****

**Application Code**

**HOSTEL MANANGENT SYSTEM**

**SUBMITTED TO: MA’AM SANEEHA AMIR**

**SUBMITTED BY:ZAMIN RAZA**

**FA21-BCS-096**

(OBJECT ORIENTED PROGRAMIING)

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.ItemEvent;

import java.awt.event.ItemListener;

import java.io.Serializable;

import java.util.ArrayList;

public class Addwindow extends JFrame implements Serializable {

JButton Submit, Exit, Back;

Integer[] roomarr = {1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,35};

Integer[] roomarr2 = {0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34};

JLabel Title, name, address, mob, bg, id, emg, uni, roomno,dateofadm;

JComboBox days, month, year, froomno;

Integer[] dayarr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 21, 23, 24, 25, 26, 27, 28, 29, 30, 31};

Integer[] montharr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};

JTextField fname, fadress, fmob, fbg, fid, femg, funi;

JCheckBox s1, s2;

JPanel p1, p2, p3,pdate;

Addwindow() {

setSize(750, 600);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

setLocationRelativeTo(null);

// setResizable(false);

// setLayout(new GridLayout(2,2));

setLayout(new BorderLayout());

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

p1.setLayout(new FlowLayout());

p2.setLayout(new GridLayout(10, 2));

p3.setLayout(new FlowLayout());

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

pdate = new JPanel(new GridLayout(1,2));

p3.setBackground(new Color(255, 158, 0));

p2.setSize(30, 40);

// for (int i =0 ; i< dayarr.length;i++){

// dayarr[i]+=1;

// }

// for (int i = 0 ; i > montharr.length ; i++){

// montharr[i]+=1;

// }

days = new JComboBox(dayarr);

month = new JComboBox(montharr);

Title = new JLabel("CAPITAL VIEW BOYS HOSTEL");

p1.add(Title);

name = new JLabel("Enter Name");

name.setSize(15, 20);

address = new JLabel("Enter Adress ");

fname = new JTextField(20);

fadress = new JTextField(20);

mob = new JLabel("Enter Mobile Number");

fmob = new JTextField(20);

bg = new JLabel("Enter blood group");

fbg = new JTextField(20);

id = new JLabel("ID");

fid = new JTextField(20);

emg = new JLabel("Emergency contact");

femg = new JTextField(20);

uni = new JLabel("Institute");

dateofadm = new JLabel("DATE OF ADMISSION");

ButtonGroup g1 = new ButtonGroup();

s1 = new JCheckBox("2-Seater", true);

s2 = new JCheckBox("3-Seater", false);

g1.add(s1);

g1.add(s2);

p2.setBackground(new Color(70, 81, 129));

ArrayList<JLabel> lab = new ArrayList<JLabel>();

lab.add(name);

lab.add(address);

lab.add(bg);

lab.add(mob);

lab.add(uni);

lab.add(emg);

lab.add(id);

lab.add(dateofadm);

days.setBackground(new Color(0, 0, 0));

month.setBackground(new Color(0, 0, 0));

days.setForeground(new Color(255, 165, 0));

month.setForeground(new Color(255, 165, 0));

p1.setBackground(new Color(27, 117, 34));

Title.setFont(new Font("ARIAL",1,22));

Title.setForeground(new Color(255, 255, 255, 255));

Font f2 = new Font("TIMES NEW ROMAN",0,18);

fname.setFont(f2);

fid.setFont(f2);

fbg.setFont(f2);

days.setFont(f2);

month.setFont(f2);

// froomno.setBackground(new Color(58,67,90));

// froomno.setForeground(new Color(255, 180, 0));

// froomno.setFont(f2);

s1.setBackground(new Color(58,67,98));

s2.setBackground(new Color(58,67,98));

s1.setForeground(new Color(255, 165, 0));

s2.setForeground(new Color(255, 165, 0));

roomno = new JLabel("Room no ");

lab.add(roomno);

Font f = new Font("Tahoma",1,16);

for (JLabel l : lab){

l.setFont(f);

l.setForeground(new Color(255, 167, 0));

}

// funi = new JLabel("Enter email");

funi = new JTextField(20);

fadress.setFont(f2);

fmob.setFont(f2);

femg.setFont(f2);

funi.setFont(f2);

pdate.add(days);

pdate.add(month);

// day.addItem(dayarr);

// froomno = new JComboBox<>();\

s1.setForeground(new Color(255, 163, 0));

s1.setBackground(new Color(59, 65, 98));

s2.setForeground(new Color(255, 163, 0));

s2.setBackground(new Color(59, 65, 98));

s1.setFont(f2);

s2.setFont(f2);

froomno = new JComboBox<>(roomarr2);

froomno.setEnabled(false);

// MyitemListener ms = new MyitemListener();

// froomno.addItemListener(ms);

// froomno.setEnabled(false);

MyActionlistener2 s = new MyActionlistener2();

s1.addActionListener(s);

s2.addActionListener(s);

// days = new JComboBox<>(dayarr);

// days.setMaximumRowCount(10);

froomno.setBackground(new Color(0, 0, 0));

froomno.setFont(f2);

froomno.setForeground(new Color(255, 183, 0));

// temail = new JTextField(20);

p2.add(name);

p2.add(fname);

p2.add(id);

p2.add(fid);

p2.add(address);

p2.add(fadress);

p2.add(mob);

p2.add(fmob);

p2.add(uni);

p2.add(funi);

p2.add(emg);

p2.add(femg);

p2.add(bg);

p2.add(fbg);

p2.add(s1);

p2.add(s2);

p2.add(roomno);

p2.add(froomno);

p2.add(dateofadm);

p2.add(pdate);

// p2.add(l4);

// p2.add(t3);

// p2.add(l5);

// p2.add(t4);

// p2.add(l6);

// p2.add(t5);

// p2.add(l7);

// p2.add(t6);

MyActionlistener as = new MyActionlistener();

Submit = new JButton("Submit");

// Submit.setEnabled(false);

// b1.addActionListener(as);

Exit = new JButton("Exit");

p3.add(Submit);

Back = new JButton("Back");

Submit.setBackground(new Color(0, 0, 0));

Submit.setForeground(new Color(255,255,255));

Submit.setFont(f2);

Back.setBackground(new Color(0, 0, 0));

Back.setForeground(new Color(255,255,255));

Back.setFont(f2);

Exit.setBackground(new Color(0, 0, 0));

Exit.setForeground(new Color(255,255,255));

Exit.setFont(f2);

Submit.addActionListener(as);

Exit.addActionListener(as);

Back.addActionListener(as);

p3.add(Exit);

p3.add(Back);

}

public class MyActionlistener implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Submit")) {

if (fname.getText().isEmpty() || fadress.getText().isEmpty() || fid.getText().isEmpty() || fmob.getText().isEmpty()

|| funi.getText().isEmpty()) {

JOptionPane.showMessageDialog(new JFrame(), "FIll the FIELDS");

} else if ((FileOperations.searchidalready(fid.getText()))) {

JOptionPane.showMessageDialog(new JFrame(), "ID ALREADY EXIST");

// System.exit(0);

}

else if(!(FileOperations.searchidalready(fid.getText()))){

//

////

ArrayList<Room> rooms = new ArrayList<Room>(36);

rooms = FileOperations.readAllRoomsfiles();

Room roomadded = new Room();

System.out.println(froomno.getSelectedIndex());

int j = froomno.getSelectedIndex();

int i =-1;

if (s1.isSelected()){

i = roomarr2[j];

} else if (s2.isSelected()) {

i= roomarr[j];

}

for (Room r : rooms) {

if (r.getNumber() == i) {

if (!(r.isfull())) {

System.out.println("1st time r ka " + r.getCurr());

FileOperations.updatecurrent(i);

System.out.println("2nd tme r ka" + r.getCurr());

roomadded = r;

roomadded.added();

System.out.println("First time room added ka" + roomadded.getCurr());

System.out.println("3r time r ka" + roomadded.getCurr());

if (roomadded != null) {

hostelite h = new hostelite(fname.getText(), fadress.getText(), fmob.getText(),

fbg.getText(), fid.getText(), femg.getText(), funi.getText(), new Date((days.getSelectedIndex() + 1), (month.getSelectedIndex() + 1), 2023), roomadded);

System.out.println(h.toString());

FileOperations.writeToFile(h);

new displaywindow(h);

dispose();

// FileOperations.writeToFile(h);

System.out.println("written successfully");

System.out.println(h.room.curr);

// FileOperations.readFile();

break;

}

}

else {

JOptionPane.showMessageDialog(new JFrame(), "NO SPACE AVAILABLE");

}

}

}

}

}

else if (e.getActionCommand().equalsIgnoreCase("Exit")) {

System.out.println("Exit successfully");

// System.out.println(days.getSelectedIndex());

// System.out.println(month.getSelectedIndex());

System.exit(1);

} else if (e.getActionCommand().equalsIgnoreCase("Back")) {

new MainWindow();

dispose();

System.out.println("back successfully");

}

}

}

public class MyitemListener implements ItemListener {

// static int i;

@Override

public void itemStateChanged(ItemEvent e) {

if (e.getStateChange() == ItemEvent.SELECTED) {

int i = roomarr[froomno.getSelectedIndex()] + 1;

}

}

// public static int selectedroomno(){

// return i;

// }

}

public class MyActionlistener2 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (s1.isSelected()) {

for (int i = 0 ; i<roomarr2.length ; i++){

//// froomno.removeItem(roomarr[i]);

froomno.addItem(roomarr2[i]);

