1.Khái niệm ReactJS?

* ReactJS là một thư viện JavaScript mã nguồn mở được phát triển bởi Facebook. Nó được sử dụng để xây dựng các ứng dụng web đơn trang (Single Page Applications - SPA) và các giao diện người dùng (User Interfaces - UI) tương tác.
* ReactJS cho phép phát triển ứng dụng web dễ dàng hơn bằng cách sử dụng các thành phần (components) tái sử dụng và quản lý trạng thái (state) của các thành phần đó.
* ReactJS được xây dựng dựa trên cơ sở của ngôn ngữ lập trình JavaScript, cho phép các nhà phát triển sử dụng nó với các thư viện khác như Redux, React Router, hay Axios để xây dựng các ứng dụng phức tạp.
* Ngoài ra, ReactJS cũng cho phép tương tác với các thư viện khác bằng cách sử dụng API.

2. ES6 - Từ khóa let và const?

Trong JavaScript ES6, let và const là từ khóa được sử dụng để khai báo biến.

**Let** được sử dụng để khai báo một biến trong một block scope, có nghĩa là biến chỉ có thể truy cập được trong phạm vi của khối lệnh mà nó được khai báo.

* function foo() {

let x = 10;

if (true) {

let x = 20;

console.log(x); // 20}

console.log(x); // 10}

**Const** cũng được sử dụng để khai báo biến, nhưng nó chỉ có thể được gán một lần và không thể được gán lại giá trị khác. Biến được khai báo bằng từ khóa const cũng có phạm vi của nó là block scope.

* function bar() {

const y = 10;

y = 20; // lỗi

const z; // lỗi}

3. ES6 - khái niệm arrow function?

Arrow function (còn được gọi là "fat arrow function" hoặc "lambda function" trong một số ngôn ngữ lập trình khác) là một tính năng mới được giới thiệu trong ECMAScript 6 (ES6) của JavaScript.

Arrow function cung cấp một cú pháp rút gọn cho khai báo hàm bằng từ khóa **function**, giúp cho việc viết mã nguồn trở nên ngắn gọn và dễ đọc hơn. Ví dụ, hãy xem xét một hàm thông thường như sau:

* function add(a, b) {

return a + b;}

Cách viết bằng arrow function sẽ giúp loại bỏ từ khóa function, giảm thiểu độ dài code và giúp cho code trở nên dễ đọc hơn:

* const add = (a, b) => a + b;

Arrow function có thể có hoặc không có tham số, cũng như có thể trả về giá trị hoặc không trả về giá trị. Nếu hàm chỉ có một tham số, bạn có thể bỏ qua cặp dấu ngoặc đơn. Ví dụ:

* const double = num => num \* 2;

Nếu hàm không có tham số, bạn cần sử dụng cặp dấu ngoặc đơn trống để biểu thị danh sách tham số:

* const sayHello = () => console.log("Hello!");

Arrow function có thể được sử dụng trong nhiều tình huống khác nhau, bao gồm việc truyền hàm vào các phương thức như Array.map(), Array.filter(), Array.reduce(), v.v. để xử lý các mảng.

4. ES6 - khái niệm Import và export modules?

**Import**: Là cách để import các module được export từ các file khác để sử dụng trong file hiện tại. Để import một module, ta sử dụng từ khóa "import" cùng với tên module cần import và đường dẫn đến file chứa module đó. Ví dụ:

* import { functionName } from './module.js';

**Export**: Là cách để export các module từ file hiện tại để có thể import và sử dụng trong các file khác. Để export một module, ta sử dụng từ khóa "export" trước khai báo tên module cần export. Ví dụ:

* export function functionName() { // code here }

Ngoài ra, ta còn có thể sử dụng default export bằng cách export một module với từ khóa "export default". Khi import, ta không cần phải sử dụng dấu ngoặc nhọn để chỉ định tên module, mà có thể sử dụng bất kỳ tên nào để đặt tên cho module đó. Ví dụ:

// export default module

* export default function() { // code here }

// import default module

* import anyName from './module.js';

