IMPLEMENTATION OF RENTAL SYSTEM

* Class USER

This class contains all the details of users. There are two main functionalities, **Login** and **SignUp.**

**Login :** To login user enters username (unique) and password.

User will be shown the functionalities of Hirer / Owner / Technician accordingly.

**SignUp :** New user first need to signup to use RENTAL SYSTEM.

static boolean login(String userName,String password){

if(DB.userDB\_pw.containsKey(userName)){

if(DB.userDB\_pw.get(userName).equals(password)){

System.out.println("Login Successful");

System.out.println("You have logged in as "+DB.user\_type.get(userName));

Main.usr\_type = DB.user\_type.get(userName);

return true;

}else{

System.out.println("Login Unsuccessful");

System.out.println("Check if password is correct");

return false;

}

}else{

System.out.println("Login Unsuccessful");

System.out.println("Check if username is correct");

return false;

}

}

static boolean signUp(String na,String ad,String uN,long c,String em,String pw,String type){

User u = new User();

if(DB.userDB\_pw.containsKey(uN)){

System.out.println("Given username is already taken. Please select another one");

return false;

}

u.setName(na);

u.setUserName(uN);

u.setContact(c);

u.setRating(0);

u.setEmail(em);

u.setPassword(pw);

Location lu = new Location();

if(DB.userDB.containsKey(u.userName)){

User u1 = DB.userDB.get(u.userName);

ArrayList<String> addr = u1.getL().getAddress();

addr.add(ad);

lu.setAddress(addr);

}else{

ArrayList<String> addr = new ArrayList<String>();

addr.add(ad);

lu.setAddress(addr);

}

u.setL(lu);

System.out.println("Account successfully created as "+type);

DB.ul\_Db.put(u.userName, lu);

DB.userDB.put(u.userName, u);

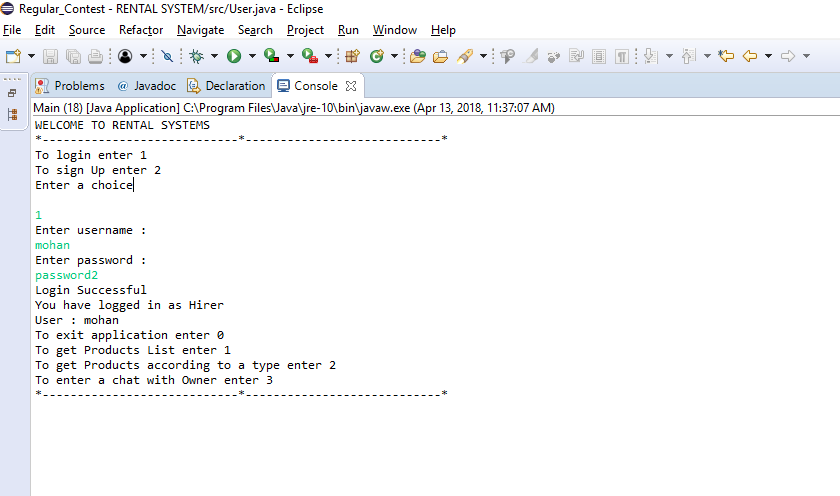
DB.userDB\_pw.put(u.userName, pw);

DB.user\_type.put(u.userName, type);

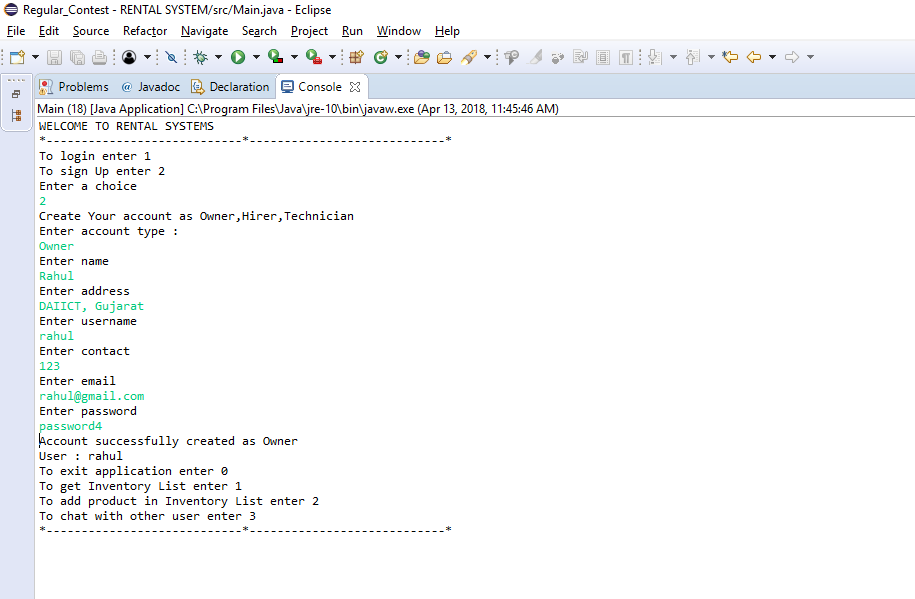
return true;

