Enunciado 2

Bienvenido a nuestra cita semanal, para no perder la costumbre. Para esta semana el equipo de iMaster ha estado siguiendo algunos tutoriales y documentaciones varias para ponerse a la par en lo que podría ser una nueva convocatoria, pero bueno, es otro tema del que hablaremos en su momento.

Como una practica del ejercicio de React, el equipo ha avanzado y ahora no solo trabaja con componentes en un único archivo, sino que ha comenzado a explorar las opciones de importación. Y exportación, también un nuevo hook, ahora contamos con el ***useState*** y con el ***useEffect.***
Como ya deben saber, una de las principales funcionalidad del ***useEffect***, es limitar o controlar el renderizado de un componente, el equipo lo ha implementado para crear una app que consuma recursos de una API para posteriormente pintarlos en pantalla. El app cuenta con un buscador y el endpoint de la API y el resto de su documentación esta disponible en el sitio web oficial de Giphy developers (https://developers.giphy.com/docs/api/endpoint#content-types).

Como es de costumbre, el contexto es el siguiente:

- Cuanto se estaba construyendo la app, se utilizó una APIKEY que provee el sitio para desarrolladores, pero al parecer alguien la borró y no había copia, entonces lo primero que se debe hacer es crear una nueva, esa APIKEY debe ser ubicada en el helper de la petición fetch.
- El limite del aplicativo, para disminuir costos en cuanto al número de elementos por petición es de 10, pero el valor esta mal establecido y no se esta aplicando.
- En el componente AddCategory se cuenta con un input que esta conectado a una variable del estado, pero no es posible escribir en el campo y por consiguiente el buscador no esta funcionando, un warning en la consola

alerta algo sobre un método onChange faltante o su manejador.

- Cuando el campo si estaba funcionando ocurrió algún otro daño, y el formulario estaba haciendo un submit, lo que hacía que la página se recargue y dejaba de mostrar los resultados de búsqueda, tal vez el método handleSubmit tenga relación con el caso.
- Algo que si ocurría y que faltaba corregir era que, al momento de hacer una búsqueda, se necesitaba limpiar el campo de búsqueda para que no quede el texto de la última búsqueda, pero como el sitio se estaba recargando no fue posible replicar el fallo, igual que el punto anterior, el método handleSubmit podría ser el causante.
- Ahora también hay errores que se recuerdan pendientes para corregir cuando la app funcionaba como por ejemplo que el texto alternativo de la imagen resultado de la búsqueda coincida con el título de la imagen y, contar con un párrafo que muestre este mismo título en el GifGridItem.

Su tarea, como ya se debe imaginar, es corregir dichos errores y levantar nuevamente el aplicativo, a continuación el equipo de iMaster le deja una imagen de como se debe ver el aplicativo a la hora de realizar una búsqueda como “Titanic”.

