

W5 - PRACTICE


JSX - Dynamic Data - Components

 At the end of this practice, you should be able to...


- ✓ Create a new **component** from HTML
- ✓ Translate HTML to **JSX**
- ✓ Understand the basic of **nested components**
- ✓ Draw a **diagram** component from some given code
- ✓ Understand how to display **data dynamically** using curly braces `{xx}` in JSX

 How to work?

- ✓ Download **the start code** from the Google classroom
- ✓ For each exercise you can either:
 - Run `npm install`
 - Or move an existing `node_modules` to the exercise folder (*fastest option!*)

 How to submit?

- ✓ **Create a repository on GitHub** with the name of this practice:
Ex: `C2-S1-PRACTICE`
- ✓ **Push your final code** on this GitHub repository (if you are lost, [follow this tutorial](#))
- ✓ Finally, submit on **Google classroom** your GitHub repository URL
Ex: `https://github.com/thebest/ C2-S1-PRACTICE.git`

 Are you lost?

You can read the following documentation to be ready for this practice:

https://www.w3schools.com/react/react_jsx.asp

https://www.w3schools.com/react/react_props.asp

<https://www.gatsbyjs.com/docs/how-to/images-and-media/importing-assets-into-files/>



EXERCISE 1

Your task is to create your first React **component**!

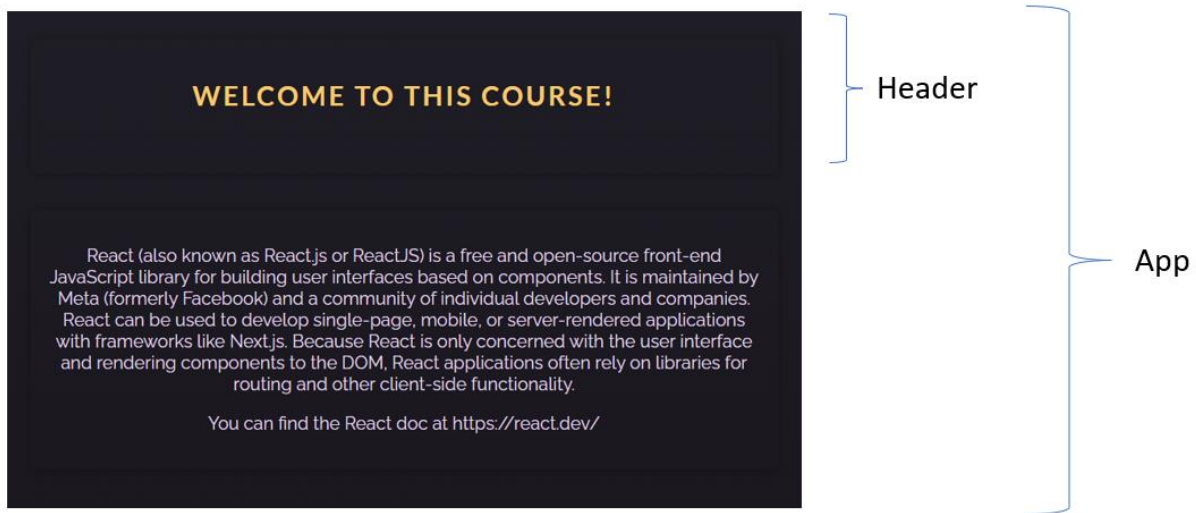
You have an App component, containing the header and the body.

- Create a component **Header** containing the header of the file.
- Change the code in the App component to use this new component

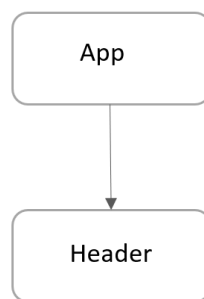
Notes:

- You can create the component directly in the App file.

The finished app could look like this:



The finished app diagram component:



EXERCISE 2

Well done!

Now your challenge is to **convert some vanilla HTML** into some React JS code!

