Conceptos Clave en el Desarrollo de Aplicaciones Web Modernas

# REST con Swagger

REST (Representational State Transfer) es un estilo de arquitectura para el diseño de servicios web que utiliza los métodos HTTP (GET, POST, PUT, DELETE) para interactuar con recursos identificados por URLs. Swagger, ahora conocido como OpenAPI, es una herramienta que permite documentar y probar estos servicios REST de manera visual y estructurada.

Funcionalidad: Permite a los desarrolladores conocer los endpoints disponibles, sus métodos, parámetros y tipos de respuesta, facilitando la integración y pruebas de las APIs.

Ejemplo: Una API REST que gestiona usuarios puede tener un endpoint GET /users para obtener todos los usuarios. Con Swagger, este endpoint se puede visualizar e incluso probar directamente desde una interfaz web.

# ReactJS

ReactJS es una biblioteca de JavaScript desarrollada por Facebook para construir interfaces de usuario (UI) basadas en componentes. Permite desarrollar aplicaciones web de manera declarativa y eficiente, enfocándose en la actualización dinámica de la interfaz a través de un DOM virtual.

Funcionalidad: React permite reutilizar componentes y manejar el estado de la aplicación de forma sencilla, promoviendo el desarrollo modular.

Ejemplo: Un componente <UserCard /> puede mostrar la información de un usuario, reutilizándose varias veces para mostrar una lista de usuarios.

# Hooks: useState, useContext, useEffect, useReducer

Los Hooks son funciones que permiten usar el estado y otras características de React sin escribir una clase.

• useState: Permite declarar variables de estado en un componente funcional.  
Ejemplo:  
const [count, setCount] = useState(0);

• useContext: Accede a valores globales definidos en el Context API.  
Ejemplo:  
const user = useContext(UserContext);

• useEffect: Ejecuta efectos secundarios como peticiones HTTP o manipulación del DOM.  
Ejemplo:  
useEffect(() => {  
 fetchData();  
}, []);

• useReducer: Maneja estados complejos con lógica similar a Redux.  
Ejemplo:  
const [state, dispatch] = useReducer(reducer, initialState);

Funcionalidad: Los hooks permiten una mejor organización del código, manteniendo la lógica del estado y el ciclo de vida dentro de componentes funcionales.

# Context API

Context API es un mecanismo nativo de React para compartir datos entre componentes sin necesidad de pasar props manualmente en cada nivel del árbol de componentes.

Funcionalidad: Se utiliza para manejar temas globales como autenticación, idioma, o estado del usuario.

Ejemplo:  
const AuthContext = createContext();  
<AuthContext.Provider value={user}>  
 <App />  
</AuthContext.Provider>

# Peticiones HTTP con Axios

Axios es una biblioteca basada en Promesas para realizar peticiones HTTP desde el navegador o Node.js. Es más sencilla y poderosa que fetch, ya que permite interceptar solicitudes, cancelar peticiones y manejar errores globalmente.

Funcionalidad: Facilita la comunicación con servidores externos para enviar o recibir datos.

Ejemplo:  
axios.get('https://api.example.com/users')  
 .then(response => setUsers(response.data));

# Rutas y navegación

En React, las rutas permiten mostrar diferentes componentes dependiendo de la URL, generalmente mediante la biblioteca react-router-dom.

Funcionalidad: Permite crear una aplicación de una sola página (SPA) donde la navegación es fluida sin recargar el navegador.

Ejemplo:  
<Routes>  
 <Route path="/home" element={<Home />} />  
 <Route path="/login" element={<Login />} />  
</Routes>

# Despliegue

El despliegue es el proceso de publicar una aplicación web para que esté disponible a los usuarios. En React, esto se puede hacer en plataformas como Vercel, Netlify o GitHub Pages.

Funcionalidad: Permite que la aplicación se ejecute en un entorno de producción accesible a través de internet.

Ejemplo: Una vez ejecutado npm run build, el contenido generado se puede subir a Netlify mediante arrastrar y soltar el directorio build.

# Referencias

Mozilla. (s.f.). Guía para el uso de fetch. https://developer.mozilla.org/es/docs/Web/API/Fetch\_API

Axios. (s.f.). Documentación oficial. https://axios-http.com

Facebook. (s.f.). React Documentation. https://reactjs.org/

OpenAPI Initiative. (s.f.). Swagger Tools. https://swagger.io/tools/

React Router. (s.f.). React Router Docs. https://reactrouter.com