**JAVA**

**HANDS-ON**

REGISTER

package com.cognizant.customer.beans;

public class Register {

private String name;

private String emailid;

private String phoneNumber;

private String password;

private int customerid;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmailid() {

return emailid;

}

public void setEmailid(String emailid) {

this.emailid = emailid;

}

public String getPhoneNumber() {

return phoneNumber;

}

public void setPhoneNumber(String phoneNumber) {

this.phoneNumber = phoneNumber;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public int getCustomerid() {

return customerid;

}

public void setCustomerid(int customerid) {

this.customerid = customerid;

}

@Override

public String toString() {

return "Register [name=" + name + ", emailid=" + emailid + ", phoneNumber=" + phoneNumber

+ ", customerid=" + customerid + "]"+"\n";

}

}

Customer imp

package com.cognizant.customers.services;

import java.sql.Date;

import java.util.List;

import com.cognizant.customer.beans.Bookings;

import com.cognizant.customer.beans.Register;

import com.cognizant.customer.beans.room;

import com.cognizant.customer.util.userExceptions;

import com.cognizant.customers.dao.customerDao;

public class Customerimp implements ICustomer{

customerDao dao=new customerDao();

@Override

public String addCustomer(Register c) throws userExceptions {

return dao.addCustomer(c);

}

@Override

public String UpdateEmail(String email, int customerid) {

// TODO Auto-generated method stub

return dao.UpdateEmail(email, customerid);

}

@Override

public int Login(String email, String password) {

// TODO Auto-generated method stub

return dao.Login(email, password);

}

@Override

public List<Bookings> quit(int bookid) {

// TODO Auto-generated method stub

return dao.quit(bookid);

}

@Override

public double fetchPrice(Bookings br) {

// TODO Auto-generated method stub

return dao.fetchPrice(br);

}

@Override

public List<room> searchRoom(Bookings b,String roomtype) {

// TODO Auto-generated method stub

return dao.searchRoom(b,roomtype);

}

@Override

public List<Bookings> BookAroom(int roomid, Bookings b, Date checkin) {

// TODO Auto-generated method stub

return dao.BookAroom(roomid, checkin, b);

}

}

EMPLOYEEIMP

package com.cognizant.customers.services;

import java.util.List;

import com.cognizant.customer.beans.Bookings;

import com.cognizant.customer.beans.Register;

import com.cognizant.customer.beans.room;

import com.cognizant.customer.util.userExceptions;

import com.cognizant.customers.dao.employeeDao;

public class Employeeimp implements IEmployee{

employeeDao dao=new employeeDao();

@Override

public String AddRoom(room r) throws userExceptions {

// TODO Auto-generated method stub

return dao.AddRoom(r);

}

@Override

public String DeleteRoom(int roomid) {

// TODO Auto-generated method stub

return dao.DeleteRoom(roomid);

}

@Override

public String UpdateRoomid(int roomid,int newroomid) {

// TODO Auto-generated method stub

return dao.UpdateRoomid(roomid, newroomid);

}

@Override–––

public List<room> getrooms() {

// TODO Auto-generated method stub

return dao.getrooms();

}

@Override

public String UpdateRoomtype(int roomid, String roomtype) {

// TODO Auto-generated method stub

return dao.UpdateRoomtype(roomid, roomtype);

}

@Override

public String Updateprice(int roomid, int price) {

// TODO Auto-generated method stub

return dao.Updateprice(roomid, price);

}

@Override

public String Updateview(int roomid, String view) {

// TODO Auto-generated method stub

return dao.Updateview(roomid, view);

}

@Override

public String UpdateRoomstatus(int roomid, String roomstatus) {

// TODO Auto-generated method stub

return dao.UpdateRoomstatus(roomid, roomstatus);

}

@Override

public List<Register> AllCustomers() {

// TODO Auto-generated method stub

return dao.AllCustomers();

}

@Override

public List<Bookings> AllBookings() {

// TODO Auto-generated method stub

return dao.AllBookings();

}

}

user Exception

package com.cognizant.customer.util;

@SuppressWarnings("serial")

public class userExceptions extends Exception{

public userExceptions(String message) {

super(message);

// TODO Auto-generated constructor stub

}

}

UTILITY

package com.cognizant.customer.util;

import java.io.File;

import java.util.List;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import javax.xml.transform.Transformer;

import javax.xml.transform.TransformerException;

import javax.xml.transform.TransformerFactory;

import javax.xml.transform.dom.DOMSource;

import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Attr;

import org.w3c.dom.Document;

import org.w3c.dom.Element;

import com.cognizant.customer.beans.Bookings;

import com.cognizant.customer.beans.Register;

import com.cognizant.customer.beans.room;

public class Utility {

public static final String xmlFilePath = "C:\\Users\\2050050\\OneDrive - Cognizant\\Desktop\\xml\\customer.xml";

public static final String xmlFilePath1 = "C:\\Users\\2050050\\OneDrive - Cognizant\\Desktop\\xml\\room.xml";

public static final String xmlFilePath2 = "C:\\Users\\2050050\\OneDrive - Cognizant\\Desktop\\xml\\booking.xml";

public static void generateXml(List<Register> list) {

try {

DocumentBuilderFactory documentFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder documentBuilder = documentFactory.newDocumentBuilder();