//

}

for (int j =0 ; j<roomarr.length ; j++){

froomno.removeItem(roomarr[j]);

}

// froomno.addItem(roomarr);

froomno.setEnabled(true);

System.out.println("2-seater room hai bhai");

} else if (s2.isSelected()) {

for (int i = 0 ; i<roomarr.length ; i++){

// froomno.removeItem(roomarr2[i]);

froomno.addItem(roomarr[i]);

}

for (int j =0 ; j<roomarr2.length ; j++){

froomno.removeItem(roomarr2[j]);

}

// froomno.addItem(roomarr2);

System.out.println("3-Seater hai bhai");

froomno.setEnabled(true);

}

}

}

}

// p3.add(b3);

// b4 = new JButton("Exit");

// b5 = new JButton("CLICK");

// MyActionlistener ae = new MyActionlistener();

// b1.addActionListener(ae);

// b2.addActionListener(ae);

// add(l1,BorderLayout.NORTH);

// add(t,BorderLayout.CENTER);

// add(l2,BorderLayout.EAST);

// add(t2,BorderLayout.CENTER);

// add(b1,BorderLayout.PAGE\_START);

// add(b2,BorderLayout.CENTER);

// add(b3,BorderLayout.WEST);

// add(b4,BorderLayout.NORTH);

// add(b5,BorderLayout.EAST);

// b1.setBackground(Color.CYAN);

// getContentPane().setBackground(Color.blue);

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.Serializable;

import java.util.ArrayList;

public class alreadyGUI extends JFrame implements Serializable {

JLabel id, mob, emg, bg, uni, adress;

JTextField fid, funi, fbg, femg, fmob, fadress;

JButton Submit, Exit, Back;

JPanel p1, p2, p3;

alreadyGUI() {

setSize(700, 400);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

setTitle("HMS");

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

setLayout(new BorderLayout());

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

p2.setLayout(new GridLayout(6, 2));

id = new JLabel("ENETER ID");

mob = new JLabel("ENTER MOBILE NUMBER");

uni = new JLabel("ENETR INSTITUTE");

emg = new JLabel("EMERGENCY CONATCT");

bg = new JLabel("BLOOD GROUP");

adress = new JLabel("ADDRESS");

ArrayList<JLabel> labels = new ArrayList<>();

labels.add(id);

labels.add(adress);

labels.add(uni);

labels.add(mob);

// labels.add(adress);

labels.add(bg);

labels.add(emg);

// labels.add(uni);

ArrayList<JTextField> fields = new ArrayList<>();

fid = new JTextField(20);

fbg = new JTextField(20);

femg = new JTextField(20);

funi = new JTextField(20);

fmob = new JTextField(20);

// fbg = new JTextFi/eld(20);

fadress = new JTextField(20);

fields.add(fid);

fields.add(fadress);

fields.add(funi);

fields.add(fmob);

fields.add(fbg);

fields.add(femg);

Submit = new JButton("Submit");

Exit = new JButton("Exit");

Back = new JButton("Back");

//

// p2.add(id);

// p2.add(fid);

// p2.add(adress);

// p2.add(fadress);

// p2.add(uni);

// p2.add(funi);

// p2.add(mob);

// p2.add(fmob);

//

for (int i = 0; i < labels.size(); i++) {

p2.add(labels.get(i));

p2.add(fields.get(i));

}

p3.add(Submit);

p3.add(Back);

p3.add(Exit);

MyActionlistener10 mov = new MyActionlistener10();

Submit.addActionListener(mov);

Back.addActionListener(mov);

Exit.addActionListener(mov);

}

public class MyActionlistener10 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Submit")) {

if (fid.getText().isEmpty() || fadress.getText().isEmpty()

|| fmob.getText().isEmpty() || funi.getText().isEmpty() || femg.getText().isEmpty()) {

JOptionPane.showMessageDialog(new JFrame(), " FILL ALL THE FIELDS");

} else {

hostelite s = FileOperations.searchreservedbyID(fid.getText());

if (s!=null) {

s.setAddress(fadress.getText());

s.setMobile(fmob.getText());

s.setEmergencyno(femg.getText());

s.setInstitue(funi.getText());

s.setBlood\_group(bg.getText());

System.out.println("write krny wala");

System.out.println(s.getId());

FileOperations.writeToFile(s);

FileOperations.readFile();

JOptionPane.showMessageDialog(new JFrame() , "ADDED SUCCESSFULLY");

}

}

} else if (e.getActionCommand().equalsIgnoreCase("Exit")) {

System.exit(0);

} else if (e.getActionCommand().equalsIgnoreCase("Back")) {

new optionalAddGUI();

dispose();

}

}

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.Serial;

import java.io.Serializable;

public class Billwindow extends JFrame implements Serializable {

JLabel byid,payment , fine , concession;

JTextField fid,fpayment , ffine , fconcession;

JButton submit, exit, back,paid;

JPanel p1, p2, p3;

Billwindow() {

setSize(700, 400);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

setLocationRelativeTo(null);

// setResizable(false);

// setLayout(new GridLayout(2,2));

setLayout(new BorderLayout());

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

p1.setLayout(new FlowLayout());

p2.setLayout(null);

p3.setLayout(new FlowLayout());

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

p2.setSize(30, 40);

p2.setBackground(new Color(24, 145, 222));

byid = new JLabel("Enter ID");

payment = new JLabel("Enter Payment recieved");

Font f1 = new Font("TIMES NEW ROMAN",1,18);

fid = new JTextField(20);

fpayment = new JTextField(20);

fine = new JLabel("FINE");

ffine = new JTextField(20);

concession = new JLabel("CONCESSION");

fconcession = new JTextField(20);

p2.add(byid);

p2.add(fid);

p2.add(payment);

p2.add(fpayment);

p2.add(fine);

p2.add(ffine);

p2.add(concession);

p2.add(fconcession);

byid.setBounds(35,50,190,70);

fid.setBounds(300,50,230,40);

payment.setBounds(35,120,190,70);

fpayment.setBounds(300,120,230,40);

fine.setBounds(35,200,180,70);

ffine.setBounds(300,200,230,40);

concession.setBounds(35,270,170,70);

fconcession.setBounds(300,270,230,40);

byid.setForeground(new Color(255,255,255));

payment.setForeground(new Color(255,255,255));

concession.setForeground(new Color(255,255,255));

fine.setForeground(new Color(255,255,255));

// fpayment.setForeground(new Color(255,255,255));

// fid.setForeground(new Color(255,255,255));

byid.setFont(f1);

fid.setFont(f1);

payment.setFont(f1);

fpayment.setFont(f1);

ffine.setFont(f1);

fconcession.setFont(f1);

concession.setFont(f1);

fconcession.setFont(f1);

fine.setFont(f1);

submit = new JButton("Submit");

exit = new JButton("Exit");

back = new JButton("Back");

paid = new JButton("Paid");

MyActionlistener3 as = new MyActionlistener3();

submit.addActionListener(as);

exit.addActionListener(as);

back.addActionListener(as);

paid.addActionListener(as);

p3.add(submit);

p3.add(exit);

p3.add(back);

p3.add(paid);

}

public class MyActionlistener3 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Back")) {

new MainWindow();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("Submit")) {

if (!(fid.getText().isEmpty())) {

double x;

//

if (!(ffine.getText().isEmpty())){

FileOperations.updatefine(fid.getText(),ffine.getText());

// System.out.println("fine para gya");

System.out.println(FileOperations.searchbyID(fid.getText()).getBill().getFine());

System.out.println(FileOperations.searchbyID(fid.getText()).totalrent());

}

if (!(fconcession.getText().isEmpty())){

FileOperations.updateconcession(fid.getText(),fconcession.getText());

}

hostelite h = FileOperations.searchbyID(fid.getText());

if (!(h==null)) {

x = h.totalrent();

System.out.println("TOTAL RENT IS "+x);

int dues =0;

if (h.getRoom().getSeater() == 2){

dues+=2000;

}

JOptionPane.showMessageDialog(new Frame(), "NAME: "+h.getName()+"\nRoom NUMBER: "+h.getRoom().getNumber()+

"\nFINE "+ h.getBill().getFine() +"\n CONCESSION " + h.getBill().getConcession()+ "\nTOTAL RENT " + x);

} else {

JOptionPane.showMessageDialog(new Frame(), "RECORD NOT FOUNDED");

}

}

} else if (e.getActionCommand().equalsIgnoreCase("Exit")) {

System.exit(0);

}

else if(e.getActionCommand().equalsIgnoreCase("Paid")) {

if (fid.getText().isEmpty() || fpayment.getText().isEmpty()) {

JOptionPane.showMessageDialog(new JFrame(), "FILL THE FIELDS");

} else {

FileOperations.updatebill(fid.getText() , Integer.parseInt(fpayment.getText()));

JOptionPane.showMessageDialog(new JFrame() ,"Paid successfully");

}

}

}

}

}

import java.io.Serializable;

public class Date implements Serializable {

int month;

int date;

int year;

Date(int a, int b, int c ){

this.month = checkmonth(b);

this.year = c;

this.date = a;

}

public int checkmonth(int testmonth){

if(testmonth> 0 && testmonth <=12){

return testmonth;

}

else {

System.out.println("Invalid month");

return 1;

}

}

public int checkday(int testday) {

int[] days = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

if (testday <= days[this.month]) {

return testday;

