Front End

Class 2

July 21, 2021

React Basics - Components

Declarative Approach – means we won't tell React that a certain html elements be created and inserted in a specific place.

Instead, we define the desired target state(s) and let react figure out the actual JS DOM instructions.

Creating a React app - https://github.com/facebook/create-react-app

Node.js allows u to run JS code outside of browser. Since React code runs in browser, we don't need node directly, but we need it to execute command for create-react-app. www.nodejs.org

npx is an npm package runner.

```
npx create-react-app my-app
cd my-app
npm start
```

React Basics – project files

All JS!
src folder has index.js is transformed and executed initially
We will write code in easy way, but it won’t run that way in browser.

Some transformations occur while running npm start
Some not regular JS syntax

```
import './index.css’;
```

Simply tells react to include the css

```
ReactDOM.render(
App />, document.getElementById('root')
);
```

React Basics – files

package.json react and react-dom are dependencies that form react library.

check out index.js

render method takes two arguments.

It has the div with id of root.

import App from ‘./App’ u can omit .js

App is a component. That is rendered inside the div.

It has a function that is exported.

Public folder has the index.html file that is used by spa. spa means only single file is exported to the browser.

React Basics – JSX

JavaScript XML

Html code in JS

Transformed code in bundle.js etc is react packaged code.

JSX code (easier to write) is transformed to code working in browser.

React Basics – How React works

Our own custom HTML instead of working with document and modifying it.

Imperitive way (cumbersome) is

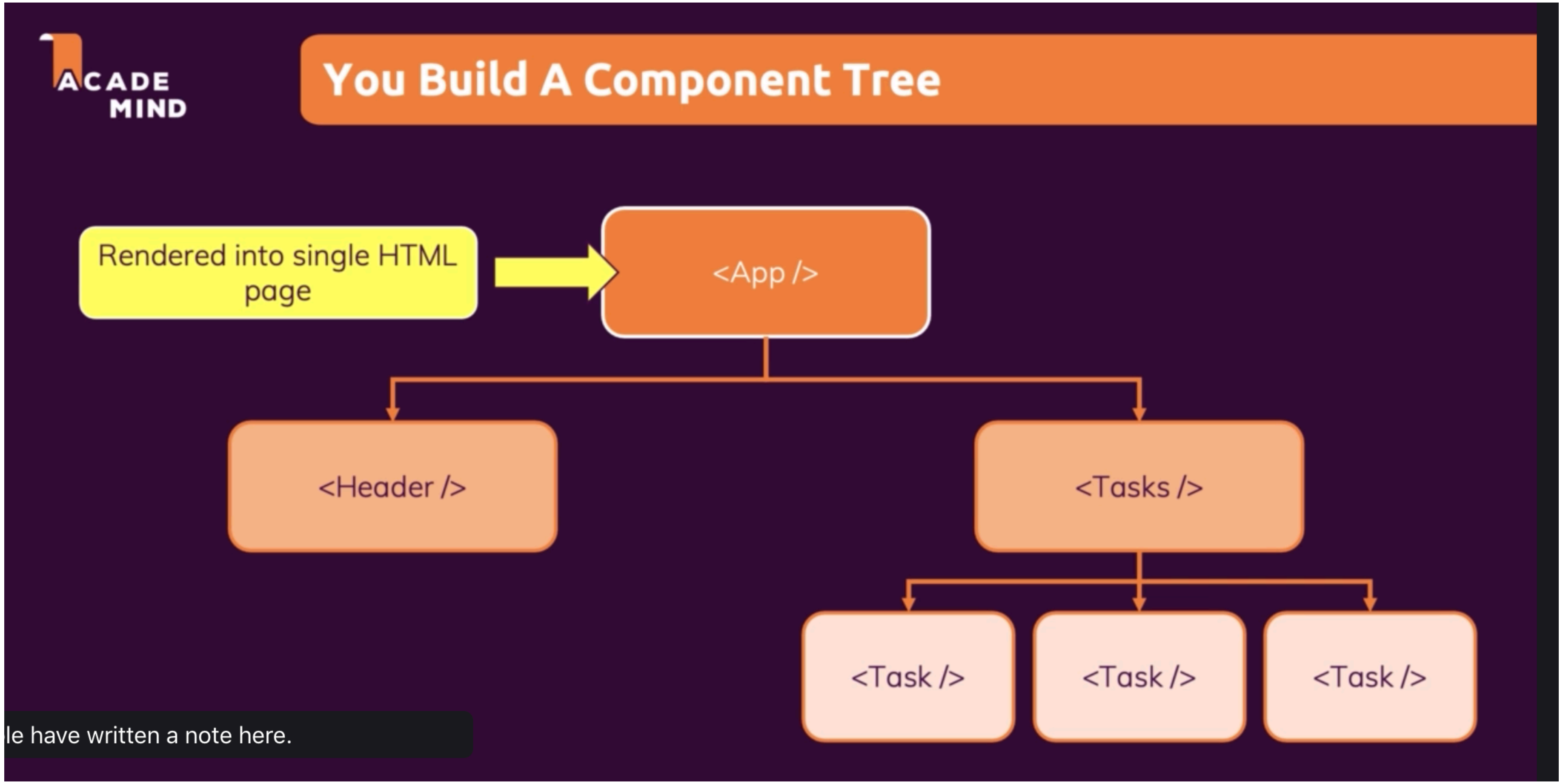
```
Const para = document.getElement(‘p’);
Para.textContent = ‘text’
Document.getElementById(‘root’).append(para);
```

