Microservicio CRUD Usuarios SPRING BOOT.

Vamos a agregar al proyecto anterior creado para el login de acceso al proyecto el CRUD de Usuarios haciendo las modificaciones que se necesiten en el front y en el back para poder realizar la conexión entre ellos.

1. CRUD de Usuario (FrontEnd).

En este módulo se debe hacer la capa de presentación por ende debemos tener en cuenta los siguientes pasos para crear la estructura del proyecto que permita solucionar lo requerido:

1. Crear las clases modelo de datos que permitan acceder a los objetos referenciados, TipoDocumento, Usuario y UsuarioResponse.
2. Agregar los encabezados de los métodos a utilizar sobre la interface IClienteTienda en donde se hace la conexión a la aplicación web.
3. Implementar los nuevos métodos definidos en la interface en la clase ClientImp.
4. Agregar los métodos al controlador que permitan desarrollar cada una de las operaciones del CRUD.
5. Las pantallas o vistas que permitan presentar los datos al usuario y de paso interactuar los métodos correspondientes.

1. Modelo de Datos.

TipoDocumento:

```java
package com.mintic.tiendafront.modelo;

public class TipoDocumento {

    private Long id;

    private String tipo;

    public Long getId() {
        return id;
    }

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

    public String getTipo() {
        return tipo;
    }

    public void setTipo(String tipo) {
        this.tipo = tipo;
    }

}
```

Usuario:

```java
package com.mintic.tiendafront.modelo;

public class Usuario {
    private Long id;

    private Long idTipoDocumento;

    private String numeroDocumento;

    private String nombre;

    private String email;

    private String password;

    private String nombreUsuario;

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public Long getId() {
        return id;
    }

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

    public Long getIdTipoDocumento() {
        return idTipoDocumento;
    }

    public void setIdTipoDocumento(Long idTipoDocumento) {
        this.idTipoDocumento = idTipoDocumento;
    }

    public String getNumeroDocumento() {
        return numeroDocumento;
    }

    public void setNumeroDocumento(String numeroDocumento) {
        this.numeroDocumento = numeroDocumento;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getNombreUsuario() {
        return nombreUsuario;
    }

    public void setNombreUsuario(String nombreUsuario) {
        this.nombreUsuario = nombreUsuario;
    }

}
```

UsuarioResponse:

```java
package com.mintic.tiendafront.modelo;

public class UsuarioResponse {

    private Long id;

    private TipoDocumento idTipoDocumento;

    private String numeroDocumento;

    private String nombre;

    private String password;

    private String email;

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    private String nombreUsuario;

    public Long getId() {
        return id;
    }

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

    public TipoDocumento getIdTipoDocumento() {
        return idTipoDocumento;
    }

    public void setIdTipoDocumento(TipoDocumento idTipoDocumento) {
        this.idTipoDocumento = idTipoDocumento;
    }

    public String getNumeroDocumento() {
        return numeroDocumento;
    }

    public void setNumeroDocumento(String numeroDocumento) {
        this.numeroDocumento = numeroDocumento;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

}
```

```
    public String getNombreUsuario() {
        return nombreUsuario;
    }

    public void setNombreUsuario(String nombreUsuario) {
        this.nombreUsuario = nombreUsuario;
    }

}
```

2. Modificar interface de Cliente.

```
package com.mintic.tiendafront.cliente;

import java.util.List;

import com.mintic.tiendafront.modelo.LoginDto;
import com.mintic.tiendafront.modelo.TipoDocumento;
import com.mintic.tiendafront.modelo.Usuario;
import com.mintic.tiendafront.modelo.UsuarioResponse;

public interface IClientTienda {

    public int login(LoginDto loginDto);

    public List<UsuarioResponse> getUsuarios();

    public UsuarioResponse nuevoUsuario(Usuario usuarioDto);

    public UsuarioResponse buscarUsuario(Long id);

    public int borrarUsuario(Long id);

    public List<TipoDocumento> getTipoDocumento();

}
```

