Programación Web Full Stack

Creando una aplicación con React

Práctica Integradora

Objetivo

Al seleccionar una tecnología para usar en el **front-end** es importante pensar en **React**. Como ya sabemos se trata de una **librería JavaScript de código abierto enfocada en la visualización y desarrollada por Facebook**. Ha sido utilizada para crear **Instagram**, así como las apps móviles de **WhatsApp** y **Uber**, entre otras. Esta librería brinda muchas ventajas en **rendimiento, modularidad y promueve un flujo de datos y eventos** muy claro, lo que facilita la planeación y el desarrollo de apps complejas.

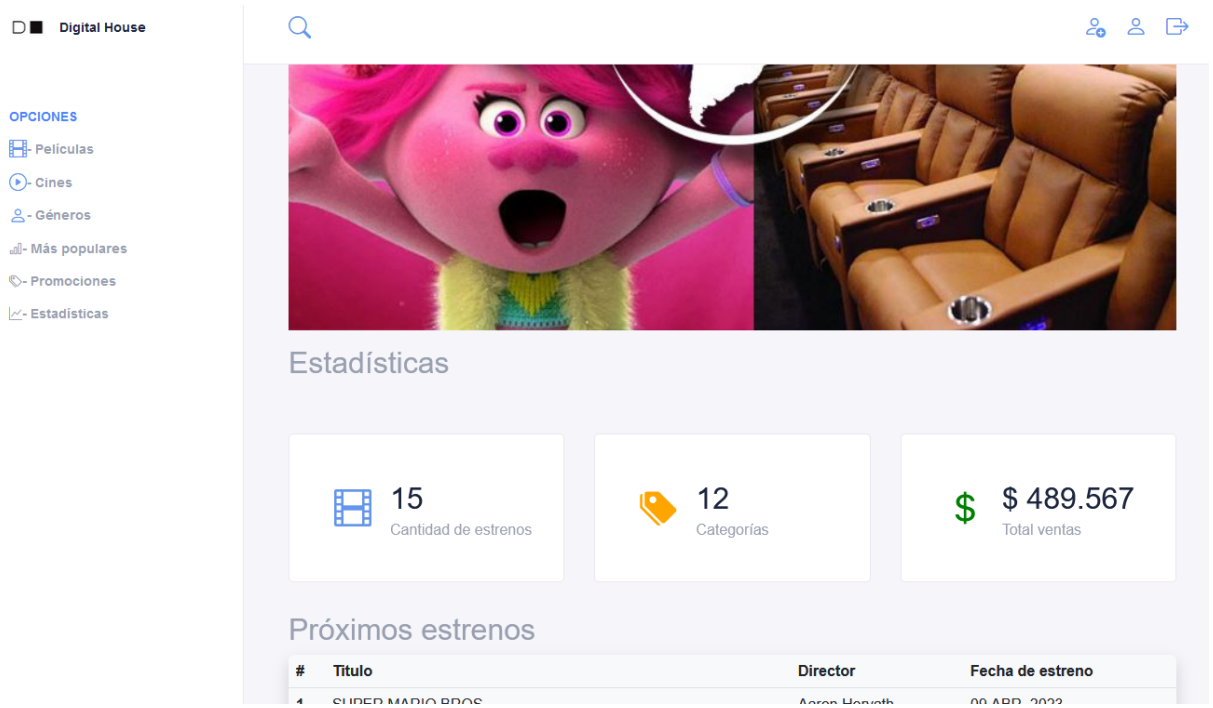
Ha llegado el momento de adentrarse en el maravilloso mundo de **React**.

En esta ejercitación tendremos la responsabilidad de "migrar" una sencilla estructura HTML + CSS, tipo tablero administrativo (dashboard) a una aplicación de **React**.

