package nurseryApp.Models;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

/\*\*

\* Created by regga on 13/01/2017.

\*/

public class Adults {

protected Integer parentID;

protected String name;

protected String surname;

protected String datOfBirth;

protected String email;

protected String mobile;

protected String address;

String postcode;

public Adults(){

}

public Adults(String name, String surname, String datOfBirth, String email, String mobile, String address, String postcode) {

this.name = name;

this.surname = surname;

this.datOfBirth = datOfBirth;

this.email = email;

this.mobile = mobile;

this.address = address;

this.postcode = postcode;

}

public Adults (Integer parentID){

Connection connection=null;

Statement statement = null;

ResultSet rs = null;

this.parentID = parentID;

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Conection succesfull for Adult:Parent");

Statement st = null;

String query = "SELECT \* FROM parentsTable WHERE parentID="+"'"+parentID+"'" ;

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

if (resultSet.next()) {

this.name=resultSet.getString("name");

this.surname=resultSet.getString("surname");

this.datOfBirth=resultSet.getString("DoB");

this.email=resultSet.getString("email");

this.mobile=resultSet.getString("mobile");

this.address=resultSet.getString("address");

this.postcode=resultSet.getString("postcode");

}else{

System.out.println("Wrong Child ID ");

return;

}

connection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

public Adults (String staff,Integer staffID){

Connection connection=null;

Statement statement = null;

ResultSet rs = null;

this.parentID = parentID;

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Conection succesfull for Adult:Staff");

Statement st = null;

String query = "SELECT \* FROM staffTable WHERE staffID="+"'"+staffID+"'" ;

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

if (resultSet.next()) {

this.name=resultSet.getString("name");

this.surname=resultSet.getString("surname");

this.datOfBirth=resultSet.getString("DoB");

this.email=resultSet.getString("email");

this.mobile=resultSet.getString("mobile");

this.address=resultSet.getString("address");

this.postcode=resultSet.getString("postcode");

}else{

System.out.println("Wrong Child ID ");

return;

}

connection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

public Integer getParentID() {

return parentID;

}

public void setParentID(Integer parentID) {

this.parentID = parentID;

}

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 getdatOfBirth() {

return datOfBirth;

}

public void setdatOfBirth(String datOfBirth) {

datOfBirth = datOfBirth;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getMobile() {

return mobile;

}

public void setMobile(String mobile) {

this.mobile = mobile;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getPostcode() {

return postcode;

}

public void setPostcode(String postcode) {

this.postcode = postcode;

}

}

package nurseryApp.Models;

/\*\*

\* Created by regga on 15/01/2017.

\*/

public class Bill {

Double balance;

String dueDate;

double discounts;

public Bill(double balance, String dueDate, double discounts) {

this.balance = balance;

this.dueDate = dueDate;

this.discounts = discounts;

}

public Bill(int parentID){

Parents parents = new Parents(parentID);

int id = parents.findOwnChildren();

Charges charges= new Charges();

this.balance=charges.getFinalBalance(id);

}

public double getBalance() {

return balance;

}

public void setBalance(double balance) {

this.balance = balance;

}

public String getDueDate() {

return dueDate;

}

public void setDueDate(String dueDate) {

this.dueDate = dueDate;

}

public double getDiscounts() {

return discounts;

}

public void setDiscounts(double discounts) {

this.discounts = discounts;

}

}

package nurseryApp.Models;

import nurseryApp.Models.DBConnection;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.scene.control.Alert;

import javafx.stage.Stage;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.Statement;

import java.text.DateFormat;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

import java.util.Locale;

import java.util.concurrent.TimeUnit;

/\*\*

\* Created by regga on 12/01/2017.

\*/

public class Charges {

//int id;

int childID;

String generatedDate;

double price;

String description;

// public Charges(int childID, int month ) {

// this.childID = childID;

// this.month = month;

// }

public Charges(int childID, String generatedDate, double price, String description) {

this.childID = childID;

this.generatedDate = generatedDate;

this.price = price;

this.description = description;

}

public Charges(){

}

// public int getId() {

// return id;

// }

//

// public void setId(int id) {

// this.id = id;

// }

public int getChildID() {

return childID;

}

public void setChildID(int childID) {

this.childID = childID;

}

public String getGeneratedDate() {

return generatedDate;

}

public void setGeneratedDate(String generatedDate) {

this.generatedDate = generatedDate;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public void addCharges(int childID, String description, float price,String todaysDay){

todaysDay = new SimpleDateFormat("yyyy-MM-dd | HH:mm:ss").format(Calendar.getInstance().getTime());

try {

Connection connection=null;

Statement statement = null;

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Connection succesfull");

PreparedStatement ps;

ps = connection.prepareStatement("INSERT INTO sessionTable (childID,description,price,todaysDay" +

"VALUES (?,?,?,?) ");

ps.setInt(1, childID);

ps.setString(2, description);

ps.setFloat(3, price);

ps.setString(4, todaysDay);

ps.executeUpdate();

ps.close();

connection.close();

System.out.println("Query Add chrages executed, connection terminated ");

}

} catch (Exception e) {

e.printStackTrace();

}

}

// String todaysDay = new SimpleDateFormat("yyyy-MM-dd | HH:mm:ss").format(Calendar.getInstance().getTime());

// DateFormat format = new SimpleDateFormat("MMMM d, yyyy", Locale.ENGLISH);

// Date date = format.parse("2016-11-09");

// Calendar cal = Calendar.getInstance();

// cal.setTime(date);

// int month = cal.get(Calendar.MONTH);

public ObservableList<Charges> chargesList (int searchID){

Connection connection=null;

Statement statement = null;

ObservableList<Charges> charges = FXCollections.observableArrayList();

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Conection succesfull");

Statement st = null;

String query = "SELECT \* FROM sessionTable WHERE child\_Id="+"'"+searchID+"'" ;

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

while (resultSet.next()) {

charges.add(new Charges(

resultSet.getInt("child\_id"),

resultSet.getString("sessionFromDate"),

resultSet.getDouble("price"),

resultSet.getString("description")

));

}

connection.close();

System.out.println("Query Create Charges executed, connection terminated ");

return charges;

}

} catch (Exception e) {

e.printStackTrace();

}

return charges;