}

**LOGIN**



**SignUp**

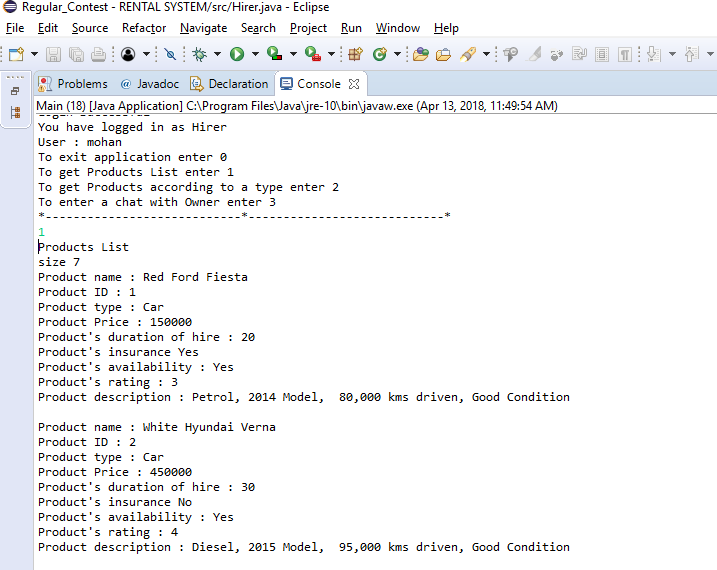


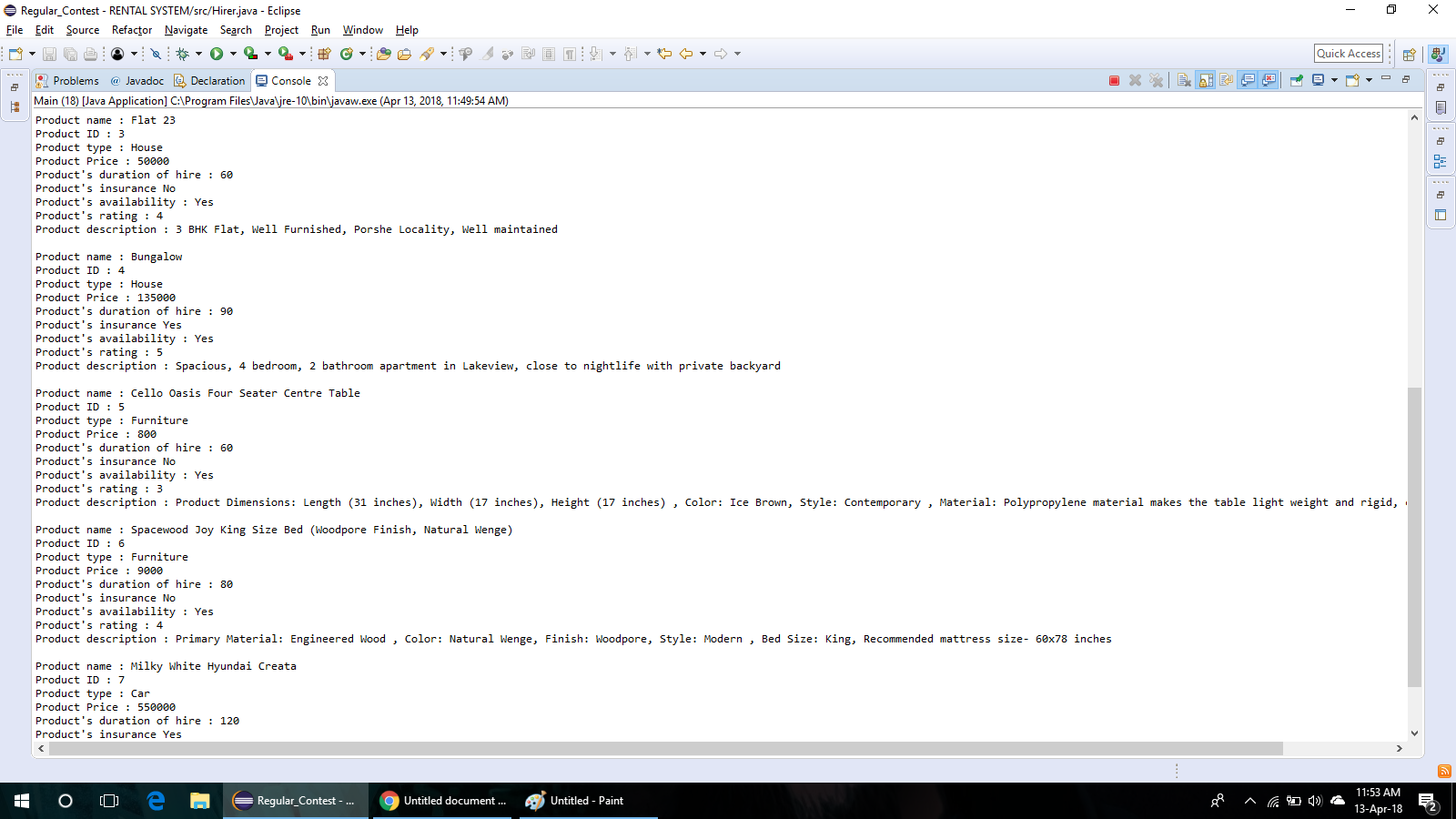
* Class Hirer

This class extends user class and has 3 main functionalities.

Functionalities :

1. Get Product List : This function shows all the products that are available to hirer.





Source code :

static void getProducts(){

Set<Integer> prid\_set = DB.pr\_DB.keySet();

System.out.println("size "+prid\_set.size());

for(int i : prid\_set){

Product p = DB.pr\_DB.get(i);

System.out.println("Product name : "+p.name);

System.out.println("Product ID : "+p.productID);