}

if (month == 2 && testday == 29 && (year % 4 == 0 || year % 4 == 0 || year % 100 != 0)) {

return testday;

} else {

System.out.println("invalid date");

return 1;

}

}

public int checkyear(int testyear){

if(testyear >= 2022){

return testyear;

}

else{

System.out.println("invalid year");

return 1;

}

}

@Override

public String toString() {

return (month +"-"+date+"-"+year);

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class Deletewindow extends JFrame{

JLabel byid;

JTextField fid;

JButton submit, exit, back,paid;

JPanel p1, p2, p3;

Deletewindow(){

setSize(700, 400);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

setLocationRelativeTo(null);

// setResizable(false);

// setLayout(new GridLayout(2,2));

setLayout(new BorderLayout());

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

p1.setLayout(new FlowLayout());

p2.setLayout(null);

p3.setLayout(new FlowLayout());

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

p2.setSize(30, 40);

p2.setBackground(new Color(112, 117, 192));

Font f1 = new Font("Times New Roman",1,18);

p1.setBackground(new Color(118, 246, 109));

byid = new JLabel("Enter ID");

fid = new JTextField(20);

byid.setFont(f1);

fid.setFont(f1);

byid.setBounds(160,100,430,60);

fid.setBounds(160,150,400,60);

p2.add(byid);

p2.add(fid);

MyActionlistener4 del = new MyActionlistener4();

submit = new JButton("Submit");

exit = new JButton("Exit");

back = new JButton("Back");

p3.add(submit);

p3.add(exit);

p3.add(back);

submit.addActionListener(del);

exit.addActionListener(del);

back.addActionListener(del);

}

public class MyActionlistener4 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Submit")){

if (!(fid.getText()==null)) {

hostelite hos = FileOperations.searchbyID(fid.getText());

FileOperations.delete(fid.getText());

FileOperations.delcurrent(hos.getRoom().number);

JOptionPane.showMessageDialog(new JFrame(), "DELETED SUCCESSFULLY");

}

else {

JOptionPane.showMessageDialog(new JFrame(),"FILL THE FIELDS");

}

} else if (e.getActionCommand().equalsIgnoreCase("Exit")) {

System.exit(1);

} else if (e.getActionCommand().equalsIgnoreCase("Back")) {

new MainWindow();

}

}

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.Serial;

import java.io.Serializable;

import java.util.ArrayList;

public class displaywindow extends JFrame implements Serializable {

JLabel[] labels = new JLabel[12];

JLabel[] labels2 = new JLabel[12];

JLabel admdate,fdate;

JLabel title;

JPanel p1,p2,p3;

JButton Back,Exit;

displaywindow(hostelite x) {

try {

setSize(700, 400);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

// setResizable(false);

// setLayout(new GridLayout(2,2));

setLayout(new BorderLayout());

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

title = new JLabel("CAPITAL VIEW BOYS HOSTEL");

title.setFont(new Font("ARIAL",1,20));

p1.add(title);

p1.setLayout(new FlowLayout());

p2.setLayout(new GridLayout(8, 2));

p3.setLayout(new FlowLayout());

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

p2.setSize(30, 40);

p2.setBackground(new Color(255, 111, 111));

labels2[1] = new JLabel(x.getName());

labels2[2] = new JLabel(x.getAddress());

labels2[3] = new JLabel(x.getId());

labels2[4] = new JLabel(x.getMobile());

labels2[5] = new JLabel(Integer.toString(x.getRoom().getNumber()));

labels2[6] = new JLabel(Integer.toString(x.getRoom().getSeater()) + " Seater");

labels2[7] = new JLabel(x.getInstitue());

labels[0] = new JLabel(" ISLAMABAD");

labels[1] = new JLabel("NAME OF HOSTELITE");

labels[2] = new JLabel("ADDRESS OF HOSTELITE");

labels[3] = new JLabel("ID OF HOSTELITE");

labels[4] = new JLabel("MOBILE NUMBER OF HOSTELITE");

labels[5] = new JLabel("ROOM NUMBER OF HOSTELITE");

labels[6] = new JLabel("ROOM TYPE OF HOSTELITE");

labels[7] = new JLabel("INSTITUTE OF HOSTELITE");

admdate = new JLabel("DATE OF ADMSSION");

fdate = new JLabel(x.getDateofadmission().date + " - " + x.getDateofadmission().month+" - "+" 2023");

Back = new JButton("Back");

Exit = new JButton("Exit");

MyActionlistener6 dis = new MyActionlistener6();

Back.addActionListener(dis);

Exit.addActionListener(dis);

p1.add(labels[0]);

for (int i = 1; i < labels.length; i++) {

if (labels[i] != null && labels2[i] != null) {

p2.add(labels[i]);

p2.add(labels2[i]);

labels[i].setFont(new Font("ARIAL",1,16));

labels2[i].setFont(new Font("ARIAL",1,16));

}

}

p2.add(admdate);

p2.add(fdate);

admdate.setFont(new Font("ARIAL",1,16));

fdate.setFont(new Font("ARIAL",1,16));

p3.add(Back);

p3.add(Exit);

} catch (NullPointerException e) {

JOptionPane.showMessageDialog(new Frame(), "Record not founded");

}

}

public class MyActionlistener6 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Back")) {

new Addwindow();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("Exit")) {

System.exit(1);

}

}

}

}

import java.io.EOFException;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

import java.util.Scanner;

public class FileOperations {

public static void writeroom(Object o){

try {

File f = new File("ALLROOMS.txt");

ObjectOutputStream oos;

if (f.exists()) {

oos = new MyobjectoutputStream(new FileOutputStream(f, true));

} else {

oos = new ObjectOutputStream(new FileOutputStream(f));

}

oos.writeObject(o);

oos.close();

} catch (Exception e) {

System.out.println("Error: ");

e.printStackTrace();

}

}

public static void readRoomFile() {

try {

ObjectInputStream is = new ObjectInputStream(new FileInputStream("ALLROOMS.txt"));

while (true) {

Room s = (Room) is.readObject();

System.out.println(s.toString());

}

} catch (ClassNotFoundException | FileNotFoundException c) {

System.out.println("Exception Found: Class not found || File not found! ");

c.printStackTrace();

} catch (EOFException e) {

System.out.println("EOF!");

} catch (IOException e) {

// e.printStackTrace();

}

}

public static void writeToFile(Object o) {

//

if (o.getClass().getName().equalsIgnoreCase("Room")) {

writeroom(o);

} else {

try {

File f = new File("ALLSTUDENTS.txt");

ObjectOutputStream oos;

if (f.exists()) {

oos = new MyobjectoutputStream(new FileOutputStream(f, true));

} else {

oos = new ObjectOutputStream(new FileOutputStream(f));

}

oos.writeObject(o);

oos.close();

} catch (Exception e) {

System.out.println("Error: ");

e.printStackTrace();

}

}

}

public static void writeToreservedFile(Object o) {

try {

File f = new File("reserved3.txt");

ObjectOutputStream oos;

if (f.exists()) {

oos = new MyobjectoutputStream(new FileOutputStream(f, true));

} else {

oos = new ObjectOutputStream(new FileOutputStream(f));

}

oos.writeObject(o);

oos.close();

} catch (Exception e) {

System.out.println("Error: ");

e.printStackTrace();

}

}

public static ArrayList<hostelite> readAllreservedfiles(){

ArrayList<hostelite> arr = new ArrayList<hostelite>();

try {

ObjectInputStream is = new ObjectInputStream(new FileInputStream("reserved3.txt"));

while (true) {

hostelite s = (hostelite) is.readObject();

arr.add(s);

}

} catch (ClassNotFoundException | FileNotFoundException c) {

System.out.println("Exception Found: Class not found || File not found! ");

c.printStackTrace();

} catch (EOFException e) {

System.out.println("EOF!");

} catch (IOException e) {

// e.printStackTrace();

}

return arr;

}

public static void readFile() {

try {

ObjectInputStream is = new ObjectInputStream(new FileInputStream("ALLSTUDENTS.txt"));

while (true) {

hostelite s = (hostelite) is.readObject();

System.out.println(s.toString());

}

} catch (ClassNotFoundException | FileNotFoundException c) {

System.out.println("Exception Found: Class not found || File not found! ");

c.printStackTrace();

} catch (EOFException e) {

System.out.println("EOF!");

} catch (IOException e) {

// e.printStackTrace();

}

}

public static void readreservedFile() {

try {

ObjectInputStream is = new ObjectInputStream(new FileInputStream("reserved3.txt"));

while (true) {

hostelite s = (hostelite) is.readObject();

System.out.println(s.toString());

}

} catch (ClassNotFoundException | FileNotFoundException c) {

System.out.println("Exception Found: Class not found || File not found! ");

c.printStackTrace();

} catch (EOFException e) {

System.out.println("EOF!");

} catch (IOException e) {

// e.printStackTrace();

}

}

public static ArrayList<Room> readAllRoomsfiles(){

ArrayList<Room> arr = new ArrayList<Room>();

try {

ObjectInputStream is = new ObjectInputStream(new FileInputStream("ALLROOMS.txt"));

while (true) {

Room s = (Room) is.readObject();

arr.add(s);

}

} catch (ClassNotFoundException | FileNotFoundException c) {

System.out.println("Exception Found: Class not found || File not found! ");

c.printStackTrace();

} catch (EOFException e) {

System.out.println("EOF!");

