UML DIAGRAM

|  |
| --- |
| Item |
| -iD: String  -type: String  -time: String  -location: String  -description: String  -phonenumber: String |
| +getID() : String  +getType() : String  +getTime() : String  +getLocation() : String  +getDescription() : String  +getPhonenumber() : String  +setType(String name) : void  +setDescription(String name) : void |

**Item class**

package missing;

public class Item {

private final String iD, time, location, phonenumber;

private String type, description;

Item(String iD, String type, String time, String location, String description, String phonenumber) {

this.iD = iD;

this.type = type;

this.time = time;

this.location = location;

this.description = description;

this.phonenumber = phonenumber;

}

String getID() {

return iD;

}

String getType() {

return type;

}

String getTime() {

return time;

}

String getLocation() {

return location;

}

String getDescription() {

return description;

}

String getPhonenumber() {

return phonenumber;

}

void setType(String type) {

this.type = type;

}

void setDescription(String description) {

this.description = description;

}

}

**Login class**

package missing;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.util.\*;

public class Login extends JFrame {

JButton blogin = new JButton("Login");

JPanel panel = new JPanel();

JTextField txuser = new JTextField(15);

JPasswordField pass = new JPasswordField(15);

Login() {

super("Please Enter Name and Password!!!");

setSize(500, 280);

setLocation(350, 300);

panel.setLayout(null);

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

textuser.setBounds(60, 28, 85, 20);

panel.add(textuser);

JLabel password = new JLabel("Password :");

password.setBounds(60, 63, 70, 20);

panel.add(password);

txuser.setBounds(150, 30, 200, 20);

pass.setBounds(150, 65, 200, 20);

blogin.setBounds(180, 100, 120, 20);

panel.add(blogin);

panel.add(txuser);

panel.add(pass);

getContentPane().add(panel);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

setResizable(false);

actionlogin();

}

public void actionlogin() {

blogin.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent ae) {

String puname = txuser.getText();

String ppaswd = pass.getText();

if (puname.equals("izzaq") && ppaswd.equals("123456")) {

Login regFace = new Login();

regFace.setVisible(false);

dispose();

} else {

JOptionPane.showMessageDialog(null, "Wrong Password / Username");

txuser.setText("");

pass.setText("");

txuser.requestFocus();

}

}

});

}

public static void main(String[] args) {

Login login = new Login();

}

}

**Source Code**

package missing;

import java.awt.Color;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.io.File;

import java.awt.FlowLayout;

import javax.swing.JTextArea;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class Missing {

ArrayList<Item> missinglist = new ArrayList<Item>();

String header[] = new String[]{"ID", "Item Type", "Time Found", "Location Found", "Description", "Phone Number"};

DefaultTableModel dtm = new DefaultTableModel(header, 1);

Missing() {

JFrame frame = new JFrame("Missing Item System");

frame.setSize(640, 600);

JLabel jlabel1 = new JLabel("ID Number :");

jlabel1.setBounds(40, 10, 90, 20);

frame.add(jlabel1);

JLabel jlabel2 = new JLabel("Item Type :");

jlabel2.setBounds(40, 32, 90, 20);

frame.add(jlabel2);

JLabel jlabel3 = new JLabel("Time Found :");

jlabel3.setBounds(40, 52, 90, 20);

frame.add(jlabel3);

JLabel jlabel4 = new JLabel("Location Found :");

jlabel4.setBounds(40, 74, 150, 20);

frame.add(jlabel4);

JLabel jlabel5 = new JLabel("Description :");

jlabel5.setBounds(40, 96, 90, 20);

frame.add(jlabel5);

JLabel jlabel6 = new JLabel("Phone Number :");

jlabel6.setBounds(40, 150, 90, 20);

frame.add(jlabel6);

JTextField jtfiD = new JTextField();

jtfiD.setBounds(150, 10, 400, 20);

frame.add(jtfiD);

JTextField jtfType = new JTextField();

jtfType.setBounds(150, 32, 400, 20);

frame.add(jtfType);

JTextField jtfTime = new JTextField();

jtfTime.setBounds(150, 54, 400, 20);

frame.add(jtfTime);

JTextField jtfLocation = new JTextField();

jtfLocation.setBounds(150, 76, 400, 20);

frame.add(jtfLocation);

JTextField jtfDescription = new JTextField();

jtfDescription.setBounds(150, 98, 400, 50);

frame.add(jtfDescription);

JTextField jtfPhonenumber = new JTextField();

jtfPhonenumber.setBounds(150, 150, 400, 20);

frame.add(jtfPhonenumber);

JButton jbuttonAdd = new JButton("INSERT");

jbuttonAdd.setBounds(200, 200, 200, 20);

frame.add(jbuttonAdd);

JButton jbuttondelete = new JButton("DELETE");

jbuttondelete.setBounds(200, 225, 200, 20);

frame.add(jbuttondelete);

JButton jbuttonsearch = new JButton("SEARCH");

jbuttonsearch.setBounds(200, 250, 200, 20);

frame.add(jbuttonsearch);

