Actividad 1 (Individual o Grupal)

Desarrollo de un *front-end* utilizando React

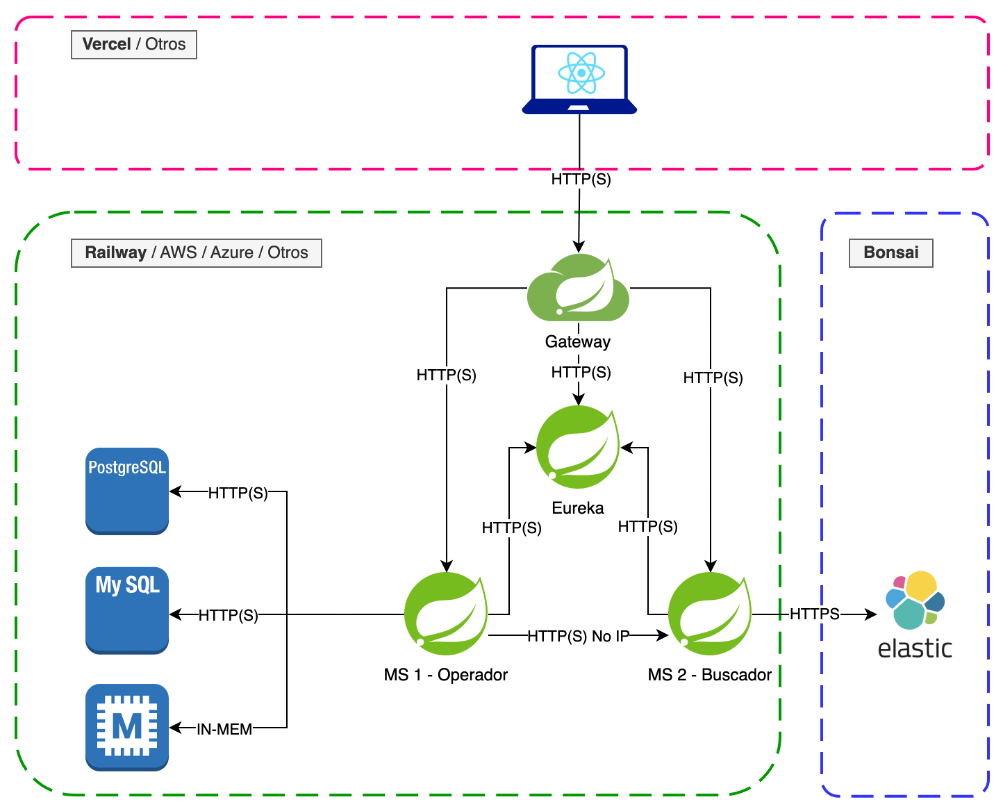
Objetivos

Las tres actividades que se llevarán a cabo en la asignatura Desarrollo Web Full Stack tienen como **objetivo:**

* Desarrollar un *front-end* haciendo uso de HTML5, CSS3, JavaScript y React.
* Desarrollar un *back-end* haciendo uso de una arquitectura orientada a microservicios, donde cada microservicio expondrá una API REST por medio de Java y Spring.
* Integrar ambas partes y desplegar el *front-end* y *back-end* tanto de forma local como pública.
* Hacer uso de bases de datos relacionales y del motor de búsqueda Elasticsearch.

Todo ello ha de estar basado en el **proyecto transversal del máster:** Relatos de papel.

**Si esta actividad se realiza en grupo, la composición del grupo debe quedar registrada en la sección de equipos de esta actividad. Dicha sección será visible unos días antes de la entrega de la actividad. No registrar el grupo podría conllevar la no evaluación de la actividad.**



Esta primera actividad consiste en la realización del ***front-end*** de la **aplicación web** que se construirá tomando como base el proyecto transversal del máster. 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 haciendo uso de los lenguajes estudiado: 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 componentes funcionales (**mínimo de diez**) y que haga uso de de JSX (por ende, HTML).
* Uso al menos de los *hooks* estudiados más importantes, useState y useEffect.
* Implementación de un *custom hook* a elección.
* Uso de **React Router** para implementar diferentes rutas en la aplicación (mínimo de cuatro).
* Aplicar estilos mediante hojas de estilo CSS. Se permite utilizar Bootstrap, Tailwind y SCSS, pero se debe seguir la **metodología BEM**.

