(lecture-4)

Date.....

- If we want to do inline Styling in JSX, we can do in form of JS object.

ex - style = {{background: #f0f0f0}}

Props → Properties

- It is something which we can pass to the component, if we want pass dynamically data to the component, we can pass it as prop.
- Functional component is normally a JS function & similarly props are just normal argument to our function.

Props ⟷ Argument

```
<RestaurantCard
  resName = "Meghana Food"
  Cuisine = "Asian"
/>
```

- React will basically take all these prop and it will wrap it inside an Object and it will pass over these as "props". This props will be an Object now.

Spiral

Date.....

```
const RestaurantCard = (props) => {
return (
<h1>{props.resName}</h1>
<h2>{props.Cuisine}</h2>
)
```

- Sometimes react developer instead of writing like this they prefer to write

```
const RestaurantCard = ({resName, Cuisine}) => {
return (
<h1>{resName}</h1>
<h2>{cuisine}</h2>
```

they destructure it on the fly → destructuring on the fly
↳ it JS, not react

- Real data comes in form of JSON.
- Config Driven UI →
- Our website is driven by data, this is known as config driven UI

Cynal

Date.....

- Controlling UI, how the UI looks like using data / using config and where that config comes from — Backend
  ↳ data changes on UI with diff. locations

```
Const RestaurantCard = ({resData}) => {
  return (
    <h3> {resData.data.name} </h3>
    <h4> {resData.data.cuisines.join(", ")} </h4>
    <h4> {resData.data.avgRating} </h4>
    <h4> {resData.data.deliveryTime} minutes </h4>
  )
```

→ But this is not a good code

```
}
```

```
const RestaurantCard = ({resData}) => {
  const {name, cuisines, avgRating, deliveryTime} =
    resData?.data
```

↳ object destructuring

This is optional chaining

```
  <h3> {cuisines.join(", ")} </h3>
  <h4> {name} </h4>
```

Spiral

Date.....

<h4> {avgRating} </h4>
<h4> {deliveryTime} </h4>

<RestaurantCard resData={resList[0]} />
<RestaurantCard resData={resList[1]} />
||
||
||

→ This is also not a good way to write.
• We are repeating it, what we can do is we can loop over this array and create this restaurant card one by one.

• For this we will write JS map function.

{ resList.map((restaurant) => (
<RestaurantCard resData={restaurant} />
))}

↑
This function is returning some piece of JSX.

Warning → Each child in a list should have a unique "key" prop.

Swiggy

Date.....

- It means that each of this list item

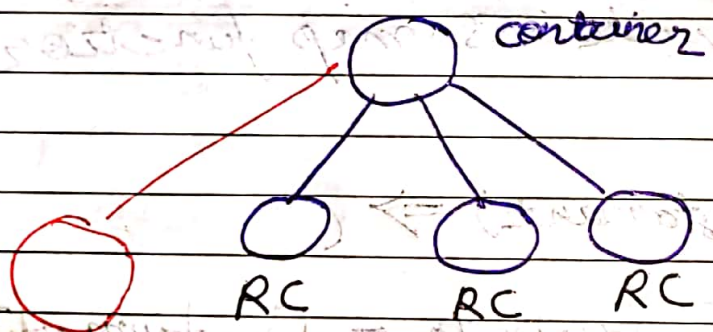
```
<RestaurantCard resData={restaurant} />
```

should be uniquely represented.

→ When we are looping onto any list we have to give key

```
<RestaurantCard key={restaurant.data.id}
resData={restaurant} />
```

Q.7 Why we need keys ?



- There are so many res-card in container, Suppose if there is a new res-card ~~came~~ which has come on the 1st place, so what react will do is, if we don't give id react will not know and react will re-render all these res-cards because react doesnot know which res-card is new, react can't uniquely identify this Restaurant cards

Spiral

Date.....

React will not know at what place new need to put that particular res-card first place, second place... etc.

- So react clears the container and re-renders all the 15 cards.
- Because react does n't know which is the new card which is added, it will treat all the res-card as same
- But if give each of them a unique id so now react exactly know that id 1, 2, 3 were already there, the new element come up in first place has id-xyz, so it will just render one restaurant over here
- It takes a big performance hit if we don't write keys.
- Some people also uses indexes as key.
- Whenever we are doing map or any for loop there is an index like 0, 1, 2, 3, 4.... and this index is the 2nd property of the map

```
{ reslist.map((restaurant, index) => (
    <RestaurantCard key={index} resData=
    {restaurant} />
))}
```

Spiral

Date.....

- This logically looks right because everytime there is a unique key for every iteration / res-card.
- But React itself says never use index as key
  → read Documentation
- using index as key is OK but not recommended
- Sometimes we don't have unique id for each card in data in that case we use index but of course it's not recommended

not using keys ↑↑↑↑↑ index as key ↑↑↑↑↑↑ unique id

Shiral