JButton jbuttonupdate = new JButton("UPDATE");

jbuttonupdate.setBounds(200, 275, 200, 20);

frame.add(jbuttonupdate);

JButton jbuttonexit = new JButton("EXIT");

jbuttonexit.setBounds(260, 490, 90, 20);

frame.add(jbuttonexit);

JTable jtable = new JTable();

jtable.setBounds(10, 320, 615, 150);

frame.add(jtable);

jtable.setModel(dtm);

JScrollPane scrollPane = new JScrollPane(jtable);

scrollPane.setBounds(10, 320, 615, 150);

new JScrollPane(scrollPane, JScrollPane.VERTICAL\_SCROLLBAR\_AS\_NEEDED, JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);

jtable.setAutoResizeMode(JTable.AUTO\_RESIZE\_OFF);

frame.add(scrollPane);

jtable.getColumnModel().getColumn(0).setPreferredWidth(70);

jtable.getColumnModel().getColumn(1).setPreferredWidth(115);

jtable.getColumnModel().getColumn(2).setPreferredWidth(125);

jtable.getColumnModel().getColumn(3).setPreferredWidth(100);

jtable.getColumnModel().getColumn(4).setPreferredWidth(120);

jtable.getColumnModel().getColumn(5).setPreferredWidth(145);

jtable.setBackground(Color.LIGHT\_GRAY);

jbuttonAdd.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent ae) {

String iD = jtfiD.getText();

String type = jtfType.getText();

String time = jtfTime.getText();

String location = jtfLocation.getText();

String description = jtfDescription.getText();

String phonenumber = jtfPhonenumber.getText();

if (description.length() < 3) {

JOptionPane.showMessageDialog(frame, "Description should contain more than 3 char!");

return;

}

if (!isInteger(iD)) {

JOptionPane.showMessageDialog(frame, "ID should only contain integer!!!");

return;

}

Item item = new Item(iD, type, time, location, description, phonenumber);

missinglist.add(item);

writeData();

}

});

jbuttondelete.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent ae) {

String iD = JOptionPane.showInputDialog("ID Number to delete?");

if (iD != null) {

System.out.println("not null");

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

if (missinglist.get(i).getID().equalsIgnoreCase(iD)) {

missinglist.remove(i);

}

}

writeData();

}

}

});

jbuttonsearch.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent ae) {

String iD = JOptionPane.showInputDialog("Enter ID Number?");

if (iD != null) {

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

if (missinglist.get(i).getID().equalsIgnoreCase(iD)) {

JOptionPane.showMessageDialog(frame, "Found!!!");

jtfiD.setText(missinglist.get(i).getID());

jtfType.setText(missinglist.get(i).getType());

jtfTime.setText(missinglist.get(i).getTime());

jtfLocation.setText(missinglist.get(i).getLocation());

jtfDescription.setText(missinglist.get(i).getDescription());

jtfPhonenumber.setText(missinglist.get(i).getPhonenumber());

return;

}

}

JOptionPane.showMessageDialog(frame, "Not Found!!!");

}

}

});

jbuttonupdate.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent ae) {

String iD = jtfiD.getText();

String type = jtfType.getText();

String time = jtfTime.getText();

String location = jtfLocation.getText();

String description = jtfDescription.getText();

String phonenumber = jtfPhonenumber.getText();

if (iD != null) {

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

if (missinglist.get(i).getID().equalsIgnoreCase(iD)) {

missinglist.get(i).setType(type);

missinglist.get(i).setDescription(description);

JOptionPane.showMessageDialog(frame, "Updated!!!");

}

}

}

writeData();

}

});

jbuttonexit.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent ae) {

frame.setVisible(false);

frame.dispose();

}

});

readData();

jtable.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

int row = jtable.getSelectedRow();

jtfiD.setText(dtm.getValueAt(row, 0).toString());

jtfType.setText(dtm.getValueAt(row, 1).toString());

jtfTime.setText(dtm.getValueAt(row, 2).toString());

jtfLocation.setText(dtm.getValueAt(row, 3).toString());

jtfDescription.setText(dtm.getValueAt(row, 4).toString());

jtfPhonenumber.setText(dtm.getValueAt(row, 5).toString());

}

});

frame.setResizable(false);

frame.setLocationRelativeTo(null);

frame.setLayout(null);

frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

public static void main(String[] args) {

Missing missing = new Missing();

Login login = new Login();

}

void readData() {

try {

File file = new File("data.txt");

file.createNewFile();

FileReader f = new FileReader("data.txt");

StringBuffer sb = new StringBuffer();

while (f.ready()) {

char c = (char) f.read();

if (c == '-') {

System.out.println(sb);

String missingArray[] = sb.toString().split(",");

Item item = new Item(missingArray[0], missingArray[1], missingArray[2], missingArray[3], missingArray[4], missingArray[5]);

missinglist.add(item);

sb = new StringBuffer();

} else {

sb.append(c);