Document document = documentBuilder.newDocument();

// root element

Element root = document.createElement("customers");

document.appendChild(root);

for(Register d:list) {

// employee element

Element customer = document.createElement("customer");

root.appendChild(customer);

// set an attribute to element

Attr attr = document.createAttribute("customerid");

attr.setValue(String.valueOf(d.getCustomerid()));

customer.setAttributeNode(attr);

//you can also use staff.setAttribute("id", "1") for this

Element name = document.createElement("name");

name.appendChild(document.createTextNode(d.getName()));

customer.appendChild(name);

// email element

Element email = document.createElement("email");

email.appendChild(document.createTextNode(d.getEmailid()));

customer.appendChild(email);

Element phonenumber = document.createElement("phonenumber");

phonenumber.appendChild(document.createTextNode(d.getPhoneNumber()));

customer.appendChild(phonenumber);

// department elements

Element password = document.createElement("password");

password.appendChild(document.createTextNode(d.getPassword()));

customer.appendChild(password);

}

// create the xml file

//transform the DOM Object to an XML File

TransformerFactory transformerFactory = TransformerFactory.newInstance();

Transformer transformer = transformerFactory.newTransformer();

DOMSource domSource = new DOMSource(document);

StreamResult streamResult = new StreamResult(new File(xmlFilePath));

// If you use

// StreamResult result = new StreamResult(System.out);

// the output will be pushed to the standard output ...

// You can use that for debugging

transformer.transform(domSource, streamResult);

System.out.println("Done creating XML File");

} catch (ParserConfigurationException pce) {

pce.printStackTrace();

} catch (TransformerException tfe) {

tfe.printStackTrace();

}

}

public static void generateRoomsXml(List<room> list) {

try {

DocumentBuilderFactory documentFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder documentBuilder = documentFactory.newDocumentBuilder();

Document document = documentBuilder.newDocument();

// root element

Element root = document.createElement("rooms");

document.appendChild(root);

for(room d:list) {

// employee element

Element customer = document.createElement("room");

root.appendChild(customer);

// set an attribute to element

Attr attr = document.createAttribute("roomid");

attr.setValue(String.valueOf(d.getRoomid()));

customer.setAttributeNode(attr);

//you can also use staff.setAttribute("id", "1") for this

Element name = document.createElement("roomtype");

name.appendChild(document.createTextNode(d.getRoomtype()));

customer.appendChild(name);

// email element

Element email = document.createElement("price");

email.appendChild(document.createTextNode(String.valueOf(d.getPrice())));

customer.appendChild(email);

Element phonenumber = document.createElement("view");

phonenumber.appendChild(document.createTextNode(d.getView()));

customer.appendChild(phonenumber);

// department elements

}

// create the xml file

//transform the DOM Object to an XML File

TransformerFactory transformerFactory = TransformerFactory.newInstance();

Transformer transformer = transformerFactory.newTransformer();

DOMSource domSource = new DOMSource(document);

StreamResult streamResult = new StreamResult(new File(xmlFilePath1));

// If you use

// StreamResult result = new StreamResult(System.out);

// the output will be pushed to the standard output ...

// You can use that for debugging

transformer.transform(domSource, streamResult);

System.out.println("Done creating XML File");

} catch (ParserConfigurationException pce) {

pce.printStackTrace();

} catch (TransformerException tfe) {

tfe.printStackTrace();

}

}

public static void generateBookingsXml(List<Bookings> list) {

try {

DocumentBuilderFactory documentFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder documentBuilder = documentFactory.newDocumentBuilder();

Document document = documentBuilder.newDocument();

// root element

Element root = document.createElement("bookings");

document.appendChild(root);

for(Bookings d:list) {

// employee element

Element customer = document.createElement("booking");

root.appendChild(customer);

// set an attribute to element

Attr attr = document.createAttribute("bookid");

attr.setValue(String.valueOf(d.getBookid()));

customer.setAttributeNode(attr);

//you can also use staff.setAttribute("id", "1") for this

Element name = document.createElement("customer\_name");

name.appendChild(document.createTextNode(d.getCustomername()));

customer.appendChild(name);

// email element

Element email = document.createElement("room\_id");

email.appendChild(document.createTextNode(String.valueOf(d.getRoomid())));

customer.appendChild(email);

Element phonenumber = document.createElement("checkin");

phonenumber.appendChild(document.createTextNode(d.getCheckin().toString()));

customer.appendChild(phonenumber);

// department elements

Element password = document.createElement("checkout");

password.appendChild(document.createTextNode(d.getCheckout().toString()));

customer.appendChild(password);

Element status = document.createElement("room\_status");

status.appendChild(document.createTextNode(d.getRoomstatus()));

customer.appendChild(status);

}

// create the xml file

//transform the DOM Object to an XML File

TransformerFactory transformerFactory = TransformerFactory.newInstance();

Transformer transformer = transformerFactory.newTransformer();

DOMSource domSource = new DOMSource(document);

StreamResult streamResult = new StreamResult(new File(xmlFilePath2));

// If you use

// StreamResult result = new StreamResult(System.out);

// the output will be pushed to the standard output ...

