# How React Works with the Browser

**Peter Kellner**

Developer, Consultant and Author

ReactAtScale.com @pkellner linkedin.com/in/peterkellner99

React separates the building and managing of components from their rendering to a device.

# React Design



## Web Browser

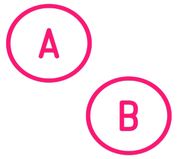
**React renders to a physical DOM which is the browser itself**

## Smart Phone

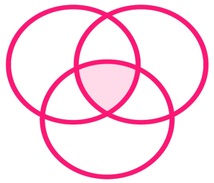
**React Native renders to smart phone like and iPhone or Android**

# Building Apps for React and React Native

There is no “write once, run everywhere” for React and React Native




Separate components required for UIs in React and React Native



You can build shared components between React and React Native

# React and React Native Code Compared

React for the web

```
function App() {
  return (
    <div>
      <b>Hello From Pluralsight!</b>
    </div>
  );
}
```

# React Components

```
function App() {
  return (
    <View
      style={{
        fontSize: 20,
        fontWeight: "bold"
      }}>
     <Text>Hello From Pluralsight!
     </Text>
   </View>
  )
}
```

React for the web

```
function App() {
  return (
    <div>
      <b>Hello From Pluralsight!</b>
    </div>
  );
}
```

The skills you develop for building React apps can be leveraged in React Native apps.

# Two Libraries Define React on the Web

## React

Creating React Elements

Creating UIs

Linking components together

## ReactDOM

Rendering elements to a browser

Renders Root Element to the DOM

ReactDOM is about the “what” to render and “where” to render it

# A Basic React App Launching

index.js

```
import ReactDOM from "react-dom";

const container =
  document.getElementById('root');
const root =
  ReactDOM.createRoot(container);

const RootComponent =
  () => <div>Hello From Pluralsight!</div>

root.render(
  <RootComponent />
);
```

# Reconciliation

## Old Virtual DOM

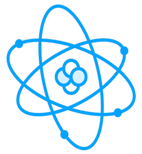
```
▼<div class="container">
  ▼<div class="row mb-1 ms-1 me-1 mt-2"> flex
    ▼<form>
      ▼<div class="row"> flex
        ▼<div class="col-7">
            <input class="px-2 mt-2 mb-2 ms-1" type="text" placeholder="Enter new task"
            value>
          </div>
        ▼<div class="col-5">
            <button class="px-2 mt-2 mb-2 ms-1">Add Item</button>
          </div>
        </div>
      </form>
    </div>
  ▼<div class="row mb-3 ms-1 me-1 mt-3"> flex
    ▼<ul class="mt-3">
        <b class="ms-3">Items:</b>
        <li class="px-2 mt-1 mb-1 ms-2">Buy Sugar</li>
        <li class="px-2 mt-1 mb-1 ms-2">Eat Carrots</li>
      </ul>
    </div>
  </div>
</div>
```

## New Virtual DOM

```
▼<div class="container">
  ▼<div class="row mb-1 ms-1 me-1 mt-2"> flex
    ▼<form>
      ▼<div class="row"> flex
        ▼<div class="col-7">
            <input class="px-2 mt-2 mb-2 ms-1" type="text" placeholder="Enter new task"
            value>
          </div>
        ▼<div class="col-5">
            <button class="px-2 mt-2 mb-2 ms-1">Add Item</button>
          </div>
        </div>
      </form>
    </div>
  ▼<div class="row mb-3 ms-1 me-1 mt-3"> flex
    ▼<ul class="mt-3">
        <b class="ms-3">Items:</b>
        <li class="px-2 mt-1 mb-1 ms-2">Return Oats</li>
        <li class="px-2 mt-1 mb-1 ms-2">Buy Sugar</li>
        <li class="px-2 mt-1 mb-1 ms-2">Eat Carrots</li>
      </ul>
    </div>
  </div>
</div>
```