¡Éxitos! 😎👍✨

Requisitos:

- **Documento provisto:** antes de iniciar revisemos en detalle el material provisto. Una vez descargado, se ve así:

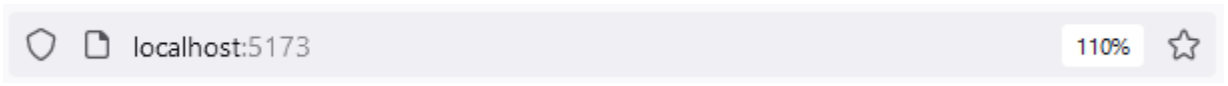


- **HTML + CSS:** estos *skills* ya hacen parte de nuestro set de herramientas, pero proponemos que, antes de iniciar, nos familiaricemos con toda la estructura del documento provisto para tener en claro por dónde ir.
- **React Basics:** para poder llevar a cabo esta misión es necesario tener en claro los conceptos de "**componente**". Si no es así, recomendamos tomarnos unos breves minutos para repasar dichos conceptos.

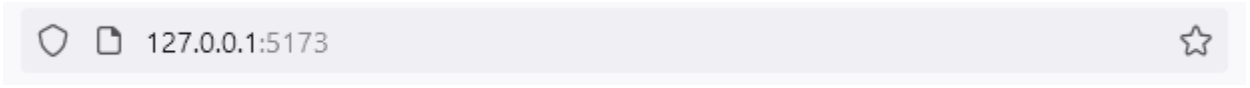
Micro desafío 1:

El objetivo principal será construir una app de **React**. Para ello podemos ejecutar desde la consola el comando: **npm create vite dashboard** y seguir los pasos ya aprendidos.

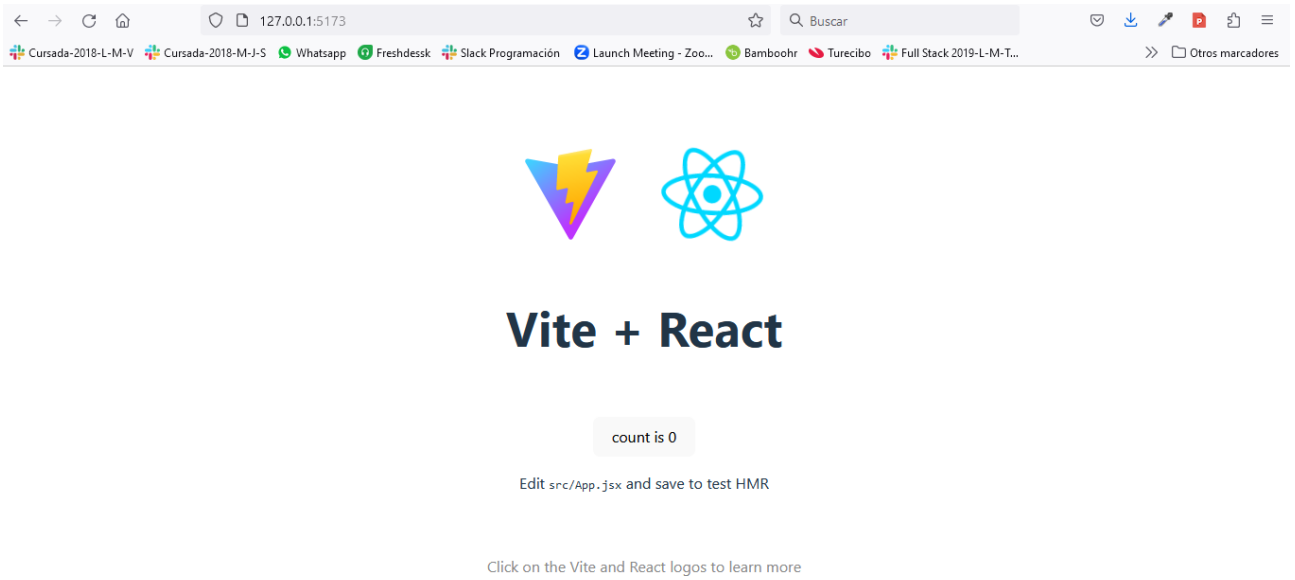
Una vez que esté la aplicación instalada, podemos ingresar a la carpeta creada **dashboard** y ejecutar el comando: **npm install** y finalmente para cargar el proyecto creado ejecutamos: **npm run dev** luego debemos ir al navegador y ejecutar:



Recuerda que también puedes cargar el proyecto, colocando en el navegador:



Si todo está bien, lograremos ver algo así:



¡Atención! En esta ejercitación **no** trabajaremos con un entorno de Node.js ni Express, ya que el enfoque principal de la misma es encararla 100% por el lado de **React**.