} catch (IOException e) {

// e.printStackTrace();

}

return arr;

}

//

public static ArrayList<hostelite> readAllfiles(){

ArrayList<hostelite> arr = new ArrayList<hostelite>();

try {

ObjectInputStream is = new ObjectInputStream(new FileInputStream("ALLSTUDENTS.txt"));

while (true) {

hostelite s = (hostelite) is.readObject();

arr.add(s);

}

} catch (ClassNotFoundException | FileNotFoundException c) {

System.out.println("Exception Found: Class not found || File not found! ");

c.printStackTrace();

} catch (EOFException e) {

System.out.println("EOF!");

} catch (IOException e) {

// e.printStackTrace();

}

return arr;

}

public static Room specificroom(int check){

ArrayList<Room> ro = new ArrayList<>();

ro = FileOperations.readAllRoomsfiles();

for (Room kamry: ro){

if (kamry.getNumber()==check){

return kamry;

}

}

return null;

}

public static hostelite searchreservedbyID(String check){

ArrayList<hostelite> d = readAllreservedfiles();

for(hostelite i : d){

if(i.getId().equalsIgnoreCase(check)){

System.out.println("mill gya");

return i;

}

}

return null;

}

public static hostelite searchbyname(String naam){

ArrayList<hostelite> p = readAllfiles();

// p = readAllfiles();

for(hostelite i : p){

if(i.getName().equalsIgnoreCase(naam)){

System.out.println("desired data");

return i;

}

else {

System.out.println("NOT FOUNDED");

}

}

return null;

}

public static boolean searchidalready(String id){

ArrayList<hostelite> p = readAllfiles();

boolean f = false;

// p = readAllfiles();

for(hostelite i : p){

if(i.getId().equalsIgnoreCase(id)){

System.out.println("ALREADY EXIST");

return true;

}

else {

System.out.println("NOT FOUNDED");

}

}

return f;

}

public static void delete(String id) {

ArrayList<hostelite> r = readAllfiles();

for (hostelite i : r) {

if (i.getId().equalsIgnoreCase(id)) {

System.out.println("founded");

r.remove(i);

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : r) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static hostelite searchbyroomno(int no){

ArrayList<hostelite> r = readAllfiles();

for(int i = 0 ; i<r.size() ; i++){

if(r.get(i).room.getNumber() == no){

return (r.get(i));

}

else{

System.out.println("Not founded");

}

}

return null;

}

public static hostelite searchbyID(String id){

ArrayList<hostelite> d = readAllfiles();

for(hostelite i : d){

if(i.getId().equalsIgnoreCase(id)){

return i;

}

}

return null;

}

public static void searchbymob(String m){

ArrayList<hostelite> d = readAllfiles();

for(hostelite i : d){

if(i.getId().equalsIgnoreCase(m)){

System.out.println(i.toString());

}

}

}

public static void updateroom(int check,int number){

ArrayList<hostelite> n = readAllfiles();

for(hostelite g : n){

if(g.getRoom().getNumber() == number){

g.setRoom(new Room(1,2,number,'B'));

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static void updatecurrent(int check){

ArrayList<Room> n = readAllRoomsfiles();

for(Room g : n){

if(g.getNumber() == check){

System.out.println("crucial" + g.getCurr());

g.setCurr(1+g.getCurr());

System.out.println("ye new curr" + g.getCurr());

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLROOMS.txt"))) {

for (Room g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static void delcurrent(int check){

ArrayList<Room> n = readAllRoomsfiles();

for(Room g : n){

if(g.getNumber() == check){

System.out.println("crucial" + g.getCurr());

g.setCurr(g.getCurr() - 1);

System.out.println("ye new curr" + g.getCurr());

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLROOMS.txt"))) {

for (Room g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static void updatefine(String check,String number){

ArrayList<hostelite> n = readAllfiles();

for(hostelite g : n){

if(g.getId().equalsIgnoreCase(check)){

g.getBill().setFine(Integer.parseInt(number));

System.out.println(g.getBill().getDues());

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static void updateconcession(String check,String number){

ArrayList<hostelite> n = readAllfiles();

for(hostelite g : n){

if(g.getId().equalsIgnoreCase(check)){

g.getBill().setConcession(Integer.parseInt(number));

// System.out.println(g.getBill().getDues());

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static void updatemob(String check,String number){

ArrayList<hostelite> n = readAllfiles();

for(hostelite g : n){

if(g.getId().equalsIgnoreCase(check)){

g.setMobile(number);

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static void updatename(String check,String name){

ArrayList<hostelite> n = readAllfiles();

for(hostelite g : n){

if(g.getId().equalsIgnoreCase(check)){

g.setName(name);

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static void updateadress(String check,String adress){

ArrayList<hostelite> n = readAllfiles();

for(hostelite g : n){

if(g.getId().equalsIgnoreCase(check)){

g.setAddress(adress);

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static void updatebill(String check, int a){

ArrayList<hostelite> n = readAllfiles();

for(hostelite g : n){

if(g.getId().equalsIgnoreCase(check)){

g.getBill().setFine(0);

g.getBill().setDues((int) (g.totalrent() - a));

System.out.println(g.getBill().getDues());

g.getBill().setRegestered(true);

if(g.totalrent() == a){

g.getBill().setPaid(true);

}

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

public static ArrayList<hostelite> allsameuni(String inst){

ArrayList<hostelite> s = new ArrayList<hostelite>();

ArrayList<hostelite> all = new ArrayList<hostelite>();

s=readAllfiles();

for (hostelite k : s){

if (k.getInstitue().equalsIgnoreCase(inst)){

all.add(k);

}

}

return all;

}

public static ArrayList<hostelite> allsameroom(String inst){

ArrayList<hostelite> s = new ArrayList<hostelite>();

ArrayList<hostelite> all = new ArrayList<hostelite>();

s=readAllfiles();

for (hostelite k : s){

if (k.getInstitue().equalsIgnoreCase(inst)){

all.add(k);

}

}

return all;

}

public static void searchreadFile() {

try {

ObjectInputStream is = new ObjectInputStream(new FileInputStream("test3.txt"));

Scanner input = new Scanner(System.in);

System.out.println("PRESS 1-SEARCH BY NAME\n 2-SEARCH BY ROOM NUMBER \n 3-SEARCH BY ID\n 4-SEARCH BY MOB \n");

int val = input.nextInt();

if(val == 1){

System.out.println("enter name you want to search");

String name = input.next();

searchbyname(name);

}

else if(val == 2) {

int num = input.nextInt();

searchbyroomno(num);

}

else if(val == 3) {

String i = input.next();

searchbyID(i);

}

else if(val == 4) {

String mob = input.next();

searchbymob(mob);

}

else{

System.out.println("invalid");

}

}

catch (FileNotFoundException c) {

System.out.println("Exception Found: Class not found || File not found! ");

c.printStackTrace();

} catch (EOFException e) {

System.out.println("EOF!");

} catch (IOException e) {

// e.printStackTrace();

}

}

public static void updatebill(hostelite h,double s){

ArrayList<hostelite> n = readAllfiles();

for(hostelite g : n){

if(g.getId().equalsIgnoreCase(h.getId())){

g.paybill(s);

break;

}

}

try {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("ALLSTUDENTS.txt"))) {

for (hostelite g : n) {

oos.writeObject(g);

}

}

}

catch(IOException e){

System.out.println("ERROR");

}

}

}

public class hostelite extends Person{

private String id;

private String emergencyno;

private String institue;

Date dateofadmission;

Room room;

Rent bill;

public hostelite(String name, String addr, String mob, String bg,String id , String eme , String inst , Date

addmission, Room room) {

super(name, addr, mob, bg);

this.dateofadmission = addmission;

this.institue = inst;

this.emergencyno = eme;

this.room = room;

this.id = id;

this.bill = new Rent(0,0,false,false);

}

public void setId(String id) {

this.id = id;

}

public void setRoom(Room room) {

this.room = room;

}

public Room getRoom() {

return room;

}

@Override

public void setAddress(String address) {

super.setAddress(address);

}

public Date getDateofadmission() {

return dateofadmission;

}

public String getEmergencyno() {

return emergencyno;

}

public void setDateofadmission(Date dateofadmission) {

this.dateofadmission = dateofadmission;

}

public String getInstitue() {

return institue;

}

public String getId() {

return id;

}

public void setInstitue(String institue) {

this.institue = institue;

}

public void setEmergencyno(String emergencyno) {

this.emergencyno = emergencyno;

}

public double totalrent () {

int roomrent;

if (this.getBill().getpaid()){

return 0;

}

if (this.room.seater == 2) {

roomrent = this.getBill().getDues() + 2000;

} else {

roomrent = this.getBill().getDues();

}

if (!(this.bill.isRegestered())) {

return (this.bill.getFine() + 4000 + (roomrent - this.bill.concession));

} else {

return (this.bill.getFine() + (roomrent - this.bill.concession));

}

}

public void paybill ( double x){

double y = this.totalrent() - x;

this.bill.setDues((int) y);

this.bill.setFine(0);

this.bill.setPaid(true);

this.bill.setRegestered(true);

}

public Rent getBill() {

return bill;

}

@Override

public String toString() {

return (super.toString() + this.emergencyno + this.institue + "your ID " +this.id

+ this.room.toString() + this.dateofadmission.toString());

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.image.ImageObserver;

import java.awt.image.ImageProducer;

public class Login extends JFrame {

JLabel title, id, password;