5. ES6 - Tìm hiểu về Class

* Trong ES6 (ECMAScript 2015) và các phiên bản mới hơn của JavaScript, bạn có thể sử dụng từ khóa "class" để định nghĩa một lớp.Là 1 bảng thiết kế của đối tượng, là sự trừu tượng hoá đối tượng
* Một lớp có thể chứa các phương thức (method) và thuộc tính (property) để định nghĩa hành vi và trạng thái của các đối tượng của lớp.
* Khi bạn tạo một đối tượng từ một lớp, đối tượng đó được gọi là một thể hiện (instance) của lớp đó.
* Class cũng hỗ trợ tính kế thừa, cho phép bạn định nghĩa một lớp con (subclass) dựa trên một lớp cha (superclass). Để kế thừa từ một lớp cha, bạn sử dụng từ khóa "extends" và nó gọi phương thức constructor của lớp cha bằng từ khóa "super".

6. ES6 - Tìm hiểu khai báo Class, Methods và Properties?

7. ES6 - Toán tử Spread và Rest?

Spread:

1. Toán tử spread được biểu thị bằng dấu ba chấm (**...**) và được sử dụng để trải một mảng thành các phần tử riêng lẻ.

* const arr1 = [1, 2, 3];

const arr2 = [4, 5, 6];

const arr3 = [...arr1, ...arr2]; // [1, 2, 3, 4, 5, 6]

1. Toán tử Spread còn được sử dụng để sao chép một mảng hoặc đối tượng.
2. Toán tử spread cũng có thể được sử dụng để truyền các phần tử của một mảng vào một hàm mà yêu cầu các đối số được truyền dưới dạng các tham số riêng lẻ.

* function sum(a, b, c) {

return a + b + c;}

const arr = [1, 2, 3];

const result = sum(...arr); // 6

**Rest**

1. Toán tử rest cũng được biểu thị bằng dấu ba chấm (...) và được sử dụng để gom các tham số thành một mảng.

* function myFunction(...args) {

console.log(args);}

myFunction(1, 2, 3, 4); // [1, 2, 3, 4]

1. Toán tử rest cũng có thể được sử dụng để định nghĩa một hàm với số lượng đối số không xác định trước đó.

* function sum(...args) {

return args.reduce((acc, val) => acc + val);}

const result = sum(1, 2, 3, 4); // 10

8.  ES6 - Kỹ thuật desctructuring?

* Cho phép bạn rút ngắn mã và truy cập các giá trị trong các đối tượng hoặc mảng. Nó cho phép bạn tách các phần tử của một mảng hoặc thuộc tính của một đối tượng ra thành các biến riêng lẻ.
* const person = { firstName: 'John', lastName: 'Doe', age: 25};

const { firstName, lastName, age } = person;

console.log(firstName); // output: John

console.log(lastName); // output: Doe

console.log(age); // output: 25

* const numbers = [1, 2, 3, 4, 5];

const [first, second, ...rest] = numbers;

console.log(first); // output: 1

console.log(second); // output: 2

console.log(rest); // output: [3, 4, 5]

9. ES6 - Khái niệm Primitive types và reference types?

* **Primitive types** là kiểu dữ liệu không thể thay đổi (immutable), nghĩa là giá trị của chúng không thể thay đổi sau khi được tạo ra. Có sáu kiểu nguyên thủy trong JavaScript: string (chuỗi), number (số), boolean (luận lý), null (giá trị null), undefined (giá trị không xác định), symbol (ký hiệu)
* **Reference types** là kiểu dữ liệu có thể thay đổi (mutable), nghĩa là giá trị của chúng có thể thay đổi sau khi được tạo ra. Reference types là các đối tượng được tạo ra bằng từ khóa new, chẳng hạn như mảng, hàm và đối tượng.
* Một khác biệt quan trọng giữa primitive types và reference types là kiểu nguyên thủy được truyền theo giá trị (passed by value), trong khi kiểu tham chiếu được truyền theo tham chiếu (passed by reference). Khi một giá trị nguyên thủy được gán cho một biến, một bản sao của giá trị được tạo ra và lưu trữ trong biến. Tuy nhiên, khi một kiểu tham chiếu được gán cho một biến, chỉ một tham chiếu đến đối tượng được lưu trữ trong biến, không phải một bản sao của đối tượng.

