Final Laboratorio IV

IMPORTANTE:

- Ejecutar el comando **npm install** para poder empezar a trabajar.
- Prestar atención a los scripts del archivo ***package.json,*** para poder iniciar la aplicación y ejecutar nuestra “api” con JsonServer.
- No todos los componentes y otros recursos están creados. Prestar atención en donde está parada la terminal al momento de crear un recurso para respetar el orden de las carpetas.
- Los módulos necesarios para los forms, diálogos emergentes o peticiones http ya están incluidos en el repositorio.
- Se debe tener cuidado con las operaciones asincrónicas perteneciente a los observables (gestionarlas adecuadamente) y las promesas.
- Se evaluará también en que carpeta se crean los recursos pedidos y como se modularizan los servicios.
- **La aplicación, para la aprobación, debe funcionar.**

La aplicación trata de un gestor de vehículos para un concesionario, donde se podrá ver el listado del inventario de vehículos, agregarlos y eliminarnos en una sola página. Ya hay usuarios en base de datos listos para poder loguearse y empezar a trabajar. Para que se puedan cumplir los requisitos de esta aplicación se necesita:

- Desarrollar los modelos e interfaces necesarios para el mapeo de los datos que trae la API.
- Crear un servicio para que se puedan hacer peticiones GET, POST, PUT y DELETE. Se debe utilizar la libreria HttpClient.
- Completar el componente de **Login** y crear los recursos necesarios para que este sea funcional. Para esto debe crear otro servicio que se encargue de las funciones correspondientes del logueo. Teniendo en cuenta que en los requerimientos de los puntos de más adelante se necesitará crear un componente **Home**, en el caso de ingresar un usuario incorrecto, se debe indicar lo sucedido en un alert, en el caso correcto se llevará al usuario al componente de Home para que vea la lista, los agregue y los elimine.
- Crear un componente **nav-bar** que tenga la opción de loguearse (si es que no está logueado) y de cerrar sesión (si es que está logueado). Esta nav-bar debe estar tanto en el componente **Landing** como en Home

- Crear el componente **Home** que debe contener otros dos componentes (crealos también): **addVehicle** y **viewVehicles.**
    - **viewVehicles** es un componente encargado de mostrar los vehículos registrados en una tabla.
    - **addVehicle** es un componente que permite crear un vehículo y agregarlo a la lista que se muestra. Cuando un vehículo se crea correctamente, debe aparecer un alert indicándolo.

- Se debe agregar al lado de cada estudiante de la tabla del componente **viewVehicles** un botón que permita eliminarlo. Pensar de que manera podemos hacerlo a través de la comunicación entre componentes padres e hijos.

Puntaje

Angular: 90 pts.
Maquetación HTML y CSS: 10 pts.