# Settingupreact

Just run underlined commands

Admin@MyLaptop MINGW64 ~

$ **cd Desktop** //or any directory you want to create

//installing vite

Admin@MyLaptop MINGW64 ~/Desktop

$ **npm create vite@latest**

> npx

> create-vite

√ Project name: ... MyFirstReactProject

√ Package name: ... myfirstreactproject

? Select a framework: » - Use arrow-keys. Return to submit.

> Vanilla

Vue

**React**

Preact

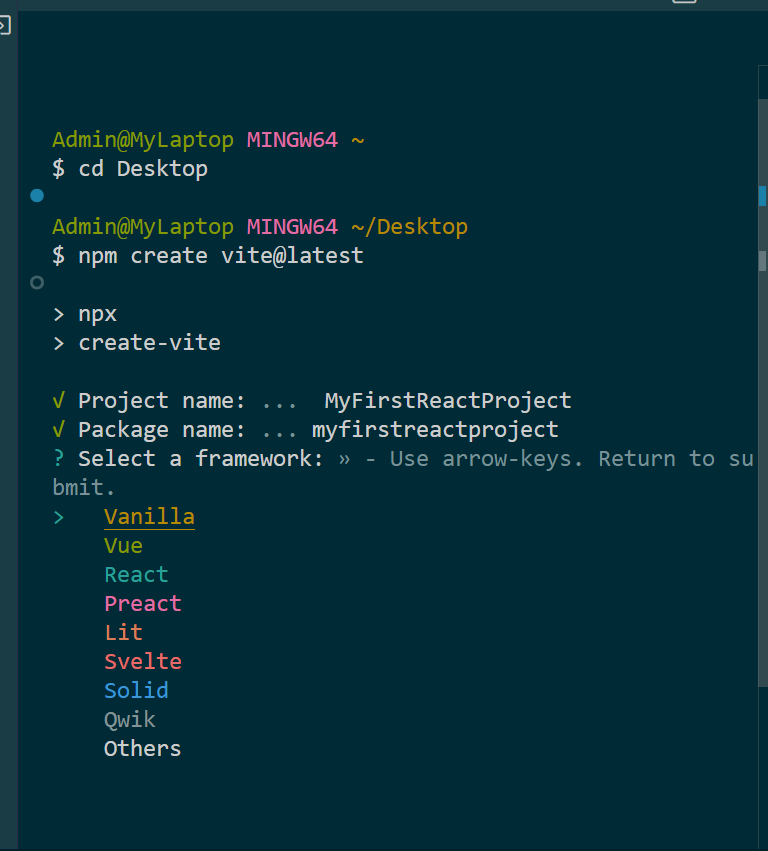
Lit

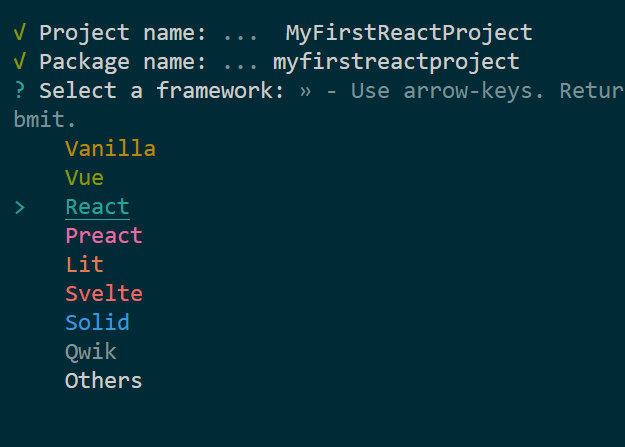
Svelte

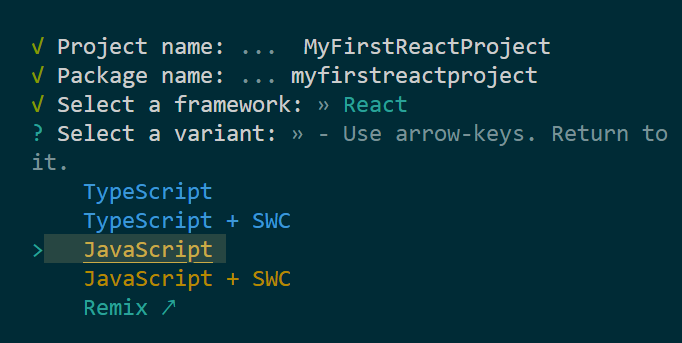
Solid

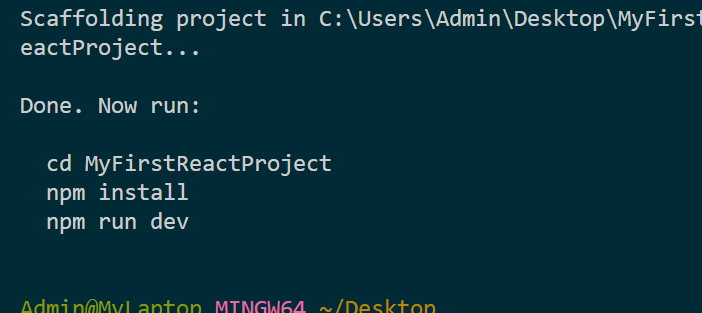