Versus declarative way (end state). React generates the instructions.

```
<div className="App">
<h2>Let's get started</h2>
<p>This is also visible</p>
</div>
```

Can see transformed code in developer tools under source.

React Basics – Creating a custom component.



le have written a note here.

React Basics – Creating a custom component.

Build a component tree, meaning how things are organized

Component consists of a function returning some jsx code.

Export the function as export default function Name

Could build in App.js Best practice to put new components in new File.

Create components folder under src to add components there.

ExpenseItem.js and write a function and export it.

Import the custom component in the place where it needs to be inserted, like App.js

<ExpenseItem></ExpenseItem> Should start with uppercase.

Build in components are lowercase.

And only one root element. Can wrap into div tag. Wrapping in brackets also improves readability.

Also import css and define className instead of class (class is reserved name in JS). Not regular html but special JSX syntax.

Single {} for dynamic data.

React Basics – Creating a custom component.

```
function ExpenseItem() {
        return <h2>Expense item</h2>;
}
export default ExpenseItem;
```

React Basics – Creating a custom component.

```
function ExpenseItem() {
return (
<div>
<div>Date</div>
<div>
<h2>Expense item</h2>
</div>
</div>
);
}
export default ExpenseItem;
```

React Basics – Creating a custom component.

```
function ExpenseItem() {
return (
<div>
<div>March 28 2021</div>
<div>
<h2>Car Insurance</h2>
<div>$297.56</div>
</div>
</div>
);
}
export default ExpenseItem;
```

React Basics – Add basic styling.

Add ExpenseItem.css

import it in js

Use className instead of class.

React Basics – Components.

Hardcoded data

Define html code once and be able to reuse it.

Need to change hardcoded data to dynamic data.

We can fake dynamic data.

JS expressions inside {}

React Basics – Components.

```
import "./ExpenseItem.css";
function ExpenseItem() {
const expenseDate = new Date(2021, 2, 28);
const expenseTitle = "Car Insurance";
const expenseAmount = 297.56;

return (
<div>
<div className="expense-item">{expenseDate}</div>
<div className="expense-item__description">
<h2>{expenseTitle}</h2>
<div className="expense-item__price">${expenseAmount}</div>
</div>
</div>
);
}
export default ExpenseItem;
```