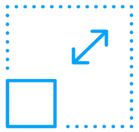
# Complex React App Reconciliation

Many elements can change during updates



Challenge is to figure out minimal diffs for updates

No diff optimization leads to O(n^3) comparisons



200 components would lead to 6 million comparisons

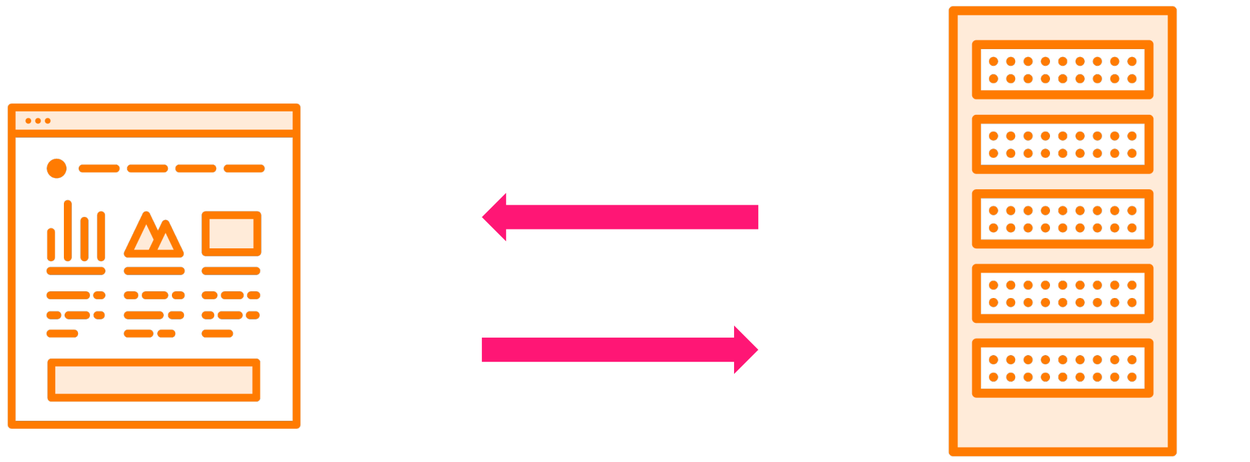
React team has optimized Reconciliation for speed

