Let's go **step by step** to create a React project using **Vite** and understand **props** with clear examples. I'll make it beginner-friendly.

---

## Step 1: Create a new React project with Vite

Open your terminal and run:

```
# Create a new Vite project
npm create vite@latest my-react-app

# Choose options:
# - Project name: my-react-app
# - Framework: React
# - Variant: JavaScript or TypeScript (I'll use JavaScript for simplicity)

cd my-react-app

# Install dependencies
npm install

# Start the development server
npm run dev
```

You should see a message like:

```
Local: http://localhost:5173/
```

Open this in your browser to see your React app running.

---

## Step 2: Understand Props in React

**Props** are inputs to React components. They allow you to pass data from a parent component to a child component.

- **Parent** → passes data
- **Child** → receives data via props

---

# Step 3: Create a simple component using props

1. In `src` folder, create a new file: `Greeting.jsx`

```
// Greeting.jsx
import React from "react";

// Props are received as a parameter
const Greeting = (props) => {
  return (
    <div>
      <h1>Hello, {props.name}!</h1>
      <p>You are {props.age} years old.</p>
    </div>
  );
};

export default Greeting;
```

2. Use this component in `src/App.jsx`

```
import React from "react";
import Greeting from "./Greeting";

function App() {
  return (
    <div>
      <h1>Welcome to My React App</h1>
      {/* Passing props to the Greeting component */}
      <Greeting name="Alice" age={25} />
      <Greeting name="Bob" age={30} />
    </div>
  );
}

export default App;
```

**# Start the development server**

```
npm run dev
```

You should see a message like:

```
Local: http://localhost:5173/
```

 Now, if you open the browser, you'll see:

```
Hello, Alice! You are 25 years old.
Hello, Bob! You are 30 years old.
```

---

# Step 4: Destructuring props (Cleaner Syntax)

Instead of `props.name` and `props.age`, you can destructure props directly in the function parameter:

```
const Greeting = ({ name, age }) => {
  return (
    <div>
      <h1>Hello, {name}!</h1>
      <p>You are {age} years old.</p>
    </div>
  );
};
```

This works **exactly the same** but is cleaner.

# Props with Event Handlers:-

Let's organize your React + Vite project with **separate files** for the `Button` component and the `App` component. Here's a clean step-by-step setup:

---

## Step 1: Project Structure

Your `src` folder will look like this:

```
src/
├── components/
│   └── Button.jsx
│
├── App.jsx
└── main.jsx
```

---

## Step 2: Create `Button.jsx`

Inside `src/components/Button.jsx`:

```
// src/components/Button.jsx
import React from "react";

const Button = ({ onClick, label }) => {
  return <button onClick={onClick}>{label}</button>;
};

export default Button;
```

---

## Step 3: Update `App.jsx`

Inside `src/App.jsx`:

```
// src/App.jsx
import React from "react";
import Button from "./components/Button";

function App() {
  const handleClick = () => {
    alert("Button clicked!");
  };

  return (
    <div style={{ padding: "20px" }}>
      <h1>React Props Example</h1>
      <Button onClick={handleClick} label="Click Me" />
    </div>
  );
}

export default App;
```

---

## Step 4: Ensure `main.jsx` is Correct

Inside `src/main.jsx` (Vite default):

```
import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";
import "./index.css"; // optional

ReactDOM.createRoot(document.getElementById("root")).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

---

## Step 5: Run the App

In the terminal:

```
npm run dev
```

Open the URL (usually `http://localhost:5173`) and you'll see your page with a button. Clicking it triggers the alert: **"Button clicked!"**