System.out.println("Product type : "+p.typeOfProduct);

System.out.println("Product Price : "+p.price);

System.out.println("Product's duration of hire : "+p.durationOfHire);

System.out.println("Product's insurance "+p.Insurance);

System.out.println("Product's availability : "+p.availability);

System.out.println("Product's rating : "+p.rating);

System.out.println("Product description : "+p.description);

System.out.println();

}

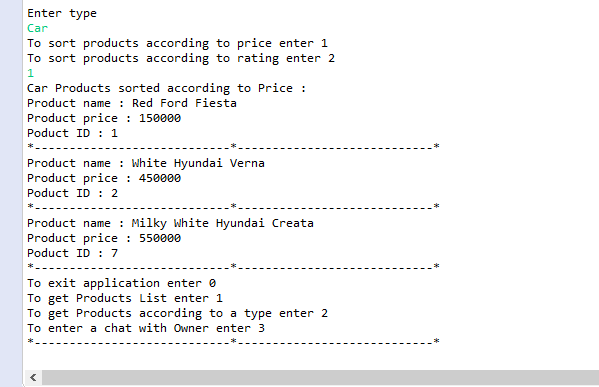
}

2) Get Products according to type : Shows products list of a particular type

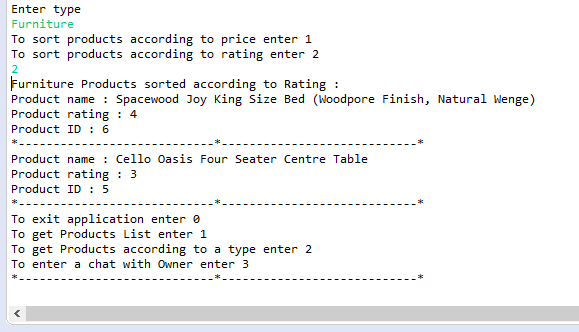
For example, type can be Car,Furniture, etc.