3. Modificar Clase Cliente de Implementación.

```java
    @Override
    public List<UsuarioResponse> getUsuarios() {

        try {
            Mono<List> response = webClient.build().get().uri(URL + "/usuarios").retrieve()
                    .bodyToMono(List.class);

            return response.block();
        } catch (Exception e) {

            return null;
        }

    }

    @Override
    public UsuarioResponse nuevoUsuario(Usuario usuarioDto) {

        try {

            UsuarioResponse u = null;
            Mono<UsuarioResponse> response = webClient.build().post().uri(URL + "/usuarios")
                    .body(Mono.just(usuarioDto), UsuarioResponse.class).retrieve().bodyToMono(UsuarioResponse.class);

            u = response.block();
            return u;

        } catch (WebClientResponseException e) {
            e.getMessage();
            System.out.println("---->" + e.getMessage());
            return null;
        }

    }
    @Override
    public UsuarioResponse buscarUsuario(Long id) {
        // TODO Auto-generated method stub
        try {

            Mono<UsuarioResponse> response = webClient.build().get().uri(URL + "/usuarios/" + id)
                    .retrieve().bodyToMono(UsuarioResponse.class);

            return response.block();
        } catch (Exception e) {

            return null;
        }

    }

    @Override
    public int borrarUsuario(Long id) {
        try {

            Mono<Integer> response = webClient.build().delete().uri(URL + "/usuarios/" + id)
                    .retrieve().bodyToMono(Integer.class);

            return response.block();

        } catch (WebClientResponseException e) {
            e.getMessage();
            System.out.println("---->" + e.getMessage());
            return 0;
        }
    }

     @Override
     public List<TipoDocumento> getTipoDocumento() {
         try {
             Mono<List> response = webClient.build().get().uri(URL + "/tipodocumento").retrieve()
                     .bodyToMono(List.class);

             return response.block();
         } catch (Exception e) {

             return null;
         }

     }


}
```

4. Modificar Controlador.

```
44    @GetMapping("/usuario")
45    public String usuario(Model model) {
46
47        Usuario usuarioEditar = new Usuario();
48        model.addAttribute("tipoDocumento", clienteTienda.getTipoDocumento());
49        model.addAttribute("usuarios", clienteTienda.getUsuarios());
50        model.addAttribute("usuarioEditar", usuarioEditar);
51        return "usuario";
52    }
53
54    @PostMapping("/usuario")
55    public String crearUsuario(Model model, Usuario usuario) {
56
57        Usuario usuarioEditar = new Usuario();
58        clienteTienda.nuevoUsuario(usuario);
59
60        model.addAttribute("tipoDocumento", clienteTienda.getTipoDocumento());
61        model.addAttribute("usuarios", clienteTienda.getUsuarios());
62        model.addAttribute("usuarioEditar", usuarioEditar);
63
64        return "usuario";
65    }
67    @GetMapping("/usuario/{id}")
68    public String actualizarUsuario(Model model, @PathVariable(name = "id") Long id) {
69
70        UsuarioResponse usuarioEditar = clienteTienda.buscarUsuario(id);
71
72        System.out.println(usuarioEditar.getNombre());
73        model.addAttribute("usuarioEditar", usuarioEditar);
74        model.addAttribute("tipoDocumento", clienteTienda.getTipoDocumento());
75        model.addAttribute("usuarios", clienteTienda.getUsuarios());
76
77        return "usuario";
78    }
79
80    @GetMapping("/eliminarusuario/{id}")
81    public String eliminarUsuario(Model model, @PathVariable(name = "id") Long id) {
82
83        Usuario usuarioEditar = new Usuario();
84        clienteTienda.borrarUsuario(id);
85
86        model.addAttribute("tipoDocumento", clienteTienda.getTipoDocumento());
87        model.addAttribute("usuarios", clienteTienda.getUsuarios());
88        model.addAttribute("usuarioEditar", usuarioEditar);
89
90        return "usuario";
91    }
92
93 }
```