}

public Double getFinalBalance(int childID){

ObservableList<Charges> newCharges = FXCollections.observableArrayList();

final double[] balance = {0};

Charges charges = new Charges();

newCharges=charges.chargesList(childID);

newCharges.forEach((price)->{ // lampda expression to generate a sum of all prices found

balance[0] = price.getPrice()+ balance[0];

System.out.println(balance[0]);

});

Double finalBalance=balance[0];

return finalBalance;

}

//

// public int daysCounter(String dateFrom,String dateTo){

// SimpleDateFormat newFormat = new SimpleDateFormat("yyyy-MM-dd");

// int counterDays=0;

// //String dateFrom = "2016-11-09";

// //String dateTo = "2016-11-10";

// try {

// Date dateA = newFormat.parse(dateFrom);

// Date dateB = newFormat.parse(dateTo);

// counterDays = (int) (dateB.getTime() - dateA.getTime());

// System.out.println ("Days: " + TimeUnit.DAYS.convert(counterDays, TimeUnit.MILLISECONDS));

//

// } catch (ParseException e) {

// e.printStackTrace();

// }

// return counterDays;

// }

}

package nurseryApp.Models;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.scene.control.Alert;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

import java.text.SimpleDateFormat;

import java.time.LocalDate;

import java.time.Period;

import java.util.Calendar;

import java.util.List;

import java.util.concurrent.TimeUnit;

/\*\*

\* Created by regga on 12/01/2017.

\*/

public class Children {

Connection connection=null;

Statement statement = null;

String userDB=null;

//Session session;

private int child\_Id;

private String name;

private String surname;

private String dateOfBirth;

private String allergy;

private int parentID;

private String category;

public Children(){

}

public Children(int child\_Id) {

this.child\_Id = child\_Id;

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Conection succesfull");

Statement st = null;

String query = "SELECT \* FROM childrenTable WHERE child\_Id="+"'"+child\_Id+"'" ;

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

if (resultSet.next()) {

this.name=resultSet.getString("name");

this.surname=resultSet.getString("surname");

this.dateOfBirth=resultSet.getString("DoB");

this.allergy=resultSet.getString("allergy");

this.parentID=resultSet.getInt("parentID");

this.category=resultSet.getString("category");

}else{

System.out.println("Wrong Child ID ");

return;

}

connection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

public int getChild\_Id() {

return child\_Id;

}

public String getName() {

return name;

}

public String getSurname() {

return surname;

}

public String getDateOfBirth() {

return dateOfBirth;

}

public String getAllergy() {

return allergy;

}

public int getParentID() {

return parentID;

}

public String getCategory() {

return category;

}

public int getChildAge(){

SimpleDateFormat newFormat = new SimpleDateFormat("yyyy-MM-dd");

int counterDays = 0;

String todaysDay = new SimpleDateFormat("yyyy-MM-dd | HH:mm:ss").format(Calendar.getInstance().getTime());

int counterYears = Period.between(LocalDate.now(), LocalDate.parse(getDateOfBirth())).getYears();

counterYears = Math.abs(counterYears);

System.out.println("Child Age: " + counterYears);

return counterYears;

}

public ObservableList<Children> createChildrenList() {

Connection connection=null;

Statement statement = null;

ResultSet rs = null;

ObservableList<Children> children = FXCollections.observableArrayList();

try {

connection = DBConnection.getConnection();

if (connection != null) {

Statement ps;

ps = connection.createStatement();

rs = ps.executeQuery("SELECT \* FROM childrenTable");

while (rs.next()) {

children.add(new Children(

rs.getInt("child\_Id")

));

System.out.println("Creating List");

}

connection.close();

System.out.println("Query Create Children executed, connection terminated ");

return children;

}

} catch (Exception e) {

e.printStackTrace();

}

return children;

}

}

package nurseryApp.Models;

import com.sun.org.apache.bcel.internal.generic.RETURN;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

/\*\*

\* Created by regga on 12/12/2016.

\*/

public class DBConnection {

public static final String JDBC\_driver = "com.mysql.jdbc.Driver";

public static final String JDBC\_url="jdbc:mysql://194.81.104.22:3306/db16442932" ;

// public static final String JDBC\_dbName="db16442932" ;

public static final String JDBC\_user = "s16442932";

public static final String JDBC\_password = "16442932";

public static Connection getConnection (){

Connection connection = null;

try{

Class.forName(JDBC\_driver);

connection = DriverManager.getConnection(JDBC\_url ,JDBC\_user,JDBC\_password);

}catch (Exception e){

e.printStackTrace();

}

return connection;

}

}

package nurseryApp.Models;

import nurseryApp.Controller.NurseryNotifications;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.Statement;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.concurrent.TimeUnit;

/\*\*

\* Created by regga on 12/01/2017.

\*/

public class NurserySession {

private Integer childID;

private String sessionFromDate;

private String sessionToDate;

private String sessionTimeRange;

private double sessionPrice;

public NurserySession(Integer childID, String sessionFromDate, String sessionToDate, String sessionTimeRange) {

this.childID = childID;

this.sessionFromDate = sessionFromDate;

this.sessionToDate = sessionToDate;

this.sessionTimeRange = sessionTimeRange;

this.sessionPrice = setSessionPrice();

}

public Integer getChildID() {

return childID;

}

public void setChildID(Integer childID) {

this.childID = childID;

}

public String getSessionFromDate() {

return sessionFromDate;

}

public void setSessionFromDate(String sessionFromDate) {

this.sessionFromDate = sessionFromDate;

}

public String getSessionToDate() {

return sessionToDate;

}

public void setSessionToDate(String sessionToDate) {

this.sessionToDate = sessionToDate;

}

public String getSessionTimeRange() {

return sessionTimeRange;

}

public void setSessionTimeRange(String sessionTimeRange) {

this.sessionTimeRange = sessionTimeRange;

}

//System.out.println();

public double setSessionPrice(){

Children newChild = new Children(childID);

int age = newChild.getChildAge();

System.out.println("Child age: "+ age);

Rates newRate = new Rates(age,getSessionTimeRange());

System.out.println("Session rate is : " + newRate.getSessionRate());

System.out.println("Session dates: " + daysCounter(getSessionFromDate(),getSessionToDate()));

double totalPrice=(newRate.getSessionRate())\*daysCounter(getSessionFromDate(),getSessionToDate());