// You can use that for debugging

transformer.transform(domSource, streamResult);

System.out.println("Done creating XML File");

} catch (ParserConfigurationException pce) {

pce.printStackTrace();

} catch (TransformerException tfe) {

tfe.printStackTrace();

}

}

}

VALIDATION

package com.cognizant.customer.util;

import java.util.Date;

public class Validation {

public static boolean nameValidation(String name) {

String arg1="[A-Za-z]+";

if( name.matches(arg1)) {

return true;

}

else {

try {

throw new userExceptions("only alphabets");

} catch (userExceptions e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

return false;

}

public static boolean numberValidation(String number) {

String arg2="[0-9]{10}";

if( number.matches(arg2)) {

return true;

}

else {

try {

throw new userExceptions("enter 10 numbers");

} catch (userExceptions e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

return false;

}

public static boolean emailValidation(String email) {

String arg3="[a-zA-Z0-9.\_%+-]+@[a-z0-9.-]+\\.[a-zA-Z]{2,4}";

if( email.matches(arg3)) {

return true;

}

else {

try {

throw new userExceptions("enter mail like:'name@gmail.com'");

} catch (userExceptions e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

return false;

}

public static boolean DateComparison(Date dte) throws userExceptions {

long now = System.currentTimeMillis();

Date sqlDate = new Date(now);

if(dte.compareTo(sqlDate)>=0) {

return true;

}

else {

throw new userExceptions("Date is not in range.\nEnter the correct date");

}

}

}

race Garden

package com.cognizant.customer.main;

import java.sql.Connection;

import java.sql.Date;

import com.cognizant.customers.services.\*;

import java.util.Scanner;

import com.cognizant.customer.beans.\*;

import com.cognizant.customer.util.ConnectionUtil;

import com.cognizant.customer.util.Utility;

import com.cognizant.customer.util.UtilityTest;

import com.cognizant.customer.util.Validation;

import com.cognizant.customer.util.userExceptions;

public class RaceGarden {

@SuppressWarnings({ "resource", "unused" })

public static void main(String[] args) {

Scanner src=new Scanner(System.in);

Register reg=new Register();

room roo=new room();

Bookings boo=new Bookings();

Connection con=ConnectionUtil.getconnection();

IEmployee employeeService=new Employeeimp();

ICustomer customerService=new Customerimp();

while(true) {

System.out.println("enter 1: employee,\n enter 2: customer \n enter 3: exit");

int n=src.nextInt();

switch(n) {

case 1:boolean emp=true;

while(emp) {

System.out.println("Employee....... ");

System.out.println("enter 1: add a room

+ "\n enter 2: delete a room"

+ "\n enter 3: update a room"

+ "\n enter 4: display all rooms"

+ "\n enter 5: display all customers"

+ "\n enter 6: show all bookings"

+ "\n enter 7: happy exit"

);

int e=src.nextInt();

switch(e) {

case 1: System.out.println("ADD A ROOM");

System.out.println("roomid");

int roomid=src.nextInt();

roo.setRoomid(roomid);

System.out.println("room type");

String roomtype=src.next();

roo.setRoomtype(roomtype);

System.out.println("price");

int price =src.nextInt();

roo.setPrice(price);

System.out.println("view");

String view=src.next();

view=view +src.nextLine();

roo.setView(view);

System.out.println("room status");

String roomstatus=src.next();

roo.setRoomstatus(roomstatus);

try {

employeeService.AddRoom(roo);

} catch (userExceptions e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

break;

case 2: System.out.println("delete a room");

System.out.println("enter a room id");

int roomid1=src.nextInt();

employeeService.DeleteRoom(roomid1);

break;

case 3:

System.out.println("update a room");

System.out.println("enter roomid");

int roomids=src.nextInt();

boolean oo=true;

while(oo){

System.out.println("enter 1:roomid\n enter 2: roomtype\n enter 3:price \n enter 4:view\n enter 5: roomstatus \n enter 6:exit");

int s=src.nextInt();

switch(s) {

case 1: System.out.println("enter new roomid");

int newroomid=src.nextInt();

employeeService.UpdateRoomid(roomids, newroomid);

break;

case 2:System.out.println("enter roomtype");

String roomtype1=src.next();

employeeService.UpdateRoomtype(roomids, roomtype1);

break;

case 3: System.out.println("enter price");

int price1=src.nextInt();

employeeService.Updateprice(roomids, price1);

break;

case 4:System.out.println("enter view");

String view2=src.next();

view2=view2+src.nextLine();

employeeService.Updateview(roomids, view2);

break;

case 5:System.out.println("enter roomstatus");

String roomstatus1=src.next();

employeeService.UpdateRoomstatus(roomids, roomstatus1);

break;

case 6:oo=false;

break;

