1. **Como conectar la base de datos.**

entrar a src/main/reources

al achivo

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

#Conexcion para Mysql

spring.datasource.url=jdbc:mysql://localhost:3306/empleadosumg

spring.datasource.username=root

spring.datasource.password=Ander2023@

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

spring.jpa.hibernate.ddl-auto=update

spring.jpa.databaseplatform=org.hibernate.dialect.MySQL8Dialect

spring.datasource.platform=mysql.

#Puerto donde se va a ejecutar el microservicio.

server.port=8089

#Logs.

spring.jpa.open-in-view=false

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

Agregar dependencias al archive pom.xml.

Codigos

Clase entity.

package com.microserver.ap.proveedores.entity;

import java.util.UUID;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.AllArgsConstructor;

@Data

@Entity

@Table(name = "proveedores")

@NoArgsConstructor

@AllArgsConstructor

public class Proveedores {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

@Column(columnDefinition = "VARCHAR(36)")

private UUID id;

@Column(nullable = false, length = 100)

private String nombre;

@Column(nullable = false, length = 200)

private String direccion;

@Column(nullable = false, length = 15)

private String telefono;

}

Clase IProveedoresImpl

package com.microserver.ap.proveedores.implementacion;

import java.util.List;

import java.util.Optional;

import java.util.UUID;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.microserver.ap.proveedores.services.IProveedores;

import com.microserver.ap.proveedores.entity.Proveedores;

import com.microserver.ap.proveedores.repository.IProveedoresRepository;

*@Service*

public class IProveedoresImpl implements IProveedores{

*@Autowired*

protected IProveedoresRepository proveedoresRepository;

*@Override*

public List<Proveedores> findAll() {

return (List<Proveedores>) proveedoresRepository.findAll();

}

*@Override*

public Proveedores save(Proveedores proveedores) {

proveedores.setId(UUID.*randomUUID*());

return proveedoresRepository.save(proveedores);

}

*@Override*

public Proveedores update(UUID id, Proveedores proveedores) {

Optional<Proveedores> optionalProveedor = proveedoresRepository.findById(id);

if (optionalProveedor.isPresent()) {

Proveedores existingProveedor = optionalProveedor.get();

existingProveedor.setNombre(proveedores.getNombre());

existingProveedor.setDireccion(proveedores.getDireccion());

existingProveedor.setTelefono(proveedores.getTelefono());

return proveedoresRepository.save(existingProveedor);

}

return null;

}

*@Override*

public Integer deleteById(UUID id) {

if (!proveedoresRepository.existsById(id)) {

return 404; // Not Found

} else {

proveedoresRepository.deleteById(id);

return 204; // No Content

}

}

}

Paquete service

Clase IProveedores

package com.microserver.ap.proveedores.services;

import java.util.List;

import java.util.UUID;

import com.microserver.ap.proveedores.entity.Proveedores;

public interface IProveedores {

List<Proveedores> findAll();

Proveedores save(Proveedores proveedores);

Proveedores update(UUID id, Proveedores proveedores);

Integer deleteById(UUID id);

}

Clase IProveedoresRepository

package com.microserver.ap.proveedores.repository;

import java.util.UUID;

import org.springframework.data.repository.CrudRepository;

import com.microserver.ap.proveedores.entity.Proveedores;

public interface IProveedoresRepository extends CrudRepository<Proveedores, UUID>{

}

Clase ProveedoresController.

package com.microserver.ap.proveedores.controller;

import java.util.List;

import java.util.UUID;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.MediaType;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.microserver.ap.proveedores.services.IProveedores;

import com.microserver.ap.proveedores.entity.Proveedores;

public class ProveedoresController {

@Autowired

protected IProveedores proveedoresService;

@GetMapping

public ResponseEntity<List<Proveedores>> getAllProveedores() {

return ResponseEntity.ok(proveedoresService.findAll());

}

@GetMapping("/findAll")

public ResponseEntity<List<Proveedores>> findAll() {

return ResponseEntity.ok(proveedoresService.findAll());

}

@PostMapping("/save")

public ResponseEntity<Proveedores> save(@RequestBody Proveedores proveedores) {

return ResponseEntity.ok(proveedoresService.save(proveedores));

}

@PutMapping(path = "/update/{id}", produces = MediaType.APPLICATION\_JSON\_VALUE, consumes = MediaType.APPLICATION\_JSON\_VALUE)

public ResponseEntity<?> update(@PathVariable("id") UUID id, @RequestBody Proveedores proveedores) {

Proveedores updateProveedor = proveedoresService.update(id, proveedores);

if (updateProveedor != null) {

return ResponseEntity.ok(updateProveedor);

} else {

return ResponseEntity.notFound().build();

}

}

@DeleteMapping("/delete/{id}")

public ResponseEntity<?> delete(@PathVariable("id") UUID id) {

Integer response = proveedoresService.deleteById(id);

if (response == 204) {

return ResponseEntity.noContent().build(); // 204 No Content

} else {

return ResponseEntity.notFound().build(); // 404 Not Found

}

}

}