Qwik

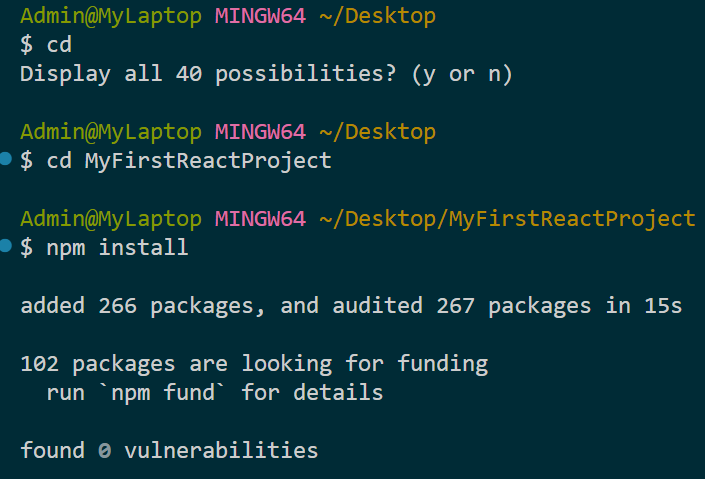
Others

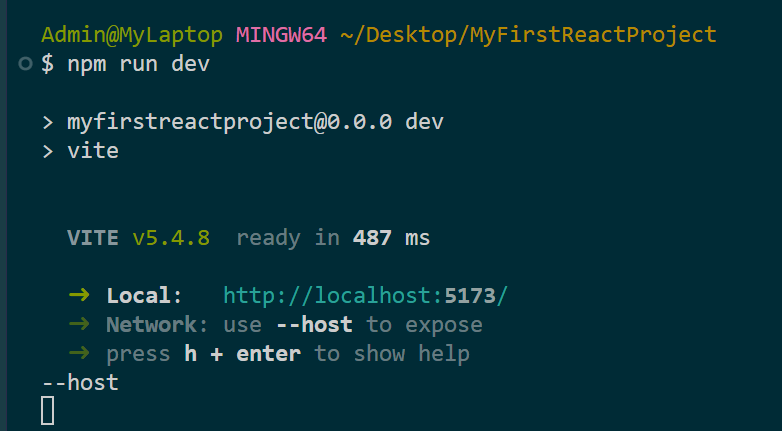




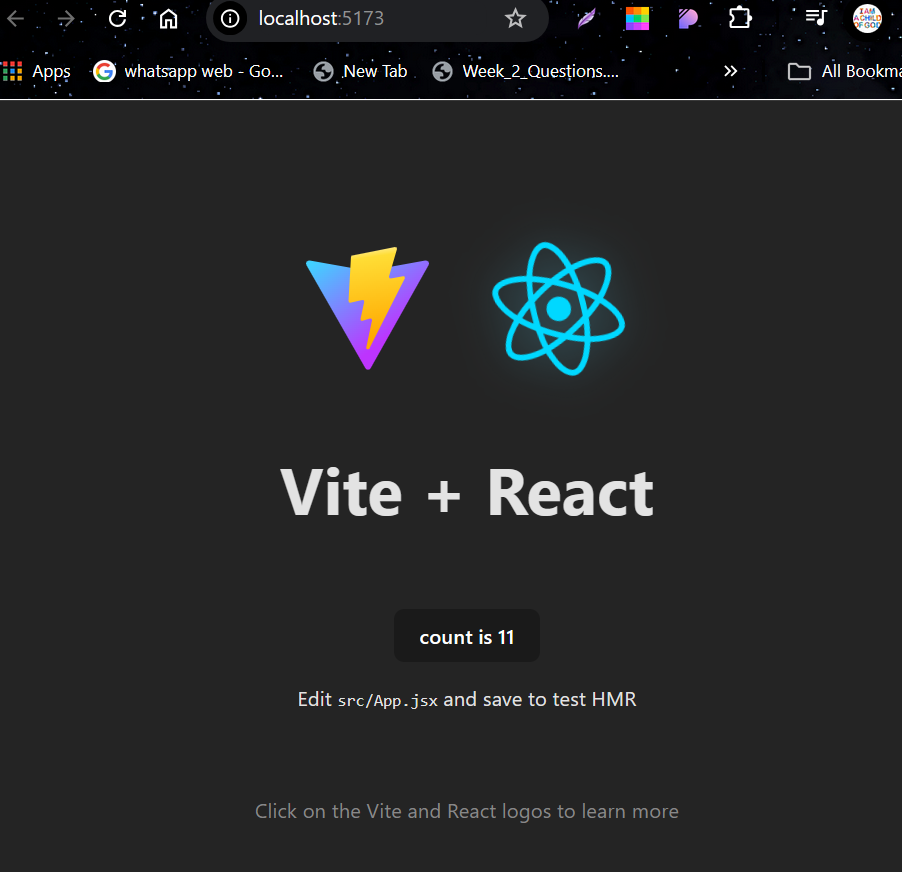








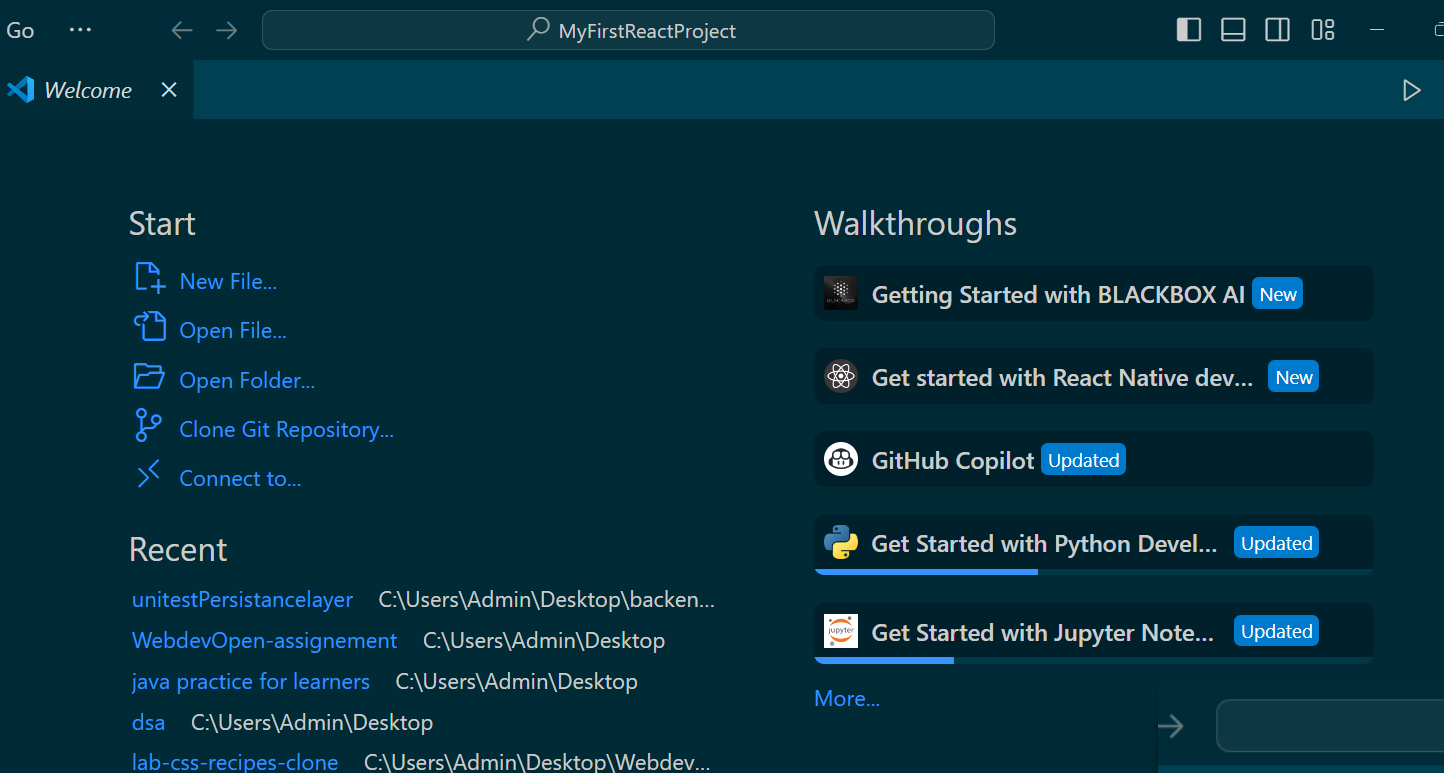




Command pallet

Ctr+shift+p

Or else click on search bar and use >





# My first app.jsx component

You can create many like app.jsx

But must import and export each jsx file to root

import React from "react"; //important

*//function names must always be capital*

*//i.e component names*

**function** App() {

*//jsx*

  return (

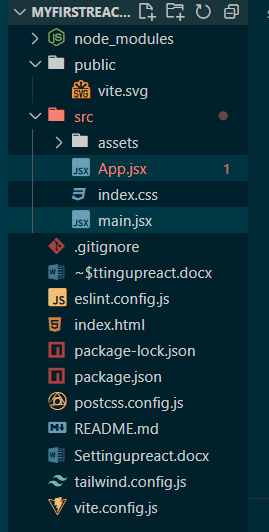
    <h1>hey</h1>