System.out.println("Total price: " + totalPrice);

sessionPrice=totalPrice;

return sessionPrice;

}

public double getSessionPrice() {

return sessionPrice;

}

public int daysCounter(String dateFrom,String dateTo) {

long differenceDays=0;

SimpleDateFormat newFormat = new SimpleDateFormat("yyyy-MM-dd");

int counterDays = 0;

try {

Date dateA = newFormat.parse(dateFrom);

Date dateB = newFormat.parse(dateTo);

counterDays = (int) (dateB.getTime() - dateA.getTime());

System.out.println("Days: " + TimeUnit.DAYS.convert(counterDays, TimeUnit.MILLISECONDS));

differenceDays = TimeUnit.DAYS.convert(counterDays, TimeUnit.MILLISECONDS)+1;

return (int) differenceDays;

} catch (ParseException e) {

e.printStackTrace();

}

return (int) differenceDays;

}

public void bookSession(){

System.out.println("Starting booking procedure for "+ getChildID());

Connection connection=null;

Statement statement = null;

try{

connection = DBConnection.getConnection();

if (connection != null){

System.out.println("Conection succesfull");

PreparedStatement ps ;

ps = connection.prepareStatement("INSERT INTO sessionTable (sessionFromDate,sessionToDate,child\_id,sessionTimeRange,price)" +

"VALUES (?,?,?,?,?) ");

ps.setString(1,getSessionFromDate());

ps.setString(2, getSessionToDate());

Children newChild=new Children(childID);

ps.setInt(3, newChild.getChild\_Id());

ps.setString(4, getSessionTimeRange());

ps.setDouble(5,getSessionPrice());

ps.executeUpdate();

ps.close();

connection.close();

System.out.println("Query executed, connection terminated , session addded ");

NurseryNotifications notifyer = new NurseryNotifications();

notifyer.showNotification("Successfully Booked");

//Stage stage =(Stage) addUser.getScene().getWindow();

//stage.close();

}

}catch (Exception e){

e.printStackTrace();

}finally {

}

}

}

package nurseryApp.Models;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

/\*\*

\* Created by regga on 12/12/2016.

\*/

public class Parents extends Adults{

Charges charges;

public Parents(String name, String surname, String datOfBirth, String email, String mobile, String address, String postcode) {

super(name, surname, datOfBirth, email, mobile, address, postcode);

}

public Parents(){

super();

}

public Parents(int parentID){

Connection connection=null;

Statement statement = null;

this.parentID = parentID;

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Conection succesfull");

Statement st = null;

String query = "SELECT \* FROM parentsTable WHERE parentID="+"'"+parentID+"'" ;

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

if (resultSet.next()) {

this.name=resultSet.getString("name");

this.surname=resultSet.getString("surname");

this.datOfBirth=resultSet.getString("DoB");

this.email=resultSet.getString("email");

this.mobile=resultSet.getString("mobile");

this.address=resultSet.getString("address");

this.postcode=resultSet.getString("postcode");

}else{

System.out.println("Wrong Parent ID ");

return;

}

connection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

public ObservableList<Adults> createAdultsList() {

Connection connection=null;

Statement statement = null;

ResultSet rs = null;

ObservableList<Adults> adults = FXCollections.observableArrayList();

try {

connection = DBConnection.getConnection();

if (connection != null) {

Statement ps;

ps = connection.createStatement();

rs = ps.executeQuery("SELECT \* FROM parentsTable");

while (rs.next()) {

adults.add(new Adults(

rs.getInt("parentID")

));

System.out.println("Parent Printing List:"+ rs.getString("name"));

}

connection.close();

System.out.println("Query Create Adult list executed, connection terminated ");

return adults;

}

} catch (Exception e) {

e.printStackTrace();

}

return adults;

}

public Parents searchParentInfo(String parentSurname, String parentDoB) throws SQLException {

Connection connection=null;

Statement statement = null;

ResultSet rs = null;

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Connection succesfull");

Statement st = null;

String query = "SELECT \* FROM parentsTable WHERE surname="+"'"+parentSurname+"'" +"AND DoB ="+"'"+parentDoB+"'" ;

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

if (resultSet.next()) {

Parents parents = new Parents(resultSet.getInt("parentID"));

return parents;

}else{

System.out.println("Parent Not Found");

}

connection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public int findOwnChildren(){

Connection connection=null;

Statement statement = null;

ResultSet rs = null;

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Connection succesfull");

Statement st = null;

String query = "SELECT \* FROM childrenTable WHERE parentID="+"'"+parentID+"'" ;

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

if (resultSet.next()) {

Children children = new Children(resultSet.getInt("child\_Id"));

return children.getChild\_Id();

}else{

System.out.println("Parent Not Found");

}

connection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

return 0;

}

}

package nurseryApp.Models;

import javafx.collections.ObservableList;

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

/\*\*

\* Created by regga on 12/01/2017.

\*/

public class PrintToFile {

String message;

String fileName;

public PrintToFile(String message, String fileName, String todaysDate) throws IOException {

this.message = message;

this.fileName = fileName;

}

//

// public PrintToFile(ObservableList<listMessages>, String fileName, String todaysDate) throws IOException {

// this.message = message;

// this.fileName = fileName;

// }

public void print(){

try {

FileWriter newFileWriter = new FileWriter(fileName,true);

PrintWriter newPrintWriter = new PrintWriter(newFileWriter);

newPrintWriter.println(message);

newPrintWriter.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

package nurseryApp.Models;

/\*\*

\* Created by regga on 12/01/2017.

\*/

public class Rates {

//All Day","Morning","Lunch","Afternoon","Pre School

int age;

String sessionRange;

double rate;

public Rates(int age,String sessionRange) {

this.age = age;

this.sessionRange = sessionRange;

}

public double getSessionRate() {

switch (sessionRange) {

case "Morning":

System.out.println("Morning choosed");

if (age < 2) {

rate = 16.5;

} else if (2 <= age | age < 3) {

rate = 15.5;

} else if (age >= 3) {

rate = 14.5;

}

break;

case "All Day":

System.out.println("All day choosed");

if (age < 2) {

rate = 35;

} else if (2 <= age | age < 3) {

rate = 34;

} else if (age >= 3) {

rate = 32;

}

break;

case "Lunch":

System.out.println("Lunch choosed");

rate = 5;

break;

case "Afternoon":

System.out.println("Afternoon choosed");

if (age < 2) {

rate = 18;

} else if (2 <= age | age < 3) {

rate = 17;

} else if (age >= 3) {

rate = 16;

}

break;

case "Pre School":

if (age < 2) {

rate = 25;

} else if (2 <= age | age < 3) {

rate = 24.5;

} else if (age >= 3) {

rate = 23.5;

}

break;

}

return rate;

}

}

package nurseryApp.Models;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

/\*\*

\* Created by regga on 21/12/2016.

