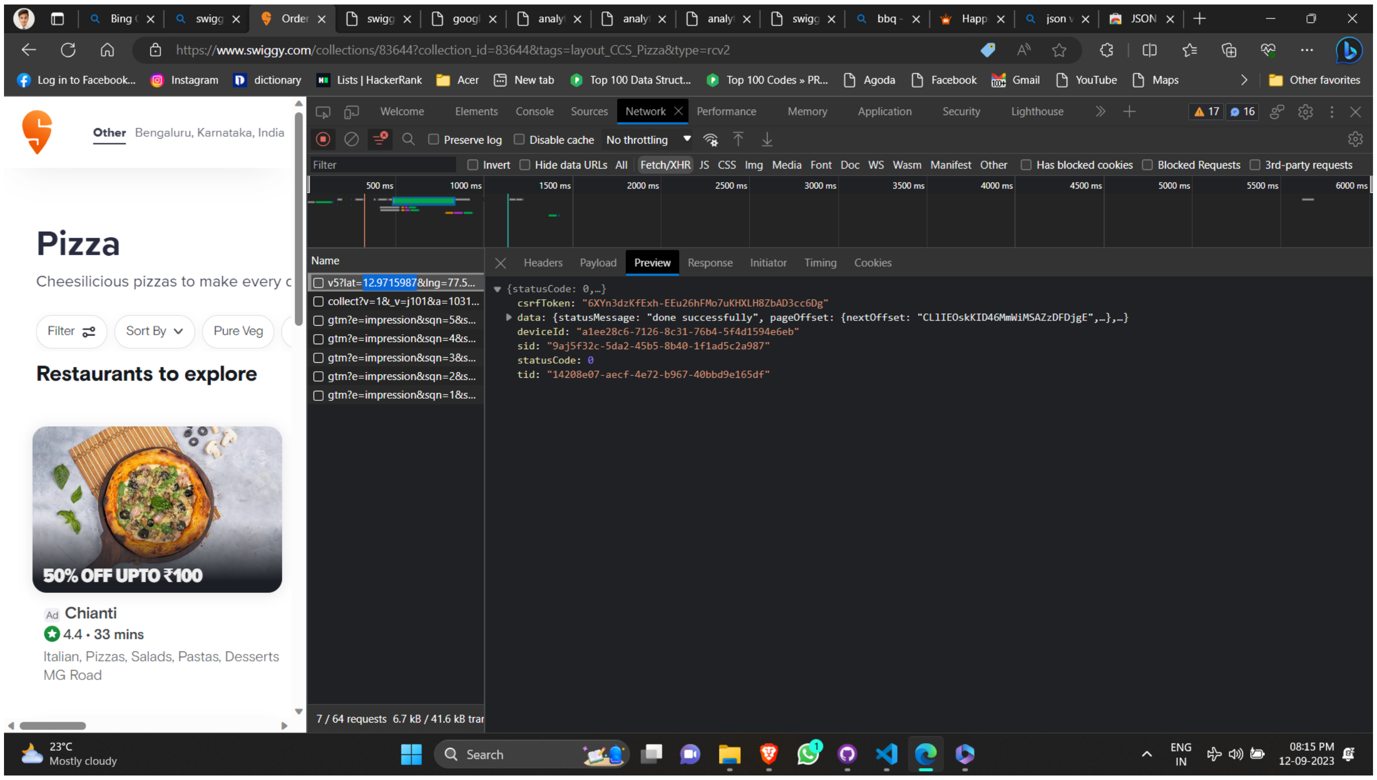
Config driven UI – System design concept

- You're planning to implement a **config-driven UI** for your food app.
- The UI will be driven by the data fetched from the backend.
- For instance, if KFC in Bangalore is offering a 50% discount on kebabs, this offer will be displayed in the app for users located in Bangalore.
- However, if KFC in Delhi does not have any such offers, users located in Delhi will not see this discount.
- This means you don't need to write separate UIs for Bangalore and Delhi. The UI will adapt based on the data it receives.
- In other words, the UI is completely dependent on the data. If the fetched data includes a discount, then and only then will it be displayed.