3) Get Products according to a particular order

* Sorting according to price



* Sorting according to rating



Source Code :

static void getProductsSortedPrice(String type){

Set<Integer> prid\_set = DB.pr\_DB.keySet();

ArrayList<Product> pr = new ArrayList<Product>();

for(int i : prid\_set){

Product p = DB.pr\_DB.get(i);

if(type.equals(p.typeOfProduct)){

pr.add(p);

}

}

Collections.sort(pr,new Comparator<Product>(){

public int compare(Product p1,Product p2){

return p1.price - p2.price;

}

});

System.out.println(type+" Products sorted according to Price : ");

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

Product p = pr.get(i);

System.out.println("Product name : "+p.name);

System.out.println("Product price : "+p.price);

System.out.println("Poduct ID : "+p.productID);

Main.drawLine();

}

}

static void getProductsSortedRating(String type){

Set<Integer> prid\_set = DB.pr\_DB.keySet();

ArrayList<Product> pr = new ArrayList<Product>();

for(int i : prid\_set){

Product p = DB.pr\_DB.get(i);

if(type.equals(p.typeOfProduct)){

pr.add(p);

}

}

Collections.sort(pr,new Comparator<Product>(){

public int compare(Product p1,Product p2){

return p2.rating - p1.rating;

}

});

System.out.println(type+" Products sorted according to Rating : ");

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

Product p = pr.get(i);

System.out.println("Product name : "+p.name);

System.out.println("Product rating : "+p.rating);

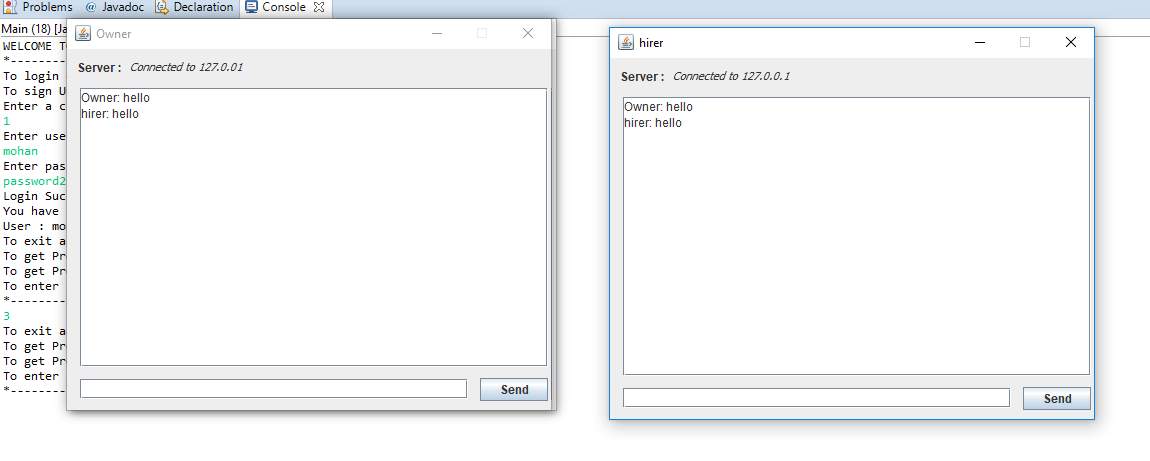
System.out.println("Product ID : "+p.productID);

Main.drawLine();

}

}

3) Chat with Owner : This function allows Hirer to chat with Owner.This function opens GUI chat-app in which we have created a server and two clients.Server is hosted on local-host.



Source code :

public StartServer() {

setResizable(false);

setTitle("Create Server");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(250, 350);

this.setLocationRelativeTo(null);

contentPane = new JPanel();

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(null);

JLabel lblName = new JLabel("Name:");

lblName.setBounds(103, 32, 38, 14);

contentPane.add(lblName);

name = new JTextField();

name.setToolTipText("viki");

name.setBounds(56, 51, 132, 20);

contentPane.add(name);

name.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e){

port.grabFocus();

}

});

name.setColumns(10);

JLabel lblPort = new JLabel("Port:");

lblPort.setBounds(106, 95, 31, 14);

contentPane.add(lblPort);

port = new JTextField();

port.setToolTipText("Ex. 8191");

port.setBounds(79, 113, 86, 20);

contentPane.add(port);

port.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e){

if(isValidPort())

showChatWindow();

else{

JOptionPane.showMessageDialog(null, "Please enter a valid port number. Ex.:8080");

port.grabFocus();

}

}

});

port.setColumns(10);

btnLogin = new JButton("Start");

btnLogin.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

if(isValidPort())

showChatWindow();

else{

JOptionPane.showMessageDialog(null, "Please enter a valid port number. Ex.:8080");

port.grabFocus();

