

Introduction to react

Single Page Applications (SPA)

These are special web apps in which we only have one HTML page that is rendered. Now any content we have to render or any interactivity we have to add on this SPA is actually controlled end to end by JS. Even managing going from one page to another is handled completely by JS, we don't make multiple html pages for different pages of the app. Interesting part is for any interactivity or shifting from one page to other, the app doesn't refresh.

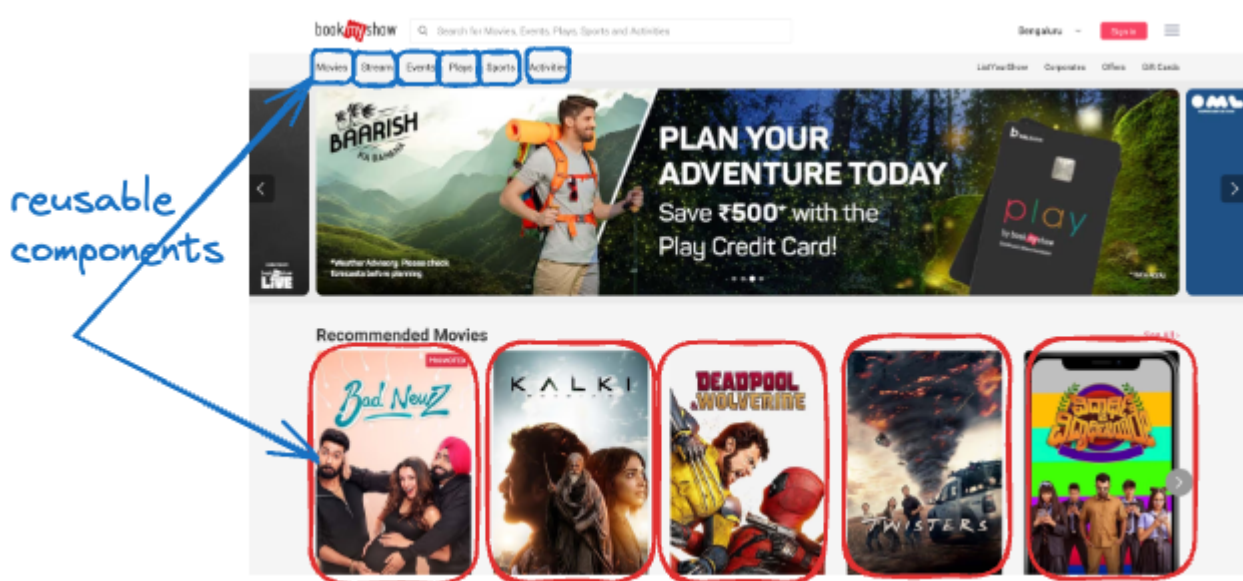
What is react and how it fits into the picture ?

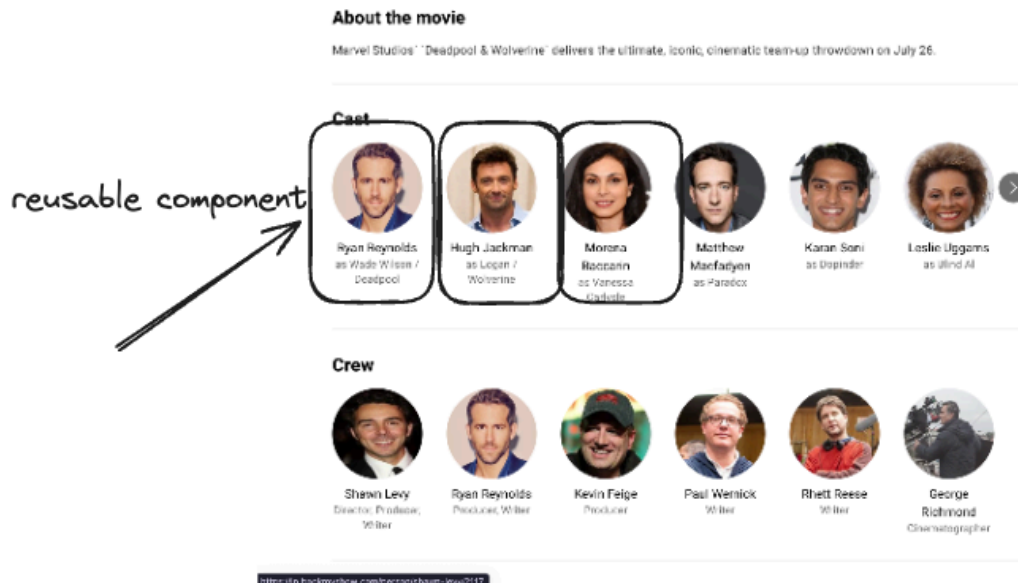
React is a JS library, it comes up with a bare minimum set of functionalities using which we can make modern SPA with ease. It doesn't bring a lot of conventions or baggage with it, and gives just the required functionalities for an SPA.