**Reconciliation step is very fast because of all the optimizations the React team has implemented.**

# Distributed Components

**Currently, only viable distributed React apps are using React Server Components.**

# Distributed React Apps



Browser

Node Server

# Distributed React Apps

# Previously, Simple HTML Page List of Numbers

EXPLORER

CLIP04

index.html U

index.js U

index.html U

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Single Page App (SPA)</title>
  <script src="index.js"></script>
</head>
<body>
  <div id="root"></div>
</body>
</html>
```

EXPLORER

CLIP04

index.html U

index.js U

index.js .../pages U

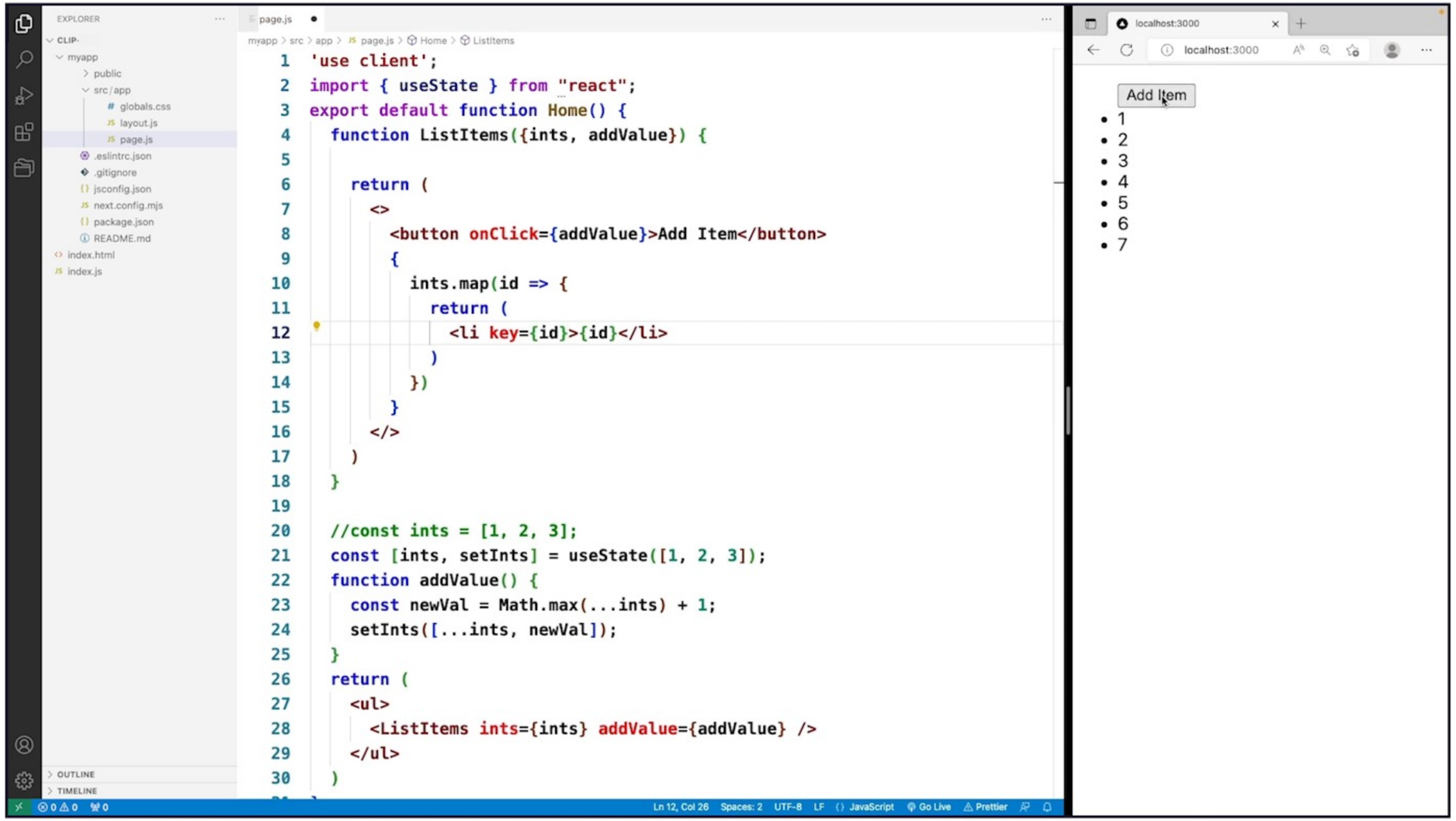
index.js ./ U

index.js > onload

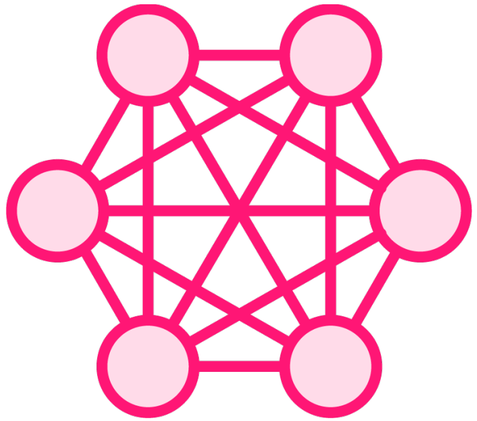
```
window.onload = () => {
  const rootElement = document.getElementById("root");
  const ints = [1, 2, 3];

  ints.forEach(i => {
    let li = document.createElement("li");
    li.innerHTML = i;
    rootElement.appendChild(li);
  });