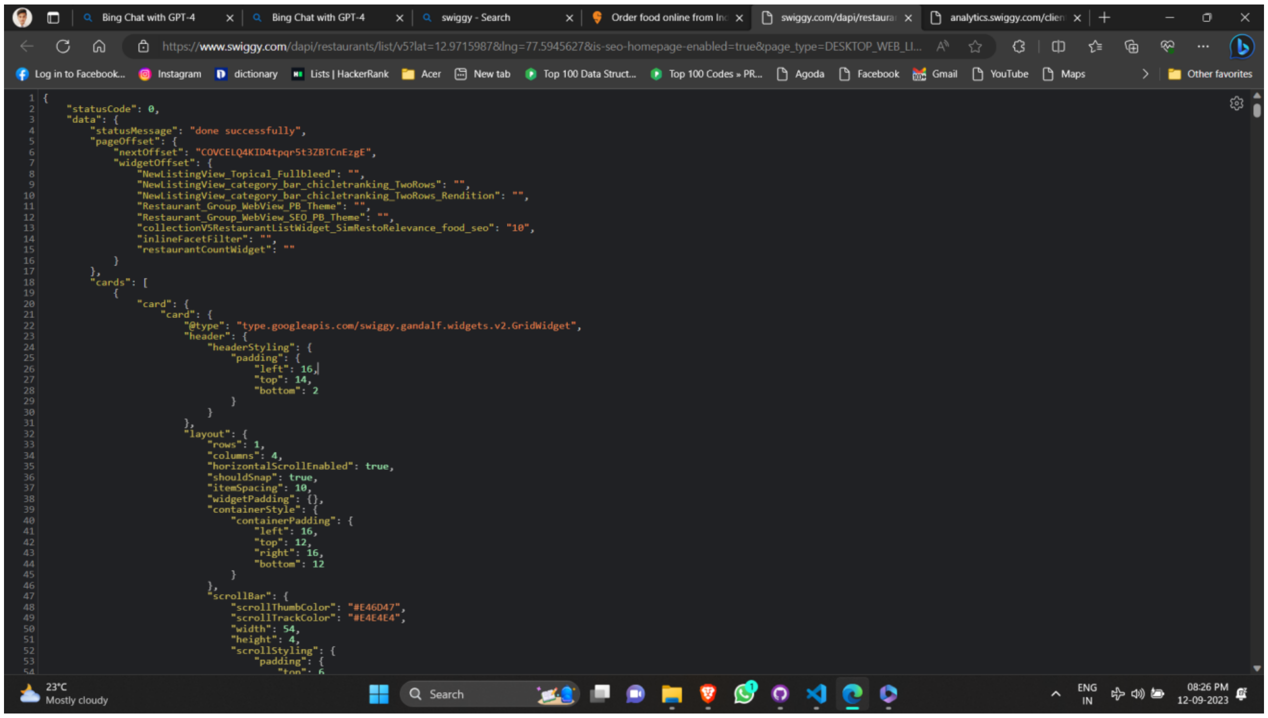
```
▼ {statusCode: 0,…}
    csrfToken: "6XYn3dzKfExh-EEu26hFMo7uKHXLH8ZbAD3cc6Dg"
  ▶ data: {statusMessage: "done successfully", pageOffset: {nextOffset: "CLlIEOskKID46MmWiMSAZzDFDjgE",…},…}
    deviceId: "a1ee28c6-7126-8c31-76b4-5f4d1594e6eb"
    sid: "9aj5f32c-5da2-45b5-8b40-1f1ad5c2a987"
    statusCode: 0
    tid: "14208e07-aecf-4e72-b967-40bbd9e165df"
```

It seems like you're exploring the Network tab in the browser's Developer Tools, which is a great way to understand how a website interacts with the server and fetches data.

When you refresh the page with the Network tab open, it records all network requests made by the page. The 'Fetch/XHR' filter shows only the requests made using Fetch or XMLHttpRequest, which are often used to load data in the background.

Opening one of these requests in a new tab can indeed show you the data returned by the server, often in JSON format. This can be very useful for understanding how the data for a website is structured and where it comes from.