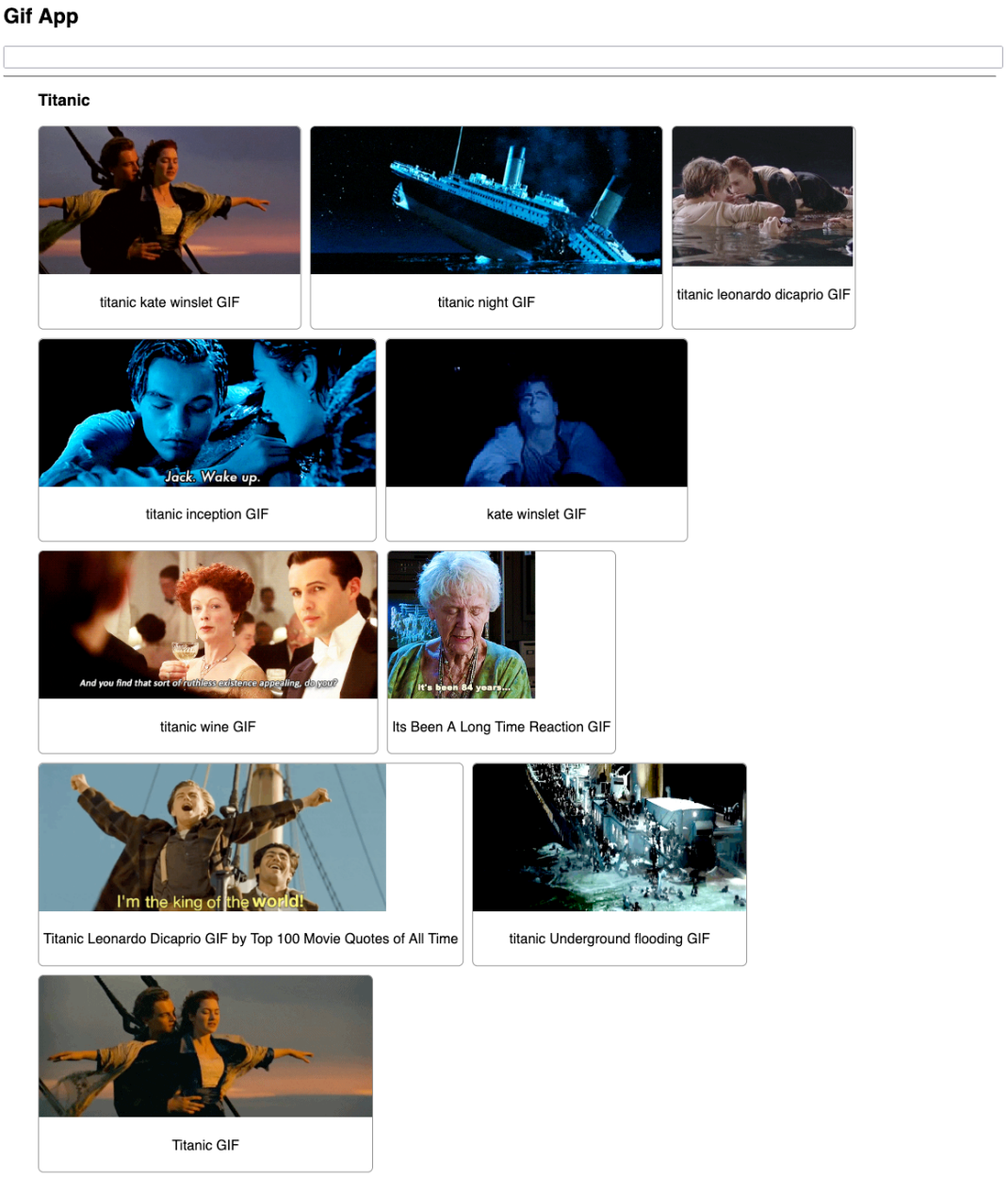

Ilustración 1 Resultado Búsqueda Titanic

Al igual que con nuestros proyectos anteriores, tenemos algunas herramientas a nuestro favor y algunos requerimientos que debemos cumplir/instalar para poder ejecutar elproyecto.

1. Debe contar con Node.js instalado en su equipo, para ello solo requiere ir al sitio oficial (https://nodejs.org/es/) y seleccionar el paquete que se ajuste a su

máquina. La interventoría recomendó utilizar la versión LTS.

2. Verifique la instalación de Node.js escribiendo en su terminal ***node -v***.
3. Una vez todo correcto, puede descomprimir el paquete de código que se le ha enviado y almacenarlo en una carpeta nombrada con su documento de identidad, todos los documentos son necesarios, no olvide incluir ninguno.
4. Con la carpeta nombrada correctamente, abra el proyecto con el editor de código de su preferencia, debe visualizar una estructura como la siguiente:

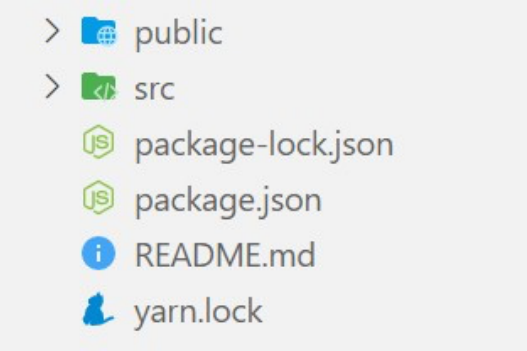

Ilustración 2 Estructura del proyecto

5. Ahora ya esta listo para ejecutar el proyecto, para ello debe abrir una terminal dentro de la carpeta del proyecto y ejecutar el comando ***npm install***.
6. Si todo esta correcto, deberá tener una nueva carpeta llamada "***node_modules***".
7. Recuerde que cuando crean que ya cumple con todos los puntos solicitados, puede ejecutar el validador, para hacerlo debe abrir una terminal dentro de la carpeta del proyecto y ejecutar el comando ***npm run test***.
8. Ahora solo resta comprimir el proyecto y enviarlo a la plataforma.
9. Antes de comprimir la carpeta nombrada con su número de documento de identidad, donde se encuentran todos los archivos, debe eliminar la carpeta "***node_modules***" del proyecto.
10. Ahora ya puede crear su archivo comprimido, el único formato permitido es ***.zip***.
11. El archivo comprimido ***.zip*** debe conservar como nombre su número de documento de identidad para proceder a la carga en la plataforma.

12. Finalmente siga las instrucciones de la plataforma para recibir una evaluación.

Recuerde que el reto puede ser subido por un único integrante del equipo, indicando el número de documento de identidad de los demás compañeros que hayan trabajado en el proyecto, separados por comas y sin puntos u otra clase de separadores, ej.

```
1111111,2222222,33333333,4444444
```

La carpeta del proyecto debe ser nombrada solo con el número de documento de identidad del usuario que hace la carga a la plataforma así la presente por todo su equipo, ej.