\*/

public class Staff extends Adults {

private String username;

private String password;

private String abailability;

public Staff(String name, String surname, String datOfBirth, String email, String mobile, String address, String postcode) {

super(name, surname, datOfBirth, email, mobile, address, postcode);

}

public Staff(){

super();

}

public String getAbailability() {

return abailability;

}

public void setAbailability(String abailability) {

this.abailability = abailability;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public ObservableList<Adults> createStaffList() {

Connection connection=null;

Statement statement = null;

ResultSet rs = null;

ObservableList<Adults> adults = FXCollections.observableArrayList();

try {

connection = DBConnection.getConnection();

if (connection != null) {

Statement ps;

ps = connection.createStatement();

rs = ps.executeQuery("SELECT \* FROM staffTable");

while (rs.next()) {

adults.add(new Adults("",

rs.getInt("staffID")

));

System.out.println("Staff Printing List:"+ rs.getString("name"));

}

connection.close();

System.out.println("Query Create staff list executed, connection terminated ");

return adults;

}

} catch (Exception e) {

e.printStackTrace();

}

return adults;

}

}

package nurseryApp.Models;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

/\*\*

\* Created by regga on 15/01/2017.

\*/

public class WaitingParents extends Parents{

String numOfChildren;

String childrenAge;

ObservableList<WaitingParents> waitingParent = FXCollections.observableArrayList();

public WaitingParents(String name, String surname,String mobile, String email,String numOfChildren, String childrenAge) {

this.name=name;

this.surname=surname;

this.mobile=mobile;

this.email=email;

this.numOfChildren= numOfChildren;

this.childrenAge=childrenAge;

}

public WaitingParents(){

}

public String getNumberOfChildren() {

return numOfChildren;

}

public void setNumberOfChildren(String numOfChildren) {

this.numOfChildren = numOfChildren;

}

public String getChildrenAge() {

return childrenAge;

}

public void setChildrenAge(String childrenAge) {

this.childrenAge = childrenAge;

}

public ObservableList<WaitingParents> createWaitingList() {

Connection connection=null;

Statement statement = null;

ResultSet rs = null;

ObservableList<Adults> adults = FXCollections.observableArrayList();

try {

connection = DBConnection.getConnection();

if (connection != null) {

Statement ps;

ps = connection.createStatement();

rs = ps.executeQuery("SELECT \* FROM waitingList");

while (rs.next()) {

waitingParent.add(new WaitingParents(

rs.getString("name"),

rs.getString("surname"),

rs.getString("mobile"),

rs.getString("email"),

rs.getString("numOfChildren"),

rs.getString("childrenAge")

));

}

connection.close();

System.out.println("Query Create waiting list executed, connection terminated ");

return waitingParent;

}

} catch (Exception e) {

e.printStackTrace();

}

return waitingParent;

}

}

package nurseryApp.Controller;

import nurseryApp.Models.DBConnection;

import nurseryApp.Models.PrintToFile;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.Statement;

import java.text.SimpleDateFormat;

import java.util.Calendar;

/\*\*

\* Created by regga on 25/12/2016.

\*/

public class AccessLog {

Connection connection=null;

Statement statement = null;

String empty = "";

String todaysDay = new SimpleDateFormat("yyyy-MM-dd | HH:mm:ss").format(Calendar.getInstance().getTime());

/\*\*

\* The method is intended for filling the database single

\* access of the user

\* @param username

\* @param employ\_ID

\*/

public void fillAccess (String username,String employ\_ID){

try{

connection = DBConnection.getConnection();

if (connection != null){

System.out.println("Connection succesfull");

PreparedStatement ps ;

ps = connection.prepareStatement("INSERT INTO sessionLog (employ\_ID,username,todaysDate )" +

"VALUES (?,?,?) ");

ps.setString(1, employ\_ID);

ps.setString(2, username);

ps.setString(3, todaysDay.toString());

ps.executeUpdate();

ps.close();

connection.close();

System.out.println("Query executed, connection terminated ");

String message = "New login for user :"+employ\_ID + " ," +" Name:" +username+", on :" + todaysDay.toString();

//String location="C:\\Users\\regga\\IdeaProjects\\NurseryApp\\src\\nurseryApp\\Resources\\logs\\Stafflog.txt";

String location="Stafflog.txt";

PrintToFile p2f=new PrintToFile(message,location,"11 jan");

p2f.print();

}

}catch (Exception e){

e.printStackTrace();

}

}

}

void searchParentEvent(ActionEvent event) throws SQLException {

System.out.println("test");

String searchDoB = parentDoB.getValue().toString();

String parentsName = parentSurname.getText();

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Conection succesfull");

Statement st = null;

String query = "SELECT \* FROM parentsTable WHERE surname="+"'"+parentsName+"'" +"AND DoB ="+"'"+searchDoB+"'" ;

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

if (resultSet.next()) {

foundParentSurname.setText(resultSet.getString("surname"));

System.out.println(resultSet.getString("surname"));

foundParentID=resultSet.getString("parentID");

foundParentName.setText(resultSet.getString("name"));

}else{

System.out.println("Parent Not Found");

Alert alert = new Alert(Alert.AlertType.INFORMATION);

alert.setTitle("NOTICE");

alert.setHeaderText("User not Found");

alert.setContentText("Please, verify parent Details");

alert.showAndWait();

}

connection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

package nurseryApp.Controller;

import com.jfoenix.controls.JFXButton;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.fxml.Initializable;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.layout.Pane;

import javafx.stage.Modality;

import javafx.stage.Stage;

import javafx.stage.StageStyle;

import nurseryApp.Models.\*;

import java.io.IOException;

import java.net.URL;

import java.sql.\*;

import java.util.ResourceBundle;

/\*\*

\* Created by regga on 18/12/2016.

