Desarrollo de un *front-end* utilizando React

Objetivos

Esta actividad consiste en la realización del *front-end* de la aplicación web. Por ello, esta actividad permite poner en práctica lo estudiado:

* Creación de páginas con HTML5.
* Aplicación de estilos con CSS3 y metodología BEM.
* Manejo de JavaScript (ECMAScript V6).
* Uso de la librería React.
  + Orientación a componentes.
  + *Hooks*.
  + React Router V6.

Pautas de elaboración

Esta actividad abarca la creación del *front-end* de una aplicación web que usa los lenguajes estudiados hasta el momento: HTML, CSS y JavaScript. Además, haremos uso de la librería React.

El *front-end* desarrollado deberá hacer uso de los principales conceptos que se han estudiado:

* Uso de la librería React para construir un frontal que utilice los componentes funcionales (mínimo diez) y que haga uso de JSX (por ende, HTML).
* Uso de los *hooks* estudiados, useState y useEffect.
* Implementación de un *custom hook* a elección.
* Uso de React Router para implementar diferentes rutas en la aplicación (como mínimo cuatro).
* Aplicar estilos mediante hojas de estilo CSS. Se permite utilizar Bootstrap, Tailwind, SCSS… Pero se debe usar la metodología BEM.

La temática de la aplicación es libre. No obstante, para ser considerada válida, la idea de la aplicación a desarrollar deberá satisfacer los siguientes requisitos (su implementación no forma parte de esta actividad, pero es necesario que la idea sea compatible con ellos):

* Debe requerir de algún tipo de base de datos, que contendrá los elementos característicos de la aplicación que se desea construir.
* Debe requerir un buscador, el cual se usará para interactuar con la base de datos.
* Debe poder realizar al menos dos tareas principales, aquellas para las que está concebida la aplicación.

Algunos de los arquetipos que se recomiendan utilizar como idea base de la aplicación son los siguientes:

* Una **tienda *online*,** es decir, una aplicación que ofrecerá a los usuarios una gran cantidad de productos para poder comprarlos. Cada producto tendrá atributos básicos como nombre, empresa asociada, descripción corta, descripción larga, imagen del producto, precio, etc.
  + Funcionalidad mínima necesaria: compra de uno o varios productos y devolución de pedidos.
  + Base de datos: contiene información de productos, compras y devoluciones.
  + Buscador: sobre los productos de la tienda (nombre, categoría, fabricante y otros atributos específicos del tipo de producto a vender).
* Una **biblioteca *online*,** es decir, una aplicación que se servirá de una colección de libros (que tendrán como atributos el nombre, autor, año de publicación, códigos ISBN de diez y trece dígitos, imagen de portada, sinopsis, críticas, etc.) para que los usuarios puedan alquilarlos por un tiempo determinado.
  + Funcionalidad mínima necesaria: alquiler de libros y extensión del plazo de alquiler.
  + Base de datos: contiene información de libros y alquileres.
  + Buscador: sobre los diferentes libros de la biblioteca (nombre, código ISBN o equivalente, categoría, autor, idioma y descripción).
* Una **plataforma de *streaming* *online*,** es decir, una aplicación similar a otras conocidas en la actualidad, que mostrará a los usuarios un gran catálogo de películas o documentales (con atributos como el nombre, director, año de publicación, sinopsis, críticas, duración, imagen de la película, etc.). En lugar de reproducir la película o documental, se reproducirá el tráiler de YouTube. Los usuarios podrán alquilar estas películas durante un tiempo determinado.
  + Funcionalidad mínima necesaria: alquiler y compra de películas.
  + Base de datos: contiene información de películas, alquileres y compras.
  + Buscador: sobre las películas disponibles (nombre, sinopsis, categoría, año, director, idioma y actores).
* Una **aplicación de inventario,** es decir, una aplicación que permitirá controlar el inventario de una tienda, taller o almacén, lo que permite listar y buscar todos aquellos elementos contenidos en una base de datos. Asimismo, permitirá realizar pedidos a un distribuidor externo para pedir *stock* de los productos que se considere.
  + Funcionalidad mínima necesaria: petición de material a proveedores y creación de pedidos para ser enviados a usuarios.
  + Base de datos: contiene información de productos, pedidos a proveedores y envíos a clientes.
  + Buscador: sobre los productos de la aplicación (nombre, categoría, fabricante y otros atributos específicos del tipo de producto).

