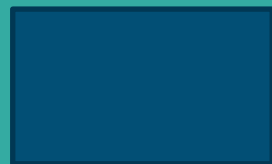


PRUEBA- TECNICA- DSJR-2022-

8 FEBRERO

MINISTERIO PÚBLICO
JULIO CAYAX





ÍNDICE

INTRODUCCIÓN	1
OBJETIVO.....	1
1. Módulos del programa y su descripción.....	2
2. Requerimientos para el uso del sistema.....	3
2.1. Sistema operativo Windows	3
2.2. Requerimientos mínimos para el uso del sistema.....	3
3. Convenciones del sistema	4
4. Base de datos	5
5. Backend	6
6. Frontend.....	9
7. Control de versiones.....	11

INTRODUCCIÓN

El presente manual es una guía completa para la gestión y administración de la prueba técnica. Este manual le permitirá al usuario aprender a utilizar todas las funcionalidades del mismo sin ninguna complicación.

OBJETIVO

Dar a conocer a los usuarios finales las características y las formas de funcionamiento de la prueba técnica, donde se les facilite a los lectores la información necesaria para utilizar el programa de manera adecuada. Al mismo tiempo brindar un documento que dé a conocer como se utiliza el programa, mediante una descripción detallada e ilustrada a través de opciones.

1. Módulos del programa y su descripción

A continuación, se listan y detallan los módulos que se implementaron dentro del programal:

- **Base de datos:** Se creo una base de datos dentro de MySQL con una tabla llamada Fiscalía.
- **Backend:** Se desarrollo mediante Java EE.
- **Frontend:** Se desarrollo mediante React JS.

2. Requerimientos para el uso del sistema




2.1. Sistema operativo Windows

- Windows 7
- Windows 8
- Windows 8.1
- Windows 10
- Versiones superiores
- Node.JS
- Java EE
- Visual Studio
- IDE de Java
- XAMPP
- MySQL

2.2. Requerimientos mínimos para el uso del sistema


- Memoria RAM 2GB mínimo
- Procesador Intel Core i3 o superior
- Disco duro de 128 GB o superior
- Monitor al criterio de la institución

3. Convenciones del sistema















Término	Significado	Icono
Botón de guardar	Al dar clic sobre un botón de guardar, almacena los datos ingresados.	
Botón de editar	Al dar clic sobre un botón de editar, modifica los datos seleccionados.	
Botón de eliminar	Al dar clic, elimina los datos seleccionados.	
Cajas de texto	Diferentes campos que deben ser llenados para guardar información.	<div>Nombre _____</div> <div>Dirección _____</div> <div>Número de teléfono _____</div>

4. Base de datos

Se realizó un modelo relacional para el desarrollo de la prueba, para la realización del modelo relacional se utilizó la herramienta Toad Data Modeler.