\*/

public class HomeController implements Initializable {

Connection connection = null;

ResultSet rs = null;

ObservableList<Adults> parents = FXCollections.observableArrayList();

ObservableList<Children> children = FXCollections.observableArrayList();

ObservableList<Adults> staff = FXCollections.observableArrayList();

// Parents table declaration

@FXML

private TableColumn<Parents, Integer> parentID = new TableColumn<>("parentID");//fromDB to table column object ,then object to column

@FXML

private TableColumn<Parents, String> name= new TableColumn<>("name");

@FXML

private TableColumn<Parents, String> surname= new TableColumn<>("surname");

@FXML

private TableColumn<Parents, String> DoB= new TableColumn<>("DoB");

@FXML

private TableColumn<Parents, String> email = new TableColumn<>("email");

@FXML

private TableColumn<Parents, String> mobile = new TableColumn<>("mobile");

@FXML

private TableColumn<Parents, String> address = new TableColumn<>("address");

@FXML

private TableColumn<Parents, String> postcode = new TableColumn<>("postcode");

@FXML

private TableView<Adults> parentsTable;

@FXML

private JFXButton refreshParents;

@FXML

void refreshParentEvent(ActionEvent event) {

parentsTable.getItems().clear();

loadParents();

}

// end parents table declaration

// Staff table declaration

@FXML

private TableColumn<Staff, Integer> staffID = new TableColumn<>("staffID");

@FXML

private TableColumn<Staff, String> staffNameTable= new TableColumn<>("name");

@FXML

private TableColumn<Staff, String> staffSurname= new TableColumn<>("surname");

@FXML

private TableColumn<Staff, String> staffAvailability= new TableColumn<>("abailability");

@FXML

private TableColumn<Staff, String> staffMobile = new TableColumn<>("mobile");

@FXML

private TableColumn<Staff, String> staffEmail = new TableColumn<>("email");

@FXML

private TableView<Adults> staffTable;

@FXML

private JFXButton refreshStaffTableButton;

@FXML

void refreshStaffTableButtonEvent(ActionEvent event) {

staffTable.getItems().clear();

loadStaff();

}

// end Staff table declaration

// children table declaration

@FXML

private TableColumn<Children, Integer> childID = new TableColumn<>("child\_Id");

@FXML

private TableColumn<Children, String> childName= new TableColumn<>("name");

@FXML

private TableColumn<Children, String> childSurname= new TableColumn<>("surname");

@FXML

private TableColumn<Children, String> childDoB = new TableColumn<>("DoB");

@FXML

private TableColumn<Children, String> childAllergy = new TableColumn<>("allergy");

@FXML

private TableColumn<Children, Integer> parentIDChildTable = new TableColumn<>("parentID");

@FXML

private TableColumn<Children, String> childCategory = new TableColumn<>("category");

@FXML

private TableView<Children> childrenTable;

@FXML

private JFXButton refreshChildTable;

@FXML

void refreshChildTableEvent(ActionEvent event) {

childrenTable.getItems().clear();

loadChildren();

}

// end children declaration

@FXML

private JFXButton registerChild;

@FXML

private Pane homePane;

@FXML

private JFXButton exitButton;

@FXML

private JFXButton invoiceButton;

@FXML

void invoiceButtonEvent(ActionEvent event) throws IOException {

windowSelector("../Views/invoice.fxml");

}

@FXML

void sessionEvent(ActionEvent event) throws IOException {

windowSelector("../Views/session.fxml");

}

@FXML

void searchEvent(ActionEvent event) throws IOException {

windowSelector("../Views/search.fxml");

}

@FXML

void sheduleEvent(ActionEvent event) throws IOException {

windowSelector("../Views/waitingList.fxml");

}

@FXML

void exitButtonEvent(ActionEvent event) {

Stage stage =(Stage) exitButton.getScene().getWindow();

stage.close();

}

@FXML

void registerChildEvent(ActionEvent event) throws IOException {

windowSelector("../Views/addChildren.fxml");

}

@FXML

void addUser(ActionEvent event) throws IOException {

windowSelector("../Views/addUser.fxml");

}

void windowSelector (String location) throws IOException {

try {

Parent parent = FXMLLoader.load(getClass().getResource(location));

Stage stage= new Stage((StageStyle.TRANSPARENT));

stage.initModality(Modality.APPLICATION\_MODAL);

stage.setScene(new Scene(parent));

stage.showAndWait();

} catch (IOException e) {

e.printStackTrace();

}

}

// public ObservableList<Person> getParent(){

// setupColumnNames();

//

//

//

// for (Person p : person

// ) {

// System.out.println(person);

//

// }

//

// return person;

// }

@Override

public void initialize(URL location, ResourceBundle resources) {

setupColumnNames();

loadParents();

loadChildren();

NurseryNotifications notifyer = new NurseryNotifications();

notifyer.showNotification("new Session Started");

Children newC= new Children(5);

}

private void loadParents() {

Parents newParents = new Parents();

parents=newParents.createAdultsList();

parentsTable.setItems(parents);

}

private void loadStaff() {

Staff newStaff = new Staff();

staff=newStaff.createStaffList();

staffTable.setItems(staff);

}

private void loadChildren() {

Children newChildren= new Children();

children= newChildren.createChildrenList();

childrenTable.setItems(children);

}

public void setupColumnNames(){

//parents

parentID.setCellValueFactory(new PropertyValueFactory<>("parentID"));

name.setCellValueFactory(new PropertyValueFactory<>("name"));

surname.setCellValueFactory(new PropertyValueFactory<>("surname"));

DoB.setCellValueFactory(new PropertyValueFactory<>("datOfBirth"));

email.setCellValueFactory(new PropertyValueFactory<>("email"));

mobile.setCellValueFactory(new PropertyValueFactory<>("mobile"));

address.setCellValueFactory(new PropertyValueFactory<>("address"));

postcode.setCellValueFactory(new PropertyValueFactory<>("postcode"));