JTextField fid;

JPasswordField fpasswrod;

JButton login, Exit;

JLabel full;

Container c;

// ImageIcon img = new ImageIcon(getClass().getResource("CUI.jpg"));

ImageIcon background = new ImageIcon("pic1.png");

Image img = background.getImage();

Login() {

setSize(500, 500);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

setLayout(null);

setTitle("CBH");

c = this.getContentPane();

title = new JLabel("CAPITAL VIEW BOYS HOSTEL");

id = new JLabel("ADMIN ID");

password = new JLabel("PASSWORD");

fid = new JTextField();

fpasswrod = new JPasswordField();

full = new JLabel(background);

full.setSize(500,500);

Font f = new Font("AGENCY FB", 1, 22);

Font f2 = new Font("ARIAL",1,16);

title.setFont(f);

id.setFont(f);

// fid.setFont(f);

fid.setFont(f2);

password.setFont(f);

// fpasswrod.setFont(f);

login = new JButton("LOGIN");

Exit = new JButton("EXIT");

c.setBackground(new Color(238, 181, 59));

title.setBounds(90, 10, 300, 40);

id.setBounds(35, 100, 120, 30);

fid.setBounds(135, 100, 200, 30);

password.setBounds(35, 150, 120, 30);

fpasswrod.setBounds(135, 150, 200, 30);

login.setBounds(57, 220, 95, 30);

Exit.setBounds(190, 220, 95, 30);

title.setBackground(new Color(225, 0, 0));

id.setBackground(new Color(0, 0, 0));

fid.setBackground(new Color(255, 255, 255));

password.setBackground(new Color(255, 0, 0));

fpasswrod.setBackground(new Color(255, 255, 255));

MyActionlistener0 acc = new MyActionlistener0();

c.add(title);

c.add(id);

c.add(password);

c.add(fid);

c.add(fpasswrod);

c.add(login);

c.add(Exit);

login.addActionListener(acc);

Exit.addActionListener(acc);

c.add(full);

setVisible(true);

}

public class MyActionlistener0 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("LOGIN")) {

if (fid.getText().equalsIgnoreCase("ZAMIN") && fpasswrod.getText().equalsIgnoreCase("12345")) {

new MainWindow();

dispose();

}

else{

JOptionPane.showMessageDialog(new JFrame() , "INVALID ID OR PASSWORD");

}

} else if (e.getActionCommand().equalsIgnoreCase("EXIT")) {

System.exit(1);

}

}

}

}

import java.util.ArrayList;

public class Main {

public static void main(String[] args) {

//---------------------------ADDING ROOMS TO ROOMFILE-------------------------------------

////

// Room[] rooms = new Room[36];

// for (int i =0 ; i < rooms.length ; i++){

// rooms[i] = new Room(i);

// if (rooms[i].number == 0 || rooms[i].number % 9 ==0){

// rooms[i].setType('O');

// }

// rooms[i].setsFloor();

// rooms[i].setsSeater();

// }

// for (Room r : rooms){

// FileOperations.writeroom(r);

// }

// ArrayList<hostelite> r = new ArrayList<hostelite>(36);

//FileOperations.readFile();

//

//ArrayList<hostelite> p = FileOperations.readAllfiles();

//for (hostelite a : p){

// System.out.println(a.getId());

//}

//

Login a = new Login();

//

}

}

import javax.imageio.ImageIO;

import javax.swing.\*;

import javax.swing.border.Border;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.image.BufferedImage;

public class MainWindow extends JFrame {

JButton addbtn,delbtn,searchbtn,billbtn,updatebtn,reserved,exit;

// ImageIcon img;

JLabel pic;

JLabel title,dev;

JTextField t,t2;

JPanel p1,p2,p3;

// ImageIcon img;

MainWindow() {

setSize(900, 700);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

// setResizable(false);

// setResizable(false);

// setLayout(new GridLayout(2,2));

setLayout(new BorderLayout());

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

p1.setLayout(new FlowLayout());

p2.setLayout(null);

p3.setLayout(new FlowLayout());

pic = new JLabel();

// Image im = Toolkit.getDefaultToolkit().createImage("pic1.png");

// ImageIcon img = new ImageIcon("pic1.png");

// pic.setIcon(img);

// pic.setBounds(620,30,300,200);

//

// p2.add(pic);

p2.setBackground(new Color(234, 185, 71));

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

title =new JLabel("CAPITAL VIEW BOYS HOSTEL");

addbtn = new JButton("ADD STUDENT RECORD");

delbtn = new JButton("DELETE STUDENT RECORD");

searchbtn = new JButton("SEARCH STUDENT RECORD");

billbtn = new JButton("BILL OF STUDENT");

updatebtn = new JButton("UPDATE STUDENT");

reserved = new JButton("RESERVE A ROOM");

exit = new JButton("EXIT");

addbtn.setBackground(new Color(25, 14, 70));

delbtn.setBackground(new Color(25,14,70));

searchbtn.setBackground(new Color(25, 14, 70));

billbtn.setBackground(new Color(25,14,70));

updatebtn.setBackground(new Color(25, 14, 70));

reserved.setBackground(new Color(25,14,70));

dev = new JLabel("Developed by Zamin Raza");

final Font FONT = new Font("Times New Roman",Font.BOLD,30);

title.setFont(FONT);

addbtn.setForeground(new Color(255, 255, 255));

addbtn.setFont(FONT);

delbtn.setForeground(new Color(255, 255, 255,232));

delbtn.setFont(FONT);

billbtn.setForeground(new Color(255, 255, 255));

billbtn.setFont(FONT);

searchbtn.setForeground(new Color(255, 255, 255,232));

searchbtn.setFont(FONT);

updatebtn.setFont(FONT);

updatebtn.setForeground(new Color(255,255,255));

reserved.setFont(FONT);

reserved.setForeground(new Color(255,255,255));

p2.setSize(500,330);

addbtn.setBounds(200,35,450,60);

delbtn.setBounds(200,115,450,60);

searchbtn.setBounds(200,195,450,60);

billbtn.setBounds(200,275,450,60);

updatebtn.setBounds(200,355,450,60);

reserved.setBounds(200,435,450,60);

exit.setBounds(360,540,100,40);

p1.add(title);

p2.add(addbtn);

p2.add(searchbtn);

p2.add(delbtn);

p2.add(billbtn);

p2.add(updatebtn);

p2.add(reserved);

p2.add(exit);

exit.setFont(new Font("TIMES NEW ROMAN",1,15));

exit.setForeground(new Color(255,255,255));

exit.setBackground(new Color(255, 0, 0));

p3.add(dev);

p2.add(reserved);

title.setForeground(new Color(255, 255, 255));

p1.setBackground(new Color(51, 150, 56));

p3.setBackground(new Color(0, 250, 250));

dev.setFont(new Font("ARIAL",Font.ITALIC , 16));

MyActionlistener ae = new MyActionlistener();

addbtn.addActionListener(ae);

delbtn.addActionListener(ae);

searchbtn.addActionListener(ae);

billbtn.addActionListener(ae);

updatebtn.addActionListener(ae);

reserved.addActionListener(ae);

exit.addActionListener(ae);

}

public class MyActionlistener implements ActionListener {

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("ADD STUDENT RECORD")) {

// new Addwindow();

new optionalAddGUI();

dispose();

}

else if (e.getActionCommand().equalsIgnoreCase("DELETE STUDENT RECORD")) {

new Deletewindow();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("SEARCH STUDENT RECORD")) {

// new Searchstdwindow();

new searchoptional();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("UPDATE STUDENT")) {

new Updatewindow();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("BILL OF STUDENT")) {

new Billwindow();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("RESERVE A ROOM"))

{

new ReservedGUI();

dispose();

}

else if (e.getActionCommand().equalsIgnoreCase("EXIT"));{

dispose();

}

}

}

}

import java.io.File;

import java.io.IOException;

import java.io.ObjectOutputStream;

import java.io.OutputStream;

public class MyobjectoutputStream extends ObjectOutputStream {

public MyobjectoutputStream(OutputStream o) throws IOException{

super(o);

}

protected MyobjectoutputStream() throws IOException, SecurityException {

super();

}

public MyobjectoutputStream(File f) throws IOException {

super();

}

@Override

public void writeStreamHeader(){

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

public class newDisplaytable extends JFrame {

// JTable t;

JTextArea t1;

JScrollPane scroll;

JPanel p1;

JButton back, exit;

Container c;

int[] arr = new int[2];

newDisplaytable(ArrayList<hostelite> x) {

c = this.getContentPane();

setSize(750, 600);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

back = new JButton("BACK");

exit = new JButton("EXIT");

c.setLayout(null);

// t=new JTable(2,5);

// p1 = new JPanel();

// add(p1);

t1 = new JTextArea(50, 50);

scroll = new JScrollPane(t1, JScrollPane.VERTICAL\_SCROLLBAR\_ALWAYS, JScrollPane.HORIZONTAL\_SCROLLBAR\_ALWAYS);

scroll.setBounds(60, 90, 500, 500);

back.setBounds(70, 500, 70, 30);

exit.setBounds(190, 500, 70, 30);

back.setBackground(new Color(0, 0, 0));

exit.setBackground(new Color(0, 0, 0));

back.setForeground(new Color(255, 255, 255));

exit.setForeground(new Color(255, 255, 255));

c.add(back);

c.add(exit);

