МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС

«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Лабораторна робота №1

з дисциплiни «Проектування інформаційних систем»

Виконав:

студент 4 курсу

групи КА-66

Дядюра О. Ю.

Прийняв:

Коновалюк М.М.

**Київ – 2020**

23. Система Больница. Врач определяет диагноз, делает назначение

Пациенту (процедуры, лекарства, операции). Назначение может выполнить

Медсестра (процедуры, лекарства) или Врач (любое назначение). Пациент

может быть выписан из Больницы, при этом фиксируется окончательный

диагноз.

Створення бази даних та її підключення за допомогою Spring Framework.

***Лістинг програми:***

**Entities**

package ua.kpi.myhospital.Entities;

import lombok.Data;

import javax.naming.Name;

import javax.persistence.\*;

import java.util.List;

@Entity

@Data

//@Table(name = "user",schema = "myhospital")

public class User {

@Id

private Integer idUser;

private String name;

private String surName;

private String role;

@OneToMany(mappedBy = "user\_prescription", fetch = FetchType.LAZY)

private List<Prescription> prescriptions;

public User() {

}

public User(Integer idUser, String login, String password, String name, String surName, String role){

this.idUser = idUser;

this.name = name;

this.surName = surName;

this.role = role;

}

public long getIduser() {

return idUser;

}

public void setIduser(Integer idUser) {

this.idUser = idUser;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getSurName() {

return surName;

}

public void setSurName(String surName) {

this.surName = surName;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

}

**package ua.kpi.myhospital.Entities;**

**import lombok.Data;**

**import javax.persistence.\*;**

@Entity

@Data

@Table(name = "prescription",schema = "myhospital")

public class Prescription {

@Id

private Integer idPrescription;

private int idDoc;

private int idPatient;

private int idExecutor;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumns({

@JoinColumn(name ="id\_doc", referencedColumnName = "id\_user"),

@JoinColumn(name="id\_executor", referencedColumnName = "id\_user"),

@JoinColumn(name="id\_patient", referencedColumnName = "id\_user")

})

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumns({

@JoinColumn(name = "id\_doc", referencedColumnName = "id\_user"),

@JoinColumn(name = "id\_executor", referencedColumnName = "id\_user"),

@JoinColumn(name = "id\_patient", referencedColumnName = "id\_user")

})

private User user\_prescription;

public Prescription() {}

public Prescription(Integer idPrescription, int idDoc, int idPatient, int idExecutor) {

this.idPrescription = idPrescription;

this.idDoc = idDoc;

this.idPatient = idPatient;

this.idExecutor = idExecutor;

}

public Integer getIdPrescription() {

return idPrescription;

}

public void setIdPrescription(Integer idPrescription) {

this.idPrescription = idPrescription;

}

public int getIdDoc() {

return idDoc;

}

public void setIdDoc(int idDoc) {

this.idDoc = idDoc;

}

public int getIdPatient() {

return idPatient;

}

public void setIdPatient(int idPatient) {

this.idPatient = idPatient;

}

public int getIdExecutor() {

return idExecutor;

}

public void setIdExecutor(int idExecutor) {

this.idExecutor = idExecutor;

}

}

package ua.kpi.myhospital.Entities;

import javax.persistence.Entity;

import javax.persistence.Id;

@Entity

public class Diagnos {

@Id

private Integer idDiagnos;

private long idPrescription;

private String currentDiagnos;

private String healthStatus;

private String kind;

public Diagnos(){}

public Diagnos( Integer idDiagnos, long idPrescription,String kind, String currentDiagnos, String healthStatus){

this.idDiagnos = idDiagnos;

this.idPrescription = idPrescription;

this.currentDiagnos = currentDiagnos;

this.kind = kind;

this.healthStatus = healthStatus;

}

public Integer getIdDiagnos() {

return idDiagnos;

}

public void setIdDiagnos(Integer idDiagnos) {

this.idDiagnos = idDiagnos;

}

public long getIdPrescription() {

return idPrescription;

}

public void setIdPrescription(Integer idPrescription) {

this.idPrescription = idPrescription;

}

public String getCurrentDiagnos() {

return currentDiagnos;

}

public void setCurrentDiagnos(String currentDiagnos) {

this.currentDiagnos = currentDiagnos;

}

public String getHealthStatus() {

return healthStatus;

}

public void setHealthStatus(String healthStatus) {

this.healthStatus = healthStatus;

}

public String getKind() {

return kind;

}

public void setKind(String kind) {

this.kind = kind;