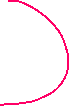
  );

*//end of jsx*

};

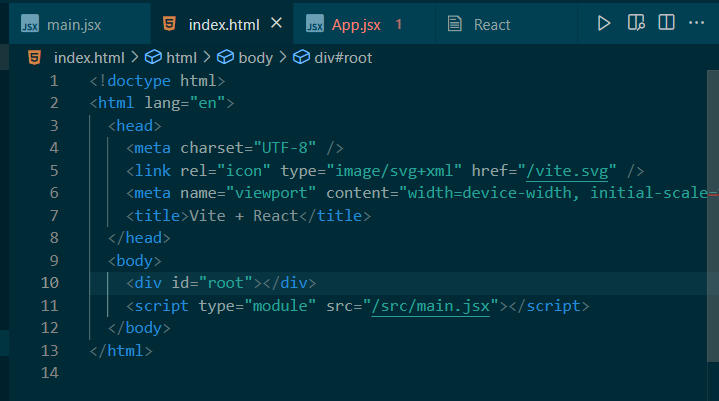
export default App; *//export name of the component*



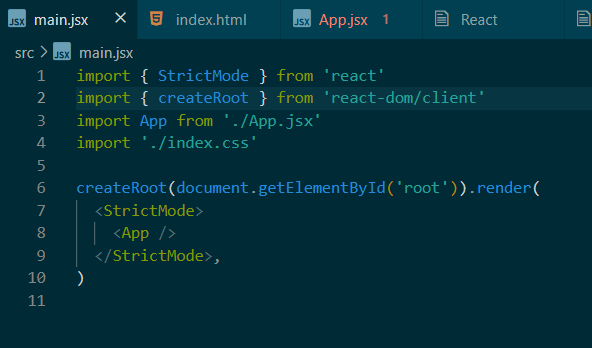


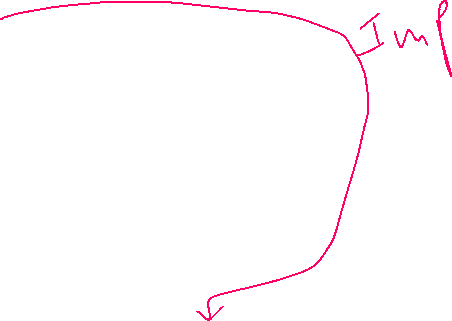
We can also change root to components or any name you wish

App.jsx is sending data to main.jsx and main.jsx is sending the component to index.html app.jsx->main.jsx->index.html









# Tailwind Css

To use Tailwind CSS with Vite in a React project, you'll need to follow a few steps.

First, you'll need to create a new React project with Vite. You can do this by running the command **npm init vite @latest** in your terminal. Then, select "React" as the framework and choose the variant you want to use (e.g., "JavaScript" or "TypeScript").

Once your project is set up, you can install Tailwind CSS by running the command

npm install -D tailwindcss postcss autoprefixer

This will install Tailwind CSS and its dependencies.

Next, you'll need to generate the configuration files for Tailwind CSS. You can do this by running the command

npx tailwindcss init -p

This will create a **tailwind.config.js** file in your project root.