};
```


Pure SPA

# React App, Numbers List with Add Button



# Previously, Simple HTML Page List of Numbers

HTML

JavaScript

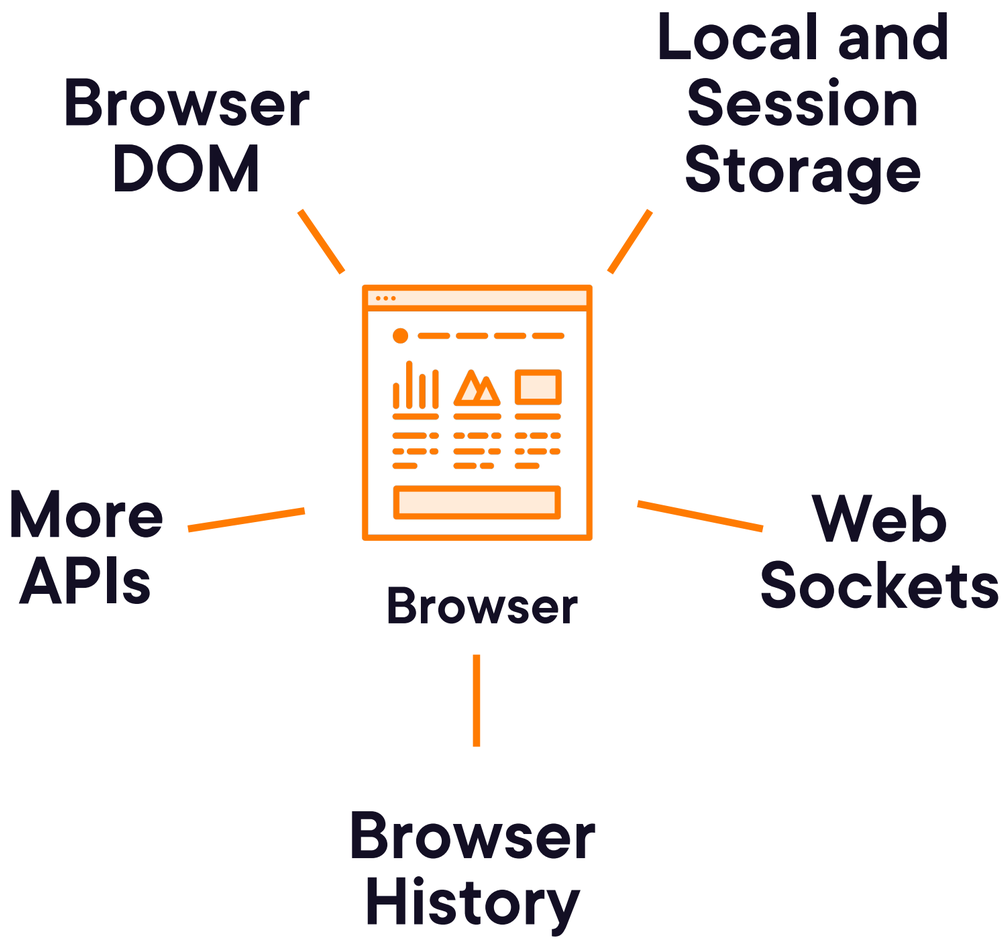
