Introducción al curso

El siguiente curso de React con Spring Boot va a estar estructurado en torno al desarrollo de aplicaciones web, centrándose específicamente en el frontend y en la integración con Spring Boot en el backend. Se tienen en cuenta los siguientes puntos clave:

* **Introducción al curso:** El curso se enfoca en el desarrollo web utilizando Spring Boot.
* **Frontend y Spring Boot:** Se abordará tanto el frontend como Spring Boot en el backend.
* **Desarrollo con React:** La mayor parte del curso se realizará utilizando React para construir aplicaciones. Se menciona la gestión del estado de la información y los datos en React.
* **Simulación de datos:** Al principio, se utilizarán datos simulados o mockup data, como arreglos de datos en JavaScript con React JSON, para desarrollar las aplicaciones.
* **Integración con datos reales:** Posteriormente, se integrarán datos reales provenientes del servidor en el backend de Spring Boot.
* **Desacoplamiento y separación de partes:** Se resalta la importancia de mantener una buena separación entre el frontend y el backend, así como una integración adecuada entre ellos.
* **Objetivos del aprendizaje:** Se enfatiza el deseo de aprender tanto el desarrollo frontend como la integración con Spring Boot en el backend.
* **Interacción con Spring Boot:** Se espera que los estudiantes adquieran habilidades para integrar el frontend desarrollado con React con el backend proporcionado por Spring Boot.

React Js

Se cubrirán los siguientes temas con relación a los componentes específicos de React:

1. **Framework/librería de interfaz de usuario:** Se introduce un gran framework o librería para trabajar con componentes de interfaz de usuario.
2. **Introducción a React:** Se ofrece una visión general de React como una biblioteca basada en componentes para trabajar con el DOM HTML. Se hace énfasis en la simplicidad de manipular el DOM utilizando etiquetas HTML y la integración total con JavaScript moderno.
3. **JavaScript completo:** Antes de profundizar en React, se ofrece una introducción completa a JavaScript, cubriendo conceptos como objetos, arreglos y funciones.
4. **Componentes en React:** Se explica cómo React se basa en el concepto de componentes y cómo estos pueden subdividirse para formar una aplicación más grande pero modular.
5. **Hooks de React:** Se exploran los diferentes hooks de React, como **useState** y **useEffect**, para manejar el estado y los efectos en la aplicación. Se presenta el uso de estos hooks en ejemplos prácticos, como una aplicación de contador y una aplicación de factura.
6. **Redux y useReducer:** Se introduce **Redux** como un estado global para la aplicación, más potente que **useState**, y se compara con **useReducer**. Se explora el uso de **Redux** en una aplicación de carrito de compras, junto con otras funcionalidades.
7. **Rutas con React Router:** Se explica cómo gestionar las rutas de la aplicación utilizando React Router, asignando componentes a rutas específicas. Se menciona la capacidad de tener rutas anidadas y su importancia en la estructura de la aplicación.
8. **Funcionalidades avanzadas:** Se abordan funcionalidades adicionales como el manejo de sesiones con JWT para el inicio de sesión, ventanas modales y validaciones de formularios. Se profundiza en el uso de **useContext** para el contexto de React y su integración con **useReducer**.
9. **Despliegue en la nube:** Finalmente, se discute el proceso de despliegue de las aplicaciones de React tanto en la nube como del backend.

Spring Boot

Los siguientes temas se cubrirán en la parte del Spring Boot del curso con relación al backend:

1. **Persistencia con Hibernate/JPA:** Se trabaja con persistencia de datos utilizando Hibernate/JPA, que es una tecnología ampliamente utilizada para mapear objetos de Java a bases de datos relacionales.
2. **MySQL como base de datos:** Se utiliza MySQL como base de datos relacional para almacenar y recuperar datos en la aplicación.
3. **Repositorios con Spring Data JPA:** Se hace uso de Spring Data JPA y los repositorios de Spring para interactuar con la base de datos de forma declarativa y simplificada.
4. **Clases de Servicios o Service para lógica de Negocio:** Se implementan clases de servicios o service para encapsular la lógica de negocio de la aplicación.
5. **Uso del patrón DTO en el Service:** Se utiliza el patrón DTO (Data Transfer Object) en el servicio para definir objetos que contienen datos a ser transferidos entre el frontend y el backend.
6. **Controladores de Spring @RestController:** Se crean controladores utilizando la anotación @RestController para manejar las solicitudes HTTP y proporcionar una API RESTful.
7. **API Restful con operaciones CRUD:** Se implementan operaciones CRUD (Create, Read, Update, Delete) en la API RESTful para interactuar con los datos almacenados en la base de datos.
8. **Validación desde el Backend:** Se realizan validaciones de datos en el backend para garantizar la integridad y consistencia de la información.
9. **Comunicación entre Frontend React y Backend Spring:** Se integra el frontend desarrollado en React con el backend de Spring para que ambos se comuniquen y trabajen juntos con datos reales.
10. **Spring Security JWT:** Se implementa seguridad en la aplicación utilizando Spring Security con JSON Web Tokens (**JWT**) para autenticación y autorización.
11. **Password encriptado con bcrypt:** Se utiliza el algoritmo **bcrypt** para encriptar las contraseñas de los usuarios y garantizar su seguridad en la aplicación.
12. **Integración de React con el login JWT en el Backend Spring:** Se realiza la integración de React con el sistema de login basado en JWT en el backend de Spring para gestionar la autenticación de los usuarios.