This is a problem of react also, that for a lot of things we have to rely on third party solutions / libraries.

React helps us to divide our UI into a set of reusable components.

So react says, that just identify any piece of UI which is reusable in your page and make it a **component**. Component is nothing but a reusable piece of UI implemented using react that we just plug and play anywhere in our UI. Components follow the DRY principle heavily.





There are more features that react brings into the picture:

- managing states in a component
- managing global states using context
- handling events
- conditional rendering
- If clubbed with a few libraries then can handle routing efficiently
- and more ...

How to get started with react ?

There are now a days a lot of tools which give you a ready to start react project out of the box.

- create-react-app
- NextJS
- Vite (getting a lot of popularity and pretty stable, fast as well)

Setup react project with Vite

To setup a new react project do the following:

1. Execute `npm create vite@latest`

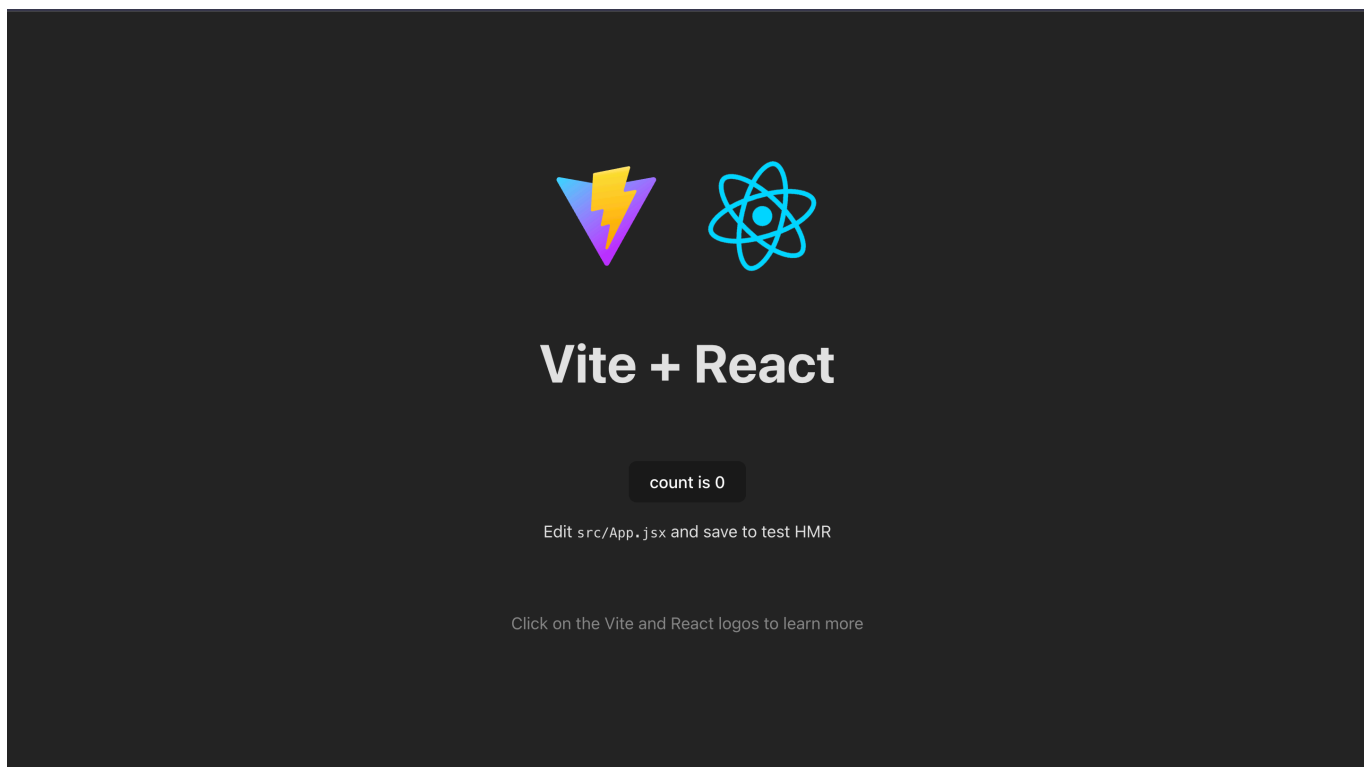
2. If after executing it asks for installation of `create-vite` say a yes