}

}

**DAO**

package ua.kpi.myhospital.DAO;

import ua.kpi.myhospital.Entities.User;

import java.util.List;

public interface UserDAO {

List<User> findAll();

User findById(Integer idUser);

List<User> findByName(String name);

void deleteById(Integer idUser);

void save(User user);

void update(User user, Integer id);

}

package ua.kpi.myhospital.DAO;

import ua.kpi.myhospital.Entities.Prescription;

import java.util.List;

public interface PrescriptionDAO {

List<Prescription> findAll();

Prescription findById(Integer idPrescription);

void deleteById(Integer idPrescription);

void save(Prescription prescription);

void update(Prescription prescription, Integer idPrescription);

}

package ua.kpi.myhospital.DAO;

import ua.kpi.myhospital.Entities.Diagnos;

import java.util.List;

public interface DiagnosDAO {

List<Diagnos> findAll();

Diagnos findById(Integer id);

void deleteById(Integer id);

void save(Diagnos diagnos);

}

**Repo:**

package ua.kpi.myhospital.Repo;

import org.springframework.data.repository.CrudRepository;

import ua.kpi.myhospital.Entities.Diagnos;

public interface DiagnosRepository extends CrudRepository<Diagnos, Integer> {

}

package ua.kpi.myhospital.Repo;

import org.springframework.data.repository.CrudRepository;

import ua.kpi.myhospital.Entities.Prescription;

public interface PrescriptionRepository extends CrudRepository<Prescription, Integer> {

}

package ua.kpi.myhospital.Repo;

import org.springframework.data.repository.CrudRepository;

import ua.kpi.myhospital.Entities.User;

import java.util.List;

public interface UserRepository extends CrudRepository<User,Integer> {

public List<User> findByName(String name);

}

**Spring Data**

package ua.kpi.myhospital.Data;

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

import org.springframework.stereotype.Repository;

import ua.kpi.myhospital.DAO.DiagnosDAO;

import ua.kpi.myhospital.Entities.Diagnos;

import ua.kpi.myhospital.Repo.DiagnosRepository;

import java.util.List;

@Repository("DiagnosRepository")

public class DiagnosData implements DiagnosDAO {

@Autowired

private DiagnosRepository diagnosRepository;

@Override

public List<Diagnos> findAll() {

return (List<Diagnos>)diagnosRepository.findAll();

}

@Override

public Diagnos findById(Integer idDiagnos) {

return diagnosRepository.findById(idDiagnos).get();

}

@Override

public void deleteById(Integer idDiagnos) {

diagnosRepository.deleteById(idDiagnos);

}

@Override

public void save(Diagnos diagnos) {

diagnosRepository.save(diagnos);

}

}

package ua.kpi.myhospital.Data;

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

import org.springframework.stereotype.Repository;

import ua.kpi.myhospital.DAO.PrescriptionDAO;

import ua.kpi.myhospital.Entities.Prescription;

import ua.kpi.myhospital.Repo.PrescriptionRepository;

import java.util.List;

@Repository("PrescriptionRepository")

public class PrescriptionData implements PrescriptionDAO {

@Autowired

private PrescriptionRepository prescriptionRepository;

@Override

public List<Prescription> findAll() {

return (List<Prescription>) prescriptionRepository.findAll();

}

@Override

public Prescription findById(Integer idPrescription) {

return prescriptionRepository.findById(idPrescription).get();

}

@Override

public void deleteById(Integer idPrescription) {

prescriptionRepository.deleteById(idPrescription);

}

@Override

public void save(Prescription prescription) {

prescriptionRepository.save(prescription);

}

@Override

public void update(Prescription prescription, Integer idPrescription) {

prescriptionRepository.save(prescription);

}

}

package ua.kpi.myhospital.Data;

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

import org.springframework.stereotype.Repository;

import ua.kpi.myhospital.DAO.UserDAO;

import ua.kpi.myhospital.Entities.User;

import ua.kpi.myhospital.Repo.UserRepository;

import java.util.List;

@Repository("UserRepository")

public class UserData implements UserDAO {

@Autowired

private UserRepository userRepository;

@Override

public List<User> findAll() {

return (List<User>) userRepository.findAll();

}

@Override

public User findById(Integer idUser) {

return userRepository.findById(idUser).get();

}

@Override

public List<User> findByName(String name) {

return userRepository.findByName(name);

}

@Override

public void deleteById(Integer idUser) {

userRepository.deleteById(idUser);

}

@Override

public void save(User user) {

userRepository.save(user);

}

@Override

public void update(User user, Integer id) {

userRepository.save(user);

}

}