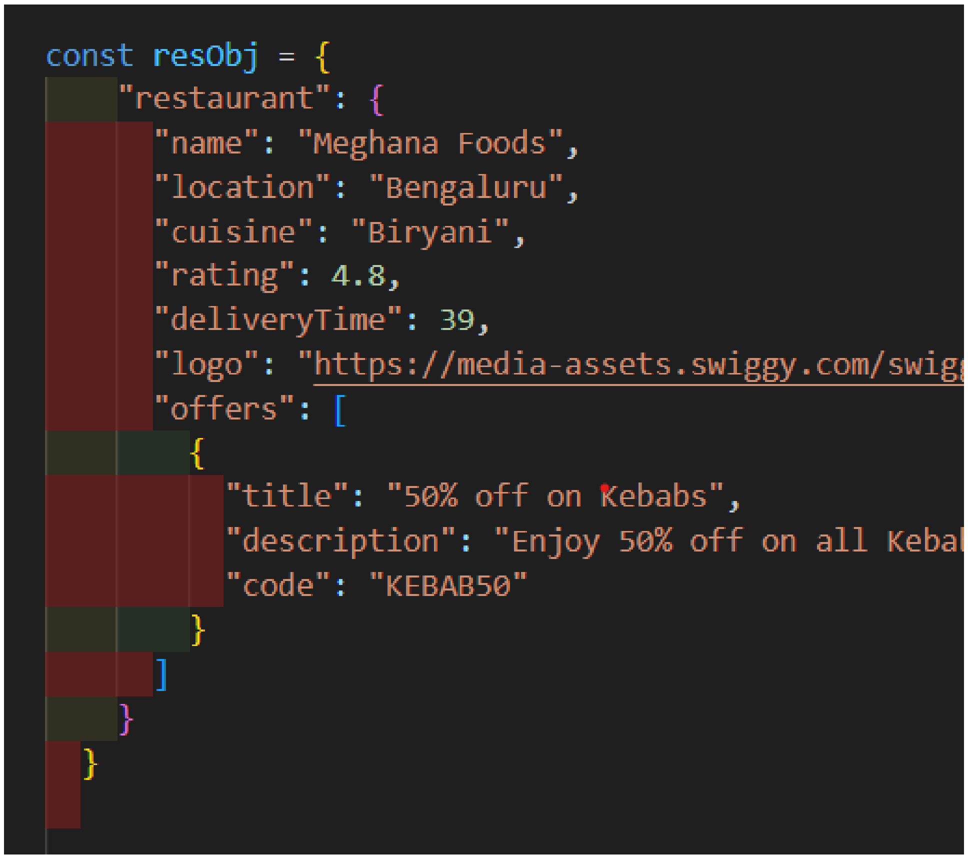
Data -


https://www.swiggy.com/dapi/restaurants/list/v5?lat=12.9715987&lng=77.5945627&is-seo-homepage-enabled=true&page_type=DESKTOP_WEB_LI...

```
{
    "statusCode": 0,
    "data": {
        "statusMessage": "done successfully",
        "pageOffset": {
            "nextOffset": "COVCELQ4KID4tpqr5t3ZBTCnEzgE",
            "widgetOffset": {
                "NewListingView_Topical_Fullbleed": "",
                "NewListingView_category_bar_chicletranking_TwoRows": "",
                "NewListingView_category_bar_chicletranking_TwoRows_Rendition": "",
                "Restaurant_Group_WebView_PB_Theme": "",
                "Restaurant_Group_WebView_SEO_PB_Theme": "",
                "collectionV5RestaurantListWidget_SimRestoRelevance_food_seo": "10",
                "inlineFacetFilter": "",
                "restaurantCountWidget": ""
            }
        },
        "cards": [
            {
                "card": {
                    "card": {
                        "@type": "type.googleapis.com/swiggy.gandalf.widgets.v2.GridWidget",
                        "header": {
                            "headerStyling": {
                                "padding": {
                                    "left": 16,
                                    "top": 14,
                                    "bottom": 2
                                }
                            }
                        },
                        "layout": {
                            "rows": 1,
                            "columns": 4,
                            "horizontalScrollEnabled": true,
                            "shouldSnap": true,
                            "itemSpacing": 10,
                            "widgetPadding": {},
                            "containerStyle": {
                                "containerPadding": {
                                    "left": 16,
                                    "top": 12,
                                    "right": 16,
                                    "bottom": 12
                                }
                            },
                            "scrollBar": {
                                "scrollThumbColor": "#E46D47",
                                "scrollTrackColor": "#E4E4E4",
                                "width": 54,
                                "height": 4,
                                "scrollStyling": {
                                    "padding": {
                                        "top": 6
```

This is how json data looks like.

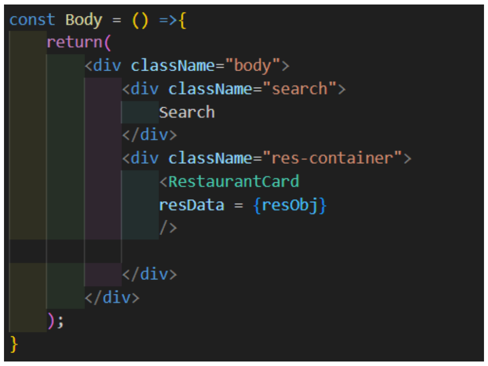
Now I want to use it in code.



```
const resObj = {
    "restaurant": {
        "name": "Meghana Foods",
        "location": "Bengaluru",
        "cuisine": "Biryani",
        "rating": 4.8,
        "deliveryTime": 39,
        "logo": "https://media-assets.swiggy.com/swig
        "offers": [
            {
                "title": "50% off on Kebabs",
                "description": "Enjoy 50% off on all Keba
                "code": "KEBAB50"
            }
        ]
    }
}
```

I created on resObj so that I can hold the data.

