Módulo 3 – Routing

# ¿Qué es?

Slide 2

Es la forma en la que el usuario puede desplazarse o visitar diferentes secciones de una aplicación web. Básicamente es una asociación de un URL, con algún componente de ReactJS. Estas URLs pueden ser:

* Estáticas, cuando no se espera que algún path de la URL cambie, por ejemplo “/user/settings”
* Dinámicas, cuando la URL contiene paths que pueden cambiar, generalmente con el objetivo de especificar la carga de algún recurso, por ejemplo: “event/view/:eventId” el cual podría ser “event/view/1” o “event/view/4”

# React Router

Slides

Es una librería de Routing para React, que nos permite definir rutas y los componentes asignados a las mismas.

Además, también podemos features adicionales como:

* Loader, funciones para cargar data al entrar a una ruta
* Actions, funciones que se disparan desde un formulario
* Error boundary, componente para customizar las pantallas en caso de errores, ya sea al renderizar, en la carga de data o al ejecutar un action.
* Nesting routes

# Configurando rutas

Slide 3 - 6

Definiremos un nuevo componente para que sea nuestra pantalla de inicio. Con un botón “Ver usuarios” que nos dirigirá a la ruta “/users”. En esto usaremos los features:

* Estructura de carpetas con buenas prácticas. Definir carpeta pages, componentes, api.
* Loader, para cargar la lista de usuarios desde una primer.
* Hook de estado de la ruta, para saber si esta o no cargando la data.

Pasos:

1. Crear carpeta pages dentro de la carpeta “src”.
2. Crear la carpeta “home” y el archivo index.tsx dentro de ella.
3. Crear el componente Layout.tsx
4. Conectar el nuevo componente con el router para que se muestre en la ruta “/” y poner como layout el nuevo componente.
5. Comprobar que la ruta “home” esté mostrando el componente nuevo.
6. Agregar un botón sencillo con el texto “Ver usuarios”.
7. Crear la caperta type para definir el type de User.
8. Crear la carpeta api y crear una función que simular la carga de los usuarios desde el Array, usar defer para estado de carga.
9. Crear la carpeta Users dentro de la carpeta pages y el archivo index.tsx con el contenido actual de App.tsx. Tomar en cuenta el uso de Suspense y Await para estado de carga.
10. Crear la ruta “/users” como hijo de la ruta “/” y asignarle el nuevo componente Users.
11. Definir el prop “Loader” para llamar a la función que simula la carga de usuarios.
12. Visualizar la carga de los usuarios.
13. Crear un nuevo componente para ErrorBoundary y definirlo como parte de la ruta “/users”.
14. Agregar el hook “useNavigation” para mostrar el estado de carga.
15. Visualizar y simular errores para comprobar los cambios.

# Componentes

En la actualidad toda aplicación, ya sea web o móvil, se empieza a diseñar por secciones, páginas y bloques. Es aquí donde se empiezan a definir las bases de cada componente, sus funcionalidades, estilos e interacción con la UI o los datos de nuestra aplicación.

En ReactJS los bloques de nuestra aplicación son los Componentes, en ellos desarrollaremos, de manera declarativa, como se renderizará el componente, si hace uso o no de “props” y sus diferentes interacciones.

Diagrama de estructura básica de una página web  
Header, Navbar, Articles, Footer

# Primer componente

Crear un componente que renderice una imagen aleatoria desde la API <http://avatar.iran.lieara.run/public>. Utiliza el componente en el archivo de entrada de ViteJs (main.tsx)

Añadir props obligatorios al componente para enviar la url y el nombre. Finalmente, renderizar el componente desde un Array de objetos con las claves url y name.