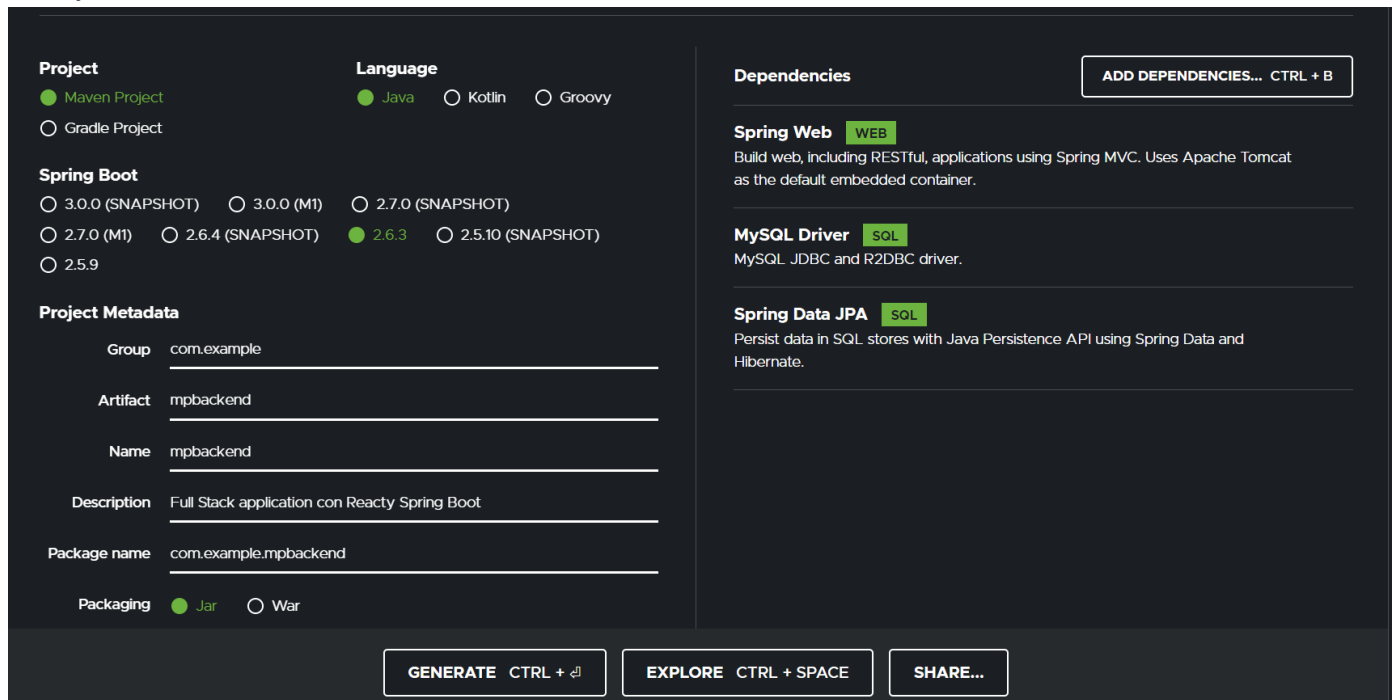
Fiscalia			
 Fiscalia_ID	Int	NN	(PK)
Nombre	Char(30)	NN	
Direccion	Char(50)	NN	
Telefono	Char(20)	NN	

Se generó el script de la base de datos y se implementó en una nueva base de datos con phpmyadmin.

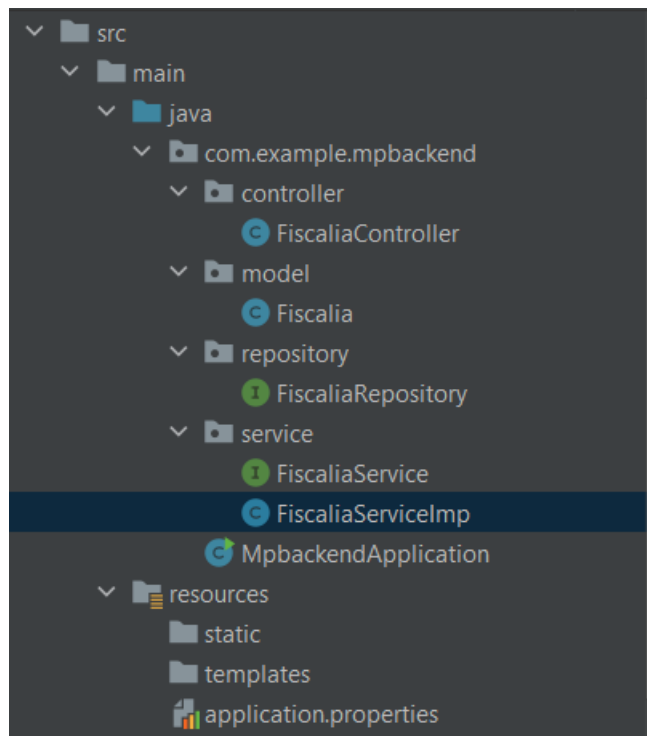
+ Opciones							
				id	address	name	number
<input type="checkbox"/>		Editar		Copiar		Borrar	1 Quetzaltenango Fiscalia 1 4569234
<input type="checkbox"/>		Editar		Copiar		Borrar	2 Mazatenango Fiscalia 2 4569234
<input type="checkbox"/>		Editar		Copiar		Borrar	3 Peten Fiscalia 3 99203912
<input type="checkbox"/>		Editar		Copiar		Borrar	6 Momostenango Fiscalia 4 2342332

5. Backend

Se crearon las dependencias para la creación del proyecto en Java EE, donde se utilizó el IDE IntelliJ DEA.



