Chapter 06 : Exploring the world

Q: What is a microservice?

A: it is an architectural and organizational approach to software development where software is composed of small independent services like database, server or a UI of the application, that communicate over well-defined APIs. These services are owned by small, self-contained teams. Microservices architectures make applications easier to scale and faster to develop, enabling innovation and accelerating time-to-market for new features. means we are dividing software into small, well-defined modules enables teams to use functions for multiple purposes.

* Benefits of Microservices:
  + Flexible Scaling
  + Easy Deployment
  + Technological Freedom
  + Reusable Code
  + Resilience

Q: What is monolith architecture ?

A: In this architecture all processes are tightly coupled and run as a single service. This means that if one process of the application experiences a spike in demand, the entire architecture must be scaled. Adding or improving a monolithic application’s features becomes more complex as the code base grows. This complexity limits experimentation and makes it difficult to implement new ideas. Monolithic architectures add risk for application availability because many dependent and tightly coupled processes increase the impact of a single process failure.

Q: Why do we need a useEffect Hook?

A: useEffect is hook provided by React. lets us to perform side effects in functional components. It accepts two arguments - a callback function that you want to call, and array of dependencies.

Initially useEffect will run after the component is rendered.

If dependency array of useEffect is empty then it will run only once after component render.

If if have state variables in dependency array , it will always run when the state variable is changed.

useEffect(() => {

setCurrentState(true)

}, [currentState] )

Q: What is optional chaining?

A: optional Chaining (?.) is good way of accessing the object keys, it prevents the application from being crashed if the key that we are trying to access is not present. If the key is not present then instead of a throwing key error, it returns undefined.

Q: What is shimmer UI

A: A Shimmer UI resembles the page's actual UI, so users will understand how quickly the web or mobile app will load even before the content has shown up. It gives people an idea of what's about to come and what's happening (while UI currently loading) when a page full of content/data takes more than 3 - 5 seconds to load. Shimmer UI is a great way for loading the applications. Instead of showing a loading circle we can design a shimmer UI for our application that is good for user experience.

Q: What is difference between js expression and js statement

A:

Expression : unit of code that can be evaluated to a value

1 + 2 // expresses

"foo".toUpperCase() // expresses 'FOO'

console.log(2) // logs '2'

isTrue ? true : false // returns us a true or false value based on isTrue value

Statement : is an instruction to perform a specific action. Such actions include creating a variable or a function, looping through an array of elements, evaluating code based on a specific condition etc

let x; // variable declaration

if () { } // if condition

Q:What is Conditional Rendering? explain with a code example.

A: When you render UI based on some condition just like we have if to decide if something will be rendered on UI or not.

// Using Ternary operator as a shorthand way or writing an if-else statement

{isLoggedIn ? (return <UserGreeting />) : (return <GuestGreeting />)};

// Using an if…else Statement

{

(if (isLoggedIn) {

return <UserGreeting />;

}else {

return <GuestGreeting />;

})

}

// Using Logical &&

{isLoggedIn && <button>Logout</button>}

## 

## 

## Q: What is CORS?

A: Cross-Origin Resource Sharing (CORS) is an HTTP-header based mechanism that allows a server to indicate any origins (domain, scheme, or port) other than its own from which a browser should permit loading resources. CORS defines a way in which a browser and server can interact to determine whether it is safe to allow the cross-origin request.

Q: What is async await

A:

**Async:** It simply allows us to write **promises-based** . Async functions will always return a value. It makes sure that a promise is returned and if it is not returned then JavaScript automatically wraps it in a promise which is resolved with its value.

const getData = async() => {

var data = "Hello World";

return data;

}

getData().then(data => console.log(data));

Await: Await function is used to wait for the promise. It could be used within the async block only. It makes the code wait until the promise returns a result. It only makes the async block wait.

const getData = async() => {

var y = await "Hello World";

console.log(y);

}

console.log(1);

getData();

console.log(2);

## Q: What is the use of const json = await data.json(); in getRestaurants()?

## A: The data object, returned by the await fetch(), is a generic placeholder for multiple data formats. so we can extract the JSON object from a fetch response by using await data.json(). data.json() is a method on the data object that lets you extract a JSON object from the data or response. The method returns a promise because we have used await keyword. so data.json() returns a promise resolved to a JSON object