Rendered in browser

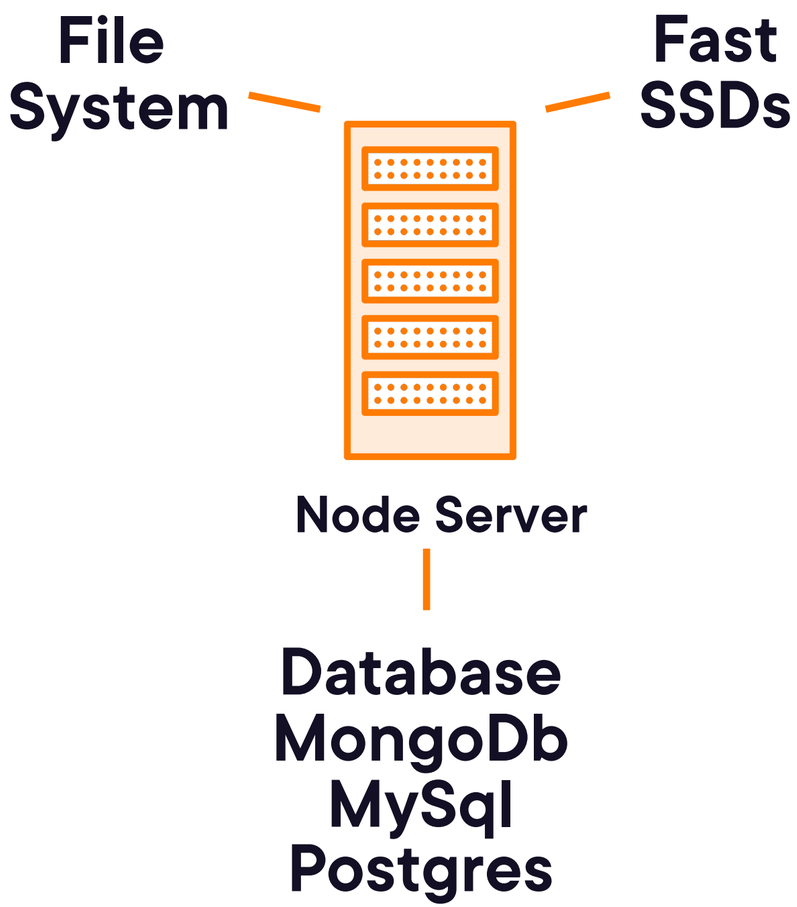
# React Server Components (RSCs)

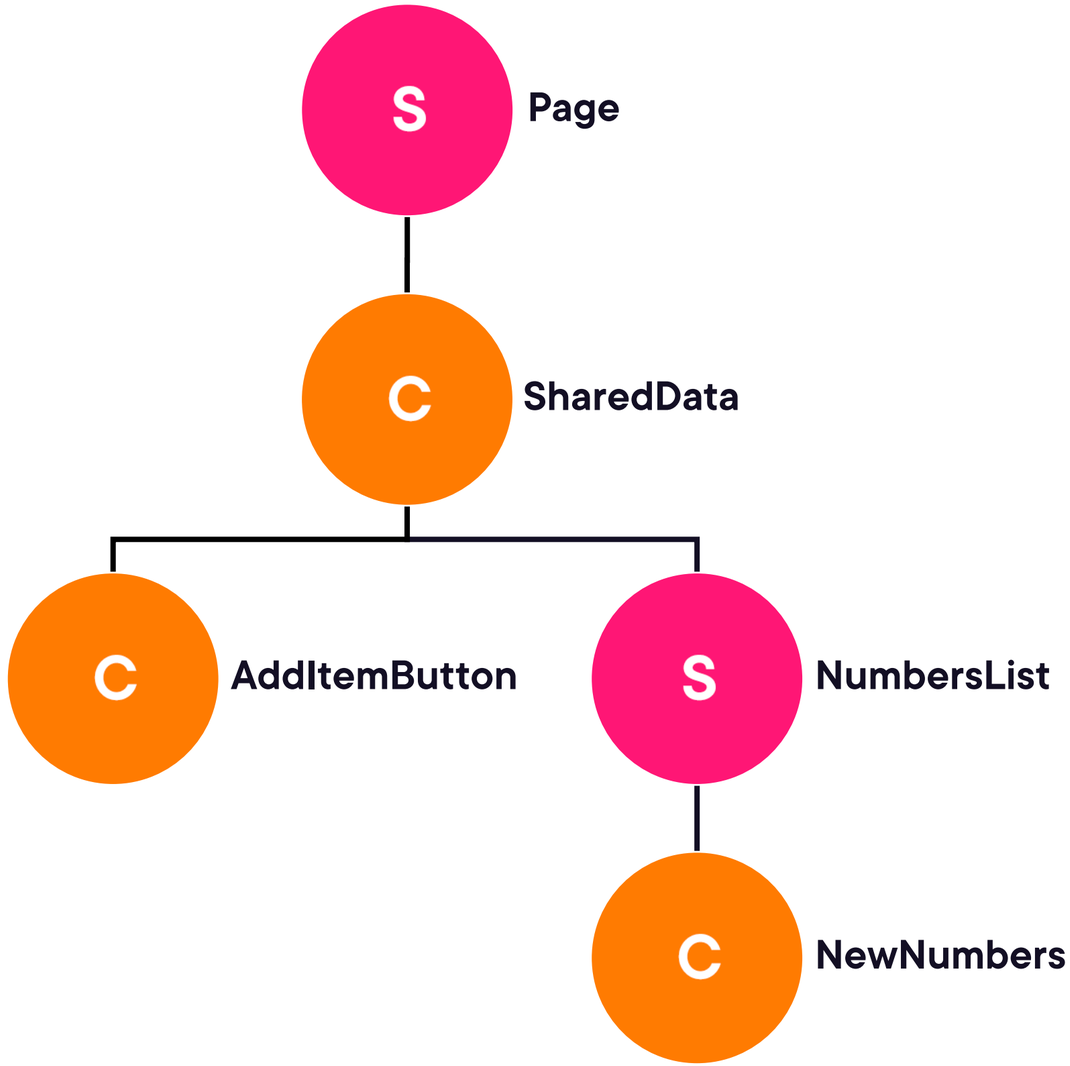
/src/app/page.js