Se crearon los paquetes de Java, controladores, modelos y servicios necesarios para la construcción interna del backend.



Se creó la conexión con la base de datos de MySQL con los parámetros necesarios.

```
#Configuracion
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost:3306/mp
spring.datasource.username =
spring.datasource.password =

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

Mediante los controladores, se crearon los parámetros necesarios para realizar las operaciones básicas de CRUD.

```
@RestController
@RequestMapping("/fiscalia")
@CrossOrigin
public class FiscaliaController {
    @Autowired
    private FiscaliaService fiscaliaService;

    @PostMapping("/add")
    public String add(@RequestBody Fiscalia fiscalia){
        fiscaliaService.saveFiscalia(fiscalia);
        return "Nueva Fiscalia agregada";
    }

    @GetMapping("/getAll")
    public List<Fiscalia> getAllFiscalias() { return fiscaliaService.getAllFiscalias(); }
```

Mediante el modelo se diseñaron los parámetros que tiene la tabla Fiscalia, tanto como la creación de sus constructores, setters y getters.

```

FiscaliaImp.java x MpbbackendApplication.java x FiscaliaController.java x application.properties x Fiscalia
private int id;
private String name;
private String address;
private String number;

public Fiscalia() {

}

//Generacion de Getters y Setters
public int getId() { return id; }

public void setId(int id) { this.id = id; }

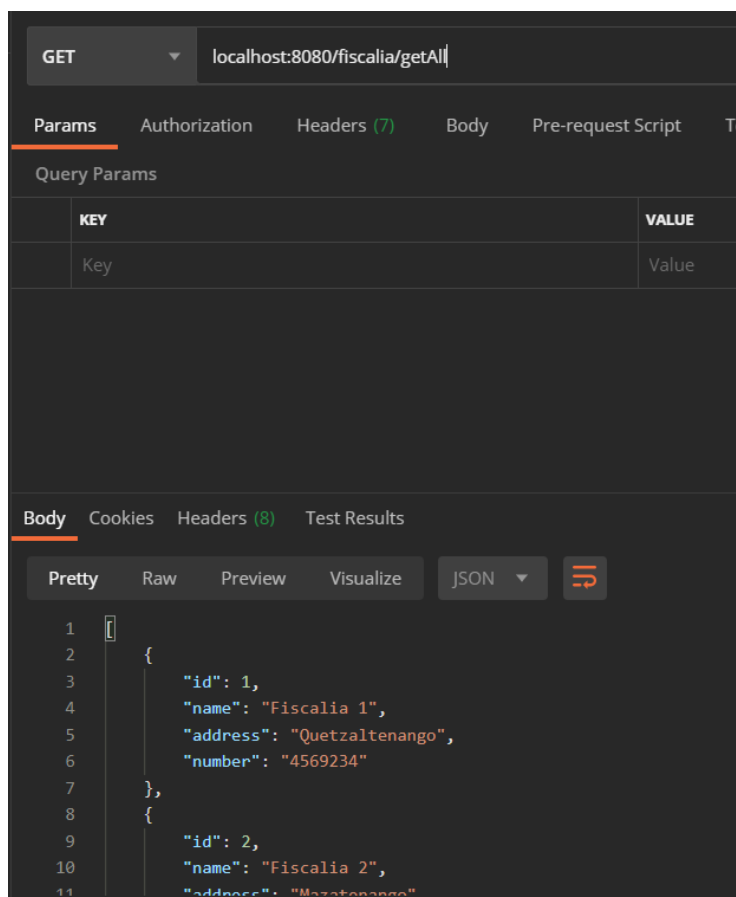
public java.lang.String getName() { return name; }

public void setName(java.lang.String name) { this.name = name; }

public java.lang.String getAddress() {

```

Se hicieron las pruebas correspondientes mediante el programa Postman, previo a desarrollar la parte de frontend.



6. Frontend

Se creó la parte de frontend con React JS, donde se utilizaron varias clases de Java. La más importante es la App.js donde se implementa la clase Fiscalia y el diseño de la app, que es un appbar.