Como puedes observar, todos los ejemplos cumplen con los requisitos que se han indicado. Es recomendable que elijas uno de estos cuatro tipos de aplicación, no obstante, puedes realizar cualquier otro tipo de aplicación siempre que cumpla con dichos criterios. En caso de elegir una idea diferente, debes asegurarte de que cumpla con los requisitos indicados. De no ser así, podrías suspender la actividad, por lo que se recomienda que lo valides con tu docente.

Para no complicar demasiado el desarrollo, no será obligatorio incluir en la aplicación ningún mecanismo de *login*. Por tanto, asumiremos que la aplicación se usa de forma anónima y que todas aquellas vistas que estén destinadas a la gestión o administración (en caso de haberlas) serán accesibles a través de una ruta diferente. No obstante, se valorará positivamente incluir esta característica.

Dado que no trabajaremos con ninguna base de datos ni *back-end* en esta actividad, para poder mostrar los datos en las pantallas, utilizaremos los datos de prueba *(mocks)*. No es necesario que el frontal realice peticiones por HTTP para obtener los datos del *back-end,* por lo que, si lo deseas, puedes usar las API de datos estáticos de prueba (públicas en internet).

Se otorgará hasta 1 punto extra si se despliega un frontal que haga uso de la herramienta Vercel (u otras similares).

La entrega consta de dos archivos:

* Una memoria en formato PDF de la actividad.
* Un archivo comprimido con el código de la aplicación, sin node\_modules.

Extensión y formato

La memoria tendrá una extensión máxima de 20 páginas y deberá incluir los siguientes apartados (en este orden):

1. Portada.
2. Índice.
3. Introducción. Se hará una introducción al contenido de la memoria, donde se indicará qué tipo de aplicación se ha decidido construir y sus motivos.
4. Componentes de React. Se listarán y explicarán, brevemente, todos los componentes de React que forman parte de la solución desarrollada.
5. *Hooks* utilizados. Se listarán todos los *hooks* utilizados en la solución desarrollada y qué componente se ha utilizado. Además, se incluirán los *hooks* personalizados que se hayan creado, indicando por qué son necesarios y el valor que aportan a la solución.
6. Vistas. Se mostrarán capturas de pantalla del *front-end* desarrollado junto con una descripción breve de su contenido, que haga énfasis en los estilos CSS que se utilizaron (los más destacables, no es necesario incluir todo el código CSS).
7. Conclusiones.
8. Referencias bibliográficas. Si procede, en formato APA.

Rúbrica

|  |  |  |  |
| --- | --- | --- | --- |
| Desarrollo de un *front-end* utilizando React | Descripción | Puntuación máxima  (puntos) | Peso  % |
| Criterio 1 | Implementación de los componentes funcionales en React (mínimo diez) que utilizan JSX | 2 | 20% |
| Criterio 2 | Uso de los *hooks* principales de React: useState y useEffect | 1 | 10% |
| Criterio 3 | Implementación de un *custom hook* que sea utilizado en algún componente funcional | 2 | 20% |
| Criterio 4 | Uso correcto de React Router para crear diferentes rutas dentro de la aplicación | 1 | 10% |
| Criterio 5 | Modificación del estilo de la aplicación mediante el uso de CSS. Uso de metodología BEM | 1 | 10% |
| Criterio 6 | Implementación de las vistas necesarias para cubrir la funcionalidad mínima de la aplicación | 2 | 20% |
| Criterio 7 | Elaboración de la memoria según las especificaciones, cuidando la presentación y la corrección ortográfica de la misma | 1 | 10% |
|  |  | **10** | **100 %** |