//children

childID.setCellValueFactory(new PropertyValueFactory<>("child\_Id"));

childName.setCellValueFactory(new PropertyValueFactory<>("name"));

childSurname.setCellValueFactory(new PropertyValueFactory<>("surname"));

childDoB.setCellValueFactory(new PropertyValueFactory<>("dateOfBirth"));

parentIDChildTable.setCellValueFactory(new PropertyValueFactory<>("parentID"));

childCategory.setCellValueFactory(new PropertyValueFactory<>("category"));

childAllergy.setCellValueFactory(new PropertyValueFactory<>("allergy"));

//Staff

staffID.setCellValueFactory(new PropertyValueFactory<>("staffID"));

staffNameTable.setCellValueFactory(new PropertyValueFactory<>("staffName"));

staffSurname.setCellValueFactory(new PropertyValueFactory<>("staffSurname"));

staffAvailability.setCellValueFactory(new PropertyValueFactory<>("staffAvailability"));

staffMobile.setCellValueFactory(new PropertyValueFactory<>("staffMobile"));

staffEmail.setCellValueFactory(new PropertyValueFactory<>("staffEmail"));

}

}

package nurseryApp.Controller;

import com.jfoenix.controls.JFXButton;

import com.jfoenix.controls.JFXListView;

import com.jfoenix.controls.JFXTextField;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.Label;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.stage.Stage;

import nurseryApp.Models.Charges;

import nurseryApp.Models.Children;

import nurseryApp.Models.DBConnection;

import nurseryApp.Models.PrintToFile;

import java.io.IOException;

import java.net.URL;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

import java.util.ResourceBundle;

public class InvoiceController implements Initializable{

String finalBalance;

NurseryNotifications notifyer = new NurseryNotifications();

Children children= new Children();

Connection connection=null;

Statement statement = null;

ObservableList<Charges> newCharges = FXCollections.observableArrayList();

@FXML

private TableColumn<Charges, Float> balanceField;

@FXML

private TableColumn<Charges, Float> extraChargesField;

@FXML

private TableColumn<Charges,Double> price= new TableColumn<>("price");

@FXML

private TableColumn<Charges,String> generatedDate= new TableColumn<>("generatedDate");

@FXML

private TableColumn<Charges,Integer> childIDTableField= new TableColumn<>("childIDTableField");

@FXML

private TableColumn<Charges,String> description = new TableColumn<>("description");

@FXML

private TableView<Charges> invoiceTableView;

@FXML

private JFXButton cancelButton;

@FXML

private Label foundChildID;

@FXML

private JFXTextField childIDField;

@FXML

private JFXButton finderButton;

@FXML

private JFXButton generateInvoice;

@FXML

private Label foundName;

@FXML

private Label totalBalancelabel;

@FXML

private JFXButton savetoFile;

@FXML

void generateInvoiceEvent(ActionEvent event) {

final double[] balance = {0};

Charges charges = new Charges();

newCharges=charges.chargesList(children.getChild\_Id());

invoiceTableView.setItems(newCharges);

newCharges.forEach((price)->{ // lampda expression to generate a sum of all prices found

balance[0] = price.getPrice()+ balance[0];

System.out.println(balance[0]);

});

finalBalance=Double.toString(balance[0]);

totalBalancelabel.setText(Double.toString(balance[0]));

}

@FXML

void savetoFileEvent(ActionEvent event) throws IOException {

String location = children.getSurname()+children.getName()+children.getChild\_Id()+".txt";

String message =("Children Surname: "+ children.getSurname()+". \n Children Name"+ children.getName()+".\n New Balance:"+ finalBalance);

PrintToFile p2f=new PrintToFile(message,location,"11 jan");

p2f.print();

System.out.println(message);

}

@FXML

void finderButtonEvent(ActionEvent event) {

Children children = getChildreDetails();

foundChildID.setText(children.getName()+" "+children.getSurname());

}

@FXML

void cancelButtonEvent(ActionEvent event) {

Stage stage =(Stage) cancelButton.getScene().getWindow();

stage.close();

notifyer.showNotification("Charges Cancelled");

}

public Children getChildreDetails() {

System.out.println("childIDFinderEvent\_ start");

Integer searchID = Integer.parseInt(childIDField.getText());

String foundChildName;

Boolean checkingID = false;

try {

connection = DBConnection.getConnection();

if (connection != null) {

System.out.println("Conection succesfull");

Statement st = null;

String query = "SELECT \* FROM childrenTable WHERE child\_Id=" + "'" + searchID + "'";

st = connection.createStatement();

ResultSet resultSet = st.executeQuery(query);

if (resultSet.next()) {

children = new Children(resultSet.getInt("child\_Id"));

return children;

} else {

notifyer.showNotification("Child Not Found");

System.out.println("Child Not Found");

}

connection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

return children;

}

@Override

public void initialize(URL location, ResourceBundle resources) {

childIDTableField.setCellValueFactory(new PropertyValueFactory<>("childID"));

generatedDate.setCellValueFactory(new PropertyValueFactory<Charges, String>("generatedDate"));

price.setCellValueFactory(new PropertyValueFactory<Charges, Double>("price"));

description.setCellValueFactory(new PropertyValueFactory<Charges, String>("description"));

}

}

package nurseryApp.Controller;

import com.jfoenix.controls.JFXButton;

import com.jfoenix.controls.JFXPasswordField;

import com.jfoenix.controls.JFXTextField;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.scene.control.Button;

import javafx.scene.control.Label;

import javafx.scene.control.PasswordField;

import javafx.scene.control.TextField;

import javafx.scene.layout.Pane;

import javafx.stage.Stage;

import nurseryApp.Models.DBConnection;

import nurseryApp.Models.Parents;

import java.awt.\*;

import java.io.IOException;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.Statement;

public class LoginController {

Connection connection=null;

Statement statement = null;

String userDB=null;

//private Parents newParents ;

@FXML

private Pane rootPane;

@FXML

private Label homeLabel;

@FXML

private JFXButton loginButton;

@FXML

private JFXButton exitButton;

@FXML

private JFXPasswordField myPassword;

@FXML

private JFXTextField usernameField;

@FXML

private Pane loginBg;

@FXML

void onEnter(ActionEvent event) throws IOException {

if (accessGranted()) {

Pane homePane= FXMLLoader.load(getClass().getResource("../Views/Home.fxml"));

rootPane.getChildren().setAll(homePane);

} else {

System.out.println("Access Denied");

