React ek mashhoor JavaScript library hai jo user interfaces banane ke liye istemal hoti hai. Yeh Facebook (jo ab Meta kehlaati hai) ne develop aur maintain ki hai. React developers ko reusable UI components banane ki sahulat deti hai, jis se complex interfaces ko efficiently banana aasaan ho jata hai. React apni flexibility, performance, aur declarative nature ke liye mashhoor hai aur yeh kisi application ke view layer par ziada focus karti hai, yani wo hissa jo user dekhta hai aur jiss ke sath interact karta hai.

React ke kuch key concepts mein shamil hain:

* **Components**: React apps components par mabni hoti hain, jo ke UI ke independent, reusable pieces hain. Components class-based ya function-based ho sakte hain aur props (data jo components ko bheja jata hai) ko accept kar sakte hain taake inka behavior control ho sake.
* **JSX (JavaScript XML)**: React mein JSX use hota hai, jo ek syntax extension hai jo aapko JavaScript ke andar HTML-jese code likhne ki ijaazat deta hai. Is se UI structure ko directly JavaScript files mein visualize aur manage karna aasaan ho jata hai.
* **State Management**: React components state rakh sakte hain aur manage kar sakte hain, jo ke wo data hai jo screen par render hone wale content ko affect karta hai. Jab ek component ki state change hoti hai, toh React efficiently UI ko update karta hai taake nayi state reflect ho.
* **Virtual DOM**: React performance ko optimize karne ke liye ek Virtual DOM ka istemal karta hai (yeh actual DOM ki memory mein representation hai). Directly DOM ke sath interact karne ke bajaye, React pehle Virtual DOM ko update karta hai aur phir sirf woh parts jo change hue hain, unhe efficiently actual DOM mein update karta hai.
* **One-Way Data Binding**: React mein data ek hi direction mein flow karta hai, yani parent se child components tak. Is se data flow predictable aur debug karna aasaan ho jata hai.

React ko aur bhi libraries ya frameworks ke sath use kiya ja sakta hai jaise state management ke liye (Redux ya Context API), routing ke liye (React Router), aur side effects ke liye (React Query). Yeh sirf single-page applications (SPAs) ke liye hi nahi, balke mobile apps (React Native ke zariye) aur server-rendered applications (Next.js ke sath) ke liye bhi ek versatile tool hai.

Next.js ek popular React framework hai jo server-side rendering (SSR) aur static site generation (SSG) ke liye use hota hai. Yeh framework Vercel ne develop kiya hai aur React applications ko aur bhi powerful aur efficient banata hai. Next.js React ke upar kaam karta hai aur kuch advanced features provide karta hai jo ek high-performance aur SEO-friendly website banane mein madadgar hain.

**DOM**

Document Object Model yaani **DOM** aik programming interface hai jo web documents ke liye use hota hai. Ye aik HTML ya XML document ki structure ko tree of objects ki tarah represent karta hai, jo JavaScript jaise programming languages ko document ke content, structure, aur style ke sath dynamically interact aur manipulate karne ki ijazat deta hai.

**React Hooks**

React Hooks wo khaas functions hain jo React 16.8 mein introduce kiye gaye thay aur ye aapko functional components mein React ki features "hook" karne dete hain. Ye aapko state manage karne, side effects perform karne, aur context use karne ki sahulat dete hain bina kisi class components ki zaroorat ke. Hooks code ko simplify karte hain aur reusability ko improve karte hain kyunke inki madad se aap stateful logic ko mukhtalif components ke darmiyan share kar sakte hain.

React Hooks functional components ko powerful, reusable, aur concise banate hain kyunke ye zyadatar cases mein lifecycle methods ki zaroorat khatam kar dete hain.

**1. useState**

* useState hook component ke andar state banane aur usay manage karne ke liye use hota hai.
* Aap isko kisi bhi variable ka initial value set karne ke liye aur usay dynamically update karne ke liye use kar sakte hain.

**Example**: Ek counter banane ke liye jo button click par barhta hai.

**2. useEffect**

* useEffect hook side effects ko manage karne ke liye use hota hai, jaise ke data fetch karna, DOM (Document Object Model) ko update karna ya event listeners add karna.
* Yeh hook Next.js mein client-side data fetching ke liye bhi use hota hai, kyunki yeh sirf tab chalta hai jab component browser mein render ho jata hai.

**3. useRouter**

* useRouter ek Next.js-specific hook hai jo ke next/router se aata hai. Yeh routing ko programmatically handle karne ke liye use hota hai.
* Is se aap current route ke parameters, URL query aur navigation ko control kar sakte hain.

**4. useContext**

* useContext global state ya shared data ko manage karne ke liye use hota hai.
* Yeh hook context ko consume karne mein madad karta hai, taake data multiple components mein bina props pass kiye share kiya ja sake. Iska istemal aksar user authentication status ya theme manage karne ke liye hota hai.

**5. Custom Hooks (Apne Hooks)**

* Custom hooks apne logic ko reuseable banane ke liye create kiye jate hain. Yeh ek hook hota hai jo dusre hooks ka istemal karta hai.
* Yeh common ya complex functionality (jaise ke data fetching) ko simplify karne mein madadgar hotay hain.

**6. useSWR (Stale-While-Revalidate)**

* useSWR ek popular hook hai jo ke SWR library ke sath aata hai aur data fetching ko asaan banata hai.
* Yeh caching, error handling aur automatic revalidation provide karta hai, jo ke efficient client-side data fetching ke liye kaam aata hai.

