Ejercitación integral DDS - Ejercicio C

- Desarrollar el **backend** y **frontend** de una aplicación utilizando 2 instancias de Visual Studio Code, una para el back en port 4000 y la otra para el front en port 3000. Se deberá generar una funcionalidad nueva a los proyectos existentes:
- El proyecto del backend está en el siguiente repositorio GIT **https://github.com/arcba/DDSBackV3.git**
- El proyecto del frontend está en el siguiente repositorio GIT **https://github.com/arcba/DDSFront.git**

Clonar ambos repositorios en la carpeta ***tmp*** de la estación de trabajo en linux. Y realizar las modificaciones solicitadas en dichos proyectos.

Importante para instalar las librerías dependientes del back y del front:

1. Eliminar (si existe) el archivo package-lock.json en ambos proyectos
2. ejecutar el comando **npm install** en ambos proyectos (misma ruta donde está el archivo **package.json**).

Tiempo de resolución: 1h 30 minutos.

Desarrollo del back: se deberá agregar una funcionalidad de consulta para un recurso nuevo "deudores"

- ✓ Desarrollar el código para conectar con la BD, crear la tabla si no existe e insertar los valores asignados. En el archivo script.sql (que está en la raíz del proyecto) están los script sql necesarios.
- ✓ Crear el modelo de sequelize que mapee la tabla deudores, revisar la estructura de la tabla dada en el punto anterior.
- ✓ Programar para la ruta deudores lo siguiente:
  - Un endpoint Deudores que mediante el verbo/método **GET** reciba un parámetro: DeudorDescripcion y utilizando el model ya desarrollado en el punto anterior recupere los registros filtrando por el parámetro

recibido. Deberá devolver todos los registros en cuyo campo DeudorDescripcion contenga el valor del parámetro recibido. Devolver todos los campos de la tabla

- La aplicación deberá registrar como middleware un **router** con la ruta "/api/deudores".

✓ Pruebe las siguientes url desde el browser:

- http://localhost:4000/api/deudores

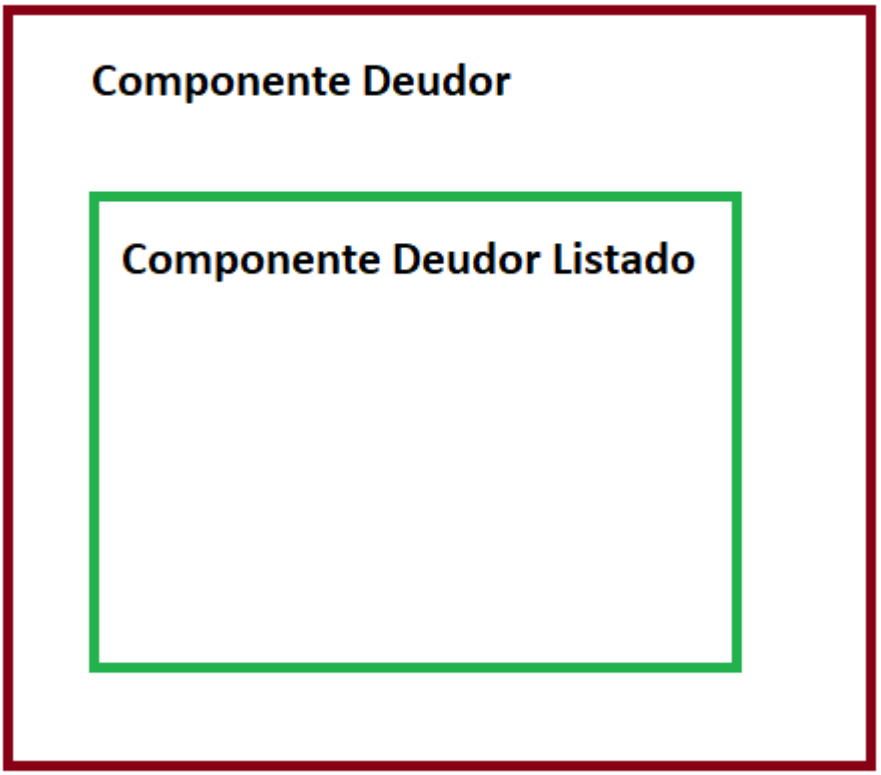
  Debe devolver todos los registros

- http://localhost:4000/api/deudores?DeudorDescripcion=Paula

  Debe devolver los registros que contengan "Paula"

Desarrollo del frontend: se deberá agregar una interface de consulta para el recurso "deudores" desarrollado en el backend

✓ Desarrollar un componente Deudor con el siguiente esquema:



✓ Agregar el código necesario para poder realizar la consulta de deudores. Se deberá realizar los componentes de react y consumir el backend desarrollado en el punto anterior.

✓ En el componente Deudor, incluir la interface de búsqueda con el campo a filtrar ("DeudorDescripcion") y el botón buscar.

