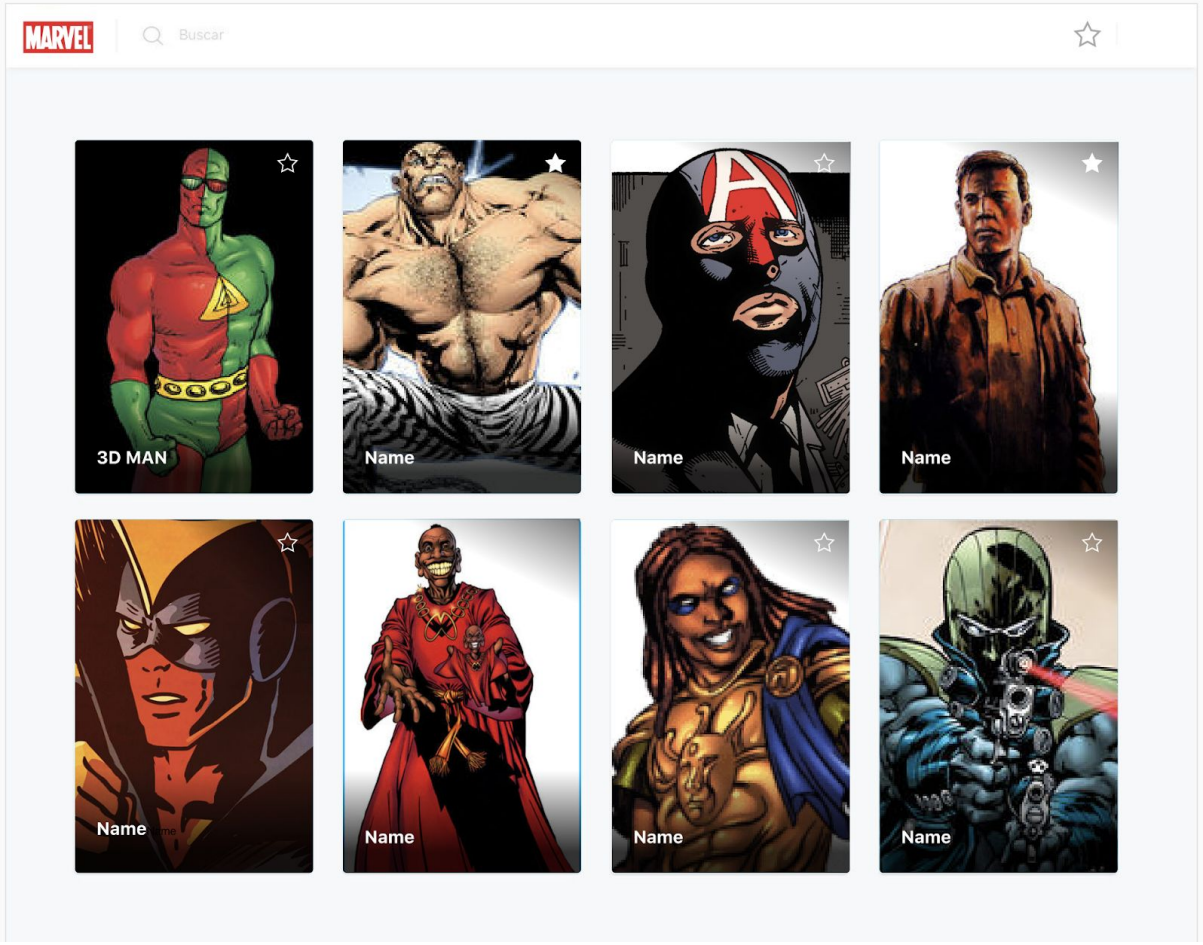
React Challenge

Part 1 - Marvel Searcher (App - Layout)

- Desarrolla una interfaz que siga el siguiente diseño adjuntado:



- Las peticiones se tienen que realizar a la siguiente URL: http://gateway.marvel.com/v1/

- El buscador tiene que tener la posibilidad de buscar los nombres de los **Personajes de Marvel en el input pero también por URL.**

- Cuando se entra la primera vez en la app web sin ningún personaje en la URL tiene que mostrar un personaje aleatorio y así por cada vez que vuelva a recargar la página.

- La búsqueda tiene que contemplar buscar por similitudes de texto y parecidos de nombre, es decir, con solo buscar “spider” debería de renderizar todas las posibilidades y matches que abarcan.

- La búsqueda tiene que tener la posibilidad de buscar por comic directamente también y si es un link que viene directamente de la página de marvel(The Amazing Spider-Man #22) tiene que visualizar un preview del cómic.

La manera que se mostrarán las búsquedas serán en forma de **cards y modales para el detalle del personaje con sus comics.**

- Al clickear un card debería ir al detalle del personaje y mostrar un listado de sus cómics ordenados por fecha.

- El buscador de marvel tiene que contemplar que por la búsqueda de la URL se pueda hacer búsqueda de los personajes mediante query strings en la url del sitio, por ejemplo: http://my.app.com/?character=”spiderman”&comic=”The Amazing Spider-Man #22” y debe renderizar el card de Spiderman que al clickearse solo debe de aparecer el comic que fue buscado(este feature debe de soportar más de un cómic por URL, así como más de un personaje)

- La búsqueda tiene que poder guardarse en un listado de favoritos y que persista en el browser para que pueda ser usada en un futuro solo dandole click a la lista de búsquedas favoritas el cual tiene la estrella en la esquina superior derecha del input de búsqueda.

Part 2 - UNIT TEST

- Cada componente creado debe de tener un test unitario que sea capaz de testear lo más importante del componente:
    - props
    - state
    - life cycles
    - render
- Al terminar el test se debe generar un snapshot con Jest para la totalidad del código.

Part 2 - Deploy

- La aplicación tiene que estar deployada en AWS o NOW.sh.

Stack

- react >=16.
- react-router-dom >= 5
- styled-components.
- Jest + Enzyme.
- **PROHIBIDO USAR CUALQUIER LIBRERÍA DE COMPONENTES COMO ANTD, REACTSTRAP, ETC.**
  (Los componentes deben ser creados desde 0 por ti mismo).
- React Context API para state management.

Conditions of Satisfaction:

- La aplicación tiene que ser totalmente responsive.
- La aplicación tiene que poder ser completamente cambiada mediante solo unas cuantas variables del theme usando styled-components.
- La página tiene que tener como máximo 1 segundo de carga por render al llegar del backend, el **performance** es importante.
- Se evaluará programación funcional si se desarrolla con hooks.
- La consola del browser debe de estar limpia de errores y advertencias.

- La estructura de carpetas y orden del proyecto es un factor importante en la sumatoria de puntos.
- **En la entrega del test se debe de compartir el repositorio donde se pueda notar commits diferenciales o branches x feature.**

  Es decir, no se puede subir toda la app en un solo commit o quedará la prueba descalificada.

Plus

- Manejar casos para cada uno de los respectivos ambientes: Prod, Dev y Staging.
- Añadir Linter + Formatter.
- La aplicación puede manejar distintos **THEMES** con styled-components u otra librería.