Inside my jsx , I was using each property name but instead now I will pass the resObject itself so that I can make use of props or array destructuring to get desired data.



```
const Body = () =>{
    return(
        <div className="body">
            <div className="search">
                Search
            </div>
            <div className="res-container">
                <RestaurantCard
                resData = {resObj}
                />

            </div>
        </div>
    );
}
```

Now that I have fiven array of objects in json.

```
const resObj = [
    {
      "name": "Meghana Foods",
      "location": "Bengaluru",
      "cuisine": "Biryani",
      "rating": 4.8,
      "deliveryTime": 39,
      "logo": "https://media-assets.swiggy.com/swiggy/image/upload/fl_lossy,f_auto,q_auto,w_660/usjgacltnt2rwla0hcpl",
      "offers": [
        {
          "title": "50% off on Kebabs",
          "description": "Enjoy 50% off on all Kebab orders today!",
          "code": "KEBAB50"
        }
      ]
    },
    {
      "name": "Sagar Ratna",
      "location": "Delhi",
      "cuisine": "South Indian",
      "rating": 4.5,
      "deliveryTime": 30,
      "logo": "https://media-assets.swiggy.com/swiggy/image/upload/fl_lossy,f_auto,q_auto,w_660/usjgacltnt2rwla0hcpl",
      "offers": []
    },
    {
      "name": "Pizza Hut",
      "location": "Mumbai",
      "cuisine": "Pizza",

      "rating": 4.2,

      "deliveryTime": 45,

      "logo": "https://media-assets.swiggy.com/swiggy/image/upload/fl_lossy,f_auto,q_auto,w_660/usjgacltnt2rwla0hcpl",

      "offers": [

        {

          "title": "Buy 1 Get 1 Free",

          "description": "Buy any medium pizza and get another medium pizza free!",

          "code": "BOGO"

        }

      ]

    }

  ]
```

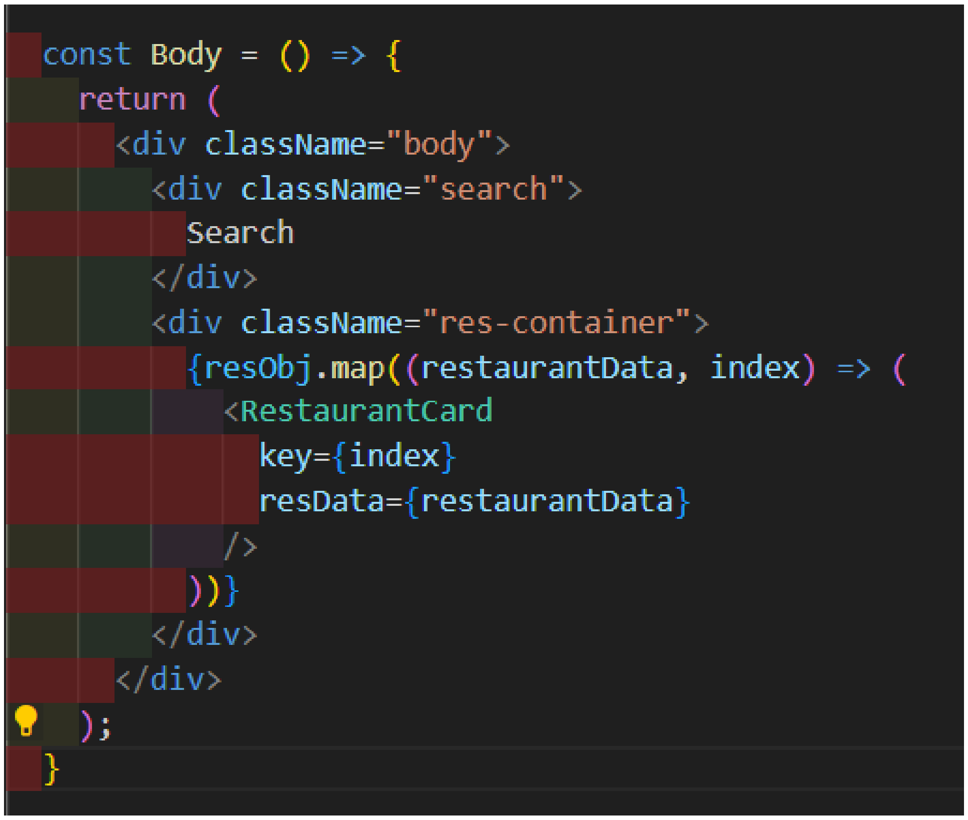
The `map` function is a built-in method in JavaScript that operates on arrays. It creates a new array by applying a function to each element of the original array.

