Spring boot crud operations example with hibernate

Step 1 : <https://start.spring.io/>

Dependencies

1. Spring Web
2. Spring Boot DevTools
3. Spring Data JPA
4. MySQL Driver

Step 2 : application.properties

server.port = 9090

jdbc.driverClassName = com.mysql.cj.jdbc.Driver

jdbc.url = jdbc:mysql://localhost:3306/securitydb

jdbc.username = root

jdbc.password = root

hibernate.dialect = org.hibernate.dialect.MySQL57InnoDBDialect

Step 3 :Database config

**import** javax.sql.DataSource;

**import** org.hibernate.SessionFactory;

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

**import** org.springframework.beans.factory.annotation.Value;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.context.annotation.PropertySource;

**import** org.springframework.jdbc.datasource.DriverManagerDataSource;

**import** org.springframework.orm.hibernate5.HibernateTransactionManager;

**import** org.springframework.orm.hibernate5.LocalSessionFactoryBean;

@PropertySource(value = { "classpath:application.properties" })

@Configuration

**public** **class** DataBaseConfig {

@Value("${jdbc.driverClassName}")

**private** String driverClass;

@Value("${jdbc.url}")

**private** String url;

@Value("${jdbc.username}")

**private** String username;

@Value("${jdbc.password}")

**private** String password;

@Value("${hibernate.dialect}")

**private** String dialect;

@Bean

**public** DataSource getDataSource() {

DriverManagerDataSource dataSource = **new** DriverManagerDataSource(url, username, password);

dataSource.setDriverClassName(driverClass);

**return** dataSource;

}

@Bean

**public** LocalSessionFactoryBean sessionFactory() {

LocalSessionFactoryBean factory = **new** LocalSessionFactoryBean();

factory.setDataSource(getDataSource());

factory.setHibernateProperties(hibernateProperties());

factory.setPackagesToScan(**new** String[] { "com.SpringBootHibernate5.entity" });

**return** factory;

}

**private** Properties hibernateProperties() {

Properties properties = **new** Properties();

properties.put("hibernate.dialect", dialect);

properties.put("hibernate.hbm2ddl.auto", "update");

properties.put("hibernate.show\_sql", "true");

properties.put("hibernate.format\_sql", "true");

**return** properties;

}

@Bean

@Autowired

**public** HibernateTransactionManager transactionManager(SessionFactory factory) {

HibernateTransactionManager transactionManager = **new** HibernateTransactionManager();

transactionManager.setSessionFactory(factory);

**return** transactionManager;

}

}

Step 4 :

**import** java.util.Date;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

**import** com.fasterxml.jackson.annotation.JsonFormat;

@Entity

@Table(name = "employee\_table")

**public** **class** Employee {

@Id

@GeneratedValue

**private** **int** id;

@Column (length = 25)

**private** String name;

@JsonFormat(shape = JsonFormat.Shape.***STRING***, pattern = "dd/MM/yyyyy")

**private** Date dob;

Step 5 : dao

**import** java.util.List;

**import** javax.persistence.criteria.CriteriaBuilder;

**import** javax.persistence.criteria.CriteriaQuery;

**import** javax.transaction.Transactional;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

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

**import** org.springframework.stereotype.Repository;

**import** com.SpringBootHibernate5.entity.Employee;

@Repository

@Transactional // should be in service layer

**public** **class** EmployeeDao {

@Autowired

**private** SessionFactory factory;

**public** **void** saveEmployee(Employee employee) {

getSession().save(employee);

}

**public** List<Employee> getEmployee() {

// create Criteria

CriteriaQuery<Employee> criteriaQuery = getSession().getCriteriaBuilder().createQuery(Employee.**class**);

criteriaQuery.from(Employee.**class**);

List<Employee> employees = getSession().createQuery(criteriaQuery).getResultList();

**return** employees;

}

**private** Session getSession() {

Session session = factory.getCurrentSession();

**if** (session == **null**) {

session = factory.openSession();

}

**return** session;

}

}

Step 6 :

@RestController

@RequestMapping("/spring-boot-orm")

**public** **class** EmployeeController {

@Autowired

**private** EmployeeDao employeeDao;

@PostMapping("/saveemployee")

**public** String saveEmployee(@RequestBody Employee employee) {

employeeDao.saveEmployee(employee);

**return** "success";

}

@GetMapping("/getAll")

**public** List<Employee> getALlPersons() {

**return** employeeDao.getEmployee();

}

}

Step 7 : exclude = HibernateJpaAutoConfiguration.**class**)

@SpringBootApplication (exclude = HibernateJpaAutoConfiguration.**class**)

**public** **class** SpringBootHibernate5Application {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(SpringBootHibernate5Application.**class**, args);

}

}