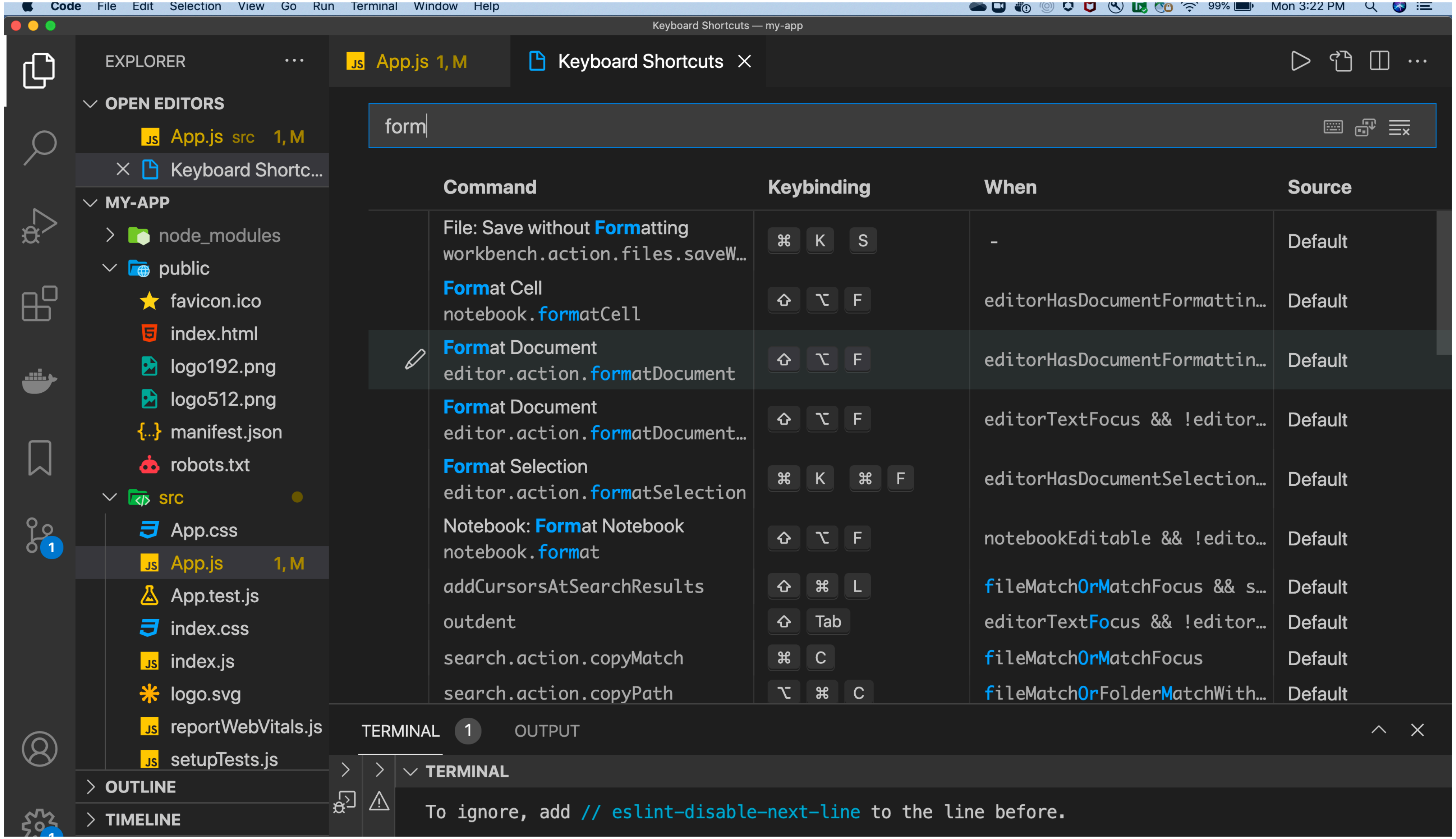
React Basics – Make reusable code. Components.

Can add ExpenseItem multiple times but same data repeats. Use parameters instead for functions.

```
import logo from "./logo.svg";
import "./App.css";
import ExpenseItem from "./components/ExpenseItem";

function App() {
return (
<div className="App">
<h2>Let's get started</h2>
<ExpenseItem />
<ExpenseItem />
<ExpenseItem />
<ExpenseItem />
</div>
);
}
```

React Basics – Creating a custom component.



form

Command	Keybinding	When	Source
File: Save without Formatting workbench.action.files.saveW…	⌘ K S	-	Default
Format Cell notebook.formatCell	⇧ ⌥ F	editorHasDocumentFormattin…	Default
Format Document editor.action.formatDocument	⇧ ⌥ F	editorHasDocumentFormattin…	Default
Format Document editor.action.formatDocument…	⇧ ⌥ F	editorTextFocus && !editor…	Default
Format Selection editor.action.formatSelection	⌘ K ⌘ F	editorHasDocumentSelection…	Default
Notebook: Format Notebook notebook.format	⇧ ⌥ F	notebookEditable && !edito…	Default
addCursorsAtSearchResults	⇧ ⌘ L	fileMatchOrMatchFocus && s…	Default
outdent	⇧ Tab	editorTextFocus && !editor…	Default
search.action.copyMatch	⌘ C	fileMatchOrMatchFocus	Default
search.action.copyPath	⌥ ⌘ C	fileMatchOrFolderMatchWith…	Default

```
To ignore, add // eslint-disable-next-line to the line before.
```

React Basics – props (properties).

Parameters for Components. Like parameters for a function.

Passing data.