```
function ServerComponent() {

  await sql.connect(sqlConfig)
  const results =
  await sql.query`select id,name from mytable`

  return (
    <ul>
      {
        results.map(rec => {
          return (
            <li key={rec.id}>{rec.name}</li>
          )
        })
      }
    </ul>
  )
}
```

# React Server Components (RSCs)






S – React Server Component
C – React Client Component

# React rendering data is all about how React facilitates browser DOM updates.

# React and Browser Events

```
localComponentState = [1, 2, 3, 4, 5]
```

```
<html>
<body>
  <ul>
    <li>1</li>
    <li>2</li>
    <li>3</li>
    <li>4</li>
    <li>5</li>
  </ul>
</body>
</html>
```

# React and Browser Events

```
localComponentState = [1, 2, 3, 4, 5, 8]
```

```
<html>
<body>
  <ul>
    <li>1</li>
    <li>2</li>
    <li>3</li>
    <li>4</li>
    <li>5</li>
    <li>8</li>
  </ul>
</body>
</html>
```

# React and Browser Events

```
localComponentState = [1, 2, 3, 4, 5, 8, 11]
```

```
<html>
<body>
  <ul>
    <li>1</li>
    <li>2</li>
    <li>3</li>
    <li>4</li>
    <li>5</li>
    <li>8</li>
    <li>11</li>
  </ul>
</body>
</html>
```

# React and Browser Events