Q1 – Research on internet and list down the **main differences** between **HTML** and **JSX** syntax

-
-
-

Q2 – The first part is to create an **empty React project** which display Hello

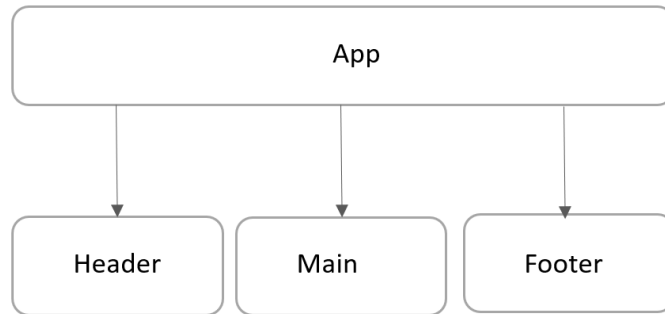
- **Create a new React project** using the following command:
`npm create vite exercise2 -- --template react`
- On the root folder, **remove** the following useless file:
`.eslintrc.cjs`
`README.md`
`.gitignore`
- On /src folder remove, **remove** the following useless file:
`/assets`
`App.css`
- Edit the `index.css` and **remove all styles**
- Edit the `App.jsx` and just write a simple code:

```
function App() {  
  return (  
    <>  
      <p>Hello</p>  
    </>  
  );  
}  
export default App;
```
- From the root folder, launch `npm install` and `npm run dev`
- You have now a very simple ReactJS code that displays Hello:

Hello

Q3 – On this second part you need to **adapt** the original HTML code to your new created project:

Your code should be composed of 4 components, as bellow:



- Create a folder /components
- In this folder create 3 additional JSX files:
 - o Header.jsx
 - o Main.jsx
 - o Footer.jsx
- Adapt the code from the original HTML code to those 4 compomers (App, header, Body and Footer)
 - o Do not forget to **export** your components to use them outside!
- Finally, you can copy the original CSS code to your new project

The finished app could look like this:



EXERCISE 3

Amazing!

Q1 - Now your challenge is to **draw a diagram component** from some existing React JS code.

1. Read the code
2. Identify components
3. Draw the diagram component (*using power point or another tool*)

ATOMIC CLOCK

The date now is:

12/13/2023, 12:12:55 PM

Did you know ?

The implementation of Greenwich Mean Time was the first step to determine the time zone of other countries in regard to GMT+0, while the concept of Coordinated Universal Time (UTC) was designed to provide a more accurate timekeeping system. Nevertheless, both of these time standards are widely used in the world for a similar purpose of time coordination. The differences in the terminology of GMT and UTC still create confusion in international cooperation. Even though UTC was introduced as a more accurate time standard, the occurrence of the leap seconds demonstrated the flaws for the universal time synchronisation.

Q2 – Let's play with dynamic data:

- In **Header**, change the title to: "The amazing atomic clock"
- In **Time** component, change the code to display only the **time** only (not **date + time**)

The date now is:
12:12:55 PM

EXERCISE 4

Amazooooome!

For this last exercise, your challenge is to provide the dynamic data for the 2 following fields:

- The value (15 dollars) converted in Dong
- The value (15 dollars) converted in Euro

Important

- You need to implement and call the functions already provided for you to convert dollar to other devices
- All inputs are disabled: we use them for display only, not to enter any value...

The screenshot shows a form titled "DEVICE CONVERSIONS" with a dark purple background. It contains three input fields. The first field is labeled "CURRENT VALUE IN DOLLARS" and contains the value "15". The second field is labeled "VALUE IN DONG" and contains the value "368400". The third field is labeled "VALUE IN EURO" and contains the value "13.8". Red arrows point from the text "Not editable !" to the three input fields. Another red arrow points from the text "Compute the values in those currencies" to the "VALUE IN DONG" and "VALUE IN EURO" fields.

| Field Label | Value |
|--------------------------|--------|
| CURRENT VALUE IN DOLLARS | 15 |
| VALUE IN DONG | 368400 |
| VALUE IN EURO | 13.8 |