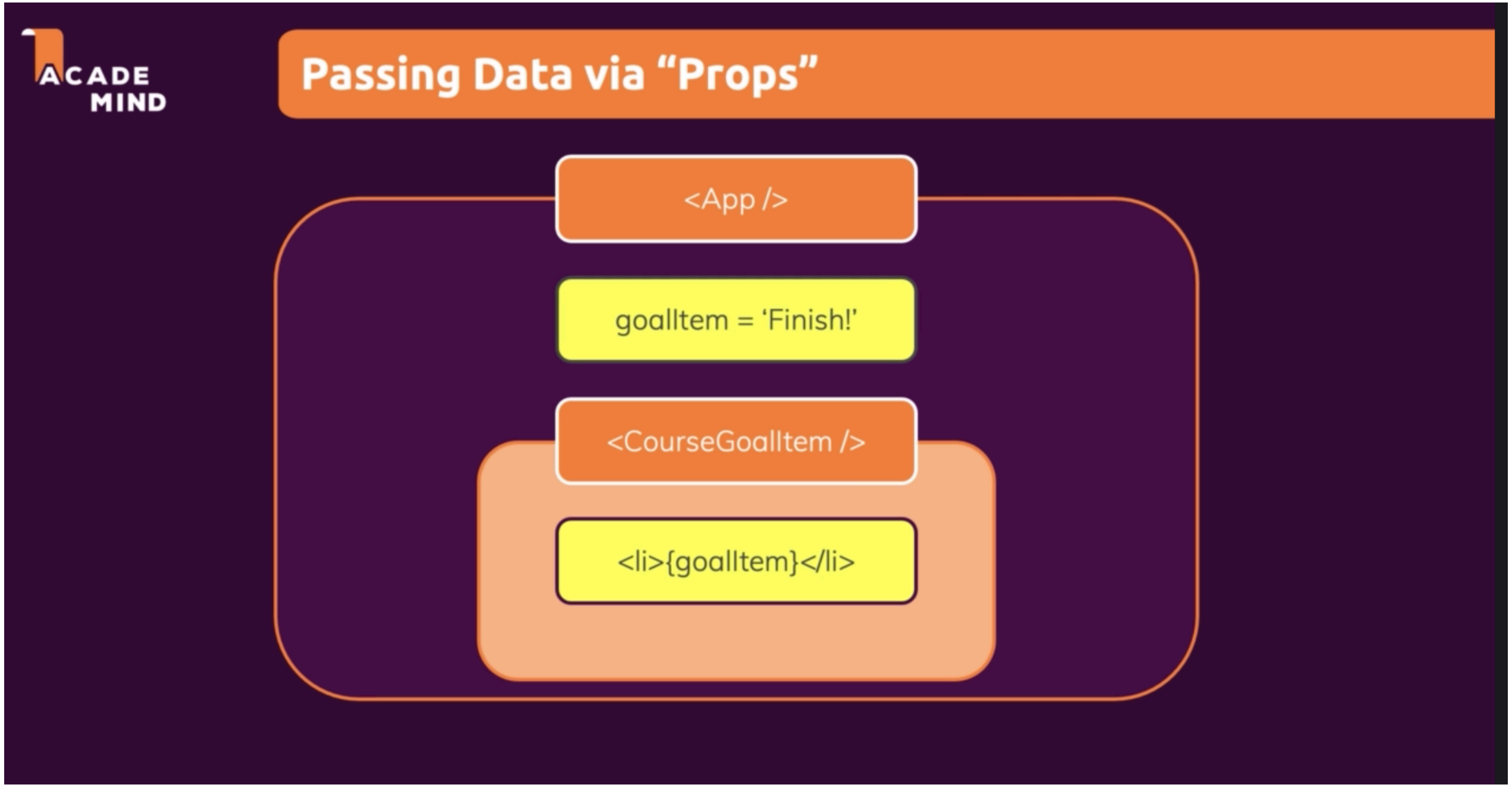
Object parameter that holds properties (props).

Key value pairs.

For example: props.title

Components reusable and configurable

React Basics – props.



Passing Data via “Props”

