Application.properties

# DataSource settings: set here your own configurations for the database

# connection. In this example we have "demo" as database name and

# "root" as username and password

spring.datasource.url=jdbc:mysql://localhost:3306/sport\_db?createDatabaseIfNotExist=true&useSSL=false&serverTimezone=UTC

spring.datasource.username=root

spring.datasource.password=\*\*\*\*\*\*

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

# Show or not log for each sql query

spring.jpa.show-sql=true

# Hibernate ddl auto (create, create-drop, update)

spring.jpa.hibernate.ddl-auto=create

# The SQL dialect makes Hibernate generate better SQL for the chosen database

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

# Naming strategy

spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl

PlayerController.java

package com.example.controller;

import java.util.List;

//import org.springframework.beans.factory.annotation.Autowired(required=true);

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

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

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.example.model.Player;

import com.example.service.PlayerService;

@CrossOrigin("http://localhost:4200")

@RestController

@RequestMapping("api/clients")

public class ClientController {

@Autowired

private ClientService clientService;

@GetMapping("")

public List<Clients> getAllClients(){

return clientService.getAllClients();

}

@PutMapping("/{id}")

public Clients update(@PathVariable Integer id ,@RequestBody Clients clients) {

return clientService.update(clients);

}

@DeleteMapping("/{id}")

public void delete(@PathVariable Integer id) {

clientService.delete(id);

}

@PostMapping

public Clients add(@RequestBody Clients clients) {

return clientService.add(clients);

}

@GetMapping("/{id}")

public Clients getClientsById(@PathVariable Integer id) {

return clientService.findById(id);

}

}

Player.java

package com.example.model;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Clients {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="CLIENT\_ID")

private Integer id;

@Column

private Integer code;

@Column

private String nom;

@Column

private String raisonSocial;

@Column

private String adress;

@Column

private Integer tel;

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public Integer getCode() {

return code;

}

public void setCode(Integer code) {

this.code = code;

}

public String getNom() {

return nom;

}

public void setNom(String nom) {

this.nom = nom;

}

public String getRaisonSocial() {

return raisonSocial;

}

public void setRaisonSocial(String raisonSocial) {

this.raisonSocial = raisonSocial;

}

public String getAdress() {

return adress;

}

public void setAdress(String adress) {

this.adress = adress;

}

public Integer getTel() {

return tel;

}

public void setTel(Integer tel) {

this.tel = tel;

}

}

PlayerRepo.java

package com.example.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.example.model.Clients;

public interface ClientRepository extends JpaRepository<Clients, Integer> {

}

PlayerService.java

package com.example.service;

import java.util.List;

import com.example.model.Clients;

public interface ClientService {

public List<Clients> getAllClients();

public Clients update(Clients clients);

public void delete(Integer id);

public Clients add(Clients clients);

public Clients findById(Integer id);

}

PlayerServiceIImpl.java

package com.example.service;

import java.util.List;

import java.util.Optional;

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

import org.springframework.stereotype.Service;

import com.example.model.Clients;

import com.example.repository.ClientRepository;

@Service

public class ClientServiceImpl implements ClientService {

@Autowired

public ClientRepository clientRepository;

@Override

public List<Clients> getAllClients() {

return clientRepository.findAll() ;

}

@Override

public Clients update(Clients clients) {

return clientRepository.save(clients);

}

@Override

public void delete(Integer id) {

clientRepository.deleteById(id);

}

@Override

public Clients add(Clients clients) {

return clientRepository.save(clients);

}

@Override

public Clients findById(Integer id) {

Optional<Clients> Clients = clientRepository.findById(id);

return Clients.isPresent() ? Clients.get() : null;

}

}