}

}

break;

case 4:

System.out.println("display all rooms");

Utility.generateRoomsXml(employeeService.getrooms());

UtilityTest.roomXmlTest();

break;

case 5: System.out.println("all customers");

Utility.generateXml(employeeService.AllCustomers());

UtilityTest.customerxml();

break;

case 6: System.out.println("display all bookings");

Utility.generateBookingsXml(employeeService.AllBookings());

UtilityTest.bookingXmlTest();

break;

case 7:emp=false;break;

}

}

break;

case 2: boolean cus=true;

while(cus) {

System.out.println("Customer......");

System.out.println("enter 1: register\n enter 2: login"

+"\n enter 3:exit");

int c=src.nextInt();

switch(c) {

case 1:

while(true) {

System.out.println("enter your name");

String name=src.next();

if(Validation.nameValidation(name)) {

reg.setName(name);

break;

}

}

while(true) {

System.out.println("enter email id");

String em=src.next();

if(Validation.emailValidation(em)) {

reg.setEmailid(em);

break;

}

}

while(true) {

System.out.println("enter phone number");

String num=src.next();

if(Validation.numberValidation(num)) {

reg.setPhoneNumber(num);

break;

}

}

System.out.println("enter password");

reg.setPassword(src.next());

System.out.println("enter customer id");

reg.setCustomerid(src.nextInt());

try {

customerService.addCustomer(reg);

} catch (userExceptions e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

break;

case 2:System.out.println("enter email");

String email=src.next();

String email1=null;

if(Validation.emailValidation(email)) {

email1 = email;

}

System.out.println("enter password");

String pass=src.next();

System.out.println(customerService.Login(email, pass));

if(customerService.Login(email1, pass)==1) {

System.out.println("login successful");

boolean opn=true;

while(opn) {

System.out.println("enter 1:email update\n enter 2: book a room\n"

+ "\n enter 3: quit\n enter 4: exit");

int l=src.nextInt();

int id = 0;

switch(l) {

case 1: for(Register d:employeeService.AllCustomers()) {

if(d.getEmailid().matches(email))

id=d.getCustomerid();

}

System.out.println("enter new email");

String email2=src.next();

email=email2;

if(Validation.emailValidation(email1)) {

customerService.UpdateEmail(email1, id);

break;

}

break;

case 2: System.out.println("Book a room");

String roomtype = null;

Date dte;

while(true) {

System.out.println("enter roomtype");

System.out.println(" roomtype : AC ,NONAC, conference, banquet");

roomtype=src.next();

System.out.println("Enter the CheckIn date ");

String dt = src.next();

dte = Date.valueOf(dt);

try {

if(Validation.DateComparison(dte)) {

boo.setCheckin(dte);

break;

}

} catch (userExceptions e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

while(true) {

System.out.println("Enter the CheckOut date");

String dt1 = src.next();

Date dte1 = Date.valueOf(dt1);

try {

if(Validation.DateComparison(dte1)) {

boo.setCheckout(dte1);

break;

}

} catch (userExceptions e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

System.out.println(customerService.searchRoom(boo,roomtype));

System.out.println("Enter the room number you want to book");

int brid = src.nextInt();

roo.setRoomid(brid);

boo.setRoomid(brid);

int rp = (int)customerService.fetchPrice(boo);

boo.setTotalprice(rp);

java.util.Date endDate = new java.util.Date(boo.getCheckout().getTime());

java.util.Date startDate = new java.util.Date(boo.getCheckin().getTime());

long diff = endDate.getTime() - startDate.getTime();

long dateDiff = (diff/(100 \* 600 \* 60 \* 24)) % 365;

int price = (int) (dateDiff\*boo.getTotalprice());

boo.setTotalprice(price);

System.out.println("Enter your employee name to confirm booking");

String epid = src.next();

boo.setCustomername(epid);

boo.setRoomstatus("booked");

System.out.println(customerService.BookAroom(brid, boo,dte));

break;

case 3: System.out.println("quit");

System.out.println("enter bookid");

int bookid=src.nextInt();

System.out.println(customerService.quit(bookid));

break;

case 4: opn =false;

break;

}

}

}

else {

try {

throw new userExceptions("enter correct email and password");

} catch (userExceptions e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

break;

case 3: cus=false;

break;

}

}

break;

case 3: System.exit(1);

break;

}

}

}

}

connectionUtil

package com.cognizant.customer.util;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class ConnectionUtil {

private static String url="jdbc:mysql://localhost:3306/java";

private static String uname="root";

private static String pass="Nanna@123";

public static Connection getconnection() {

Connection con=null;

try {

con=DriverManager.getConnection(url,uname,pass);

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return con;

}

}

Customer Dao

package com.cognizant.customers.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import com.cognizant.customer.beans.Bookings;

import com.cognizant.customer.beans.Register;

import com.cognizant.customer.beans.room;

import com.cognizant.customer.util.ConnectionUtil;

import com.cognizant.customer.util.userExceptions;

public class customerDao {

Connection con=ConnectionUtil.getconnection();

public String addCustomer(Register c) throws userExceptions {

String sql="select count(\*) from customer where customerid=?";