}

}

/\* Initialize Home Screen \*/

@FXML

void loginButtonEvent(ActionEvent event) throws IOException {

if (accessGranted()) {

Pane homePane= FXMLLoader.load(getClass().getResource("../Views/Home.fxml"));

rootPane.getChildren().setAll(homePane);

} else {

System.out.println("Access Denied");

}

}

/\* Exit Login Screen \*/

@FXML

void exitButtonEvent(ActionEvent event) {

Stage stage =(Stage) exitButton.getScene().getWindow();

stage.close();

}

private boolean accessGranted()

{

String newUser = usernameField.getText();

String newPass = myPassword.getText();

boolean access = false;

try{

connection = DBConnection.getConnection();

if (connection != null){

System.out.println("Conection succesfull");

Statement st=null;

String query="SELECT \* FROM staffTable WHERE username="+"'"+newUser+"'" ;

st= connection.createStatement();

ResultSet resultSet=st.executeQuery(query);

if (resultSet.next()){

userDB = resultSet.getString("username");

String employID = resultSet.getString("staffID");

String passDB = resultSet.getString("password");

if ((userDB.equals(newUser)) &&(passDB.equals(newPass))){

System.out.println("Access Granted");

AccessLog newAccess = new AccessLog();

newAccess.fillAccess(userDB,employID);

return true;

}else{

System.out.println("wrong credentials:"+userDB +" "+passDB);

String errorMsg="try again";

System.out.println("Access Denied");

//homeLabel.setTextFill(Color.red);

homeLabel.setText("Access Denied");

usernameField.setPromptText(errorMsg);

return false;

}

}else{

homeLabel.setText("Access Denied");

System.out.println("Wrong Password");

}

}

}catch (Exception e){

e.printStackTrace();

}finally {

}

return false;

}

}

package nurseryApp.Controller;

import javafx.geometry.Pos;

import javafx.util.Duration;

/\*\*

\* Created by regga on 12/01/2017.

\*/

public class NurseryNotifications {

public String getMessage() {

return message;

}

public void setMessage(String message) {

this.message = message;

}

String tittle;

String message;

public void showNotification(String message){

org.controlsfx.control.Notifications sessionNotification= org.controlsfx.control.Notifications.create()

.title(tittle)

.text(message)

.darkStyle()

.hideAfter(Duration.seconds(5))

.position(Pos.BOTTOM\_RIGHT)

.onAction(e-> {

System.out.println("notification has been clicked");

});

sessionNotification.showConfirm();

}

}

package nurseryApp.Controller;

/\*\*

\* Created by regga on 15/01/2017.

\*/

import com.jfoenix.controls.JFXButton;

import com.jfoenix.controls.JFXTextField;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.scene.Parent;

import javafx.scene.control.DatePicker;

import javafx.scene.control.Label;

import javafx.stage.Stage;

import nurseryApp.Models.Bill;

import nurseryApp.Models.Charges;

import nurseryApp.Models.Children;

import nurseryApp.Models.Parents;

import java.sql.SQLException;

public class SearchController {

@FXML

private JFXTextField childrenIDSearch;

@FXML

private JFXTextField parentSurnameSearch;

@FXML

private Label searchingStatusField;

@FXML

private JFXButton searchParentInfo;

@FXML

private JFXButton exitSearchButton;

@FXML

private JFXTextField childrenRhField;

@FXML

private JFXTextField ChildrenSurnameField;

@FXML

private JFXTextField parentDuedateField;

@FXML

private DatePicker parentDoBSearch;

@FXML

private JFXTextField addChargesField;

@FXML

private JFXTextField childrenNameField;

@FXML

private JFXTextField parentMobileField;

@FXML

private JFXTextField parentBalanceField;

@FXML

private JFXButton searchChildrenInfo;

@FXML

private JFXTextField childrenIDField;

@FXML

private JFXTextField parentEmailField;

@FXML

private JFXTextField childrenCategoryField;

@FXML

private JFXButton addChargesButton;

@FXML

private JFXTextField parentSurnameField;

@FXML

private JFXTextField parentDobField;

@FXML

private JFXTextField childrenAllergyField;

@FXML

private JFXTextField childrenDoBField;

@FXML

private JFXTextField parentDiscountField;

@FXML

private JFXTextField parentAddressField;

@FXML

private JFXTextField parentPoscodeField;

@FXML

private JFXTextField childrenNaneSearch;

@FXML

private JFXTextField parentNameField;

@FXML

void addChargesButton(ActionEvent event) {

Charges charges= new Charges();

//charges.addCharges(child);

}

@FXML

void searchChildrenInfoEvent(ActionEvent event) {

Children children=new Children(Integer.parseInt(childrenIDSearch.getText()));

fillChildrenFields(children);

}

@FXML

void searchParentInfoEvent(ActionEvent event) throws SQLException {

Parents parents= new Parents();

fillparentFields(parents.searchParentInfo(parentSurnameSearch.getText(),parentDoBSearch.getValue().toString()));

}

private void fillparentFields(Parents parents) {

parentNameField.setText(parents.getName());

parentSurnameField.setText(parents.getSurname());

parentDobField.setText(parents.getdatOfBirth());

parentEmailField.setText(parents.getEmail());

parentMobileField.setText(parents.getMobile());

parentAddressField.setText(parents.getAddress());

parentPoscodeField.setText(parents.getPostcode());

Children children = new Children(parents.findOwnChildren());

Bill bill = new Bill(parents.getParentID());

parentBalanceField.setText(String.valueOf(bill.getBalance()));

}

@FXML

void exitSearchButtonEvent(ActionEvent event) {

Stage stage =(Stage) exitSearchButton.getScene().getWindow();

stage.close();

}

public void fillChildrenFields(Children children){

childrenNaneSearch.setText(children.getName()+" ,"+ children.getSurname());

childrenIDField.setText(String.valueOf(children.getChild\_Id()));

childrenNameField.setText(children.getName());

ChildrenSurnameField.setText(children.getSurname());

childrenDoBField.setText(children.getDateOfBirth());

childrenCategoryField.setText(children.getCategory());

childrenRhField.setText(children.getAllergy());

childrenAllergyField.setText(children.getAllergy());

Parents parents= new Parents(children.getParentID());

fillparentFields(parents);

}

}

package nurseryApp.Controller;

import com.jfoenix.controls.JFXButton;

import com.jfoenix.controls.JFXTextField;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.\*;

import javafx.stage.Stage;

import nurseryApp.Models.Children;

import nurseryApp.Models.DBConnection;

import nurseryApp.Models.NurserySession;

import java.net.URL;

import java.sql.\*;

import java.util.ResourceBundle;

/\*\*

\* Created by regga on 31/12/2016.