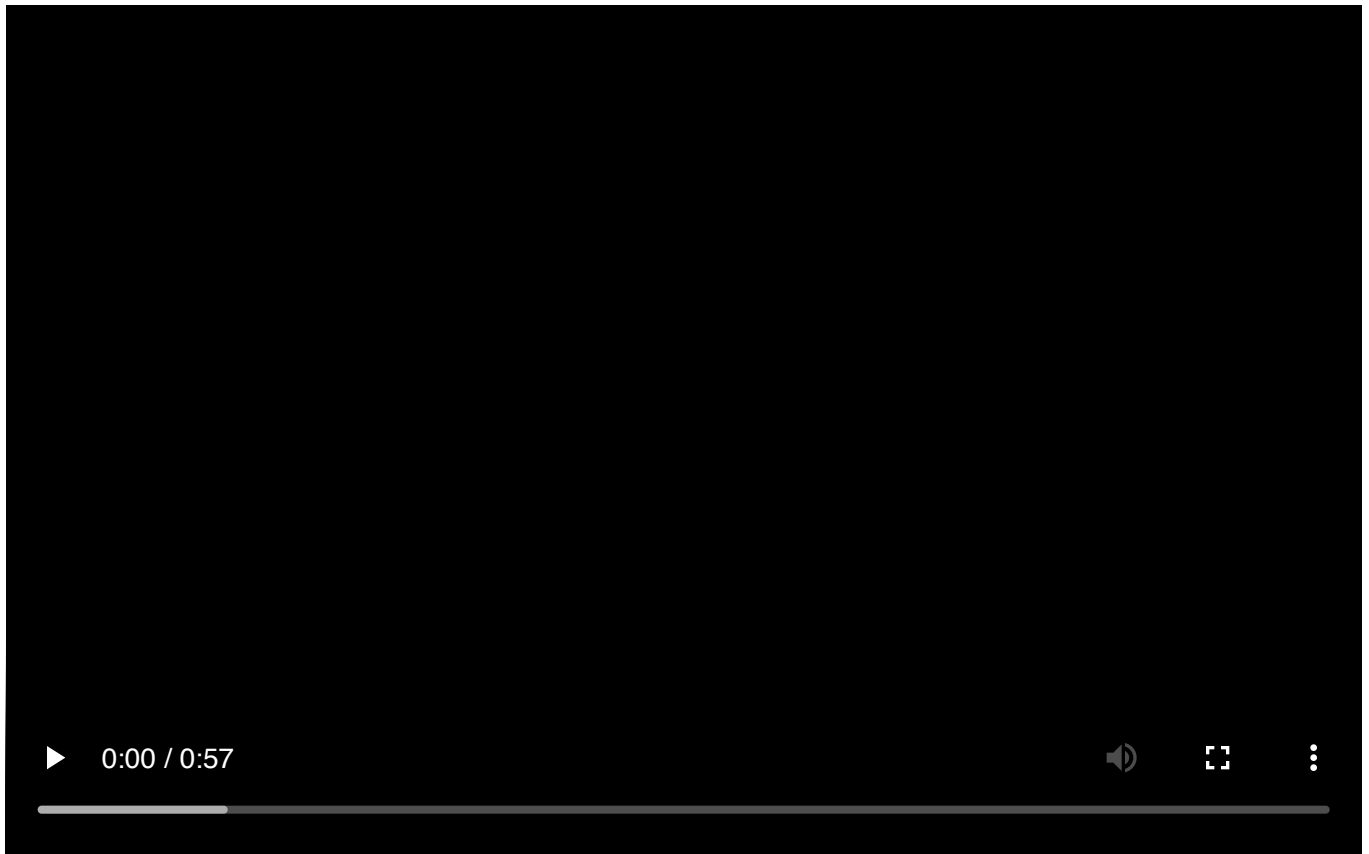
```
Need to install the following packages:
create-vite@5.4.0
Ok to proceed? (y) y
```

