APLICACIÓN GESTIÓN DE PEDIDOS

A continuación, debido a la falta de tiempo material para realizar la práctica, se detallan los aspectos que no se han podido desarrollar.

1.- Autenticación: Cada llamada que se vaya a realizar en nuestra aplicación deberá comprobar que el usuario se encuentra logado en el sistema. Para ello, se podría utilizar la librería JWT de Angular. Dicha librería nos permite que cualquier solicitud que se realice mediante HttpClient, incluirá un token en el encabezado de la llamada. Al incluir el modulo de JWT, deberemos indicar la forma de obtener dicho token (por ej. en la sesión del navegador). Dicho token deberá haberse generado en la parte back cuando el usuario se haya logado correctamente y se retornará al cliente. Este token servirá de acuerdo entre la parte back y la parte cliente.

2.- Pestaña Gestión de pedidos: Tendremos un listado de pedidos en el cual cada fila contendría, dependiente del estado del mismo, una botonera de posibles acciones.

Si el pedido está Emitido, se mostrará un botón para cancelarlo siempre y cuando la fecha del sistema sea la misma que la del pedido.

Si el pedido está Entregado, se mostrará un botón para pagarlo.

Si el pedido está Pagado o Cancelado no mostraremos ningún botón.

Para el cambio de estado de pedidos se solicitará confirmación al usuario.

Para facilitar la consulta de los pedidos, se habilitará encima del listado un filtro que permita obtener al usuario solo los registros que necesite:

-Se mostrará un Checkbox para indicar si solo se quiere obtener los pedidos que coincidan con el usuario logado en el sistema.

-Se mostrará un dataList con los posibles estados, de manera que se puedan seleccionar uno, varios o ninguno.

-Se incluye el botón de búsqueda para realizar la llamada al backend con los filtros informados.

3.- Creación pedidos: Para la creación de pedidos, dentro de la pestaña de de gestión de pedidos se habilitaría un botón de creación de pedido el cual nos llevaría mediante navegación horizontal a un formulario. Este formulario contendría los campos usuario, tres combobox que se cargarian con los posibles valores obtenidos de la tabla maestra; y un botón de crear pedido. A la hora de crear el pedido, se leerá la fecha y hora del sistema para comprobar si la fecha en la que se servirá es la de hoy o la de mañana. Se establecerá por defecto el estado Emitido.

4.- Proceso de entrega de pedidos: Se configura mediante la funcionalidad Scheduled de Spring, que todos los días a las 13:00 (configurado mediante cron).

5.- Para la creación de nuevos platos, se habilitará una pestaña distinta para la gestión de estos productos. De esta forma se podrá hacer una gestión de los platos ya creados, modificarlos o eliminarlos según desee el usuario

6.- Con respecto a la parte backend, se crean los servicios Rest en la clase PedidappController.

Esta clase controller se comunicará con una capa manager donde se realizará toda la lógica de negocio necesaria. Será la capa manager la que tenga acceso a la capa de lógica de datos (DAO) para la comunicación con la base de datos.

7.- Para la realización de pruebas unitarias, se utilizará JUnit.

Se creará un test unitario para cada función que se pueda realizar desde la parte cliente:

Crear pedidos, Cancelar pedido, Pagar pedido, Crear plato, Modificar Plato, Eliminar Plato, Lanzar proceso de entrega de pedidos, etc.