In the **tailwind.config.js** file, you'll need to specify the paths to your template files. You can do this by adding the following code to the **content** section:

javascript

EditCopy code

module.exports = {

content: [

**"./index.html",**

**"./src/\*\*/\*.{js,ts,jsx,tsx}",**

],

theme: {

extend: {},

},

plugins: [],

}

Finally, you'll need to add the Tailwind CSS directives to your CSS file. You can do this by adding the following code to your **index.css** file:

css

EditCopy code

@**tailwind base;**

**@tailwind components;**

**@tailwind utilities;**

That's it! You should now be able to use Tailwind CSS in your React project.

Here's an example of how you can use Tailwind CSS in your **App.js** file:

javascript

EditCopy code

1import React from 'react';

2

3function App() {

4 return (

5 <div className="container mx-auto p-4">

6 <h1 className="text-3xl font-bold underline">Hello World!</h1>

7 </div>

8 );

9}

10

11export default App;

This code uses Tailwind CSS classes to style the **<h1>** element.

If you have any questions or need further help, feel free to ask!

# ShortCuts

1. **Option 1: Using ES7+ React/Redux/React-Native snippets Extension**
2. **Install the extension**:
   * In VS Code, go to the Extensions view (Ctrl+Shift+X).
   * Search for **"ES7+ React/Redux/React-Native snippets"**.
   * Install the extension.
3. **Generate React boilerplate**: Once the extension is installed, you can use a variety of shortcuts. To create a full functional React component boilerplate, type:

Copy code

## rfce

and press Enter. This will generate:

jsx

Copy code

import React from 'react';

function ComponentName() {

return (

<div>

</div>

);

}

export default ComponentName;

The rfce snippet is short for "React Functional Component with Export."

1. **Other Useful Snippets from the Extension:**

* rfc: React functional component
* rafce: React arrow function component with export
* rafc: React arrow function component

# Shorcut to Test websites on mobile

# With IP address

To access your Vite React app from your mobile device using your computer's IP address, follow these simple step-by-step instructions:

1. **Step 1: Ensure Both Devices Are on the Same Network**

Make sure both your **computer** (where you are running the Vite app) and your **mobile device** are connected to the same Wi-Fi network.

1. **Step 2: Find Your Computer’s Local IP Address**

You'll need your computer's local IP address to access it from your mobile.

* **On Windows**:
  1. Press Windows + R, type cmd, and press Enter to open Command Prompt.
  2. Type ipconfig and press Enter.
  3. Look for the section Wireless LAN adapter (if using Wi-Fi) or Ethernet adapter (if using a wired connection).
  4. Find the IPv4 Address, which will look like 192.168.x.x. This is your local IP address.
* **On macOS**:
  1. Open the **Terminal**.
  2. Type ifconfig and press Enter.
  3. Look for inet under en0 (Wi-Fi) or eth0 (Ethernet), which will show something like 192.168.x.x. This is your local IP address.

1. **Step 3: Modify Vite to Allow Network Access**

By default, Vite serves your app only on localhost. To allow access from other devices on your network, follow these steps:

1. Open the **vite.config.js** file (in the root of your project).
2. Add the following configuration:

javascript

Copy code

export default {

server: {

host: true, // This will allow Vite to listen on all addresses, including your local IP.

port: 5174 // Optional: You can specify the port if needed (default is 5174).

}

}

import { defineConfig } from 'vite'

import react from '@vitejs/plugin-react'

*// https://vite.dev/config/*

export default defineConfig({

  plugins: [react()],

  server: {

    host: true, *// This will allow Vite to listen on all addresses, including your local IP.*

    port: 5174  *// Optional: You can specify the port if needed (default is 5174).*

  }

})

This allows Vite to serve the site on your local IP address, not just localhost.

1. **Step 4: Restart Your Vite Server**

* Stop your current Vite development server by pressing Ctrl + C in the terminal.
* Restart it by running:

bash

Copy code

npm run dev

Now, Vite should be running on your **computer's IP address**, e.g., 192.168.x.x:5174.

1. **Step 5: Access the Website from Your Mobile**
2. Open the browser on your mobile device.
3. Type the URL using your computer's local IP address and the Vite port (5174 by default).

Example:

arduino

Copy code

http://192.168.x.x:5174

Replace 192.168.x.x with your actual computer’s IP address found in Step 2.

1. **Step 6: (Optional) Check Firewall Settings**

If your mobile device cannot connect to your Vite app, your firewall might be blocking the connection.