In your code, `resObj` is an array of restaurant data. The `map` function is being used to create a new array of `RestaurantCard` components, one for each restaurant in the `resObj` array.

Here's what's happening in each part of the `map` function:

- `(restaurantData, index) => {...}`: This is the function that's applied to each element of the `resObj` array. It takes two parameters:
- `restaurantData`: The current element being processed in the array. In this case, it's an object containing data for a restaurant.
- `index`: The index of the current element being processed in the array. This is optional and is being used here to provide a unique `key` prop to each `RestaurantCard`.
- `<RestaurantCard key={index} resData={restaurantData} />`: This is what the function returns for each element in the `resObj` array. It's a `RestaurantCard` component with two props:
- `key`: A special string attribute that React uses to identify which items have changed, are added, or are removed. It's important to include keys when creating arrays of elements in React.
- `resData`: The data for one restaurant. This prop is used inside the `RestaurantCard` component to display the restaurant information.

I hope this helps! Let me know if you have any other questions or issues.

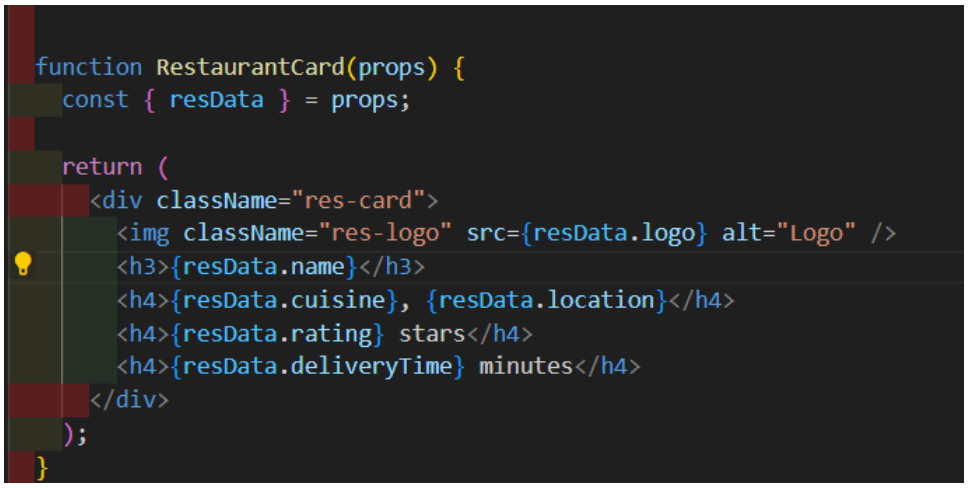


```
const Body = () => {
  return (
    <div className="body">
      <div className="search">
        Search
      </div>
      <div className="res-container">
        {resObj.map((restaurantData, index) => (
          <RestaurantCard
            key={index}
            resData={restaurantData}
          />
        ))}
      </div>
    </div>
  );
}
```

Here I have used map to get each object in array (each obj represent once restorant):

After that we can see the map function creates a jsx obj for each restorant and we know map returns an array

That will be passed as props in restorant cars functional component.



```
function RestaurantCard(props) {
  const { resData } = props;

  return (
    <div className="res-card">
      <img className="res-logo" src={resData.logo} alt="Logo" />
      <h3>{resData.name}</h3>
      <h4>{resData.cuisine}, {resData.location}</h4>
      <h4>{resData.rating} stars</h4>
      <h4>{resData.deliveryTime} minutes</h4>
    </div>
  );
}
```

Also you noticed.

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

You’re absolutely correct! In React, the key prop is used to uniquely identify each element in an array or list. It’s important for helping React understand which items have changed, are added, or are removed.

Keys should be given to the elements inside an array to give the elements a stable identity. The best way to pick a key is to use a string that uniquely identifies a list item among its siblings. Most often you would use IDs from your data as keys.

When you don’t provide a key prop, React will give a warning in the console because it wants to encourage developers to provide keys for list items for the reasons you mentioned. Without keys, React might need to re-render more components than necessary when the list changes, which could lead to performance issues.

So while your code will still work without keys (hence it’s a warning and not an error), it’s considered best practice to include them. It’s one of those things that can help keep your application efficient and bug-free in the long run.

Some people use index to key. You can directly give give restaurant.id also but its best to have index as key.

But react itself says never use index as key. Its not recommended . Its an antipattern.

Not using key – not acceptable

Using index as keys – not recommended

Use unique key if present , if not use index.