

Chapter 04 - Talk is cheap, show me the code

Building Food villa (similar to swiggy.com)

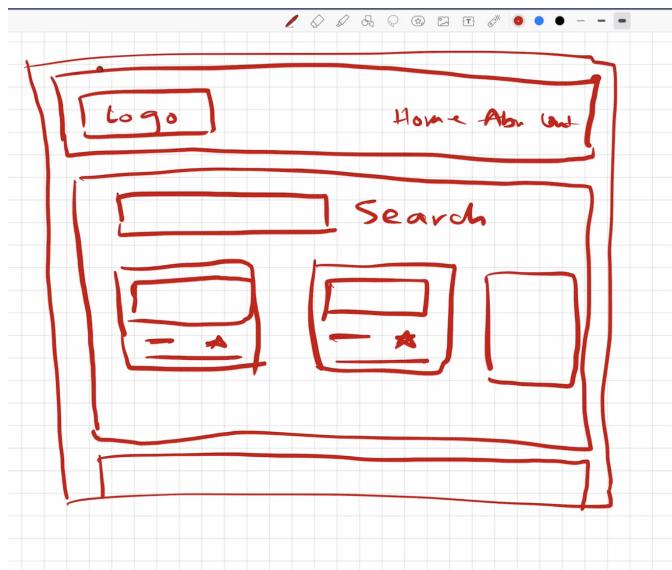
Akshay's Code:

Bitbucket

👉 <https://bitbucket.org/namastedev/namaste-react-live/src/21cb5ff0115f544b19edef37f813f8a5e4f3714b/>

▼ Steps

- First step - Planning (using good notes app)



- Prepare AppLayout

```
JS App.js > [o] AppLayout
19  };
20
21  const AppLayout = () =>{
22    return (
23      /**
24       Header
25       - Logo
26       - Nav Items(Right Side)
27       - Cart
28       Body
29       - Search bar
30       - RestrauntList
31         - RestaurantCard
32           - Image
33           - Name
34           - Rating
35           - Cusines
36       Footer
37         - links
38
39     */
40   );
41 }
```

- Code for each component

▼ React Fragments:

- Any piece of JSX component we write - there can be only one parent. But add extra div is ugly So we can use React.Fragment
- React.Fragment is a component exported by React

```
const jsx = (
  <React.Fragment>
    <h1>JSX</h1>
    <h1>Second JSX</h1>
  </div>
);
```

- This will not add additional node in DOM. Its like an empty tag. We can try this way (empty tag) too

```
// React.Fragment  
// JSX - one parent  
const jsx = (  
    
    <h1>JSX</h1>  
    <h1>Second JSX</h1>  
  </>  
)
```

▼ Styling in React

- Inline styling

```
const styleObj = {  
  backgroundColor: "red",  
};  
  
// Inline styling in React  
const jsx = (  
  <div style={styleObj}>  
    <h1>JSX</h1>  
    <h1>Second JSX</h1>  
  </div>  
)
```

- Other way is to use className and write style in index.css
- Another way: Other way using external library: Tailwind, Bootstrap, materialUI etc

▼ Config Driven UI

- UI is driven by config returned by backed api
Eg: How website look in different cities
Eg: swiggy shows carousel with offers specific to cities

```
//Config Driven UI
const config = [
  {
    You, now + Uncommitted changes
    type: "carousel",
    cards: [
      {
        offerName: "50% off"
      },
      {
        offerName: "No Delivery Charge"
      }
    ]
  }
]
```

▼ Props

- Props means properties - Passing data into component

```
const Body = () => {
  You, 1 minute ago • Uncommitt
  return (
    <div className="restaurant-list">
      <RestrauntCard restaurant={restrautList[0]} />
      <RestrauntCard restaurant={restrautList[1]} />
      <RestrauntCard restaurant={restrautList[2]} />
      <RestrauntCard restaurant={restrautList[3]} />
      <RestrauntCard restaurant={restrautList[4]} />
      <RestrauntCard restaurant={restrautList[5]} />
    </div>
  );
}
```