10. ES6 - Khái niệm Array functions?

1. Array functions là một tập hợp các hàm được cung cấp bởi ES6 để làm việc với các mảng trong JavaScript. Các hàm này giúp bạn thực hiện các tác vụ phổ biến trên mảng như lọc, sắp xếp, ánh xạ, giảm, phù hợp và nhiều hơn nữa một cách dễ dàng và rõ ràng hơn.
2. Một số Array functions phổ biến trong ES6 bao gồm:

* map() - tạo một mảng mới bằng cách thực hiện một hành động trên mỗi phần tử của mảng ban đầu.
* filter() - lọc các phần tử của mảng ban đầu dựa trên một điều kiện nhất định và trả về một mảng mới chỉ chứa các phần tử thoả mãn điều kiện đó.
* reduce() - tính tổng các phần tử trong mảng ban đầu hoặc thực hiện một hành động phức tạp hơn trên các phần tử của mảng.
* forEach() - lặp lại mỗi phần tử của mảng và thực hiện một hành động trên từng phần tử.
* find() - trả về phần tử đầu tiên của mảng thoả mãn điều kiện được chỉ định.
* some() - kiểm tra xem có phần tử nào trong mảng thoả mãn điều kiện được chỉ định hay không.
* every() - kiểm tra xem tất cả các phần tử trong mảng có thoả mãn điều kiện được chỉ định hay không.

20. Toolchain?

1. **Toolchain** là chuỗi công cụ tích hợp trong quá trình phát triển giúp đem đến trải nghiệm tốt nhất cho người dùng và nhà phát triển như:

* Mở rộng quy mô cho nhiều file và component.
* Sử dụng thư viện của bên thứ ba từ npm.
* Phát hiện sớm các lỗi thường gặp.
* Chỉnh sửa trực tiếp CSS và JS đang được phát triển.
* Tối ưu hóa đầu ra cho production.

1. **Một số toolchain phổ biến**

* [npm](https://www.npmjs.com/) tên đầy đủ là Node Package Manager, một trình quản lý các thư viện trong môi trường Node.js. Các thư viện được đóng gói dưới dạng package và được lập trình viên đóng góp, chia sẻ trên npm registry (cơ sở dữ liệu của npm)
* Cú Pháp: npm instal (tên thư viện muốn cài)
* npx (Node Package Execute), là một công cụ để chạy package mà không cần phải cài đặt. Hãy xem một ví dụ dưới đây để hình dung được lợi ích mà npx đem lại cho chúng ta:
* Cú pháp: npx create-react-app app-name

21.Khái niệm Client Side Rendering và Server Side Rendering?

* **Server Side Rendering:** Bạn request một trang web, server xử lý nội dung thành HTML, return lại cho browser hiển thị lại lên màn hình.=> nhược điểm:
* Trang web phải xử lý lại hoàn toàn và load lại từ đầu nếu chỉ có một thay đổi nhỏ trong nội dung. (Ví dụ tiêu đề thay đổi …)
* Việc xử lý nội dung HTML khiến hao tốn tài nguyên server, gây chậm trễ khi xử lý các request khác.
* Lượng request lên server rất nhiều, do mọi tác vụ đều phải xử lý lại trên server và render lại HTML
* TTFB (Time To First Byte) cao do cần phải xử lý trên server.
* **Client-side rendering (CSR)** đề cập đến quá trình render các trang web hoàn toàn trong trình duyệt của khách hàng bằng cách sử dụng JavaScript. Việc sinh nội dung … đều được thực hiện ngay trên client, server chỉ có nhiệm vụ trả mã nguồn JS và dữ liệu thô cho client=> khắc phục được 1 số nhược điểm của SSR. Tuy nhiên vẫn có 1 số nhược điểm:
* Lượng dữ liệu lần đầu load về khá nặng
* Nội dung hiển thị ra lần đầu chậm hơn so với SSR, tuy nhiên những lần sau thì quá nhanh do việc xử lý ngay trên client.
* SEO bị ảnh hưởng, do nội dung web được sinh trên client, khiến crawler của của Search engine không tiếp xúc được nội dung.
* Gần đây mới có Google giải quyết được vấn đề này, tuy nhiên dù Google nắm phần lớn thị phần tìm kiếm thì tại những quốc gia khác nhau, sẽ có ảnh hưởng nhất định.

22.