React.JS

What is react?

React is a JavaScript library for building user interfaces.

Developed in 2011 and released in 2013.now version 18

What is SPA?

SPA stands for single-page application. It’s a web application or website that interacts with the user by dynamically rewriting the current page rather than loading an entire new page

What is CDN?

CDN Stands for content delivery network, it’s a network of distributed servers that deliver pages, content, or files to users based on their geographic locations, allowing developers to include these libraries in their project without having to host them on their own servers

CDN determines geolocation based on our request’s IP address and this request redirects to the nearest most suitable server, it will help to reduce latency and faster access to the content you requested

Virtual DOM

It's a concept implemented by React to improve the performance of web applications. It’s a lightweight copy of the actual Dom. virtual Dom allows react to minimize the direct Dom manipulation, whenever a component state is changed, react updates its virtual Dom and compares the updated virtual Dom with the previous virtual Dom version, and then calculates the best way to make changes in real Dom, minimizing operation and improving performance

React Element

It's an object that describes the Dom node, elements are the smallest building block of the react app.

This node represents what you want to see on the screen in react

Components

A Component is one of the core building blocks of React. They have the same purpose as JavaScript functions and return HTML. Components make the task of building UI much easier.

A UI is broken down into multiple individual pieces called components. You can work on components independently and then merge them all into a parent component which will be your final UI.

components can be class-based or functional,

State

state is a built-in object used to store mutable data within a component. It allows components to keep track of changing information and automatically re-render when the state is updated.

It is a set of data that determines the behavior of the component and renders its output. when the state of a component changes, the component re-renders to reflect those changes

Props

Props means Properties. it’s input data for the component that is received from the parent component. Props allow you to pass data from one component to another.

They are read-only and should not be changed within the child component. Any changes to props should be done in the parent component. if you need to change the value, you should lift the state up to the parent component and pass down a new prop value.

React and React Dom

REACT - provides a framework for defining and managing components in a virtual environment

REACT DOM – It bridges the gap between virtual Dom and real dom. allowing react component to be displayed or rendered on the web page

Why CDN links are not a preferred way to bring React and React-DOM to the project.

- CDN loads the entire library, it contains both used and unused packages,
- Harder to keep track of library versions and update them,
- CDN links require internet, which can be made slower our project due to the need for library fetching, The local module bundled with our app will remove this dependency and enable it via offline
- if CDN experiences downtime or network issues it will affect our application

NPM

It's a package manager for node js. Allows developers to install, share, and manage packages or libraries of code to use in their projects.

Package.JSON

It contains metadata about the project, such as name, version,description, dependencies, etc.

Package lock Json

It automatically generates by npm when dependencies are installed. It locks the version of each package's dependency to ensure that the same versions are installed on all machines. also, it prevents unexpected version changes, ensuring that the application behaves consistently across different installations

Dependency

It is the package or library that is installed on our project, they are specified in package.json

3 types of dependency are regular, development, optional

1. regular dependency

These are required for the application to run in production. E.g.: REACT.Express,react-dom,

2. development dependency

These are only needed in during the development and testing of the application, and not required for the application to run in production
EG:parcel,nodemon,postcss,eslint
3. optional dependency
These are not required for the application to run, but if present it can provide additional functions and features

Bundlers

Bundlers are tools that take separate modules and their dependencies and bundle them together into a single file that is optimized for the browser. bundlers features are,

1. module support – commonjs, and, es modules, it enables without worrying compatibility issue
2. code splitting – bundlers can also split code into smaller chunks allowing load necessary parts
3. asset management -can handle assets like images, fonts, CSS
4. Tree shaking – feature removes unused code from the final bundle, makes the bundle smaller and faster
5. Loaders and plugins

Transform the files before they are added to the bundle. Eg: the babel-loader can transpile JSX and es6+ modules code to es5, the CSS loader can process CSS files, file loader can manage import and export

Plugins: extend the capabilities of bundlers allowing them to customize the build process extensively

Can include plugins for bundle optimization, environment variable injection, etc...

6. HMR (Hot module replacement)

It is a feature that updates modules in the browser at runtime without needing a full refresh,

This means we can see the changes instantly while maintaining the application state

7. cross-platform compatibility

Bundlers can ensure code compatibility in different browsers and platforms

JSX

Stands for javascript XML, it's a syntax extension for javascript recommended by React. JSX allows you to write HTML elements in JavaScript.without creating an element and appending child calls

JSX syntax is similar to making it easier for developers to understand the structure of ul

JSX code is understood to react with the help of transpilation(convert one lag version to another)

Babel is an example of a transpiler. It also ensures the browser compatibility with modern js es6+ syntax into older versions that are compatible with a wider range of browsers

Component lifecycle

The component lifecycle of react refers to the series of events that occur from the starting and ending procedure of a component. The lifecycle is divided into 3 phases

1. mounting phase

When the component is being created and inserted into the Dom

- Constructor () - will be called before the component mount
- render() – it examines props and state
- component did mount() – immediately invoked after component is mounted

IF THE CHILD COMPONENT IS INSIDE THE PARENT THEN CHILD 1 IN ORDER IT WILL BATCH UP THE RENDER GROUP IT WILL NOT MANIPULATEE DIRECTLY DOM INSTEAD CHANGE IN VIRTUAL DOM THEN IT WILL EXECUTE IN ORDER AND THE LAST PARENT WILL MOUNT

2. updating phase()

Ouccures when a component is being re-rendered

*render(): Called during updates as it is in the mounting phase.