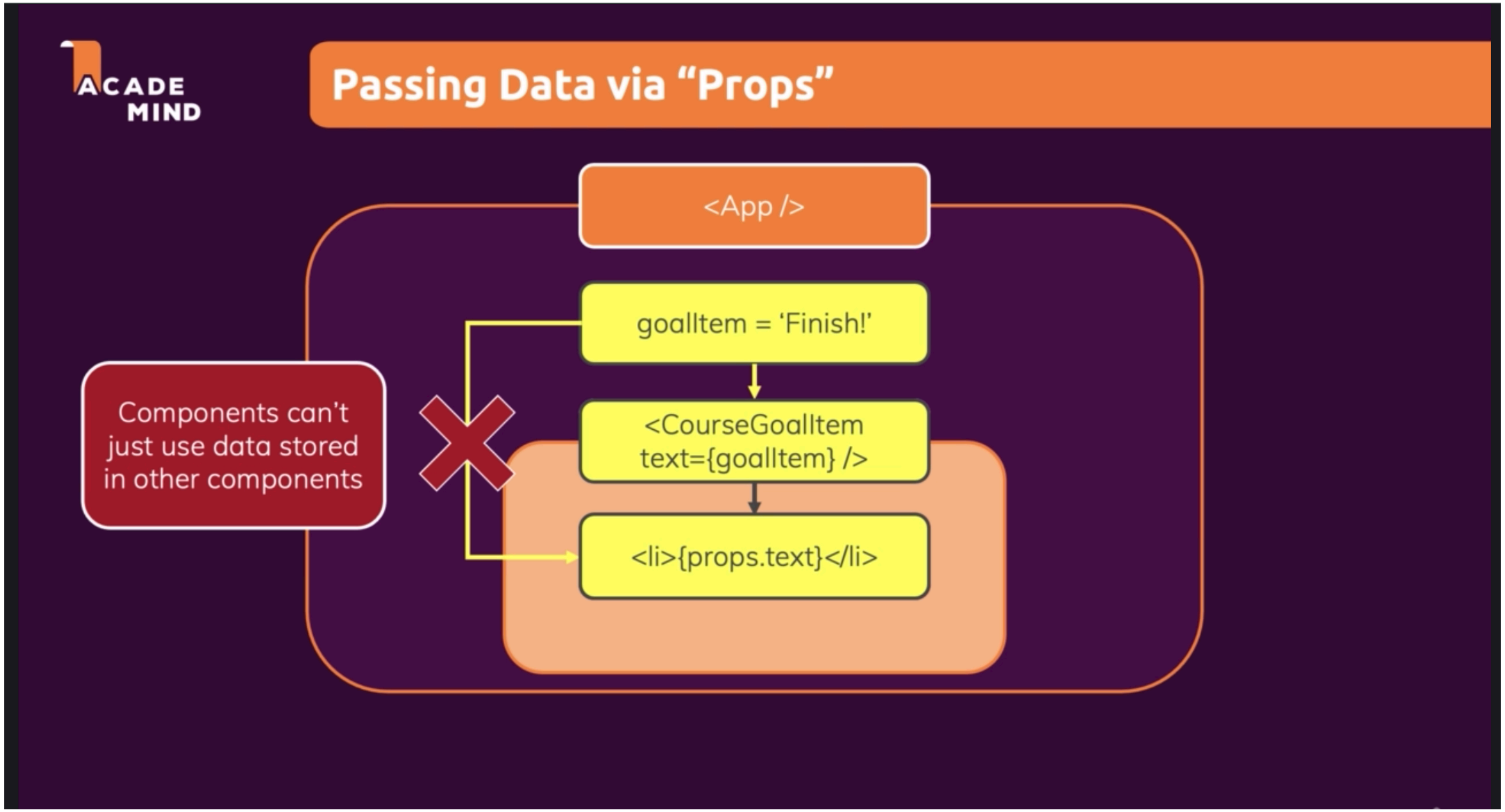
<App />

goalItem = ‘Finish!’

<CourseGoalItem />

<li>{goalItem}</li>

React Basics – props.



Passing Data via "Props"

Components can't just use data stored in other components

React Basics – props (properties).

Use the text file and copy the expense objects into App.js

Need to pass the data from expenses into ExpenseItem

Props concept to the rescue!

{} can be used within the attributes as well.

React Basics – props (properties).

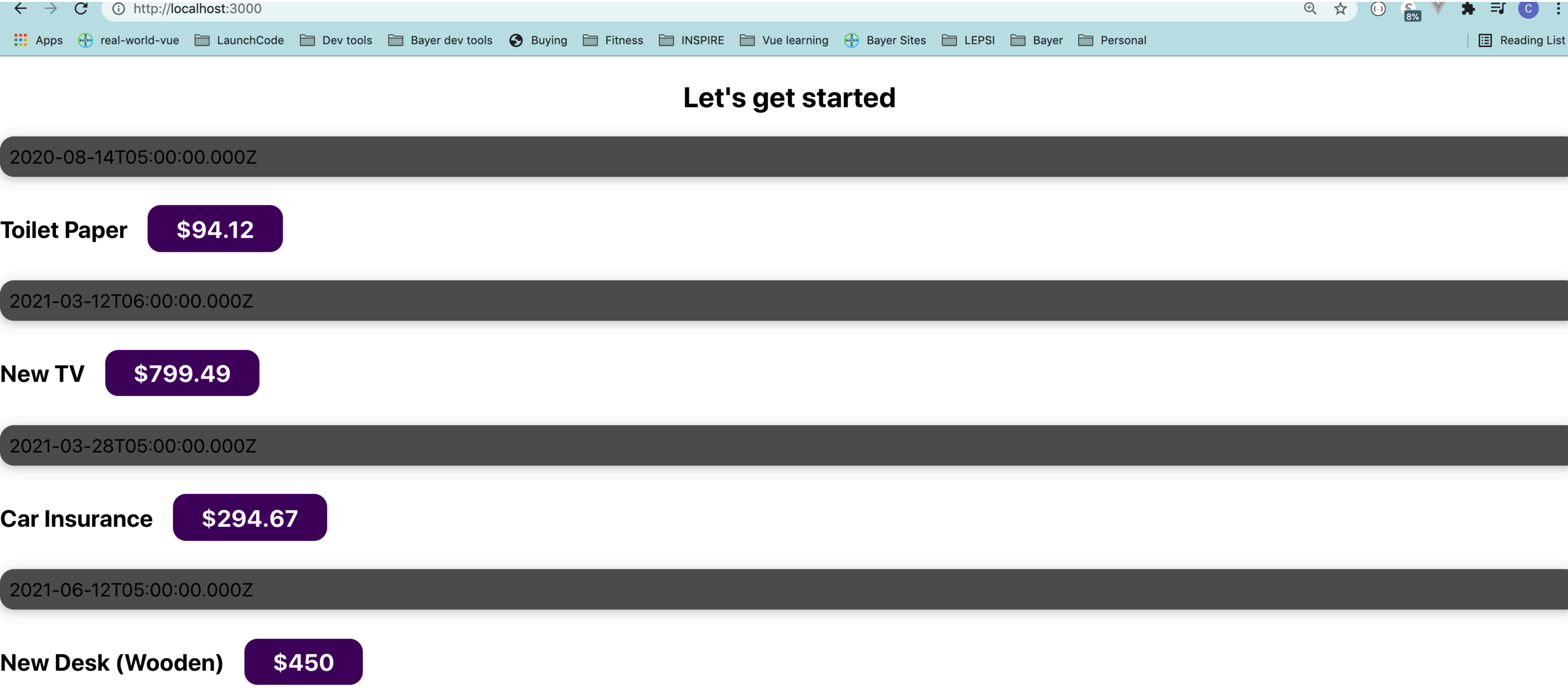
```
import logo from "./logo.svg";
import "./App.css";
import ExpenseItem from "./components/ExpenseItem";

function App() {
const expenses = [
{
id: "e1",
title: "Toilet Paper",
amount: 94.12,
date: new Date(2020, 7, 14),
},
{ id: "e2", title: "New TV", amount: 799.49, date: new Date(2021, 2, 12) },
{
id: "e3",
title: "Car Insurance",
amount: 294.67,
date: new Date(2021, 2, 28),
},
{
id: "e4",
title: "New Desk (Wooden)",
amount: 450,
date: new Date(2021, 5, 12),
},
];
```