// c.add(t1);

c.add(scroll);

t1.setText("STUDENT DATA");

t1.setFont(new Font("TIMES NEW ROMAN", 1, 12));

t1.setEditable(false);

for (hostelite check :x) {

String str = "\nNAME : " + check.getName() + "\nADDRESS " + check.getAddress() + "\n MOBILE: " + check.getMobile() +

"\nROOM NUMBER " + check.room.getNumber();

t1.setText(t1.getText() + "\n" + str);

}

MyActionlistenerdis fg = new MyActionlistenerdis();

back.addActionListener(fg);

exit.addActionListener(fg);

}

public class MyActionlistenerdis implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Back")) {

new Searchstdwindow();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("Exit")) {

System.exit(1);

}

}

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.Serializable;

public class optionalAddGUI extends JFrame implements Serializable {

JButton reservedadd, addnew , EXIT,BACK;

Container c;

optionalAddGUI() {

setSize(500, 500);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

c = this.getContentPane();

setVisible(true);

setLayout(new FlowLayout());

reservedadd = new JButton("ALREADY RESERVED");

addnew = new JButton("NEW HOSTELITE");

Font fontbt = new Font("TIMES NEW ROMAN",1,17);

EXIT = new JButton("EXIT");

BACK = new JButton("BACK");

reservedadd.setBounds(85,60,220,120);

addnew.setBounds(85,260,220,120);

EXIT.setBounds(85,400,100,70);

BACK.setBounds(220,400,100,70);

c.setBackground(new Color(47, 76, 107));

reservedadd.setBackground(new Color(255, 184, 0));

addnew.setBackground(new Color(255, 183, 0));

EXIT.setBackground(new Color(0, 0, 0));

BACK.setBackground(new Color(0, 0, 0));

addnew.setFont(fontbt);

reservedadd.setFont(fontbt);

BACK.setFont(fontbt);

EXIT.setFont(fontbt);

setLayout(null);

MyActionlistener9 opt = new MyActionlistener9();

c.add(reservedadd);

c.add(addnew);

c.add(EXIT);

c.add(BACK);

reservedadd.addActionListener(opt);

addnew.addActionListener(opt);

EXIT.addActionListener(opt);

BACK.addActionListener(opt);

}

public class MyActionlistener9 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("ALREADY RESERVED")) {

new alreadyGUI();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("NEW HOSTELITE")) {

new Addwindow();

dispose();

}

else if (e.getActionCommand().equalsIgnoreCase("EXIT")) {

System.exit(1);

}

else if (e.getActionCommand().equalsIgnoreCase("BACK")) {

new MainWindow();

dispose();

}

}

}

}

//

import java.io.Serializable;

public class Person implements Serializable {

String name;

String address;

String mobile;

String blood\_group;

public Person(String a , String b , String c , String d){

name = a;

address = b;

mobile = c;

blood\_group = d;

}

public String getName() {

return name;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public void setName(String name) {

this.name = name;

}

public String getBlood\_group() {

return blood\_group;

}

public String getMobile() {

return mobile;

}

public void setBlood\_group(String blood\_group) {

this.blood\_group = blood\_group;

}

public void setMobile(String mobile) {

this.mobile = mobile;

}

@Override

public String toString() {

return "Person{" +

"name='" + name + '\'' +

", address='" + address + '\'' +

", mobile='" + mobile + '\'' +

", blood\_group='" + blood\_group + '\'' +

'}';

}

}

import java.io.Serializable;

public class Rent implements Serializable {

boolean paid;

int fine;

int dues;

int concession;

boolean regestered;

Rent(int fine,int conc,boolean reg, boolean paid){

this.concession=conc;

this.dues = 13000;

this.fine =fine;

this.regestered =reg;

this.paid = paid;

}

public boolean isPaid() {

return paid;

}

public boolean isRegestered() {

return regestered;

}

public int getDues() {

return dues;

}

public boolean getpaid(){

return this.paid;

}

public int getFine() {

return fine;

}

public int getConcession() {

return concession;

}

public void setDues(int dues) {

this.dues = dues;

}

public void setFine(int fine) {

this.fine = fine;

}

public void setConcession(int concession) {

this.concession = concession;

}

public void setPaid(boolean paid) {

this.paid = paid;

}

public void setRegestered(boolean regestered) {

this.regestered = regestered;

}

@Override

public String toString() {

return "Rent{" +"\n"+

"paid= " + paid +"\n"+

"fine=" + fine +"\n"+

"dues=" + dues +"\n"+

"concession=" + concession +"\n"+

"regestered=" + regestered +"\n"+

'}';

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.ItemEvent;

import java.awt.event.ItemListener;

import java.io.Serializable;

import java.util.ArrayList;

public class ReservedGUI extends JFrame implements Serializable {

JButton Submit, Exit, Back;

Integer[] roomarr = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35};

Integer[] roomarr2 = {0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34};

JLabel Title, name, mob, id, roomno;

JComboBox days, month, year, froomno;

JTextField fname, fmob, fid;

JCheckBox s1, s2;

JPanel p1, p2, p3;

ReservedGUI() {

setSize(700, 800);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

// setResizable(false);

// setLayout(new GridLayout(2,2));

setLayout(new BorderLayout());

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

p1.setLayout(new FlowLayout());

p2.setLayout(new GridLayout(6, 2));

p3.setLayout(new FlowLayout());

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

p2.setSize(30, 40);

Title = new JLabel("CAPITAL VIEW BOYS HOSTEL");

p1.add(Title);

name = new JLabel("Enter Name");

name.setSize(15, 20);

// address = new JLabel("Enter Adress ");

fname = new JTextField(20);

// fadress = new JTextField(20);

mob = new JLabel("Enter Mobile Number");

fmob = new JTextField(20);

// bg = new JLabel("Enter blood group");

// fbg = new JTextField(20);

id = new JLabel("ID");

fid = new JTextField(20);

// emg = new JLabel("Emergency contact");

// femg = new JTextField(20);

// uni = new JLabel("Institute");

ButtonGroup g1 = new ButtonGroup();

s1 = new JCheckBox("2-Seater", true);

s2 = new JCheckBox("3-Seater", false);

g1.add(s1);

g1.add(s2);

p2.setBackground(new Color(56, 55, 55));

ArrayList<JLabel> lab = new ArrayList<JLabel>();

lab.add(name);

// lab.add(address);

// lab.add(bg);

lab.add(mob);

// lab.add(uni);

// lab.add(emg);

lab.add(id);

p1.setBackground(new Color(255, 222, 0));

Title.setFont(new Font("ARIAL", 1, 22));

Title.setForeground(new Color(255, 0, 0, 200));

Font f2 = new Font("TIMES NEW ROMAN", 0, 18);

fname.setFont(f2);

fid.setFont(f2);

// fbg.setFont(f2);

Font f = new Font("Tahoma", 1, 16);

for (JLabel l : lab) {

l.setFont(f);

l.setForeground(new Color(255, 255, 255));

}

// funi = new JLabel("Enter email");

// funi = new JTextField(20);

roomno = new JLabel("Room no ");

// fadress.setFont(f2);

fmob.setFont(f2);

// femg.setFont(f2);

// funi.setFont(f2);

// day.addItem(dayarr);

// froomno = new JComboBox<>();\

froomno = new JComboBox<>(roomarr2);

froomno.setEnabled(false);

// MyitemListener ms = new MyitemListener();

// froomno.addItemListener(ms);

// froomno.setEnabled(false);

MyActionlistener7 s = new MyActionlistener7();

s1.addActionListener(s);

s2.addActionListener(s);

// days = new JComboBox<>(dayarr);

// days.setMaximumRowCount(10);

// temail = new JTextField(20);

p2.add(name);

p2.add(fname);

p2.add(id);

p2.add(fid);

// p2.add(address);

// p2.add(fadress);

p2.add(mob);

p2.add(fmob);

// p2.add(uni);

// p2.add(funi);

// p2.add(emg);

// p2.add(femg);

// p2.add(bg);

// p2.add(fbg);

p2.add(s1);

p2.add(s2);

p2.add(roomno);

p2.add(froomno);

// p2.add(l4);

// p2.add(t3);

// p2.add(l5);

// p2.add(t4);

// p2.add(l6);

// p2.add(t5);

// p2.add(l7);

// p2.add(t6);

MyActionlistener8 as = new MyActionlistener8();

Submit = new JButton("Submit");

// Submit.setEnabled(false);

// b1.addActionListener(as);

Exit = new JButton("Exit");

p3.add(Submit);

Back = new JButton("Back");

Submit.addActionListener(as);

Exit.addActionListener(as);

Back.addActionListener(as);

p3.add(Exit);

p3.add(Back);

}

public class MyActionlistener8 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Submit")) {

if (fname.getText().isEmpty() || fid.getText().isEmpty() || fmob.getText().isEmpty()

) {

JOptionPane.showMessageDialog(new JFrame(), "FIll the FIELDS");

} else {

//

ArrayList<Room> rooms = new ArrayList<Room>(36);

rooms = FileOperations.readAllRoomsfiles();

Room roomadded = new Room();

System.out.println(froomno.getSelectedIndex());

int j = froomno.getSelectedIndex();

int i = -1;

if (s1.isSelected()) {

i = roomarr2[j];

} else if (s2.isSelected()) {

i = roomarr[j];

}

for (Room r : rooms) {

if (r.getNumber() == i) {

if (!(r.isfull())) {

System.out.println("1st time r ka " + r.getCurr());

FileOperations.updatecurrent(i);