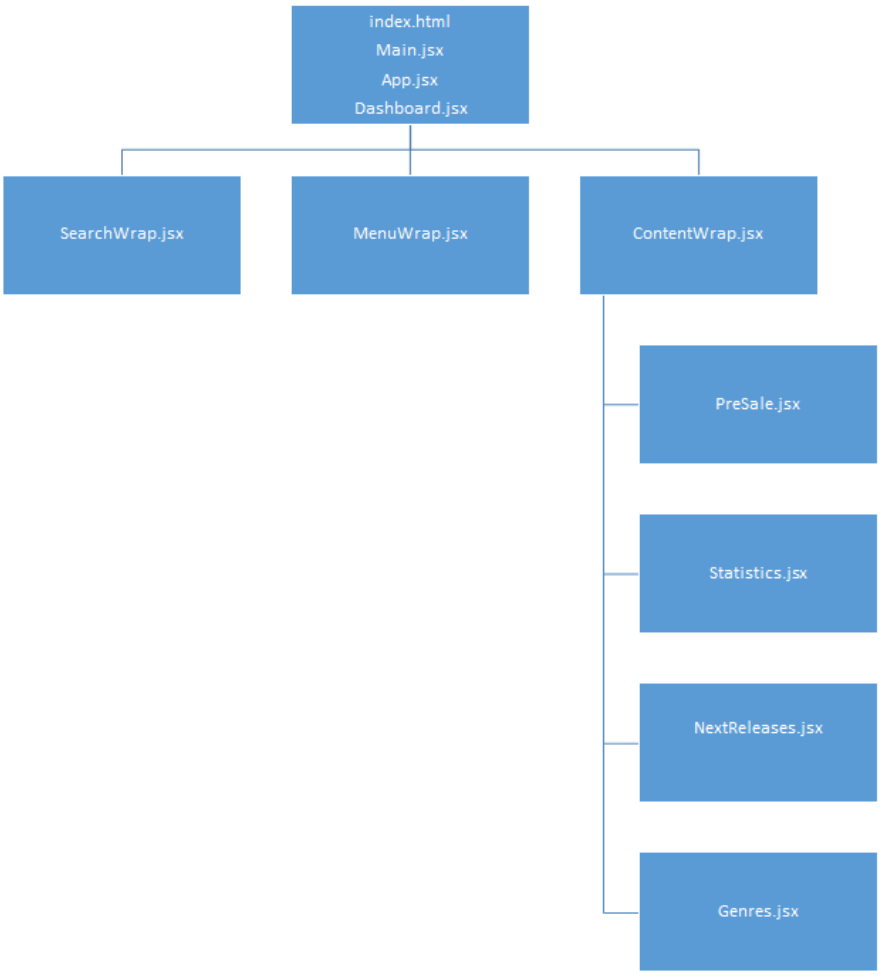
Micro desafío 2:

Una vez esté la aplicación instalada, deberemos proceder a crear los componentes. Tengamos en cuenta que los componentes que generemos deberán estar dentro la carpeta **/src** y preferiblemente dentro de una carpeta llamada **/components**.