React Basics – props (properties).

Keys have to match up.

```
import "./ExpenseItem.css";
function ExpenseItem(props) {
return (
<div>
<div className="expense-item">{props.date.toISOString()}</div>
<div className="expense-item__description">
<h2>{props.title}</h2>
<div className="expense-item__price">${props.amount}</div>
</div>
</div>
);
}
export default ExpenseItem;
```



Let's get started

2020-08-14T05:00:00.000Z

Toilet Paper **$94.12**

2021-03-12T06:00:00.000Z

New TV **$799.49**

2021-03-28T05:00:00.000Z

Car Insurance **$294.67**

2021-06-12T05:00:00.000Z

New Desk (Wooden) **$450**

React Basics – props (properties).

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date/toLocaleString

```
{props.date.toLocaleString("en-US", { month: "long" })}
```

Or use helper variables

React Basics – props (properties).

```
import "./ExpenseItem.css";
function ExpenseItem(props) {
const month = props.date.toLocaleString("en-US", { month: "long" });
const day = props.date.toLocaleString("en-US", { day: "2-digit" });
const year = props.date.getFullYear();

return (
<div className="expense-item">
<div>
<div>{month}</div>
<div>{day}</div>
<div>{year}</div>
</div>
<div className="expense-item__description">
<h2>{props.title}</h2>
<div className="expense-item__price">${props.amount}</div>
</div>
</div>
);
}
export default ExpenseItem;
```

React Basics – Splitting Components.

Expense Item component could be split into two components.

ExpenseDate

```
function ExpenseDate(props) {
const month = props.date.toLocaleString("en-US", { month: "long" });
const day = props.date.toLocaleString("en-US", { day: "2-digit" });
const year = props.date.getFullYear();

return (
<div>
<div>{month}</div>
<div>{day}</div>
<div>{year}</div>
</div>
);
}

export default ExpenseDate;
```

React Basics – Splitting Components.

```
import "./ExpenseItem.css";
import ExpenseDate from "./ExpenseDate";
function ExpenseItem(props) {
const month = props.date.toLocaleString("en-US", { month: "long" });
const day = props.date.toLocaleString("en-US", { day: "2-digit" });
const year = props.date.getFullYear();

return (
<div className="expense-item">
<ExpenseDate date={props.date} />
<div className="expense-item__description">
<h2>{props.title}</h2>
<div className="expense-item__price">${props.amount}</div>
</div>
</div>
);
}
export default ExpenseItem;
```