*componentDidUpdate() invoked after the update occurs

3.unMountingPhase ()

When the component is being removed from the dom

*component will unmount() – invoked immediately before a component is unmounted/removed

NOTE: similarly use effect can done in this three-phase

.putting dependency will call only the 1st-time component rendered

.putting dependency with a specified state will call when that state gets updated

.putting return in use effect and make code, inside return that will execute when the page disappears

Client-side routing

Loaded in a single page application Each route only changes the components no need for a network call

Server-side routing

Network call will fetch the data and that server will decide what to expose or what to be displayed or rendered

Difference between SERVER server-side routing and server-side rendering

SSR is a rendering strategy to improve load times and SEO by generating by HTML on the server

Where server-side routing is about managing navigations and URLs on the server to serve the appropriate content for each request

.rendering -just render the content from the server

.routing-server will fix each endpoint to what are the content to be rendered in each route

State management

It's the process of managing the state of an application.in framework react,vue, angular. State management involves handling data that controls the behavior and appearance of the application

- local state management=managing state within a component bw parent and child component

To manage these builtin capabilities – useState hooks

- global state management = managing global state easily by using libraries like redux,

Otherwise also managed by the use of context hooks

HOOKS

a hook is a function that allows you to use state and other React features in functional components

Hooks Provide access to states and other react features In functional components.

1.useState

allow functional components to manage state, we can initialize a state with a value and it returns as an array 1st value name of the state second is to state manage

2.useEffect

similar to life cycle methods in class components, it has 2 arguments cl fundtion and optional array of dependency componentDidMount, componentDidUpdate, and componentWillUnMount.

writing area is componentdid update section

without optional dependency call for every renders

with optionalarray with empty calls initial render

with optional array with state act as componentdidupdate it will call every time that state changes

by writing a call back function in return make It as Componentwillunmount

3.useContext()

Allow subscribing to react context

4,useRef

Accessing dom directly within functional components or keeping track of previous values without causing re-rendering

Controlled and uncontrolled components

Imagine a form input data

Controlled = form data is handled by a react components state, any change occurs in form event handlers will update that in the components state

Uncontrolled =data handled by Dom itself without relying on react state, changes of form not tracked by react state. instead of using ref access no need to track state

Stateful components

Also known as class components manage their own state by using this. State and this .setState()

Stateless components

Also known as functional components. it does not have a state. receive data via pros and renders based on that data

Lifting state up

It’s a process of moving the state from a child component to its parent component,

Using where multiple components need to share and synchronize the same state. when the state needs to be managed at a higher level in the component hierarchy for better control and organization of the code.

Props drilling

Process of passing props from a parent component to deeply nested child components

To avoid this use techniques such as context or stage-management library redux,zustand

1 parent 10 child 10 child needed props should travel through all child components

Context API

It is a feature in React that allows sharing data between components without passing props through the component tree

Api contains 2 main parts

1. provider

Wrap a part of your component tree and pass down a value to all within that tree

2. consumer

It allows to access the value passed down by the provider

Optimization technique

Technique which is used to improve performance by reducing unnecessary computation and re-renders

Techniques: lazy loading,useMemo, UseCallback, code splitting

Lazy loading

It will avoid nonessential resources at the initial time of page load. Suspense to load components only when they needed

Code splitting

The technique used to split code into smaller bundles .it helps to reduce initial load time. react lazy and suspense can be used for code splitting as well

UseMemo

It is a hook to memorize the result of a computation. On subsequent render, it will check if any of the dependencies of the use memo function changed. if not changed react reuses the memorized results. if changed react again call that function to compute the new result

useCallback

it also a hook similar to the use of Memo but here it will memorize the function, not the result. it returns a memorized version of a callback function.

Diff algorithm

it is the algorithm that used for comparing the virtual doms versions

Reconciliation

It is the process of rect used to update UI by after applying diff algorithm

React fiber

It is a complete rewrite of the reconciliation algorithm, introduced in react 16, fiber improvement the react to handle asynchronous rendering.

createContext

used to create a new context object. initial setup for defining global objects

usecontext hook

used to access the value of context from within the functional components. Making to access without prop drilling

consumer

in older versions with in-class components. The consumer component is used to subscribe to the context changes. components wrapped in consumer can access the context value

provider

used to wrap part of applications where the context value should be accessible .it will be distributed to all components within the provider's tree.

components can either use usecontext hook or consumer component to access the context value

REDUX TOOLKIT

create a slice and connect to store

wrap the store with components by provider

use the selector to enable the store values and usedispatch function to trigger the action for splice changes

Statefull components

Class-based components are the traditional approach for state management in

React. They inherit from React's built-in `Component` class, allowing them to have their own internal state that can change over time. Additionally, class components have lifecycle methods that run at different stages of the component's existence,

Stateless components

Functional components are the simpler option in React. They act like functions that take data (props) and return what to display on the screen (JSX). Unlike class components, they don't manage their own state by default, making them lightweight and easier to reason about. With React Hooks (available since version 16.8), functional components can now also manage state

React Fiber

A reimplementation of React's core algorithm designed to enable incremental rendering and improve responsiveness. It can prioritize tasks, allowing for more efficient updates and better handling of complex UI interactions.

Pure component

A **Pure Component** in React is a component that "SHOULDCOMPONENTUPDATE" lifecycle method to prevent unnecessary re-renders, optimizing performance.(it avoid unaffected areas rerenders ,if a parent render all child automatically renders so it can prevent unwanted renders). the same thing we can implement in functional components with REACT.MEMO