PreparedStatement pstmt;

try {

pstmt=con.prepareStatement(sql);

pstmt.setInt(1,c.getCustomerid());

ResultSet rs=pstmt.executeQuery();

int count=0;

while(rs.next()) {

count=rs.getInt(1);

}

if(count==0) {

String sql1="insert into customer values(?,?,?,?,?)";

try {

PreparedStatement pstmt1=con.prepareStatement(sql1);

pstmt1.setInt(1, c.getCustomerid());

pstmt1.setString(2, c.getName());

pstmt1.setString(3, c.getEmailid());

pstmt1.setString(4, c.getPhoneNumber());

pstmt1.setString(5, c.getPassword());

System.out.println(pstmt1.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

else {

throw new userExceptions("please enter the different customerid");

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success";

}

public String UpdateEmail(String email,int customerid) {

String sql3="update customer set emailid = ? where customerid=?";

PreparedStatement pstmt1;

try {

pstmt1 = con.prepareStatement(sql3);

pstmt1.setString(1, email);

pstmt1.setInt(2, customerid);

System.out.println(pstmt1.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success";

}

public int Login(String email,String password) {

int count=0;

String sql="select count(\*) from customer where emailid=? and password1=?";

PreparedStatement pstmt;

try {

pstmt = con.prepareStatement(sql);

pstmt.setString(1,email);

pstmt.setString(2,password);

ResultSet rs=pstmt.executeQuery();

while( rs.next()) {

count=rs.getInt(1);

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return count;

}

// List<room> list1=new ArrayList<room>();

// public List<room> allrooms(Bookings b){

//

// String sql="select rooms.roomid,roomtype,price,view,roomstatus from rooms where rooms.roomid not in (select roomid from booking where checkout > ?)";

// PreparedStatement pstmt;

// try {

// pstmt = con.prepareStatement(sql);

// pstmt.setDate(1,b.getCheckin());

// ResultSet rs=pstmt.executeQuery();

//

//

// while(rs.next()) {

// room r=new room();

// r.setRoomid(rs.getInt(1));

// r.setRoomtype(rs.getString(2));

// r.setPrice(rs.getInt(3));

// r.setView(rs.getString(4));

// r.setRoomstatus(rs.getString(5));

// list1.add(r);

//

// }

//

// } catch (SQLException e) {

// // TODO Auto-generated catch block

// e.printStackTrace();

// }

// return list1;

//

// }

public List<room> searchRoom(Bookings b,String roomtype) {

List<room> rl = new ArrayList<room>();

String ssql = "select rooms.roomid,roomtype,price,view,roomstatus from rooms where roomtype=? and rooms.roomid not in (select roomid from booking where checkout > ? and checkin < ?) ";

try {

PreparedStatement sp = con.prepareStatement(ssql);

sp.setString(1, roomtype);

sp.setDate(2, b.getCheckin());

sp.setDate(3, b.getCheckout());

ResultSet rs = sp.executeQuery();

while (rs.next()) {

room r = new room();

r.setRoomid(rs.getInt(1));

r.setRoomtype(rs.getString(2));

r.setPrice(rs.getInt(3));

r.setView(rs.getString(4));

r.setRoomstatus(rs.getString(5));

rl.add(r);

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return rl;

}

public List<Bookings> BookAroom(int roomid,Date checkin,Bookings b) {

List<Bookings> list=new ArrayList<Bookings>();

String sql1="insert into booking(customername,roomid,checkin,checkout,totalprice,roomstatus) values(?,?,?,?,?,?)";

PreparedStatement pstmt;

try {

pstmt = con.prepareStatement(sql1);

pstmt.setString(1, b.getCustomername());

pstmt.setInt(2, b.getRoomid());

pstmt.setDate(3,b.getCheckin());

pstmt.setDate(4, b.getCheckout());

pstmt.setInt(5, b.getTotalprice());

pstmt.setString(6, b.getRoomstatus());

pstmt.executeUpdate();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

String sql3="select bookid,customername,roomid,checkin,checkout,totalprice,roomstatus from booking where roomid=? and checkin=?";

PreparedStatement pstmt2;

try {

pstmt2=con.prepareStatement(sql3);

pstmt2.setDate(2, (java.sql.Date) checkin);

pstmt2.setInt(1, roomid);

ResultSet rs=pstmt2.executeQuery();

while(rs.next()) {

b.setBookid(rs.getInt(1));

b.setCustomername(rs.getString(2));

b.setRoomid(rs.getInt(3));

b.setCheckin(rs.getDate(4));

b.setCheckout(rs.getDate(5));

b.setTotalprice(rs.getInt(6));

b.setRoomstatus(rs.getString(7));

list.add(b);

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return list;

}

public List<Bookings> quit(int bookid) {

String sql1="update booking set roomstatus='quit' where bookid=?";

PreparedStatement pstmt;

try {

pstmt=con.prepareStatement(sql1);

pstmt.setInt(1, bookid);

pstmt.executeUpdate();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

List<Bookings> list=new ArrayList<Bookings>();

