1. **React là gì?**

* Là thư viện JS front-end được phát triển bởi Facebook dùng để xây dựng ứng dụng Single Page

1. **State trong React**

* Là object chứa thông tin có thể thay đổi trong suốt thời gian tồn tại của component

1. **JSX là gì?**

* Là cú pháp mở rộng của JS viết theo kiểu XML

1. **Prop vs State vs global state(redux)**

* Props được truyền từ component cha sang component con và giá trị của props không thể thay đổi được
* State: có thể thay đổi giá trị, chỉ dùng trong component chứa state đó
* Global state: có thể thay đổi giá trị, được sử dụng bởi nhiều component khác nhau

1. **Class component vs Function component?**

* Class component có thể sử dụng local state và các lifecycle hook
* Function component chỉ nhận vào props và hiển thị lên UI

1. **Real DOOM vs Virtual DOOM**

* DOOM ảo là bản sao của real DOOM, react sử dụng DOOM ảo để tăng hiệu xuất vì khi state thay đổi, react chỉ load lại những phần có dữ liệu bị thay đổi chứ không load lại toàn bộ component lớn.

1. **Render là gì?**

* Render là quá trình xảy ra khi state của component thay đổi và component cần cập nhật lại dữ liệu mới

1. **Var, let, const**

* Let tạo ra biến có thể truy cập trong block chứa biến đó
* Var tạo ra biến có thể truy cập trong function chứa biến đó, có thể dẫn tới hoisting
* Const dùng khai báo hằng số

1. **Server side render vs Client side render**

* Server side render: logic thực hiện phần lớn ở server (render, validation, authentication, routing,..)
  + Ưu: load nhanh, chạy được trên phần lớn các trình duyệt
  + Nhược: trang phải load lại nhiều lần, nặng server, tốn băng thông vì server gửi dữ liệu lớn
* Client side render: render HTML, CSS thực hiện ở trình duyệt (client). logic đơn giản thực hiện ở client (routing, validation,..), còn logic phức tạp vẫn nằm ở server(data process, report).
  + Ưu: Page chỉ load khi user chuyển trang hoặc thêm dữ liệu, giảm tải xử lí logic ở server, giảm băng thông do chỉ lấy json và dữ liệu cần thiết
  + Nhược: load lần đầu tiên chậm, project phải chia thành 2 phần front-end và back-end, ko chạy được nếu disable JS

1. **React vs Angular**

* React chỉ dùng phần View trong mô hình MVC còn angular thì dùng toàn bộ mô hình MVC
* React là server side rendering còn Angular là Client-side rendering
* React dùng DOOM ảo còn Angular dùng DOOM thật
* React dùng one-way binding (dữ liệu truyền 1 chiều từ view sang component hoặc ngược lại), Angular dùng two-wat binding (thay đổi dữ liệu từ component qua view và ngược lại)

1. **Life cycle của component trong Reactjs**

* Được tạo ra (Mouting – được thêm vào DOOM) (thực thi 1 lần)
* Qua nhiều thay đổi (Updating) (thực hiện nhiều lần)
* Bị hủy bỏ (Unmouting – bỏ ra khỏi DOOM) (thực thi 1 lần)

1. **Arrow function**

* Là hàm vô danh, cách khai báo ngắn gọn so với function

1. **Component và element**

* Component nhận vào giá trị đầu vào và trả về element

1. **Reactjs và ReactNative**

* React là thư viện front-end JS còn ReactNative là mobile framework

1. **setTimeOut, setInterval**

* setTimeOut: thực thi block code sau 1 khoảng thời gian nhất định (thực hiện 1 lần duy nhất)
* setInterval: thực thi block code liên tục sao 1 khoảng thời gian nhất định

1. **Callback**

* Là hàm sẽ thực thi sao khi 1 hàm khác đã thực thi xong

1. **Promise**

* Là cơ chế thực thi tác vụ bất đồng bộ

1. **Custom hook**

* Là hook tự định nghĩa, tên bắt đầu với “use”.
* Có thể sử dụng những hook khác (react hook – useState, useEffect hoặc custom hook khác)
* Custom hook dùng để tách riêng phần logic khỏi UI, và chia sẻ logic này cho nhiều component khác nhau.

1. **Redux**

* Là thư viện quản lý state có thể dự đoán được dựa trên khái niệm store, để cập nhật store, phải dispatch 1 action tới store thông qua reducer. Reducer nhận vào state hiện tại và action vừa được dispatch để trả ra state mới.

1. **useRef**

* Luôn luôn giữ state mới nhất nhưng không render giá trị đó lên

1. **Stateless component**

* Là component không có state, dùng khi component không cần tới react life cycle

1. **Truyền parameter vào event handler hoặc callback**

* Sử dụng arrow function (hàm vô danh, ngắn gọn hơn so với function) để bọc quanh event handler và truyền tham số vào.

1. **Ngăn component hiển thị trong react**

* Trả về null ở hàm render()

1. **Super trong constructor**

* Dùng super để sử dụng tham chiếu của keyword this.

1. **Key là gì khi hiển thị danh sách**

* Key giúp xác định item nào đã thay đổi, thêm mới hoặc xóa bỏ.
* Nên sử dụng key để cung cấp định danh cho các phần tử

1. **Tại sao cần viết hoa component**

* Component ko phải là phần tử DOOM mà là hàm tạo, nên nếu ko được viết hoa có thể dễ gây nhầm lẫn

1. **Mô tả cách xử lý event trong react**

* Bắt các sự kiện onClick, onChange, ta bọc {} quanh function xử lí sự kiện này

1. **Hàm setState trong react là bất đồng bộ hay đồng bộ**

* Hàm setState trong react là bất đồng bộ
* Vì react cố tình chờ khi tất cả component gọi tới setState trước khi re-render để tăng hiệu xuất và tránh render ko cần thiết

1. **Mục đích của tham số thứ 2 (hàm callback) trong setState?**

* Tại vì setState là hàm bất đồng bộ nên hàm callback này dùng để xử lí 1 logic nào đó sau khi setState kết thúc và component được hiển thị

1. **Condition render**

{var >= 2 &&

(block code)

}

1. **Presentational component vs container component**

* Container component: là component chứa component khác, và cũng là nguồn dữ liệu cho các component khác
* Presentational component: là component chỉ nhận dữ liệu thông qua props và hiển thị, các state của component này thường liên quan đến giao diện (ẩn hiện button, text,…)

1. **Ngăn chặn default của event trong react**

* Sử dụng preventDefault với tham số là e
* I’m thanh and I’m have over a years of experience as a front-end developer. And I have take part in some projects like: farna ecommerce platform for special industry is pharmacy and logistic. And class event management is using to manage class and member information in each class as well as manage document and event.