**7. getServerSideProps aur getStaticProps**

* Yeh hooks nahi hain lekin Next.js mein specific data fetching functions hain.
* getServerSideProps ko server-side rendering ke liye use kiya jata hai, jo page ko request ke waqt data ke sath render karta hai. getStaticProps ko static site generation ke liye use kiya jata hai jo build time par page ko generate karta hai.

In hooks ka istemal Next.js applications mein efficient aur interactive components banane ke liye hota hai aur ye development ko asaan aur tez banate hain.

Next.js aur React mein, **props** wo data ya information hote hain jo ek component dusre component ko pass karta hai, khaaskar jab ek parent component apne child component ko kuch data dena chahta hai. Yeh "properties" ka short form hai aur components ke darmiyan data transfer karne ka ek tareeqa hai.

Props immutable (badalne ke qabil nahi) hote hain, yani child component ko ye data receive karne ke baad ismein directly koi tabdeeli nahi kar sakta — wo sirf usay read ya display kar sakta hai.

### Props ke istemal ka maqsad:

Props ka maqsad reusable aur dynamic components banana hai, jise ek hi component ko mukhtalif data ke sath bar-bar use kiya ja sake.

**JavaScript (JS)** ek mashhoor aur powerful programming language hai jo web development mein istemal hoti hai. Yeh language web pages ko interactive banane ke liye use hoti hai, jese ke buttons pe click karne par action lena, animations banana, data validate karna, aur dynamic content load karna.

JavaScript ka istemal front-end (browser) aur back-end (server) dono taraf ho sakta hai. Browser par yeh language HTML aur CSS ke sath mil kar kaam karti hai taake users ke liye interactive aur responsive web pages banaaye ja sakein.

### JavaScript ki kuch khas baatein:

1. **Client-Side aur Server-Side**: JavaScript ko client-side scripting ke liye use kiya jata hai, jo ke browser mein chalta hai. Lekin ab Node.js ke zariye yeh server-side programming ke liye bhi use hota hai.
2. **Interactive Web Pages**: JavaScript ki madad se web pages mein animations, sliders, pop-ups aur user interactions banaye ja sakte hain.
3. **DOM Manipulation**: JavaScript se DOM (Document Object Model) ko directly manipulate kiya ja sakta hai, yani HTML aur CSS ke elements ko update kar sakte hain bina page ko reload kiye.
4. **Event Handling**: JavaScript events ko handle kar sakta hai jese button click, form submit, ya page load.
5. **Programming Concepts**: JavaScript mein basic programming concepts hote hain, jese variables, functions, loops, arrays, aur objects jo powerful applications banane mein madadgar hain.

JavaScript aik versatile language hai jo web applications ko asaan aur powerful banati hai aur har modern browser mein supported hoti hai.

**Object-Oriented Programming (OOP)** aik programming paradigm hai jo code ko organized aur reusable banata hai. Ismein key concepts shamil hain jaise:

1. **Classes**: Blueprint ya template jo objects banane ke liye use hota hai.
2. **Objects**: Class ka specific instance jo properties aur methods ko contain karta hai.
3. **Inheritance**: Aik class doosri class ke properties aur methods ko inherit kar sakti hai, jisse code reuse aur hierarchy establish hoti hai.
4. **Encapsulation**: Class ke data ko protect karna taake woh sirf specific methods ke zariye access ho sake.
5. **Polymorphism**: Ek method ya function ko mukhtalif objects mein mukhtalif tareeqe se implement karna.
6. **Abstraction**: Unnecessary details ko chhupa kar sirf important details ko highlight karna.

TypeScript mein OOP concepts ka istemal aapko powerful, structured, aur maintainable applications banane ki sahulat deta hai.

**TypeScript (TS)** Microsoft ki banayi hui aik programming zaban hai jo **JavaScript** ka ek **superset** hai. Isme **static typing** aur additional syntax include hota hai jo developers ko bugs jaldi pakarne, code ko asani se samajhne aur editor tools ko enhance karne mein madad karta hai.

TypeScript ki khas features:

1. **Static Typing**: TypeScript mein aap variables, function parameters, aur return values ke liye types define kar sakte hain jo ke compile time par type-related bugs ko avoid karne mein madad karta hai.
2. **Type Inference**: Agar aap explicitly types declare na bhi karein, TypeScript aksar types ko khud infer kar leta hai, jis se type annotations kam zaroori hote hain.
3. **Object-Oriented Features**: TypeScript classes, interfaces, inheritance aur doosri object-oriented programming ki features ko support karta hai, jo complex applications likhne mein asani paida karta hai.
4. **Improved Tooling**: TypeScript developer experience ko enhance karta hai behtar autocompletion, navigation aur refactoring tools ke saath, khaaskar VS Code jaise editors mein.
5. **JavaScript Compatibility**: Kyunke TypeScript ek superset hai, koi bhi valid JavaScript code TypeScript mein bhi valid hota hai. TypeScript compile hokar plain JavaScript mein convert hota hai, is liye ise har jagah use kiya ja sakta hai jahan JavaScript chalti hai.

Bade projects aur mashhoor frameworks, jaise ke **Angular**, **Next.js**, aur **React** mein TypeScript ka istemal hota hai, jisse code ki quality aur maintainability mein behtari aati hai.