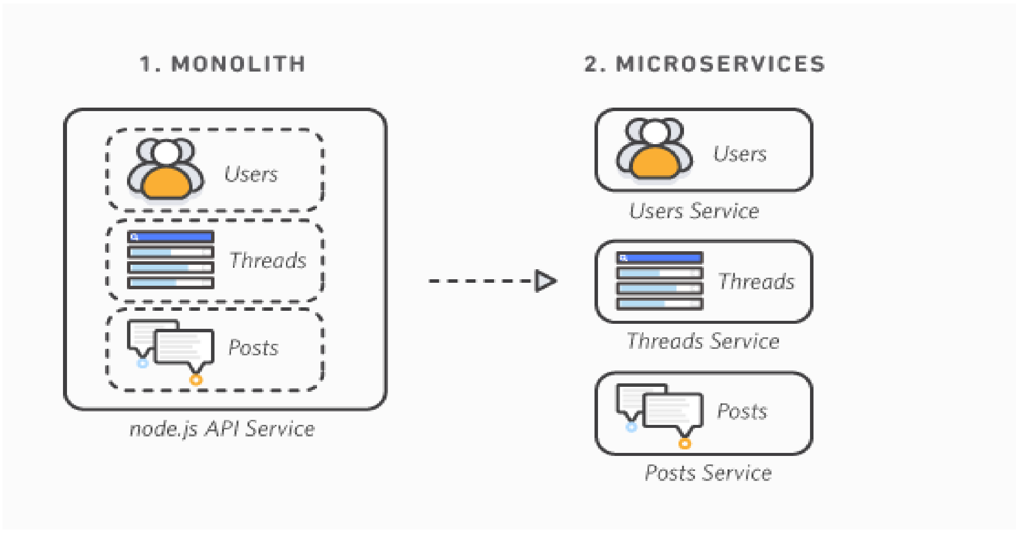
- **What is a Microservice?**
  A microservices architecture is a type of application architecture where the application is developed as a collection of services. It provides the framework to develop, deploy, and maintain microservices architecture diagrams and services independently.

- **What is Monolith architecture?**
  Monolithic architecture is a traditional software design approach where all the components of an application are tightly coupled and deployed as a single unit.

- **What is the difference between Monolith and Microservice?**
  Monolithic applications typically consist of a client-side UI, a database, and a server-side application. Developers build all of these modules on a single code base.

  On the other hand, in a distributed architecture, each microservice works to accomplish a single feature or business logic. Instead of exchanging data within the same code base, microservices communicate with an API.

- **Why do we need a useEffect Hook?**

  The `useEffect` Hook in React is essential for managing side effects within functional components. Side effects are actions that occur outside of rendering the component, such as:

  Fetching data: Making API calls to retrieve data from external sources.

- **What is Optional Chaining?**

  Optional Chaining is a concise and safe way to access properties of an object that might be null or undefined.

  Here's how it works:

  - **Syntax:** `?.`
  - **Purpose:** To safely access nested properties without causing errors.

- **What is Shimmer UI?**

  Shimmer UI is a visual technique used in user interfaces to provide a sense of progress and engagement while content is loading. It replaces the actual content with a placeholder that mimics its shape and layout, often using a gradient animation that creates a shimmering effect.

  Implementation of Shimmer UI:

  Shimmer UI can be implemented using various techniques, including:

  - **CSS Animations:** Using CSS gradients and animations to create the shimmering effect.
  - **JavaScript Libraries:** Utilizing libraries like React Shimmer or Vue Shimmer for easier implementation.

- **What is the difference between JS expression and JS statement**

  JavaScript Expressions

- Produce a value: Expressions are pieces of code that evaluate to a single value.
- Examples:
  - `2 + 3` (evaluates to 5)
  - `'hello' + ' world'` (evaluates to 'hello world')
  - `x * y` (evaluates to the product of x and y)
  - `true || false` (evaluates to true)

  JavaScript Statements

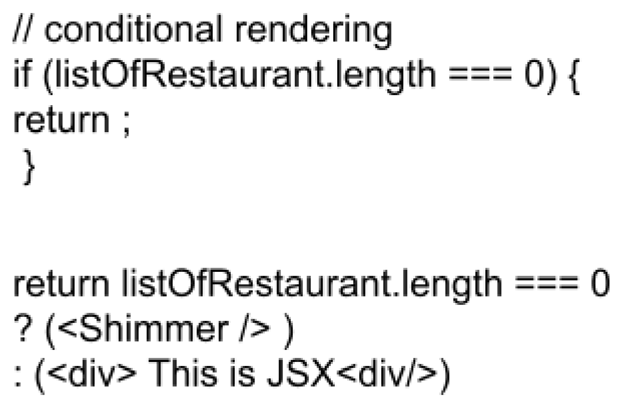
- **Perform actions:** Statements are instructions that tell the JavaScript engine to do something.
- **Complete units of execution:** They often end with a semicolon (`;`).
- **Examples:**
  - `let x = 5;` (declares a variable and assigns a value)
  - `console.log('Hello, world!');` (prints a message to the console)

   - `if (x > 0) { ... }` (conditionally executes a block of code)
   - `for (let i = 0; i < 10; i++) { ... }` (loops through a block of code)

- **What is Conditional Rendering, explain with a code example**
  Conditional Rendering in React allows you to display different parts of your UI based on certain conditions. This is crucial for creating dynamic and interactive user interfaces.

  How it works:

- Conditional statements: Utilize `if`, `else if`, and `else` statements to determine which parts of the UI should be displayed.
- Ternary operator: A concise way to conditionally render elements using the `condition ? true : false` syntax.



```
// conditional rendering
if (listOfRestaurant.length === 0) {
return ;
 }


return listOfRestaurant.length === 0
? (<Shimmer /> )
: (<div> This is JSX<div/>)
```

- **What is CORS?**
  CORS (Cross-Origin Resource Sharing)
  is a mechanism that allows web pages to make requests to servers on a different domain than the one that served the web page.

- **What is async and await?**

  Async:

   - Declares a function as asynchronous.
   - An async function always returns a promise.
   - Even if the function doesn't explicitly return a value, it implicitly returns a promise that resolves to `undefined`.

Await:

- Used only within an `async` function.
- Pauses the execution of the async function until the promise it's waiting for resolves.
- Once the promise resolves, `await` returns the resolved value.

- **What is the use of `const json = await data.json();` in getRestaurants()**

  The line of code `const json = await data.json();` is a crucial part of working with data fetched from a network in JavaScript, particularly when using the `fetch` API. Let's break it down:

- **`data`:** This variable likely holds a `Response` object. The `Response` object is returned by the `fetch()` method after successfully fetching data from a URL.
- **`.json()`:** This is a method of the `Response` object. It reads the response body as JSON (JavaScript Object Notation) and returns a promise that resolves to the parsed JavaScript object.
- **`await`:** This keyword can only be used within an `async` function. It pauses the execution of the `async` function until the promise returned by `data.json()` resolves. In other words, it waits for the JSON data to be parsed before proceeding.
- **`const json`:** This declares a constant variable named `json` to store the parsed JavaScript object that is resolved by the promise.