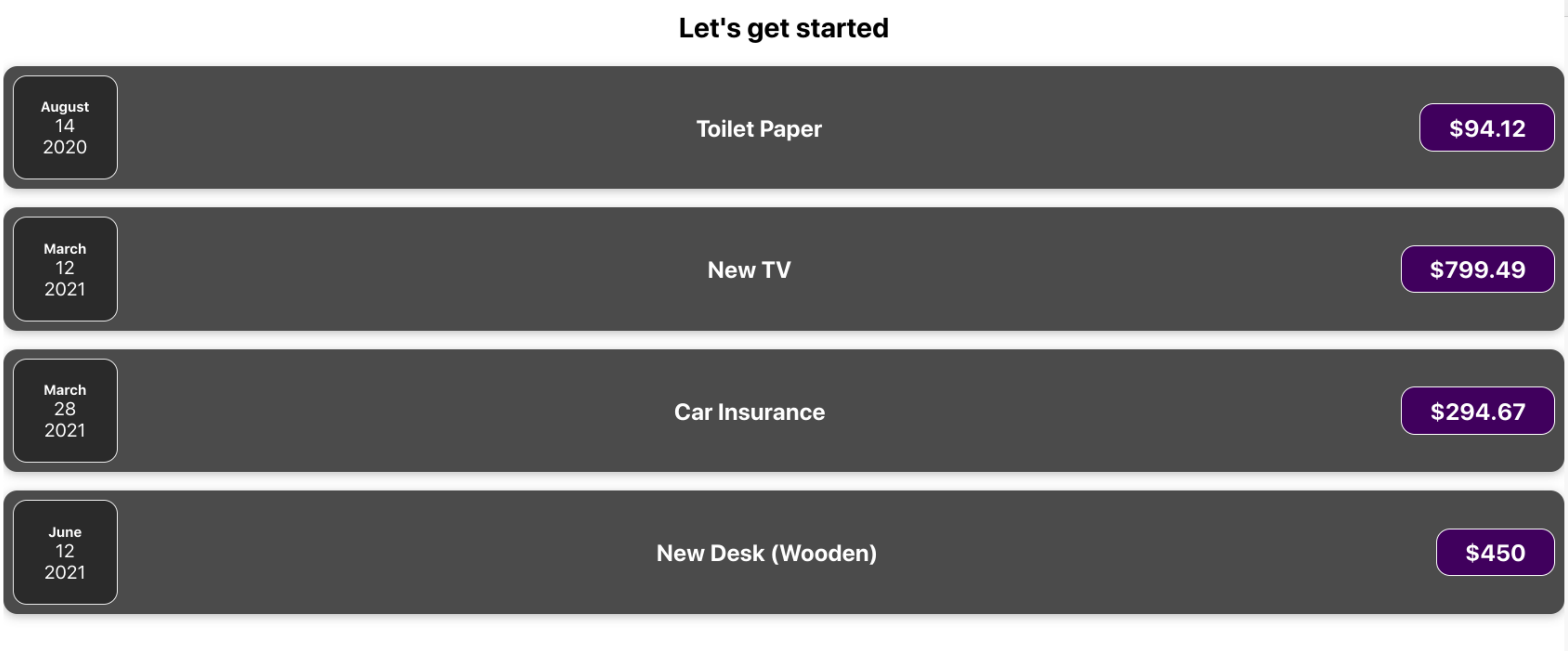
React Basics – Splitting Components with CSS.

```
import "./ExpenseDate.css";
function ExpenseDate(props) {
const month = props.date.toLocaleString("en-US", { month: "long" });
const day = props.date.toLocaleString("en-US", { day: "2-digit" });
const year = props.date.getFullYear();

return (
<div className="expense-date">
<div className="expense-date__month">{month}</div>
<div className="expense-year">{day}</div>
<div className="expense-day">{year}</div>
</div>
);
}

export default ExpenseDate;
```

React Basics – Splitting Components with CSS.



Let's get started

August 14 2020	Toilet Paper	$94.12
March 12 2021	New TV	$799.49
March 28 2021	Car Insurance	$294.67
June 12 2021	New Desk (Wooden)	$450

React Basics – Assignment 1

- Keep expenses Array
- Create a new component which renders the expense items and that component is rendered in App.js
- Can call it Expenses.
- It should wrap all the expense items in a div and use className=”expenses” from Expenses.css