La **temática** de la aplicación girará en torno **al proyecto transversal del máster.** Así pues, la **funcionalidad mínima necesaria** que se debe cubrir es la siguiente:

* **Vista de acceso:** muestra una landing. Se redireccionará al usuario a la vista de página principal a los cinco segundos si no realiza ninguna acción.
* **Vista de página principal:** muestra una página principal donde se puedan encontrar algunos de los libros que se venden en la aplicación web. Se dispondrá de una barra de búsqueda en la parte superior que permitirá en el futuro buscar por cualquier término asociado a un libro (autor, código de libro, título de libro…). A los efectos de esta actividad, únicamente filtrará por título de libro. Al hacer clic en un libro, se redirigirá a la vista de libro.
* **Vista de libro:** muestra los detalles de un libro en concreto y permite añadirlo al carrito.
* **Carrito:** tanto en la vista de la página principal como en la de libro, será visible el carrito (ya sea en todo momento o mediante una animación desplegable), donde estarán todos aquellos libros que el usuario haya decidido comprar. Este podrá eliminar libros del carrito o proceder al *checkout.*
* **Vista de checkout:** el usuario verá un resumen de los libros que va a comprar y, si está conforme, podrá proceder al pago. A los efectos de esta actividad, el pago satisfactorio consiste en mostrar una alerta de que el pedido se ha realizado, vaciar el carrito y redirigir a la vista de la página principal (en este orden).

Para no complicar el desarrollo, no será obligatorio incluir en la aplicación ningún **mecanismo de *login*.** Asumiremos que la aplicación se usa de forma anónima.

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

La entrega consistirá **un único archivo ZIP** con los siguientes archivos:

* **Videomemoria obligatoria** en formato MP4 de la actividad.
* Archivo TXT con la **URL del proyecto desplegado** en Vercel u otro.
* **Proyecto con el código** de la aplicación sin node\_modules.

¿Qué es una vídeo memoria?

La vídeo memoria es una grabación donde explicarás con detalle el *software* que has desarrollado. Si estás muy orgulloso del *software* que has construido, ¡podrías incluso publicar este vídeo de forma pública como parte de tu porfolio!

En esta ocasión, la videomemoria tendrá una duración máxima de **diez minutos** y deberá visitar los siguientes aspectos de tu actividad:

* **Componentes de React:** se listarán y explicarán brevemente todos los componentes de React que forman parte de la solución desarrollada. Hablarás aquí también del enrutador que has utilizado.
* ***Hooks* utilizados:** se listarán todos los *hooks* utilizados en la solución desarrollada y en qué componentes. Además, se incluirán aquí los *hooks* personalizados que se hayan creado, indicando por qué son necesarios y el valor que aportan a la solución.
* **Vistas:** mostrarás cómo el proyecto puede arrancarse de forma local. Una vez desplegado en local, se mostrarán las diferentes vistas 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 hablar de todo el código CSS).
* **Despliegue:** se mostrará el proyecto desplegado en Vercel correctamente.

Rúbrica

|  |  |  |  |
| --- | --- | --- | --- |
| Criterio | Descripción | Puntuación máxima  (puntos) | Peso  % |
| Criterio 1 | Implementación de componentes funcionales en React (mínimo de diez) haciendo uso de JSX. | 0,5 | 5 % |
| Criterio 2 | Uso de los *hooks* principales de React: useState y useEffect. | 0,5 | 5 % |
| Criterio 3 | Implementación de un *custom hook* que sea utilizado en algún componente funcional. | 1 | 10 % |
| 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 y funcionamiento correcto de las vistas necesarias para cubrir la funcionalidad mínima de la aplicación. | 4,5 | 45 % |
| Criterio 7 | Elaboración de la videomemoria obligatoria según las especificaciones, ajustándose a la duración aproximada y cubriendo los aspectos solicitados. | 1 | 10 % |
| Criterio 8 | Despliegue en Vercel | 0,5 | 5 % |
|  |  | **10** | **100 %** |