String sql3="select bookid,customername,roomid,checkin,checkout,totalprice,roomstatus from booking where bookid=? ";

PreparedStatement pstmt2;

try {

pstmt2=con.prepareStatement(sql3);

pstmt2.setInt(1, bookid);

ResultSet rs=pstmt2.executeQuery();

while(rs.next()) {

Bookings b=new Bookings();

b.setBookid(rs.getInt(1));

b.setCustomername(rs.getString(2));

b.setRoomid(rs.getInt(3));

b.setCheckin(rs.getDate(4));

b.setCheckout(rs.getDate(5));

b.setTotalprice(rs.getInt(6));

b.setRoomstatus(rs.getString(7));

list.add(b);

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return list;

}

public double fetchPrice(Bookings br) {

double price = 0;

String ssql = "select price from rooms where roomid =? ";

try {

PreparedStatement sps = con.prepareStatement(ssql);

sps.setInt(1, (int) br.getRoomid());

ResultSet rs = sps.executeQuery();

rs.next();

price = rs.getDouble(1);

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

return price;

}

}

employee Dao

package com.cognizant.customers.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import com.cognizant.customer.beans.Bookings;

import com.cognizant.customer.beans.Register;

import com.cognizant.customer.beans.room;

import com.cognizant.customer.util.ConnectionUtil;

import com.cognizant.customer.util.userExceptions;

public class employeeDao {

Connection con=ConnectionUtil.getconnection();

public String AddRoom(room r) throws userExceptions {

String sql="select count(\*) from rooms where roomid=?";

PreparedStatement pstmt;

try {

pstmt=con.prepareStatement(sql);

pstmt.setInt(1,r.getRoomid());

ResultSet rs=pstmt.executeQuery();

int count=0;

while(rs.next()) {

count=rs.getInt(1);

}

if(count==0) {

String sql1="insert into rooms values(?,?,?,?,?)";

try {

PreparedStatement pstmt1=con.prepareStatement(sql1);

pstmt1.setInt(1, r.getRoomid());

pstmt1.setString(2, r.getRoomtype());

pstmt1.setInt(3, r.getPrice());

pstmt1.setString(4, r.getView());

pstmt1.setString(5, r.getRoomstatus());

System.out.println(pstmt1.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

else {

throw new userExceptions("please enter the different roomid");

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success" ;

}

public String DeleteRoom(int roomid) {

String sql1="select count(\*) from rooms where roomid=?";

PreparedStatement pstmt1;

try {

pstmt1=con.prepareStatement(sql1);

pstmt1.setInt(1,roomid);

ResultSet rs=pstmt1.executeQuery();

int count=0;

while(rs.next()) {

count=rs.getInt(1);

}

if(count==1) {

String sql="delete from rooms where roomid=?";

try {

PreparedStatement pstmt=con.prepareStatement(sql);

pstmt.setInt(1, roomid);

System.out.println(pstmt.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

else {

try {

throw new userExceptions("please enter correct roomid");

} catch (userExceptions e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success";

}

public String UpdateRoomid(int roomid,int newroomid) {

String sql1="select count(\*) from rooms where roomid=?";

PreparedStatement pstmt1;

try {

pstmt1=con.prepareStatement(sql1);

pstmt1.setInt(1,roomid);

ResultSet rs=pstmt1.executeQuery();

int count=0;

while(rs.next()) {

count=rs.getInt(1);

}

if(count==1) {

String sql="update rooms set roomid=? where roomid=?";

PreparedStatement pstmt;

try {

pstmt = con.prepareStatement(sql);

pstmt.setInt(1, newroomid);

pstmt.setInt(2, roomid);

System.out.println(pstmt.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

else {

try {

throw new userExceptions("please enter correct roomid");

} catch (userExceptions e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success";

}

public String UpdateRoomtype(int roomid, String roomtype) {

String sql3="update rooms set roomtype = ? where roomid=?";

PreparedStatement pstmt1;

try {

pstmt1 = con.prepareStatement(sql3);

pstmt1.setString(1, roomtype);

pstmt1.setInt(2, roomid);

System.out.println(pstmt1.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success";

}

public String Updateprice(int roomid, int price) {

String sql4="update rooms set price = ? where roomid=?";

PreparedStatement pstmt2;

try {

pstmt2 = con.prepareStatement(sql4);

pstmt2.setInt(1, price);

pstmt2.setInt(2, roomid);

System.out.println(pstmt2.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success";

}

public String Updateview(int roomid, String view) {

String sql5="update rooms set view = ? where roomid=?";

PreparedStatement pstmt3;

try {

pstmt3 = con.prepareStatement(sql5);

pstmt3.setString(1, view);

pstmt3.setInt(2, roomid);

System.out.println(pstmt3.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success";

}

public String UpdateRoomstatus(int roomid, String roomstatus) {

String sql6="update rooms set roomstatus = ? where roomid=?";