\*/

public class SessionController implements Initializable{

Connection connection=null;

Statement statement = null;

String userDB=null;

String foundParentID = null;

int foundChildID;

int fouundChildDoB;

NurseryNotifications notifyer = new NurseryNotifications();

@FXML

private DatePicker fromDate;

@FXML

private DatePicker toDate;

@FXML

private JFXButton bookSession;

@FXML

private JFXButton exitbookSession;

@FXML

private ChoiceBox<String> timeRange;

@FXML

private TextField childIDFinder;

@FXML

private JFXTextField foundTextField;

@FXML

private JFXTextField foundTextField2;

@FXML

void childIDFinderButtonEvent(ActionEvent event) {

if (childIDFinder.getText().equals("") ){

notifyer.showNotification(" Please Fill all details");

}

else{

int id=Integer.parseInt(childIDFinder.getText());

Children children = new Children(id);

foundTextField.setText(children.getName());

foundTextField2.setText(children.getSurname());

}

}

@FXML

void exitbookSessionEvent(ActionEvent event) {

Stage stage =(Stage) exitbookSession.getScene().getWindow();

stage.close();

}

@FXML

public void childIDFinderEvent (ActionEvent event) throws SQLException {

}

@FXML

void bookSessionEvent(ActionEvent event) {

if ( (childIDFinder.getText() != "" )&& (foundTextField.getText() != null) && (fromDate != null) && (toDate != null)&&(timeRange != null)){

int id=Integer.parseInt(childIDFinder.getText());

String from=fromDate.getValue().toString();

String to=toDate.getValue().toString();

String range=timeRange.getValue().toString();

NurserySession newSession = new NurserySession(id,from,to,range);

System.out.println("Price in Button:"+ newSession.getSessionPrice());

newSession.bookSession();

}

else{

notifyer.showNotification(" Please Fill all details");

}

}

@Override

public void initialize(URL location, ResourceBundle resources) {

timeRange.getItems().addAll("All Day","Morning","Lunch","Afternoon","Pre School");

}

}

package nurseryApp.Controller;

/\*\*

\* Created by regga on 15/01/2017.

\*/

import com.jfoenix.controls.JFXButton;

import com.jfoenix.controls.JFXTextField;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.stage.Stage;

import nurseryApp.Models.Charges;

import nurseryApp.Models.DBConnection;

import nurseryApp.Models.WaitingParents;

import java.net.URL;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.Statement;

import java.util.ResourceBundle;

public class WaitingListController implements Initializable {

ObservableList<WaitingParents> waitingparents = FXCollections.observableArrayList();

@FXML

private JFXTextField childrenAgeField;

@FXML

private JFXButton exitWaitingButton;

@FXML

private TableColumn<WaitingParents, String> numChildColumn = new TableColumn<>("numOdChildren");

@FXML

private TableColumn<WaitingParents, String> surnameColumn = new TableColumn<>("surname");

@FXML

private TableColumn<WaitingParents,String> numberColumn = new TableColumn<>("mobile");

@FXML

private JFXButton refreshWaitingListButton;

@FXML

private JFXTextField parentEmailField;

@FXML

private JFXTextField numChildrenField;

@FXML

private JFXTextField parentNumberField;

@FXML

private JFXTextField parentSurnameField;

@FXML

private JFXButton addWaitiningButton;

@FXML

private TableColumn<WaitingParents,String> nameColumn = new TableColumn<>("name");

@FXML

private JFXTextField parentNameField;

@FXML

private TableColumn<WaitingParents, String> emailColumn = new TableColumn<>("email");

@FXML

private TableView<WaitingParents> waitingTable;

@FXML

void refreshWaitingListButton(ActionEvent event) {

WaitingParents waitingParents = new WaitingParents();

waitingparents= waitingParents.createWaitingList();

waitingTable.setItems(waitingparents);

}

@FXML

void addWaitiningButtonEvent(ActionEvent event) {

insertData();

}

@FXML

void exitWaitingButton(ActionEvent event) {

Stage stage =(Stage) exitWaitingButton.getScene().getWindow();

stage.close();

}

void insertData (){

Connection connection=null;

Statement statement = null;

String empty = "";

try{

connection = DBConnection.getConnection();

if (connection != null){

System.out.println("Conection succesfull");

PreparedStatement ps ;

ps = connection.prepareStatement("INSERT INTO waitingList (name,surname,mobile,email,numOfChildren,childrenAge)" +

"VALUES (?,?,?,?,?,?) ");

ps.setString(1, parentNameField.getText());

ps.setString(2, parentSurnameField.getText());

ps.setString(3, parentNumberField.getText());

ps.setString(4, parentEmailField.getText());

ps.setString(5, numChildrenField.getText());

ps.setString(6, childrenAgeField.getText());

ps.executeUpdate();

ps.close();

connection.close();

System.out.println("Query executed: insert new waiting list candidate, connection terminated ");

}

}catch (Exception e){

e.printStackTrace();

}

}

@Override

public void initialize(URL location, ResourceBundle resources) {

nameColumn.setCellValueFactory(new PropertyValueFactory<>("name"));

surnameColumn.setCellValueFactory(new PropertyValueFactory<>("surname"));

numberColumn.setCellValueFactory(new PropertyValueFactory<>("mobile"));

emailColumn.setCellValueFactory(new PropertyValueFactory<>("email"));

numChildColumn.setCellValueFactory(new PropertyValueFactory<>("numOfChildren"));

}

}

package nurseryApp;

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.stage.Stage;

import javafx.stage.StageStyle;

public class Main extends Application {

@Override

public void start(Stage primaryStage) throws Exception{

Parent root = FXMLLoader.load(getClass().getResource("Views/login.fxml"));

primaryStage.initStyle(StageStyle.TRANSPARENT);

primaryStage.setScene(new Scene(root, 800, 600));

primaryStage.show();

}

public static void main(String[] args) {

launch(args);

}

}