```
localComponentState = [1, 2, 3, 4, 5, 8, 11, 14]
```

```
<html>
<body>
  <ul>
    <li>1</li>
    <li>2</li>
    <li>3</li>
    <li>4</li>
    <li>5</li>
    <li>8</li>
    <li>11</li>
    <li>14</li>
  </ul>
</body>
</html>
```

**React component state changes could also come from external events like a database.**

# React and Accessing Databases


**SQLite**

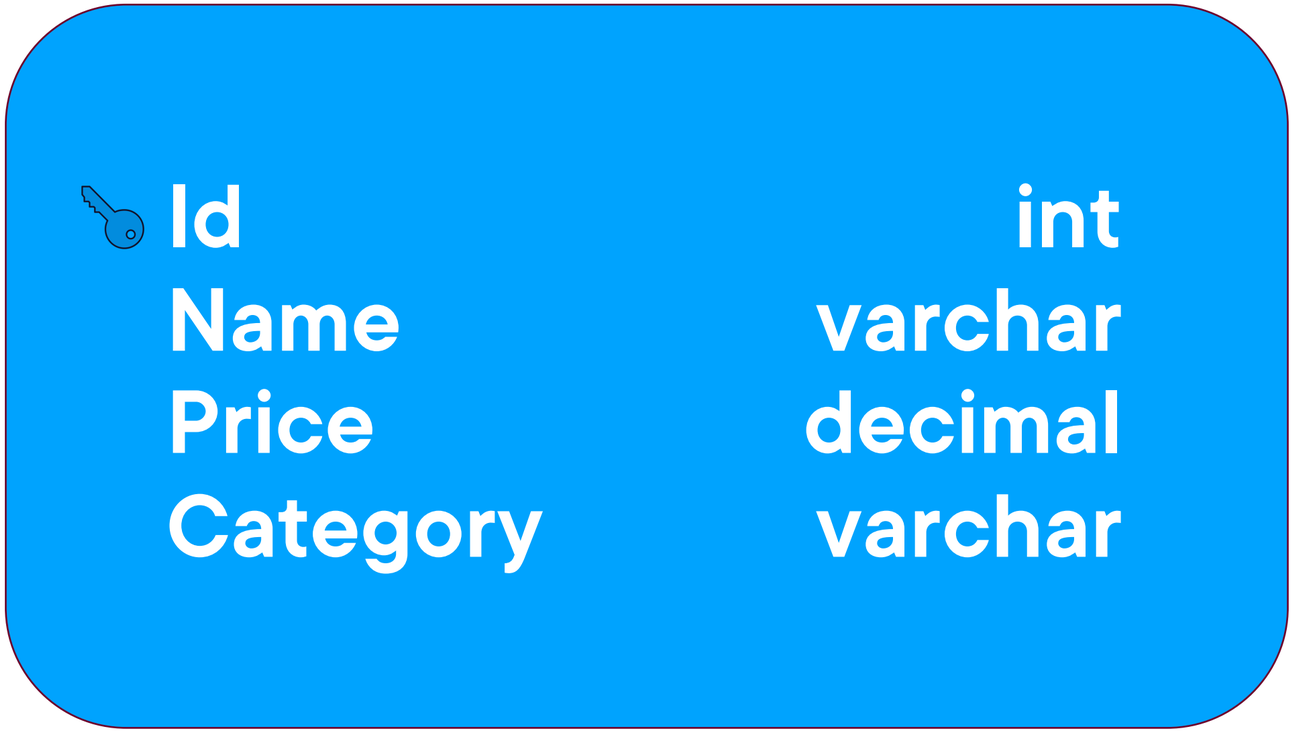
Database running on a local server

# React and Accessing Databases


**SQLite**
Database running on a local server

**Enterprise Database**
Could be any an enterprise database potentially running in the cloud

# SQLite Products Table



# No way to access database directly from the browser with a SPA

No way to access database directly from the browser with a SPA

Most common way to access a database from a browser is with the REST protocol

# What does a React app look like that renders data from a database?

# What does a React app look like that renders data from a database?

# React Server Components run directly on a Node Server.

# React Client Components can work with Suspense and async with no required local state.

# Important!

# Suspense in client-side React is only available in React 19 Release Candidate

# Choosing React

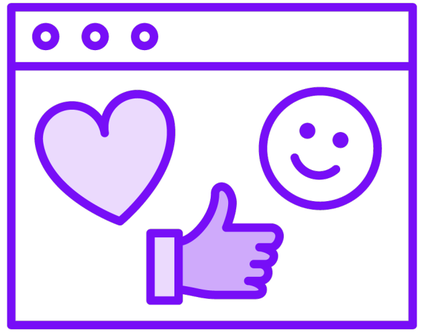
**Good understanding**

Learned what React is and how it fits into web development

# Choosing React

**Good understanding**

Learned what React is and how it fits into web development



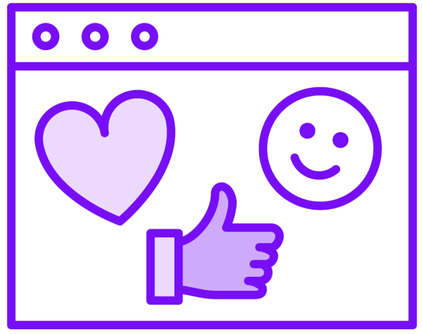
**Gained appreciation**

How React is good for you as a developer and your users

# Choosing React

**Good understanding**

Learned what React is and how it fits into web development



**Gained appreciation**

How React is good for you as a developer and your users

**My experience**

React has been a big win for me personally with my projects

# Final Thoughts


Fast to learn – easy to use

Unopinionated

Declarative

Easy to bind data to

# Final Thoughts


Fast to learn – easy to use

Unopinionated

Declarative

Easy to bind data to

Easy to build reusable components