- How do pass props to Functional component?
 - Funtional component is just a Javascript function, nothing special. So we pass props as params and arguments.
 - Props is just a nomenclature in React, at the end its just a function parameter
 - Difference is that React wraps up all the properties into one object called props.**

```
<RestrauntCard restaurant={restrautList[0]} hello="world"/>
```

```
▼ {restaurant: {...}, hello: 'world'} ⓘ
  hello: "world"
  ▶ restaurant: {type: 'restaurant', data: {...}, subtype: 'basic'}
  ▶ [[Prototype]]: Object
```

- We need not call it as props, can be different name too

```
const RestrauntCard = (props) => {
  return (
    <div className="card">
      <img
        src={
          "https://res.cloudinary.com/swiggy/image/upload/fl_llossy,f_auto,q_60/restrautList[1].data?.cloudinaryImageId"
        }
      />
      <h2>{restrautList[0].data?.name}</h2>
      <h3>{restrautList[0].data?.cuisines.join(", ") }</h3>
      <h4>{restrautList[0].data?.lastMileTravelString} minutes</h4>
    </div>
  );
};
```

- We can also restructure the prop and use only needed field

```
const RestrauntCard = ({restaurant}) => {
  console.log(props);
  return (
    <div className="card">
      <img
```

- We can also use spread operator

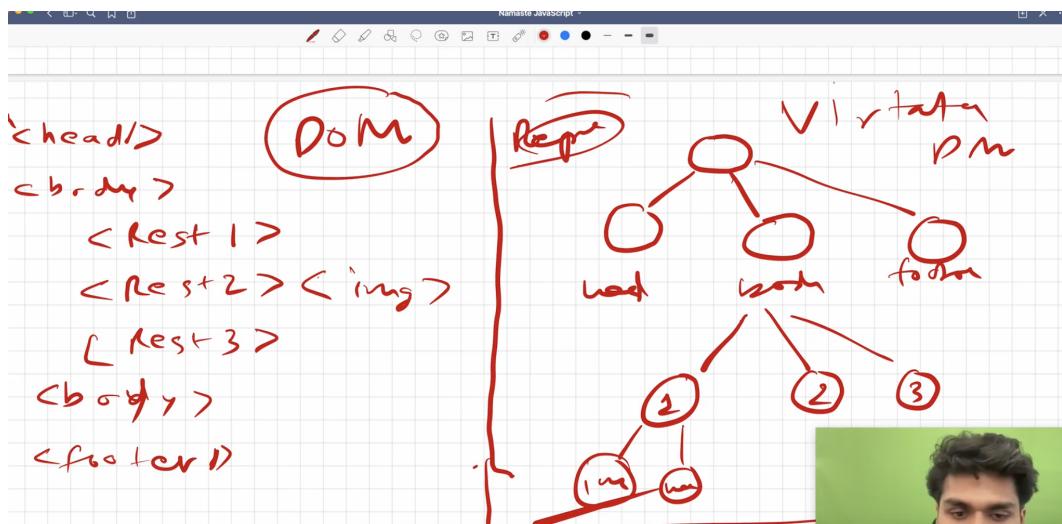
```
//props - properties
const Body = () => {
  return (
    <div className="restaurant-list">
      <RestrauntCard {...restrauntList[0].data} />
      <RestrauntCard {...restrauntList[1].data} />
      <RestrauntCard {...restrauntList[2].data} />
      <RestrauntCard {...restrauntList[3].data} />
      <RestrauntCard {...restrauntList[4].data} />
      <RestrauntCard {...restrauntList[5].data} />
    </div>
  );
};
```

- We can use map to render components many times in loop, passing props using map