PreparedStatement pstmt4;

try {

pstmt4 = con.prepareStatement(sql6);

pstmt4.setString(1, roomstatus);

pstmt4.setInt(2, roomid);

System.out.println(pstmt4.executeUpdate());

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return "Success";

}

public List<room> getrooms() {

List<room> list=new ArrayList<room>();

String sql="select roomid,roomtype,price,view,roomstatus from rooms";

PreparedStatement pstmt;

try {

pstmt = con.prepareStatement(sql);

ResultSet rs=pstmt.executeQuery();

while(rs.next()) {

room r=new room();

r.setRoomid(rs.getInt(1));

r.setRoomtype(rs.getString(2));

r.setPrice(rs.getInt(3));

r.setView(rs.getString(4));

r.setRoomstatus(rs.getString(5));

list.add(r);

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return list;

}

public List<Register> AllCustomers() {

// TODO Auto-generated method stub

List<Register> list=new ArrayList<Register>();

String sql="select customerid,name1,emailid,phonenumber,password1 from customer";

PreparedStatement pstmt;

try {

pstmt = con.prepareStatement(sql);

ResultSet rs=pstmt.executeQuery();

while(rs.next()) {

Register c=new Register();

c.setCustomerid(rs.getInt(1));

c.setName(rs.getString(2));

c.setEmailid(rs.getString(3));

c.setPhoneNumber(rs.getString(4));

c.setPassword(rs.getString(5));

list.add(c);

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return list;

}

public List<Bookings> AllBookings() {

List<Bookings> list=new ArrayList<Bookings>();

String sql3="select bookid,customername,roomid,checkin,checkout,totalprice,roomstatus from booking";

PreparedStatement pstmt2;

try {

pstmt2=con.prepareStatement(sql3);

ResultSet rs=pstmt2.executeQuery();

while(rs.next()) {

Bookings b=new Bookings();

b.setBookid(rs.getInt(1));

b.setCustomername(rs.getString(2));

b.setRoomid(rs.getInt(3));

b.setCheckin(rs.getDate(4));

b.setCheckout(rs.getDate(5));

b.setTotalprice(rs.getInt(6));

b.setRoomstatus(rs.getString(7));

list.add(b);

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return list;

}

}

ROOM

package com.cognizant.customer.beans;

public class room {

private int roomid;

private String roomtype;

private int price;

private String view;

private String roomstatus;

@Override

public String toString() {

return "room [roomid=" + roomid + ", roomtype=" + roomtype + ", price=" + price + ", view=" + view

+ ", roomstatus=" + roomstatus + "]"+"\n";

}

public int getRoomid() {

return roomid;

}

public void setRoomid(int roomid) {

this.roomid = roomid;

}

public String getRoomtype() {

return roomtype;

}

public void setRoomtype(String roomtype) {

this.roomtype = roomtype;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getView() {

return view;

}

public void setView(String view) {

this.view = view;

}

public String getRoomstatus() {

return roomstatus;

}

public void setRoomstatus(String roomstatus) {

this.roomstatus = roomstatus;

}

}

Utility test

package com.cognizant.customer.util;

import java.io.File;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.DocumentBuilder;

import org.w3c.dom.Document;

import org.w3c.dom.NodeList;

import org.w3c.dom.Node;

import org.w3c.dom.Element;

public class UtilityTest {

public static final String xmlFilePath = "C:\\Users\\2050050\\OneDrive - Cognizant\\Desktop\\xml\\customer.xml";

public static final String xmlFilePath1 = "C:\\Users\\2050050\\OneDrive - Cognizant\\Desktop\\xml\\room.xml";

public static final String xmlFilePath2 = "C:\\Users\\2050050\\OneDrive - Cognizant\\Desktop\\xml\\booking.xml";

public static void customerxml() {

try {

File inputFile = new File(xmlFilePath);

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

Document doc = dBuilder.parse(inputFile);

doc.getDocumentElement().normalize();

System.out.println("Root element :" + doc.getDocumentElement().getNodeName());

NodeList nList = doc.getElementsByTagName("customer");

System.out.println("----------------------------");

for (int temp = 0; temp < nList.getLength(); temp++) {

Node nNode = nList.item(temp);

System.out.println("\nCurrent Element :" + nNode.getNodeName());

if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

Element eElement = (Element) nNode;

System.out.println("customerid : "

+ eElement.getAttribute("customerid"));

System.out.println(" customer name: "

+ eElement

.getElementsByTagName("name")

.item(0)

.getTextContent());

System.out.println("email : "

+ eElement

.getElementsByTagName("email")

.item(0)

.getTextContent());

System.out.println("phone number: "

+ eElement

.getElementsByTagName("phonenumber")

.item(0)

.getTextContent());

System.out.println("password "

+ eElement

.getElementsByTagName("password")

.item(0)

.getTextContent());

}

}

} catch (Exception e) {

e.printStackTrace();

}

}

public static void roomXmlTest() {

try {

File inputFile = new File(xmlFilePath1);

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

Document doc = dBuilder.parse(inputFile);

doc.getDocumentElement().normalize();