5. Crear Plantilla de Presentación (usuario.html).

```
<!DOCTYPE html>
<html xmlns="https://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1">
<link
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
    rel="stylesheet"
    integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
    crossorigin="anonymous">
<script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
    crossorigin="anonymous"></script>
<title>Menu Principal</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-light bg-light">
        <a class="navbar-brand" href="/menu">Inicio</a>

        <div class="collapse navbar-collapse" id="navbarText">
            <ul class="navbar-nav mr-auto">
                <li class="nav-item active"><a class="nav-link" href="#">Usuarios</a></li>
                <li class="nav-item"><a class="nav-link" href="#">Clientes</a>
                </li>
                <li class="nav-item"><a class="nav-link" href="#">Proveedores</a></li>

                <li class="nav-item"><a class="nav-link" href="#">Ventas</a></li>

                <li class="nav-item"><a class="nav-link" href="#">Reportes</a></li>
                <li class="nav-item"><a class="nav-link" href="#">Consolidado</a></li>
            </ul>

        </div>
    </nav>
   <br>
   <h1 style="text-align: center;">Gestionar Usuarios</h1>
   <div class="container">
      <div class=" col-xl-2 col-lg-3 col-md-4 col-sm-6" id="formulario">
         <form method="post" th:action="@{/usuario}" th:object="${usuarioEditar}">
            <div class="form-group">
               <label>tipo Documento :</label>
               <select name="idTipoDocumento" class="form-select" th:value="${usuarioEditar.idTipoDocumento}">
                  <option value="0">Seleccionar</option>
                     <option th:each="tipos : ${tipoDocumento}" th:text="${tipos.tipo}" th:value="${tipos.id}"
                     th:attr="selected=${tipos.id==usuarioEditar.idTipoDocumento?true:false}"></option>
               </select>
            </div>
            <div class="form-group">
               <input type="hidden" name="id" th:field="*{id}">

               <label for="numero"> numero:</label>
               <input type="text" name="numeroDocumento" id="numero"
                  th:field="*{numeroDocumento}" class="form-control" />
            </div>
            <div class="form-group">
               <label>nombre:</label><input type="text" name="nombre"
                  class="form-control" th:field="*{nombre}" />
            </div>
            <div class="form-group">
               <label>Email:</label><input type="text" name="email"
                  class="form-control" th:field="*{email}" />
            </div>
            <div class="form-group">
               <label> nombre usuario:</label> <input type="text"
                  name="nombreUsuario" class="form-control"
                  th:field="*{nombreUsuario}" />
            </div>
                  <div class="form-group">
                     <label>password:</label> <input type="password" name="password"
                        class="form-control" th:field="*{password}" />
                  </div>

                  <br>
                  <button type="submit" class="btn btn-primary">Guardar</button>
            </form>
         </div>
         <br> <br>
```

```
        <table class="table">
            <thead>
                <tr>
                    <th>#</th>
                    <th>Tipo documento</th>
                    <th>Numero</th>
                    <th>Nombre</th>
                    <th>Nombre usuario</th>
                    <th colspan="2">Operaciones</th>
                </tr>
            </thead>
            <tbody>
                    <tr th:each ="usuario : ${usuarios}">
                        <td th:text="${usuario.id}"></td>
                        <td th:text="${usuario.idTipoDocumento.tipo}"></td>
                        <td th:text="${usuario.numeroDocumento}"></td>
                        <td th:text="${usuario.nombre}"></td>
                        <td th:text= "${usuario.nombreUsuario}"></td>
                        <td><a class="btn btn-danger"
                            th:href="@{'/eliminarusuario/' + ${usuario.id}}">Eliminar</a></td>
                        <td><a class="btn btn-success" th:href="@{'/usuario/' + ${usuario.id}}">Actualizar</a></td>
                    </tr>
            </tbody>
        </table>
    </div>
</body>
</html>
```

2. CRUD de Usuario (BackEnd).

Clase de transferencia de objetos:

