**Análisis de REACT:**

**Aplicación de Películas.**

Estructura y organización de las carpetas y ficheros de créate-react-app:

**Node\_modules:** Aquí guardamos todas las dependencias de nodejs

**Public:** Se almacenan los archivos públicos.

**Src:** Aquí va el código fuente.

Afuera de todo, tenemos los archivos .gitignore, package.json, README.md, estos son archivos de configuración del proyecto, sirve para ignorar dependencias, y para listar dependencias y para dejar comentarios útiles del proyecto.

**SRC/App.tsx\_or\_App.js:**  Es el archivo main del proyecto, aquí invocaremos todos nuestros componentes, creados con anterioridad en otras páginas. La estructura básica de una pagina en react, es importar las páginas y librerías que están en nuestro proyecto, son necesarias para poder usar nuestros componentes.

Crear una función, seguidamente necesitamos crear una **función** con su nombre, mas abajo creamos un **return** con sus paréntesis, luego creamos una sección, ya sea div o un fragmento <>Aquí van todos los componentes</>, finalmente se exporta, así: **export default App;**

function App() {

variables y lógica.

Return (

<>

</>

)

}

Componentes: para crear componentes, debemos crear una carpeta llamada components, y ahí deja todas nuestras secciones de la aplicación. Por ejemplo, **componentes/películas/listaPeliculas,**

**App.css:** esta página contiene código css, que tiene que ser importado para ser usado en otra página.

**App.tsx > ReactDOM.render** ¿Qué es? R/. Aquí renderizamos las páginas de react.

**Interface:** una interfaz, es un contrato, el cual se debe cumplir con los anexos que ahí vienen,

1. Se pueden categorizar tipos
2. Siempre hay un cadenero que verifica que seas parte del club.

**Funcionalidad de carga** o espera: Para lograr esta funcionalidad, usaremos **useEffect**, ¿Por qué es necesaria la funcionalidad de carga?, porque la información viene de un web api, y esto tarda un poco, por lo que necesitamos poner un relleno antes, para hacer esto usaremos **timeOut**, y este es un efecto secundario del componente, es por eso que useEffect es perfecto para este trabajo.

Cuando se escucha la frase, quiero cambiar una variable para que se muestre en pantalla, se tiene que pensar en estados, con los estados cuando cambiamos el valor de una variable, podemos renderizar el componente.

Cuando queremos crear varios estados en la lógica de nuestro programa, podemos usar un if – else para hacerlo.