* **On Windows**:
  1. Go to Control Panel > Windows Defender Firewall.
  2. Click "Allow an app or feature through Windows Defender Firewall."
  3. Ensure that Node.js (or any program related to Vite) is allowed through the firewall for both Private and Public networks.
* **On macOS**:
  1. Go to System Preferences > Security & Privacy > Firewall.
  2. Click "Firewall Options" and ensure that your development tools (e.g., Node.js, terminal) are allowed.

1. **Step 7: Test It!**

* On your mobile, visit the IP address you found earlier and you should see your Vite React app running on your phone's browser!

If you run into any issues, feel free to ask for help on a specific step!

4o

1. **You said:**

import { defineConfig } from 'vite' import react from '@vitejs/plugin-react' // https://vitejs.dev/config/ export default defineConfig({ plugins: [react()], }) modify it for step 3

1. **ChatGPT said:**

ChatGPT

To modify your vite.config.js for network access as described in **Step 3**, you need to add the server configuration to allow Vite to listen on all IP addresses and not just localhost. Here's how you can update your vite.config.js:

1. **Modified vite.config.js**

javascript

Copy code

import { defineConfig } from 'vite'

import react from '@vitejs/plugin-react'

// https://vitejs.dev/config/

export default defineConfig({

plugins: [react()],

server: {

host: true, // This allows access using your local network IP address

port: 5174, // Optional: Use the default port (5174) or any port you prefer

},

})

1. **Explanation:**

* **host: true**: This tells Vite to listen on all network interfaces, including your computer’s IP address, making the app accessible via the IP address from other devices on the same network.
* **port: 5174**: Specifies the port (optional, as Vite defaults to 5174 if you don't set it).

1. **After modifying this file:**
2. Save the file.
3. Stop your current Vite development server by pressing Ctrl + C in the terminal.
4. Restart the server using:

bash

Copy code

npm run dev

Now, your Vite app will be accessible from your mobile device at http://<your-computer-ip>:5174.

# **CONDITIONAL RENDERING**

We can add data contionally how

import React from 'react';

import card1 from '../assets/card1.png';

import card2 from '../assets/card2.png';

import card3 from '../assets/card3.png';

import card4 from '../assets/card4.png';

**const** carddata = [

     {

        src: card1,

        title: "Declarative",

        description: "React makes it painless to create interactive UIs",

        instock:true

    },



     {

        src: card2,

        title: "Components",

        description: "Build encapsulated components that manage their state",

        instock:false



    },

     {

        src: card3,

        title: "Single-Way",

        description: "A set of immutable values are passed to the compmnent's",

        instock:true

    },



     {

        src: card4,

        title: "JSX",

        description: "Statically-types designed to run on modern browers",

        instock:false

    }



];

**function** Cards() {

  return (

    <>

    <div className="w-full h-full  gap-20 bg-slate-50 flex align-middle justify-center items-center flex-wrap">

            {   *//this bracket is must*

             carddata.map((card,index)**=>**{

**const** {src,title,description}=card;

                 return ( <>

                            <div className="flex flex-col w-[11rem] justify-center items-center" key={index}>

                              <img className="w-full object-cover mix-blend-color-burn" src={src} alt={title}/>

                              <h1 className="text-3xl font-semibold">{title}</h1>

                              <p className="text-1xl text-left pl-4 text-[#343F5C]">{description}</p>

                              <button className={`${card.instock?"bg-blue-500 text-blue-50":"bg-pink-600 text-blue-50"} text-2xl text-green-950 w-fit h-fit shadow-md p-1 drop-shadow-sm accent-transparent rounded-lg self-start ml-4`}>



                                {card.instock?"instock":"out of stock"}

                              </button>



                            </div>

                            </>

                        )

})

            }

      </div>

    </>

  )

}

export default Cards

if you wanna add css with conditionally while fetching objects

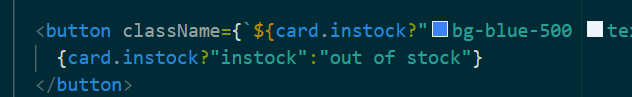


Use className={` …………… `} es6 way using js

${………….}..es6 js you can js anywhere

<button className={`${card.instock?"bg-blue-500 text-blue-50"**:**"bg-pink-600 text-blue-50"} text-2xl text-green-950 w-fit h-fit shadow-md p-1 drop-shadow-sm accent-transparent rounded-lg self-start ml-4`}>

Similary for button text





# EVENT HANDLING





Always do it inside array function

Many events are available use it

ex



# USE-STATE