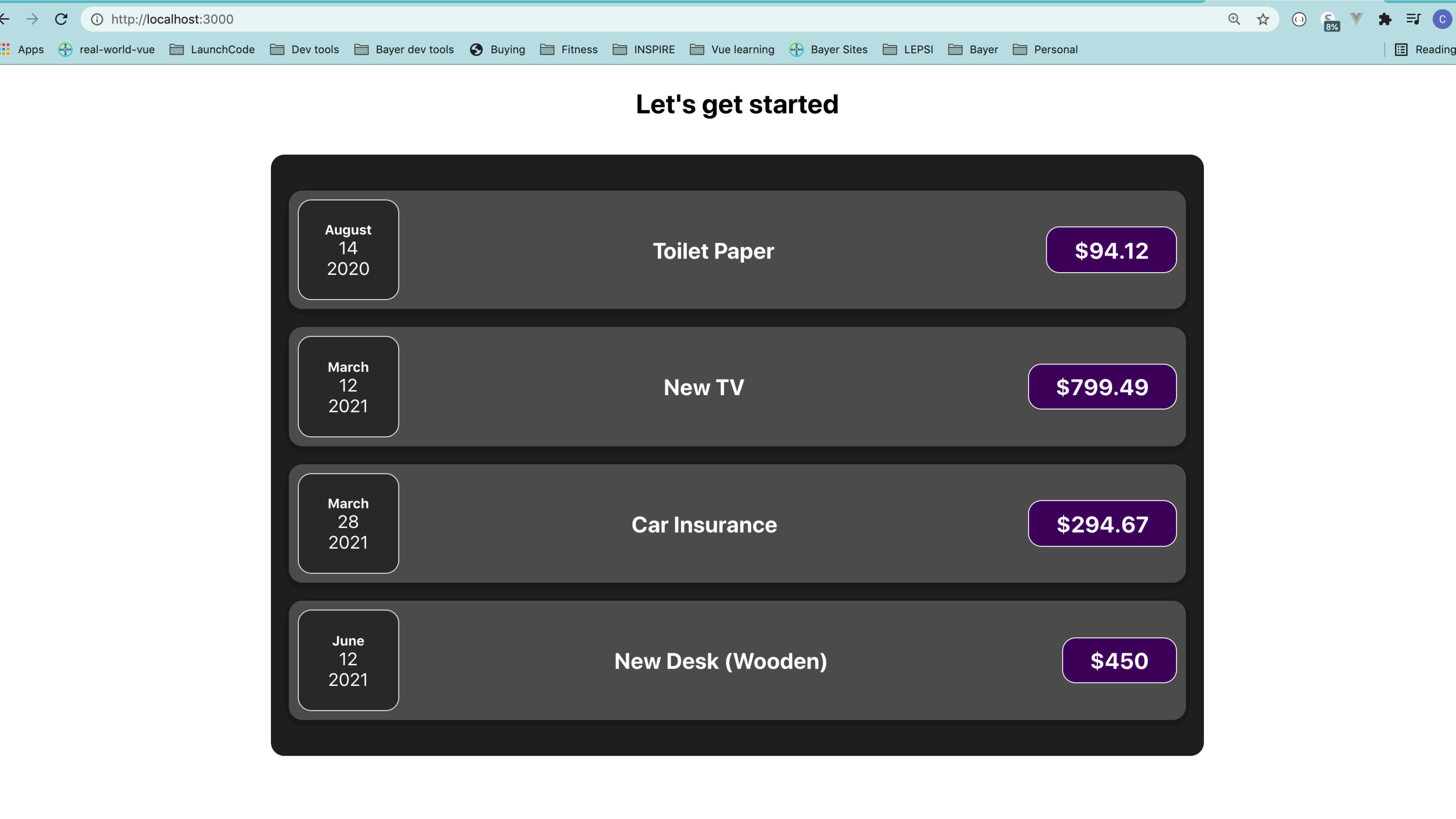
React Basics – Assignment 1

```
import "./Expenses.css";
import ExpenseItem from "./ExpenseItem";
function Expenses(props) {
const expenses = props.expenses;
return (
<div className="expenses">
<ExpenseItem
title={expenses[0].title}
amount={expenses[0].amount}
date={expenses[0].date}
/>
<ExpenseItem
title={expenses[1].title}
amount={expenses[1].amount}
date={expenses[1].date}
/>
<ExpenseItem
title={expenses[2].title}
amount={expenses[2].amount}
date={expenses[2].date}
/>
<ExpenseItem
title={expenses[3].title}
amount={expenses[3].amount}
date={expenses[3].date}
```

React Basics – Assignment 1

```
.expenses {
padding: 1rem;
background-color: rgb(31, 31, 31);
margin: 2rem auto;
width: 50rem;
max-width: 95%;
border-radius: 12px;
}
```

-



Let's get started

August 14 2020	Toilet Paper	$94.12
March 12 2021	New TV	$799.49
March 28 2021	Car Insurance	$294.67
June 12 2021	New Desk (Wooden)	$450

Project

https://learn.launchcode.org/courses/306/pages/project-rubric-all-gas-included

Links

- https://jsbin.com/?js,output
- https://reactjs.org/
- https://github.com/chetna23/JavaScriptRefresher
- https://github.com/facebook/create-react-app
- https://www.udemy.com/course/react-the-complete-guide-incl-redux/learn/lecture/25595350#overview
- https://egghead.io/courses/the-beginner-s-guide-to-react
- https://www.cronj.com/blog/javascript-es7-es8-new-features/
-
- Extensions: Prettier, Babbel JavaScript, Live Server - Ritwick Dey