En el backend se debe construir una estructura que soporte la información recibida del front y a su vez lo que se envía de respuesta, debemos crear una clase encargada de hacer la transferencia de objetos llamada UsuarioDto con un contenido como el que se aprecia en la figura:

```
package com.mintic.tiendaback.modelo;

public class UsuarioDto {
    private Long id;

    private Long idTipoDocumento;

    private String numeroDocumento;

    private String nombre;

    private String password;

    private String nombreUsuario;

    private String email;


    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public Long getId() {
        return id;
    }

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

    public Long getIdTipoDocumento() {
        return idTipoDocumento;
    }

    public void setIdTipoDocumento(Long idTipoDocumento) {
        this.idTipoDocumento = idTipoDocumento;
    }

    public String getNumeroDocumento() {
        return numeroDocumento;
    }

    public void setNumeroDocumento(String numeroDocumento) {
        this.numeroDocumento = numeroDocumento;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
```

```
66
67      public String getNombreUsuario() {
68          return nombreUsuario;
69      }
70
71      public void setNombreUsuario(String nombreUsuario) {
72          this.nombreUsuario = nombreUsuario;
73      }
74
75  }
```

Interface de Servicio.

```
1  package com.mintic.tiendaback.servicio;
2
3  import java.util.List;
4
5  import org.springframework.http.ResponseEntity;
6
7  import com.mintic.tiendaback.modelo.UsuarioDto;
8  import com.mintic.tiendaback.modelo.Usuario;
9  import com.mintic.tiendaback.modelo.LoginDto;
10
11
12 /*
13  * Aqui se definen los metodos que se van a utilizar (el contrato)
14  * */
15 public interface IUsuarioService {
16
17     int login(LoginDto usuarioDto);
18
19     List<Usuario> getUsuarios();
20
21     Usuario nuevoUsuario(UsuarioDto usuarioDto);
22
23     Usuario buscarUsuario(Long id);
24
25     int borrarUsuario(Long id);
26
27     ResponseEntity<?> ingresar(LoginDto usuarioDto);
28
29 }
```

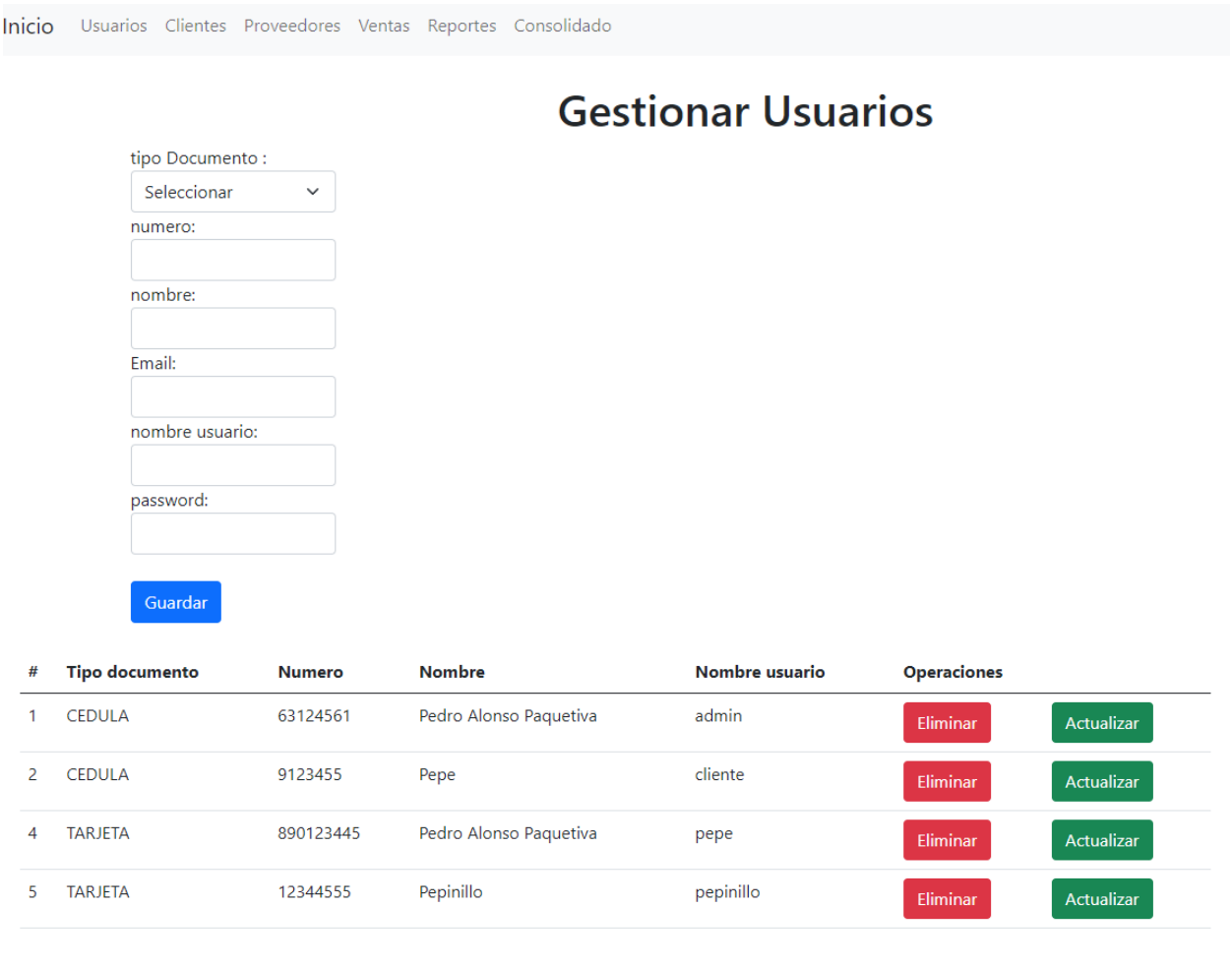
Implementación de Métodos en Servicio.

```
38      @Override
39      public List<Usuario> getUsuarios() {
40
41          return (List<Usuario>) iUsuario.findAll();
42      }
43
44      @Override
45      public Usuario nuevoUsuario(UsuarioDto usuarioDto) {
46
47          Usuario usuario = new Usuario();
48          TipoDocumento td = new TipoDocumento();
49          td.setId(usuarioDto.getIdTipoDocumento());
50
51          if(usuarioDto.getId()!=null) {
52              usuario.setId(usuarioDto.getId());
53          }
54
55          usuario.setIdTipoDocumento(td);
56          usuario.setNumeroDocumento(usuarioDto.getNumeroDocumento());
57          usuario.setNombre(usuarioDto.getNombre());
58          usuario.setNombreUsuario(usuarioDto.getNombreUsuario());
59          usuario.setPassword(usuarioDto.getPassword());
60          usuario.setEmail(usuarioDto.getEmail());
61          return iUsuario.save(usuario);
62      }
```

```java
    @Override
    public int borrarUsuario(Long id) {
        iUsuario.deleteById(id);
        return 1;
    }

    @Override
    public Usuario buscarUsuario(Long id) {

        Usuario usuario = null;

        usuario = iUsuario.findById(id).orElse(null);
        if (usuario == null) {
            return null;
        }

        return usuario;
    }
```

