dependency** **array** in your useEffect.

#### 

### 

### Challenge 4 Manage application state

#### Concepts

* React Context

#### Task summary

Add a checkout page that displays all the items in the user’s cart, as well as the total cost. Include a checkout button that displays an alert to the user thanking them for their purchase.

View the [live demo](https://react-simple-shop.pages.dev/items) of the example site to see what this should look like. Add a few items to your cart, then view the checkout page.

#### Tips

* Use the Context examples and exercises you’ve worked through previously to help you.

**Extension challenges**

Try out these additional challenges to stretch yourself if you have extra time.

#### Extension 1

Apply a 10% discount at checkout if the user has more than £60 worth of items in their cart.

#### Extension 2

Allow the user to choose the quantity of each item when adding it to their cart.  
*If using <input type=”number”>, you’ll need to understand* [*how to react to input with state.*](https://beta.reactjs.org/learn/reacting-to-input-with-state)

#### Extension 3

Persist the items in the user’s cart via [localStorage](https://www.w3schools.com/html/html5_webstorage.asp).