}

}

});

btnLogin.setBounds(84, 239, 75, 23);

contentPane.add(btnLogin);

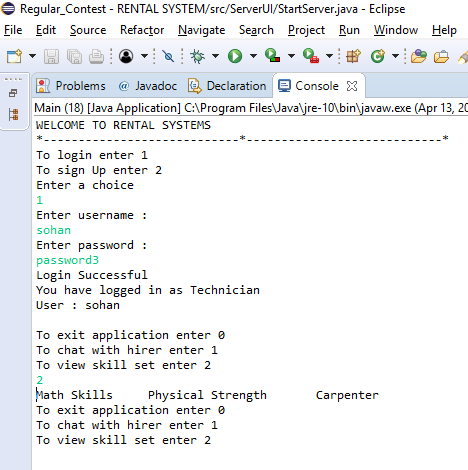
}

* Class Technician

This class extends user class and has 2 main functionalities.

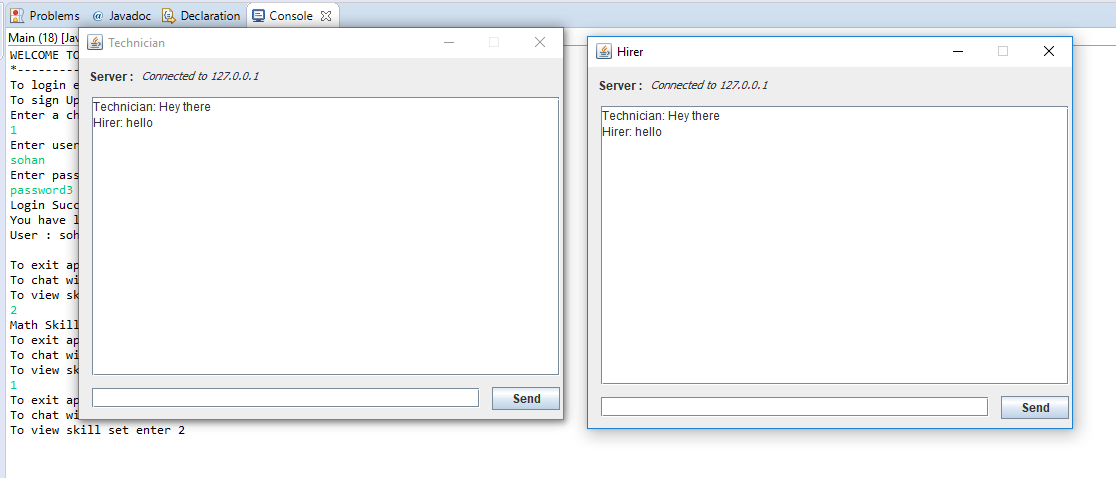
Functionalities are :

1. To add skill set of Technician : This functionality allows technician to add another skill to his skill-set.



2. Chat with hirer : This method allows technician and hirer to chat

with each other.



Source code :

public StartServer() {

setResizable(false);

setTitle("Create Server");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(250, 350);

this.setLocationRelativeTo(null);

contentPane = new JPanel();

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(null);

JLabel lblName = new JLabel("Name:");

lblName.setBounds(103, 32, 38, 14);

contentPane.add(lblName);

name = new JTextField();

name.setToolTipText("viki");

name.setBounds(56, 51, 132, 20);

contentPane.add(name);

name.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e){

port.grabFocus();

}

});

name.setColumns(10);

JLabel lblPort = new JLabel("Port:");

lblPort.setBounds(106, 95, 31, 14);

contentPane.add(lblPort);

port = new JTextField();

port.setToolTipText("Ex. 8191");

port.setBounds(79, 113, 86, 20);

contentPane.add(port);

port.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e){

if(isValidPort())

showChatWindow();

else{

JOptionPane.showMessageDialog(null, "Please enter a valid port number. Ex.:8080");

port.grabFocus();

}

}

});

port.setColumns(10);

btnLogin = new JButton("Start");

btnLogin.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

if(isValidPort())

showChatWindow();

else{

JOptionPane.showMessageDialog(null, "Please enter a valid port number. Ex.:8080");

port.grabFocus();

}

}

});

btnLogin.setBounds(84, 239, 75, 23);

contentPane.add(btnLogin);

}