Implementación de Métodos en Controlador.

```java
    @PostMapping("/usuarios")
    public Usuario usuario(@RequestBody UsuarioDto usuarioDto) {

        return iUsuario.nuevoUsuario(usuarioDto);
    }

    @GetMapping("/usuarios")
    public List<Usuario> listarUsuarios() {

        return iUsuario.getUsuarios();
    }

    @GetMapping("/usuarios/{id}")
    public Usuario buscarUsuarioId(@PathVariable Long id) {
        return iUsuario.buscarUsuario(id);
    }

    @DeleteMapping("/usuarios/{id}")
    public int eliminarUsuario(@PathVariable Long id) {

        return iUsuario.borrarUsuario(id);
    }

    @GetMapping("/tipodocumento")
    public List<TipoDocumento> listarTipoDocumento() {

        return (List<TipoDocumento>) iTipoDocumento.findAll();
    }
```


Inicio
Usuarios
Clientes
Proveedores
Ventas
Reportes
Consolidado
Gestionar Usuarios
tipo Documento :
Seleccionar
numero:
nombre:
Email:
nombre usuario:
password:
Guardar
Tipo documento Numero Nombre Nombre usuario Operaciones
1 CEDULA 63124561 Pedro Alonso Paquetiva admin Eliminar Actualizar
2 CEDULA 9123455 Pepe cliente Eliminar Actualizar
4 TARJETA 890123445 Pedro Alonso Paquetiva pepe Eliminar Actualizar
5 TARJETA 12344555 Pepinillo pepinillo Eliminar Actualizar