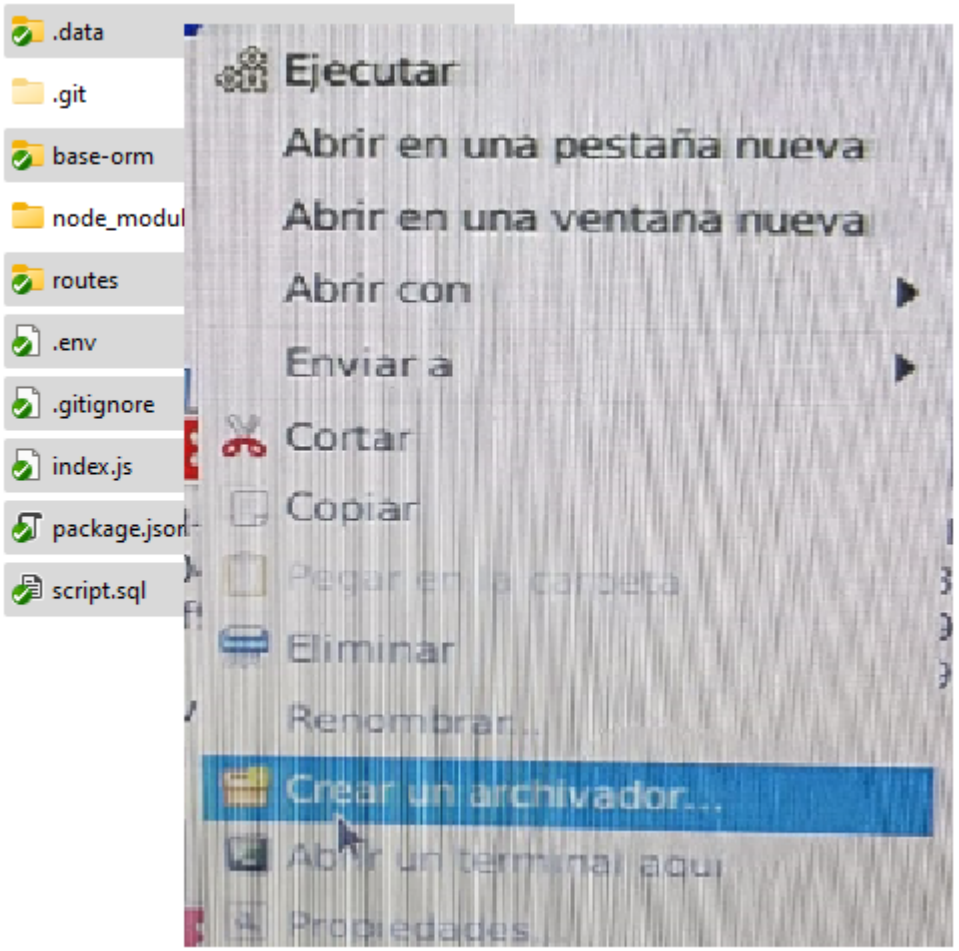
Deberá cumplir los siguientes requisitos:

- Consumir vía axios el endpoint del back
- Utilizar para el formulario de búsqueda la librería react-hook-form

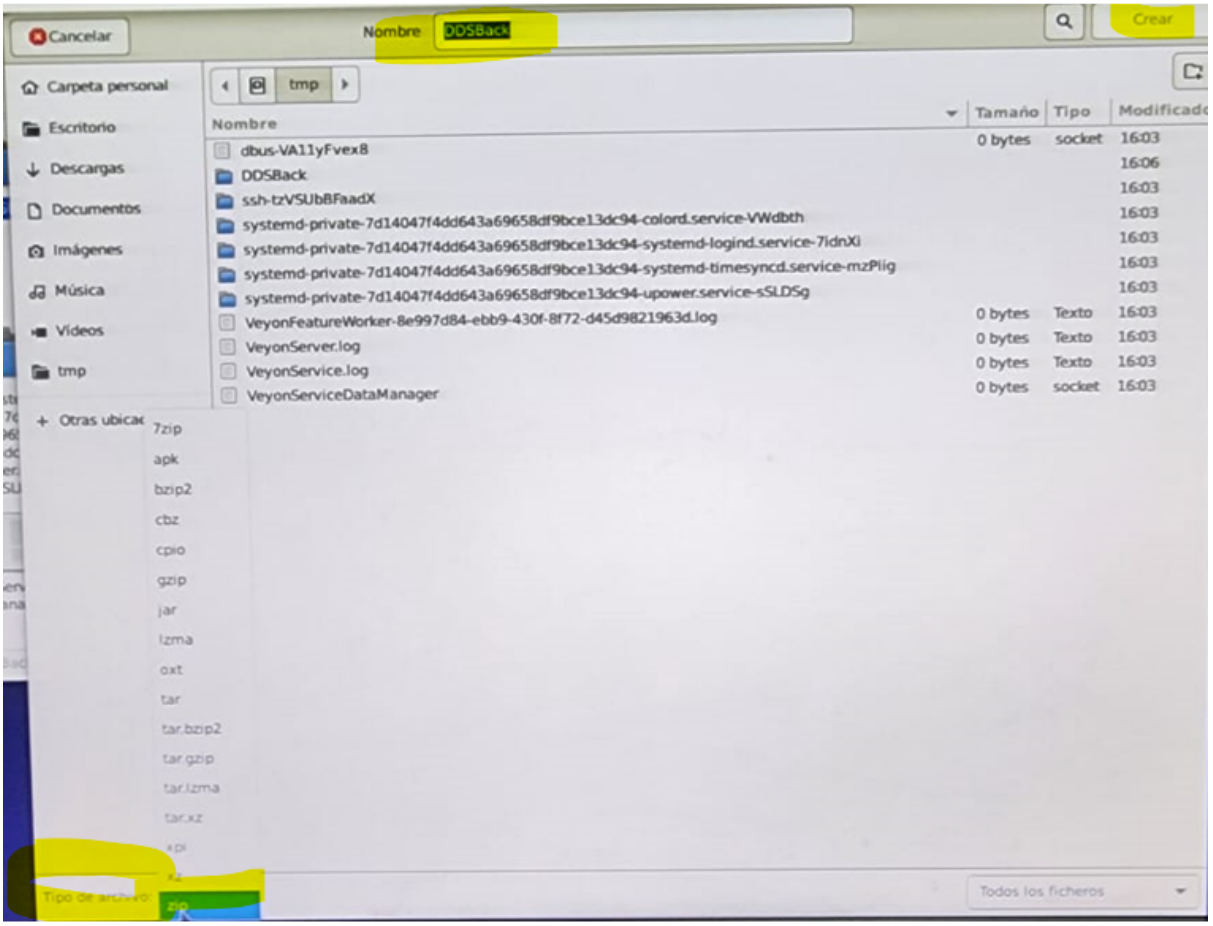
- ✓ En el componente DeudorListado incluir una tabla de html para mostrar los resultados de la búsqueda, incluya en la misma todos los registro y sus campos, (no se pide paginación)