System.out.println("2nd tme r ka" + r.getCurr());

roomadded = r;

roomadded.added();

System.out.println("First time room added ka" + roomadded.getCurr());

System.out.println("3r time r ka" + roomadded.getCurr());

if (roomadded != null) {

hostelite h = new hostelite(fname.getText(), null, fmob.getText(), null, fid.getText(), null, null, new Date(02, 2, 2023), roomadded);

System.out.println(h.toString());

FileOperations.writeToreservedFile(h);

new displaywindow(h);

// FileOperations.writeToFile(h);

System.out.println("written successfully");

System.out.println(h.room.curr);

// FileOperations.readFile();

}

break;

} else {

JOptionPane.showMessageDialog(new JFrame(), "NO SPACE AVAILABLE");

}

}

}

}

}

else if (e.getActionCommand().equalsIgnoreCase("Exit")) {

System.out.println("Exit successfully");

System.exit(1);

} else if (e.getActionCommand().equalsIgnoreCase("Back")) {

dispose();

new MainWindow();

System.out.println("back successfully");

}

}

}

public class MyitemListener implements ItemListener {

// static int i;

@Override

public void itemStateChanged(ItemEvent e) {

if (e.getStateChange() == ItemEvent.SELECTED) {

int i = roomarr[froomno.getSelectedIndex()] + 1;

}

}

// public static int selectedroomno(){

// return i;

// }

}

public class MyActionlistener7 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (s1.isSelected()) {

for (int i = 0; i < roomarr2.length; i++) {

//// froomno.removeItem(roomarr[i]);

froomno.addItem(roomarr2[i]);

//

}

for (int j = 0; j < roomarr.length; j++) {

froomno.removeItem(roomarr[j]);

}

// froomno.addItem(roomarr);

froomno.setEnabled(true);

System.out.println("2-seater room hai bhai");

} else if (s2.isSelected()) {

for (int i = 0; i < roomarr.length; i++) {

// froomno.removeItem(roomarr2[i]);

froomno.addItem(roomarr[i]);

}

for (int j = 0; j < roomarr2.length; j++) {

froomno.removeItem(roomarr2[j]);

}

// froomno.addItem(roomarr2);

System.out.println("3-Seater hai bhai");

froomno.setEnabled(true);

}

}

}

}

import java.io.Serializable;

public class Room implements Serializable {

int floor;

int number;

int seater;

boolean reserved;

int curr;

char type;

public int getFloor() {

return floor;

}

public Room(int a, int b,int c, char d){

this.floor=a;

this.number = b;

this.seater = c;

this.type = d;

}

public Room(int num){

this.number = num;

}

public Room(){

this.floor = 0;

this.number = 0;

this.seater =0;

this.type='a';

}

public boolean isfull(){

return (this.curr==this.seater?true:false);

}

public void added(){

curr++;

}

public void setType(char type) {

this.type = type;

}

public int getNumber() {

return number;

}

public char getType() {

return type;

}

public void setNumber(int number) {

this.number = number;

}

public int getSeater() {

return seater;

}

public void setSeater(int seater) {

this.seater = seater;

}

public void setsFloor() {

if (this.number>= 0 && this.number<9){

this.floor = 0;

}

else if( this.number>=9 && this.number <=17){

this.floor = 1;

}

else if( this.number>=18 && this.number <=26){

this.floor = 2;

}

else if( this.number>=27 && this.number <=35){

this.floor = 3;

}

}

public void setsSeater(){

if(this.type != 'O' && this.number % 2 == 0){

this.seater = 2;

}

else if(this.type != 'O' && this.number % 2 != 0){

this.seater = 3;

}

}

public int getCurr() {

return curr;

}

public void setCurr(int curr) {

this.curr = curr;

}

public void setFloor(int floor) {

this.floor = floor;

}

@Override

public String toString() {

return "Room{" +"\n"+

"floor=" + floor +"\n"+

"number=" + number +"\n"+

"seater=" + seater +"\n"+

"type=" + type +"\n"+

"current=" + curr+

'}';

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

public class Roomdetail extends JFrame {

Container c;

JLabel floor,number,curr,type;

JButton back;

JLabel ffloor,fnumber,fcurr,ftype;

Roomdetail(Room r){

setSize(500,500);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setTitle("ROOM DETAILS");

setLayout(null);

c = this.getContentPane();

c.setLayout(null);

floor = new JLabel("FLOOR : ");

number = new JLabel("ROOM NUMBER : ");

curr = new JLabel("CURRENT STUDENTS :");

type = new JLabel("CAPACITY : ");

back=new JButton("Back");

back.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Back")){

new searchoptional();

dispose();

}

}

});

ffloor = new JLabel(Integer.toString(r.getFloor()));

fnumber = new JLabel(Integer.toString(r.getNumber()));

fcurr = new JLabel(Integer.toString(r.getCurr()));

ftype = new JLabel(Integer.toString(r.getSeater()));

floor.setBounds(30,60,180,30);

ffloor.setBounds(200,60,180,30);

number.setBounds(30,120,180,30);

fnumber.setBounds(200,120,180,30);

curr.setBounds(30,180,180,30);

fcurr.setBounds(200,180,180,30);

type.setBounds(30,240,180,30);

ftype.setBounds(200,240,180,30);

back.setBounds(200,320,120,30);

back.setForeground(new Color(233,2,2));

back.setBackground(new Color(255,255,255));

Font f = new Font("AGENCY FB",1,20);

ArrayList<JLabel> l = new ArrayList<>(8);

l.add(floor);

l.add(ffloor);

l.add(number);

l.add(fnumber);

l.add(curr);

l.add(fcurr);

l.add(type);

l.add(ftype);

// c.add(fnumber);

for (JLabel p : l){

c.add(p);

p.setFont(f);

}

c.add(back);

setVisible(true);

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class searchoptional extends JFrame {

JButton reservedadd, addnew, EXIT, BACK;

JLabel roomno;

JTextField t;

Container c;

searchoptional() {

setSize(650, 600);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

c = this.getContentPane();

setVisible(true);

setLayout(new FlowLayout());

reservedadd = new JButton("ROOM DETAIL");

addnew = new JButton("SEARCH STUDENTS");

roomno = new JLabel("ROOM NO ");

t = new JTextField();

Font fontbt = new Font("TIMES NEW ROMAN", 1, 17);

EXIT = new JButton("EXIT");

BACK = new JButton("BACK");

reservedadd.setBounds(20, 60, 200, 40);

addnew.setBounds(245, 60, 200, 40);

roomno.setBounds(100,140,140,30);

t.setBounds(100, 200, 200, 30);

EXIT.setBounds(70, 400, 100, 40);

BACK.setBounds(250, 400, 100, 40);

c.setBackground(new Color(47, 76, 107));

reservedadd.setBackground(new Color(255, 184, 0));

addnew.setBackground(new Color(255, 183, 0));

EXIT.setBackground(new Color(0, 0, 0));

BACK.setBackground(new Color(0, 0, 0));

EXIT.setForeground(new Color(255,255,255));

BACK.setForeground(new Color(255,255,255));

roomno.setForeground(new Color(255,255,255));

addnew.setFont(fontbt);

reservedadd.setFont(fontbt);

roomno.setFont(fontbt);

BACK.setFont(fontbt);

EXIT.setFont(fontbt);

setLayout(null);

MyActionlisteneropt opt = new MyActionlisteneropt();

c.add(reservedadd);

c.add(addnew);

c.add(roomno);

c.add(t);

c.add(EXIT);

c.add(BACK);

reservedadd.addActionListener(opt);

addnew.addActionListener(opt);

EXIT.addActionListener(opt);

BACK.addActionListener(opt);

}

public class MyActionlisteneropt implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("ROOM DETAIL")) {

if (!(t.getText().isEmpty())){

Room h = FileOperations.specificroom(Integer.parseInt(t.getText()));

new Roomdetail(h);

dispose();

}

else {

JOptionPane.showMessageDialog(new JFrame() , "ENTER ROOM NUMBER");

}

} else if (e.getActionCommand().equalsIgnoreCase("SEARCH STUDENTS")) {

new Searchstdwindow();

dispose();

} else if (e.getActionCommand().equalsIgnoreCase("EXIT")) {

System.exit(1);

} else if (e.getActionCommand().equalsIgnoreCase("BACK")) {

new MainWindow();

dispose();

}

}

}

}

import com.sun.jdi.ArrayReference;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.Serializable;

import java.lang.reflect.Array;

import java.util.ArrayList;

public class Searchstdwindow extends JFrame implements Serializable {

JCheckBox byname , byuni,byid,byroomno,byunpaid,bypaid;

JLabel name,uni,id,roomno,unpaid,paid;

JTextField fname,fid,froom,funi,funpaid,fpaid;

JButton Submit,Exit,Back;

JPanel p1,p2,p3;

Searchstdwindow(){

setSize(700, 400);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

setLocationRelativeTo(null);

// setResizable(false);

// setLayout(new GridLayout(2,2));

setLayout(new BorderLayout());

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

p1.setLayout(new FlowLayout());

p2.setLayout(new GridLayout(9,1));

p3.setLayout(new FlowLayout());

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

p2.setSize(30, 40);

p2.setBackground(new Color(255, 187, 0));

byname = new JCheckBox("SEARCH BY NAME");

byid = new JCheckBox("SEARCH BY ID");