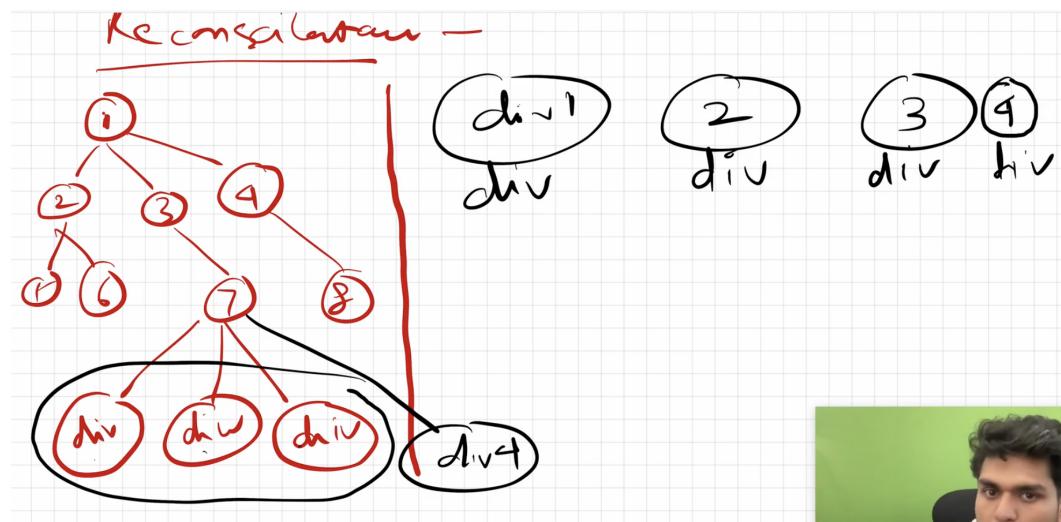
```
//props - properties
const Body = () => {
  return (
    <div className="restaurant-list">
      {restrauntList.map((restaurant) => {
        return <RestrauntCard {...restaurant.data} />;
      })}
    </div>
  );
};
```

▼ Virtual DOM, Reconciliation, Keys, React Fibre

- Virtual DOM is a software engineering concept - We keep a representation of DOM with us in our code - Virtual DOM. Read Doc
<https://legacy.reactjs.org/docs/faq-internals.html>



- Why do we need VDOM?
 - For Reconciliation in React
- What is Reconciliation?
 - Read DOC: <https://legacy.reactjs.org/docs/reconciliation.html>
 - Algorithm that React uses to diff one tree from other, and it determines what needs to be changes in UI & what does not need to be changed



- If 7 is changed, not everything needs to be rerendered. Only 7 and its children

- What happens if new div is added to 7 - React will know which div got changed, so it will rerender all. But If we add key - React will render only new div
- So give Key

```
const Body = () => {
  return (
    <div className="restaurant-list">
      {restrauntList.map((restaurant) => {
        return <RestrauntCard {...restaurant.data} key={restaurant.data.id} />;
      })}
    </div>
  );
}
```

- Why we should not use index as our key: Read Doc:
<https://robinpokorny.com/blog/index-as-a-key-is-an-anti-pattern/>

```
<div className="RestaurantList">
  {restrauntList.map((restaurant, index) => {
    return <RestrauntCard {...restaurant.data} key={[index]} />;
  })}
</div>
```

We don't recommend using indexes for keys if the order of items may change. This can negatively impact performance and may cause issues with component state. Check out Robin Pokorny's article for an in-depth explanation on the negative impacts of using an index as a key. If you choose not to assign an explicit key to list items then React will default to using indexes as keys.

Here is an in-depth explanation about why keys are necessary if you're interested in learning more.

If you no key - use index key

```
// no key (not acceptable) <<<<<< index key(last option) <<<< unqie key (best practice)
```

- Why React is fast?
 - React uses VDOM. Uses Reconciliation, which has diff algorithm and rerenders only the portions that changed in trees.
- What is React Fiber

- New reconciliation engine added in React 16. Its responsible for Diff
- <https://github.com/acdlite/react-fiber-architecture>

▼ Test data in code

- Its from swiggy website network tab
- Just fyi - These food images are from cloudfront - and we can just get Id from api & append

```
src={
  "https://res.cloudinary.com/swiggy/image/upload/fl_llossy,f_auto,q_auto,w_508,h_320,c_fill/" +
  restrautList[1].data?.cloudinaryImageId
}
```

- We can use firebase api too

Code Output:

localhost:1234

Home About Contact Cart

KFC American, Snacks, Biryani 6.1 kms minutes	Domnik Pizza Pizzas, Italian, Fast Food, Snacks, Beverages 0.6 kms minutes	FOOD PLANET RESTAURANT Indian, Chinese, Tandoor, Thalis, Fast Food 0.6 kms minutes	Burger King Burgers, American 6.3 kms minutes	Annapurna Andhra Mess South Indian, Biryani, North Indian 1.3 kms minutes	Uncle Ji Restaurant North Indian, Snacks, Beverages 0.8 kms minutes
--	---	---	--	--	--

Footer