- ✓ Agregar al componente menú de la aplicación, el acceso al nuevo componente.

Para la entrega del ejercicio, realizar dos archivos .ZIP, uno para cada proyecto, sin incluir la carpeta **node_modules**: front_legajo.zip y back_legajo.zip (reemplazar legajo por su numero de legajo) y el día del parcial indicará donde subir o enviar dichos archivos.
Los archivos ZIP se deberán realizar **SIN** la carpeta **node_modules**. Para ello dentro de la carpeta donde se encuentra el proyecto, Seleccionar todos los archivos y carpetas (menos node_modules) y del menú flotante seleccionar **Crear un Archivador**



Indicar el formato de compresión (margen inferior izquierdo), seleccionando **ZIP**, e indicar el nombre y presionar el botón CREAR



TIPS PARA RESOLUCIÓN:

Observaciones para el Laboratorio:

- equipos debian con problemas al ejecutar npm install
  deberan ejecutar el comando: npm config delete registry (elimina el proxy golum)

Observaciones back:

- El back ya está configurado para trabajar en el puerto 4000 (ver index.js)

```
const port = 4000;
app.listen(port, () => {
  console.log(`Servidor iniciado en el puerto ${port}`);
});
```

- recordar que el backend no inicia el browser, si quiero testearlo debo abrir el browser explícitamente o usar postman
- Si la tabla pymes ya existe, debemos borrarla para que se autogenere y se ejecute el nuevo código que permita crear la tabla artículos.
- en Sequelize las validaciones de las propiedades de los modelos solo se usan en altas y modificaciones… para una consulta no hacen falta implementarlas.
- el back podemo ejecutarlo con: npm run dev
- en relación al script.sql, copiar la instrucción insert completa, es decir desde el insert hasta el final, observe que a diferencia de artículos, no se pasa el campo IdDeudor el cual al ser autonumérico se genera solo. Conclusión se puede pasar el Id autonumérico como está en articulo o no como está en deudores.

```
insert into deudores (DeudorDescripcion, MontoAdeudado) values
('Juan Pérez', 5320),
('María Gómez', 97853),
('Carlos López', 654987),
('Ana Ramírez', 62378),
('Luis González', 5486),
('Laura Martínez', 75321),
('Pedro Rodríguez', 42356),
```

- ***Paso a paso simulacro back:***

  1 - modificar sqlite-init.js (para crear y cargar la tabla según script.sql). No olvidar exportar la definición de deudor al finalizar

  2 - modificar sequelize-init.js (para crear el modelo deudor)

  3 - crear el /routes/deudores.js (para crear el método GET que reciba el parámetro y devuelva los registros que coincidan)

  4 - modificar el index.js para que se monte la ruta creada en punto anterior (acordarse de importarla)

Observaciones Front:

- El front arranca por defecto en el puerto 3000
- Asegúrese al probar el front que el código del back está en ejecución, porque el front va a consumir el back.
- Por simplicidad todos los componentes están en una única carpeta!
- No se pide implementar servicios, pero no está mal si así lo hiciera.
- ***paso a paso simulacro front***
  1 - crear Deudores.jsx desde Articulos.jsx
  2 - crear ListadoDeudores.jsx desde ListadoArticulos.jsx
  3 - agregar a Menu.jsx el link a deudores en html
  4 - en app.js importar deudores.jsx y agregarlo Route en el html

```
<Route path="/deudores" element={<Deudores />} />
```