// byroomseat = new JCheckBox("SEARCH BY ROOMSEAT");

byuni = new JCheckBox("SEARCH BY ISTITUTE");

byroomno = new JCheckBox("SEARCH BY ROOM NUMBER");

// bypaid = new JCheckBox("SEARCH FOR PAID HOSTELITES");

// byuni = new JCheckBox("SEARCH FOR PAID HOSTELITES");

// bybill = new JCheckBox(" ");

Font f = new Font("TIMES NEW ROMAN",1,20);

ButtonGroup c1 = new ButtonGroup();

c1.add(byid);

c1.add(byroomno);

c1.add(byname);

c1.add(byuni);

// c1.add(bypaid);

// c1.add(byunpaid);

byid.setBackground(new Color(255, 184, 0));

byroomno.setBackground(new Color(255, 180, 0));

byuni.setBackground(new Color(255, 184, 0));

byname.setBackground(new Color(255, 186, 0));

// bypaid.setBackground(new Color(255, 184, 0));

// byunpaid.setBackground(new Color(255, 186, 0));

byid.setForeground(new Color(0, 0, 0));

byuni.setForeground(new Color(0, 0, 0));

byname.setForeground(new Color(0, 0, 0));

byroomno.setForeground(new Color(0, 0, 0));

// bypaid.setForeground(new Color(0, 0, 0));

// byunpaid.setForeground(new Color(0, 0, 0));

// c1.add(byroomseat);

// c1.add(byuni);

// c1.add(bybill);

ArrayList<JCheckBox> ar1 = new ArrayList<>(4);

ar1.add(byname);

ar1.add(byroomno);

ar1.add(byuni);

// ar1.add(byroomseat);

ar1.add(byid);

// ar1.add(bypaid);

// ar1.add(byunpaid);

// ar1.add(bybill);

MyActionlistener2 m2 = new MyActionlistener2();

for (JCheckBox s: ar1){

p2.add(s);

s.addActionListener(m2);

}

name = new JLabel("ENTER NAME");

roomno = new JLabel("ENETR ROOM NO");

// roomseat = new JLabel("ENTER SEATER");

uni = new JLabel("ENTER UNIVERSITY/INSTITUTE");

id = new JLabel("ENTER ID");

fname = new JTextField(20);

froom = new JTextField(20);

fid = new JTextField(20);

// fseat = new JTextField(20);

funi = new JTextField(20);

ArrayList<JLabel> ar2 = new ArrayList<>(4);

ar2.add(name);

ar2.add(id);

ar2.add(roomno);

ar2.add(uni);

// ar2.add(roomseat);

ArrayList<JTextField> ar3 = new ArrayList<>(5);

ar3.add(fname);

ar3.add(fid);

// ar3.add(fseat);

ar3.add(froom);

ar3.add(funi);

for (int i = 0 ; i<ar2.size() ; i++){

p2.add(ar2.get(i));

p2.add(ar3.get(i));

ar3.get(i).setEnabled(false);

ar2.get(i).setFont(f);

ar2.get(i).setForeground(new Color(0, 0, 0));

ar3.get(i).setFont(f);

}

Submit = new JButton("Search");

Exit = new JButton("Exit");

Back = new JButton("Back");

MyActionlistener ma = new MyActionlistener();

Submit.addActionListener(ma);

Exit.addActionListener(ma);

Back.addActionListener(ma);

Submit.setBounds(new Rectangle(14,20));

p3.add(Submit);

p3.add(Exit);

p3.add(Back);

}

public class MyActionlistener implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Back")){

new MainWindow();

}

else if (e.getActionCommand().equalsIgnoreCase("Exit")){

System.exit(1);

}

else if (e.getActionCommand().equalsIgnoreCase("Search")){

if (byname.isSelected()){

if (fname.getText().isEmpty()){

JOptionPane.showMessageDialog(new JFrame(),"Enter name to be searched");

}

else{

new displaywindow(FileOperations.searchbyname(fname.getText()));

dispose();

}

}

else if (byroomno.isSelected()){

if (froom.getText().isEmpty()){

JOptionPane.showMessageDialog(new JFrame(),"Enter room no to be searched");

}

else{

// new newDisplaytable(froom.getText(),2);

// dispose();

ArrayList<hostelite> allsameroom = new ArrayList<hostelite>();

allsameroom= FileOperations.allsameuni(funi.getText());

new newDisplaytable(allsameroom);

dispose();

// for (hostelite s : allsameroom){

// System.out.println(s.getName());

// }

//

//

//// System.out.println("Searched");

}

}

else if (byuni.isSelected()){

if (funi.getText().isEmpty()){

JOptionPane.showMessageDialog(new JFrame(),"Enter institute to be searched");

}

else{

ArrayList<hostelite> allsame = new ArrayList<hostelite>();

allsame= FileOperations.allsameuni(funi.getText());

new newDisplaytable(allsame);

// for (hostelite s : allsame){

// System.out.println(s.getName());

// }

// System.out.println("Searched");

}

}

else if (byid.isSelected()){

if (fid.getText().isEmpty()){

JOptionPane.showMessageDialog(new JFrame(),"Enter Id to be searched");

}

else{

System.out.println(fid.getText());

new displaywindow(FileOperations.searchbyID(fid.getText()));

System.out.println("Searched");

}

}

}

}

}

public class MyActionlistener2 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (byname.isSelected()) {

fname.setEnabled(true);

funi.setEnabled(false);

fid.setEnabled(false);

froom.setEnabled(false);

fname.setBackground(new Color(241, 79, 79));

fname.setForeground(new Color(255,255,255));

} else if (byid.isSelected()) {

fname.setEnabled(false);

funi.setEnabled(false);

fid.setEnabled(true);

froom.setEnabled(false);

} else if (byroomno.isSelected()) {

fname.setEnabled(false);

funi.setEnabled(false);

fid.setEnabled(false);

froom.setEnabled(true);

}

else if(byuni.isSelected()){

fname.setEnabled(false);

funi.setEnabled(true);

fid.setEnabled(false);

froom.setEnabled(false);

}

}

}

}

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

public class Updatewindow extends JFrame {

JLabel title, name,address,mob,byid;

JTextField fname,fadress,fmob,fid;

JButton submit, exit, back, paid;

JPanel p1, p2, p3;

Updatewindow() {

setSize(700, 400);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

// setResizable(false);

// setLayout(new GridLayout(2,2));

setLayout(new BorderLayout());

p1 = new JPanel();

p2 = new JPanel();

p3 = new JPanel();

p1.setLayout(new FlowLayout());

p2.setLayout(new GridLayout(4, 2));

p3.setLayout(new FlowLayout());

add(p1, BorderLayout.NORTH);

add(p2, BorderLayout.CENTER);

add(p3, BorderLayout.SOUTH);

p2.setSize(30, 40);

p2.setBackground(new Color(27, 39, 73));

Font f = new Font("TIMES NEW ROMAN",1,18);

byid = new JLabel("Enter ID");

fid = new JTextField(20);

name = new JLabel("Enter Name");

name.setSize(15, 20);

address = new JLabel("Enter Adress ");

fname = new JTextField(20);

fadress = new JTextField(20);

mob = new JLabel("Enter Mobile Number");

fmob = new JTextField(20);

title = new JLabel("CAPITAL VIEW BOYS HOSTEL");

title.setFont(f);

title.setBackground(new Color(17, 32, 47));

p1.add(title);

title.setForeground(new Color(255, 0, 0));

ArrayList<JTextField> tf = new ArrayList<JTextField>();

ArrayList<JLabel> JB = new ArrayList<JLabel>();

p3.setBackground(new Color(26, 138, 33));

JB.add(byid);

tf.add(fid);

JB.add(name);

tf.add(fname);

JB.add(address);

tf.add(fadress);

JB.add(mob);

tf.add(fmob);

for (int i = 0 ; i<tf.size() ; i++){

p2.add(JB.get(i));

JB.get(i).setFont(f);

JB.get(i).setForeground(new Color(255,255,255));

p2.add(tf.get(i));

tf.get(i).setFont(f);

tf.get(i).setForeground(new Color(0, 0, 0));

}

submit = new JButton("Submit");

exit = new JButton("Exit");

back = new JButton("Back");

p3.add(submit);

p3.add(exit);

p3.add(back);

MyActionlistener5 up = new MyActionlistener5();

submit.addActionListener(up);

back.addActionListener(up);

exit.addActionListener(up);

}

public class MyActionlistener5 implements ActionListener {

//

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equalsIgnoreCase("Submit")){

if(fid.getText().isEmpty()){

JOptionPane.showMessageDialog(new JFrame(),"Eneter ID");

}

else{

if (!(fname.getText().isEmpty())){

if(!(FileOperations.searchbyID(fid.getText())==null)){

System.out.println("nahi milla");

FileOperations.updatename(fid.getText(),fname.getText());

JOptionPane.showMessageDialog(new JFrame(),"UPDATED SUCCESSFULLY");

}

else{

JOptionPane.showMessageDialog(new JFrame(),"RECORD NOT FOUNDED");

}

}

if (!(fmob.getText().isEmpty())){

FileOperations.updatemob(fid.getText(),fmob.getText());

}

if (!(fadress.getText().isEmpty())){

FileOperations.updateadress(fid.getText(),fadress.getText());

}

}

} else if (e.getActionCommand().equalsIgnoreCase("Exit")) {

System.exit(1);

} else if (e.getActionCommand().equalsIgnoreCase("Back")) {

new MainWindow();

}

}

}

}