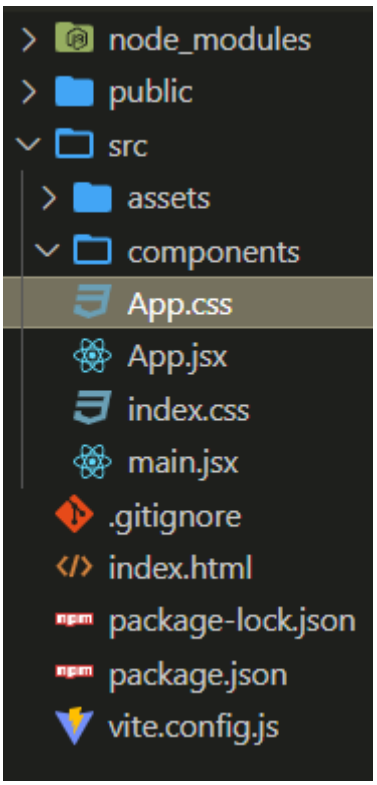
Por ello, antes de iniciar, debemos preguntarnos:

- ¿Qué partes de la interfaz pueden ser componentes?
- ¿Cuántos componentes se deben generar?
- Si un componente se parece mucho a otro, ¿hay alguna manera de hacer un solo componente y definir su aspecto visual de alguna forma?

De todas maneras, a continuación te entregamos una propuesta de referencia:



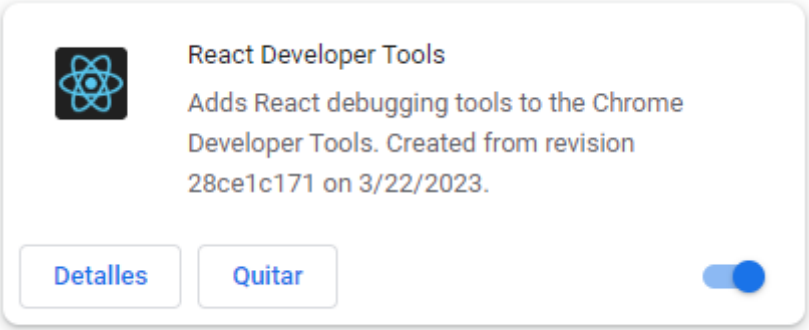
Adicionalmente, para tener una guía de la organización del proyecto, te proporcionamos la siguiente estructura de carpetas, recuerda crear los componentes: **SearchWrap.jsx** , **MenuWrap.jsx** y **ContentWrap.jsx**, dentro de la carpeta **components.**



Recordemos que para la creación de los **componentes sin estado (*stateless*)** debemos pasar, del **Material provisto**, las porciones de códigos necesarias, así como los respectivos archivos **css** (**/src/assets/css/**) y las **imágenes** (**/src/assets/img**). Ten presente que el código porcionado está escrito en HTML5, por lo tanto no se te olvide considerar el uso de los recursos de JSX. Si deseas ampliar información sobre estos recursos, puedes dar click aquí.

Micro desafío 3:

Una vez que crees los componentes, puedes ingresar a tu navegador y procede a instalar la extensión de las **React Developer Tools.**



Luego de instalarlas trata de revisar y familiarizarte con los recursos que esta extensión te ofrece para visualizar de una manera muy cómoda los componentes creados y la relación entre ellos.

Bonus Track:

Si lograstes realizar toda la práctica, lo cual estoy seguro que si, una buena idea es subdividir el componente creado: **/components/ContentWrap.jsx** en cuatro **componentes** independientes:

1. **/components/PreSale.jsx**
2. **/components/Statistics.jsx**
3. **/components/NextReleases.js**
4. **/components/Genres.jsx**

¡A usar la imaginación!

Conclusión

Con esta práctica pudimos comprobar la versatilidad que nos ofrece **React** por medio de la creación de los **componentes**. Estos son micro entidades independientes y autosuficientes que describen una parte de su interfaz de usuario. La **IU** de una aplicación se puede dividir en componentes más pequeños, donde cada componente tiene su propio código, estructura y API.

Facebook, por ejemplo, tiene miles de piezas de funcionalidad conectadas entre sí. Aquí hay un dato interesante: Facebook comprende más de 30,000 componentes... y el número sigue creciendo. La arquitectura de componentes le permite pensar en cada pieza de forma aislada. Cada componente puede actualizar todo a su alcance, sin preocuparse por cómo afecta a otros componentes.

¡Hasta la próxima!