3. Then it will ask for the project name, give any name here. Then select the React library option with either JS or TS whatever you want.


```
✓ Project name: ... ReactStarterProject
✓ Package name: ... reactstarterproject
✓ Select a framework: > React
✓ Select a variant: > JavaScript
```


4. And that's it you have a brand new react project with you.
5. Go inside the react project by doing `cd <name of the project>`
6. Install dependencies using `npm install`.
7. Run the react server using `npm run dev`
8. Then go to your browser and open `localhost:5173` and you will get this:








Let's discuss the project structure:


>  node_modules


>  public


>  src


 .eslintrc.cjs


 .gitignore

 index.html

 package-lock.json

 package.json

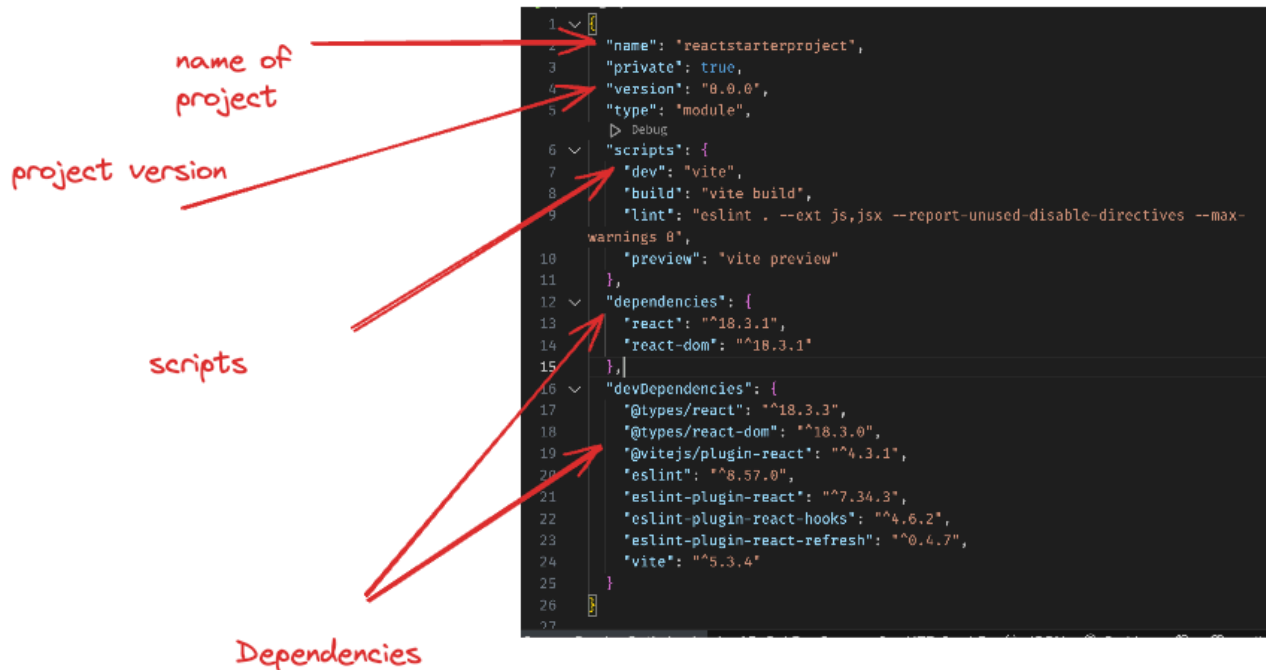
 README.md

 vite.config.js

Let's go over every folder and file one by one:

- **package.json** - This is indeed the most important file of the whole project, why ? It contains metadata about your project. It tells important details of the project like name of

the project, version of the project (self configurable), it also tells the dependencies which our project has, apart from that a couple of scripts are mentioned which are designated to do a particular job. For example: `dev` script written is executed when we execute `npm run dev`.



- **node_modules** : This folder contains the downloaded dependency codes. For example: we have eslint react etc as dependency of the project so we need to download them, so node_modules folder is the place where we download them.
- **package-lock.json**: Now the dependencies we installed also have inner dependencies, so this file keeps the version details of all the dependencies and inner dependencies. So, if we loose node_modules we can download the same version of everything because of package0lock.json .
- **gitignore**: This file contains all the files and folder we don't want to upload on github once we upload the project on github.
- **eslinttrc**: This file keeps the eslint configuration.
- **vite.config.js**: This file keeps the vite configurations.
- **public**: Any static file like pdfs, images etc goes here.
- **src**: Your actual react code is present.

To learn about pure react check this [link](#)