```
mpfrontend > src > JS App.js > ...
1  import './App.css';
2  import AppBar from './components/Appbar'
3  import Fiscalia from './components/Fiscalia';
4
5  function App() {
6    return (
7      <div className="App">
8        <AppBar/>
9        <Fiscalia/>
10     </div>
11   );
12 }
13
14 export default App;
```

Se creó una clase de Java llamada Fiscalia.js, donde se hace conexión con las rutas de backend. Se utilizan funciones como useState, para guardar los datos de la caja de texto y se utiliza Material UI para darle una interfaz vistosa a la aplicación.

```
//Ingreso de variables
const [name, setName] = React.useState('')
const [address, setAddress] = React.useState('')
const [number, setNumber] = React.useState('')
//Ver todos los datos
const [fiscalias, setFiscalias] = React.useState([])

const handleClick = (e) => {
  e.preventDefault()
  const fiscalia = { name, address, number }
  console.log(fiscalia)
  //Agregar Ruta Backend
  fetch("http://localhost:8080/fiscalia/add", {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(fiscalia)
  }).then(() => {
    console.log("Nueva Ficalia Agregada")
  })
}
```

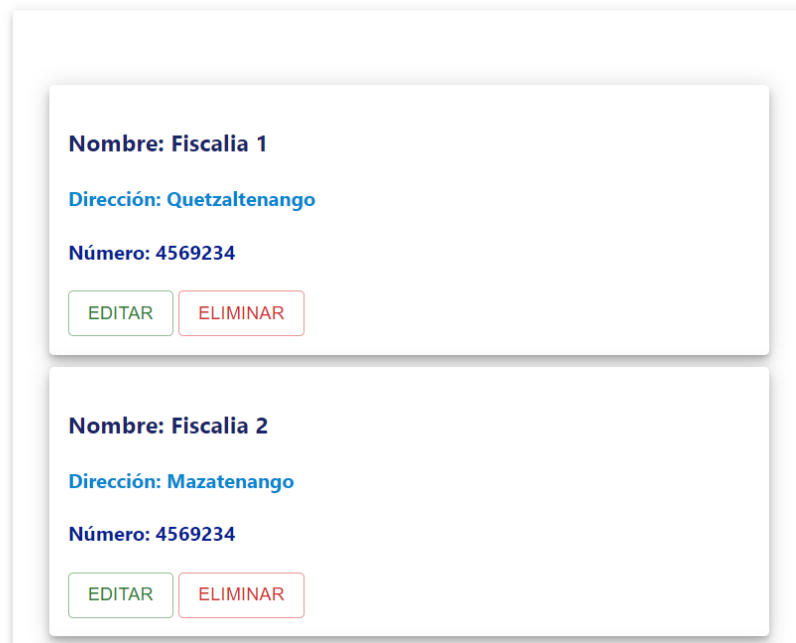
Se instalaron paquetes de Material UI, para el desarrollo de la interfaz, donde se diseñó una interfaz vistosa al usuario, donde se deben ingresar los datos de las fiscalías.



The screenshot shows a web application interface with a blue header bar. On the left of the header is a hamburger menu icon, and in the center is the text 'Ministerio Público'. On the right is a search bar with a magnifying glass icon and the placeholder text 'Buscar...'. Below the header, there is a white card titled 'Ingreso de Fiscalías'. Inside the card, there are three input fields labeled 'Nombre', 'Dirección', and 'Número de teléfono'. Below these fields is a blue button labeled 'GUARDAR'.

También se desarrolló de una forma básica y fácil para el usuario la visualización de los datos.

Listado General



The screenshot shows a web application interface with a white background. At the top, there is a section titled 'Listado General'. Below this title, there are two white cards, each representing a fiscalía entry. The first card shows 'Nombre: Fiscalía 1', 'Dirección: Quetzaltenango' (in blue), and 'Número: 4569234'. Below the text are two buttons: 'EDITAR' (green) and 'ELIMINAR' (red). The second card shows 'Nombre: Fiscalía 2', 'Dirección: Mazatenango' (in blue), and 'Número: 4569234'. Below the text are two buttons: 'EDITAR' (green) and 'ELIMINAR' (red).

7. Control de versiones

Se utilizó la plataforma GitHub para la creación del repositorio público, donde se llevó constancia de todos los commits y del desarrollo de la aplicación fullstack.

<https://github.com/Jul10CC/PRUEBA-TECNICA-DSJR-2022-1-JULIO-CAYAX.git>