System.out.println("Root element :" + doc.getDocumentElement().getNodeName());

NodeList nList = doc.getElementsByTagName("room");

System.out.println("----------------------------");

for (int temp = 0; temp < nList.getLength(); temp++) {

Node nNode = nList.item(temp);

System.out.println("\nCurrent Element :" + nNode.getNodeName());

if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

Element eElement = (Element) nNode;

System.out.println("roomid : "

+ eElement.getAttribute("roomid"));

System.out.println(" room type: "

+ eElement

.getElementsByTagName("roomtype")

.item(0)

.getTextContent());

System.out.println("price: "

+ eElement

.getElementsByTagName("price")

.item(0)

.getTextContent());

System.out.println("view: "

+ eElement

.getElementsByTagName("view")

.item(0)

.getTextContent());

}

}

} catch (Exception e) {

e.printStackTrace();

}

}

public static void bookingXmlTest() {

try {

File inputFile = new File(xmlFilePath2);

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

Document doc = dBuilder.parse(inputFile);

doc.getDocumentElement().normalize();

System.out.println("Root element :" + doc.getDocumentElement().getNodeName());

NodeList nList = doc.getElementsByTagName("booking");

System.out.println("----------------------------");

for (int temp = 0; temp < nList.getLength(); temp++) {

Node nNode = nList.item(temp);

System.out.println("\nCurrent Element :" + nNode.getNodeName());

if (nNode.getNodeType() == Node.ELEMENT\_NODE) {

Element eElement = (Element) nNode;

System.out.println("bookid : "

+ eElement.getAttribute("bookid"));

System.out.println(" customer name: "

+ eElement

.getElementsByTagName("customer\_name")

.item(0)

.getTextContent());

System.out.println("room id : "

+ eElement

.getElementsByTagName("room\_id")

.item(0)

.getTextContent());

System.out.println("checkin: "

+ eElement

.getElementsByTagName("checkin")

.item(0)

.getTextContent());

System.out.println("checkout :"

+ eElement

.getElementsByTagName("checkout")

.item(0)

.getTextContent());

System.out.println("room status: "

+ eElement

.getElementsByTagName("room\_status")

.item(0)

.getTextContent());

}

}

} catch (Exception e) {

e.printStackTrace();

}

}

}

I employee

package com.cognizant.customers.services;

import java.util.List;

import com.cognizant.customer.beans.Bookings;

import com.cognizant.customer.beans.Register;

import com.cognizant.customer.beans.room;

import com.cognizant.customer.util.userExceptions;

public interface IEmployee {

public String AddRoom(room r)throws userExceptions ;

public String DeleteRoom(int roomid);

public String UpdateRoomid(int roomid,int newroomid);

public String UpdateRoomtype(int roomid,String roomtype);

public String Updateprice(int roomid,int price);

public String Updateview(int roomid,String view);

public String UpdateRoomstatus(int roomid,String roomstatus);

public List<room> getrooms();

// public void CheckStatus();

public List<Bookings> AllBookings();

public List<Register> AllCustomers();

}

I customer

package com.cognizant.customers.services;

import java.sql.Date;

import java.util.List;

import com.cognizant.customer.beans.Bookings;

import com.cognizant.customer.beans.Register;

import com.cognizant.customer.beans.room;

import com.cognizant.customer.util.userExceptions;

public interface ICustomer {

public String addCustomer(Register c) throws userExceptions;

public int Login(String email,String password);

public String UpdateEmail(String email,int customerid);

public List<Bookings> BookAroom(int roomid,Bookings b,Date checkin);

public List<room> searchRoom(Bookings b,String roomtype);

public List<Bookings> quit(int bookid);

public double fetchPrice(Bookings br) ;

}

Bookings

package com.cognizant.customer.beans;

import java.sql.Date;

public class Bookings {

private int bookid;

private String customername ;

private int roomid ;

private Date checkin ;

private Date checkout ;

private int totalprice ;

private String roomstatus ;

@Override

public String toString() {

return "Bookings [bookid=" + bookid + ", customername=" + customername + ", roomid=" + roomid + ", checkin="

+ checkin + ", checkout=" + checkout + ", totalprice=" + totalprice + ", roomstatus=" + roomstatus

+ "]";

}

public int getRoomid() {

return roomid;

}

public void setRoomid(int roomid) {

this.roomid = roomid;

}

public Date getCheckin() {

return checkin;

}

public void setCheckin(Date startdate) {

this.checkin = startdate;

}

public Date getCheckout() {

return checkout;

}

public void setCheckout(Date checkout) {

this.checkout = checkout;

}

public String getRoomstatus() {

return roomstatus;

}

public void setRoomstatus(String roomstatus) {

this.roomstatus = roomstatus;

}

public String getCustomername() {

return customername;

}

public void setCustomername(String customername) {

this.customername = customername;

}

public int getTotalprice() {

return totalprice;

}

public void setTotalprice(int totalprice) {

this.totalprice = totalprice;

}

public int getBookid() {

return bookid;

}

public void setBookid(int bookid) {

this.bookid = bookid;

}

}