}

}

dtm.setRowCount(0);

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

Object[] objs = {missinglist.get(i).getID(), missinglist.get(i).getType(), missinglist.get(i).getTime(), missinglist.get(i).getLocation(), missinglist.get(i).getDescription(), missinglist.get(i).getPhonenumber()};

dtm.addRow(objs);

}

} catch (IOException e) {

}

}

private void writeData() {

try (FileWriter f = new FileWriter("data.txt")) {

StringBuilder sb = new StringBuilder();

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

sb.append(missinglist.get(i).getID() + "," + missinglist.get(i).getType() + "," + missinglist.get(i).getTime() + "," + missinglist.get(i).getLocation() + "," + missinglist.get(i).getDescription()+ "," + missinglist.get(i).getPhonenumber()+ "-");

}

f.write(sb.toString());

f.close();

} catch (IOException e) {

return;

}

dtm.setRowCount(0);

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

Object[] objs = {missinglist.get(i).getID(), missinglist.get(i).getType(), missinglist.get(i).getTime(), missinglist.get(i).getLocation(), missinglist.get(i).getDescription(), missinglist.get(i).getPhonenumber()};

dtm.addRow(objs);

}

}

public boolean isInteger(String str) {

if (str == null) {

return false;

}

int length = str.length();

if (length == 0) {

return false;

}

int i = 0;

if (str.charAt(0) == '-') {

if (length == 1) {

return false;

}

i = 1;

}

for (; i < length; i++) {

char c = str.charAt(i);

if (c < '0' || c > '9') {

return false;

}

}

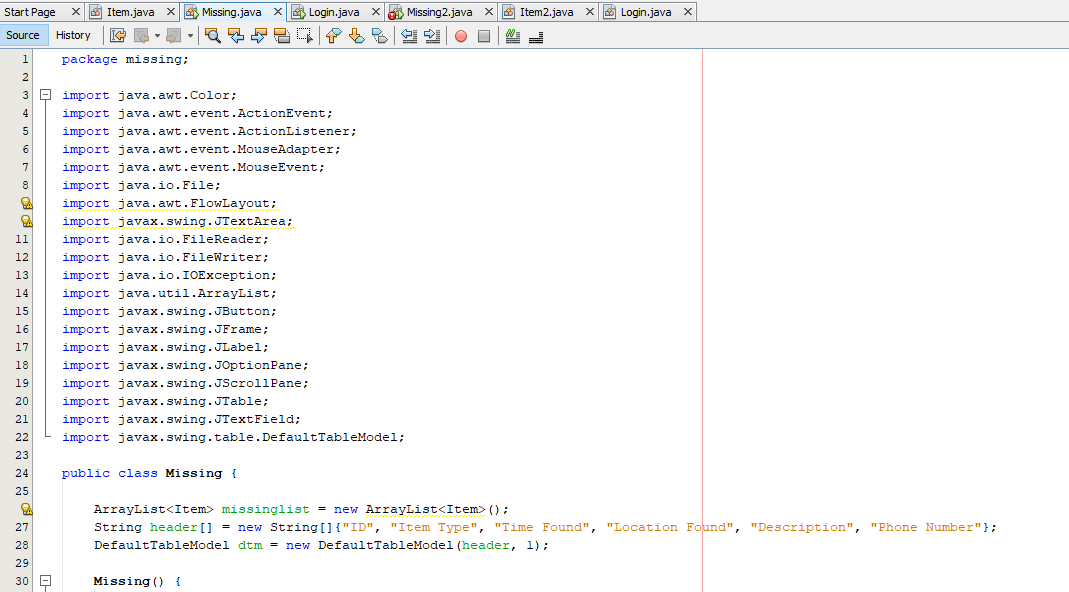
return true;

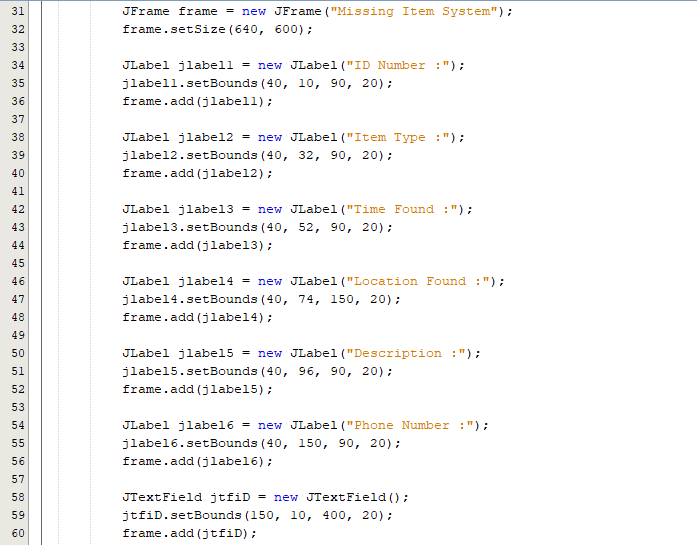
}

}

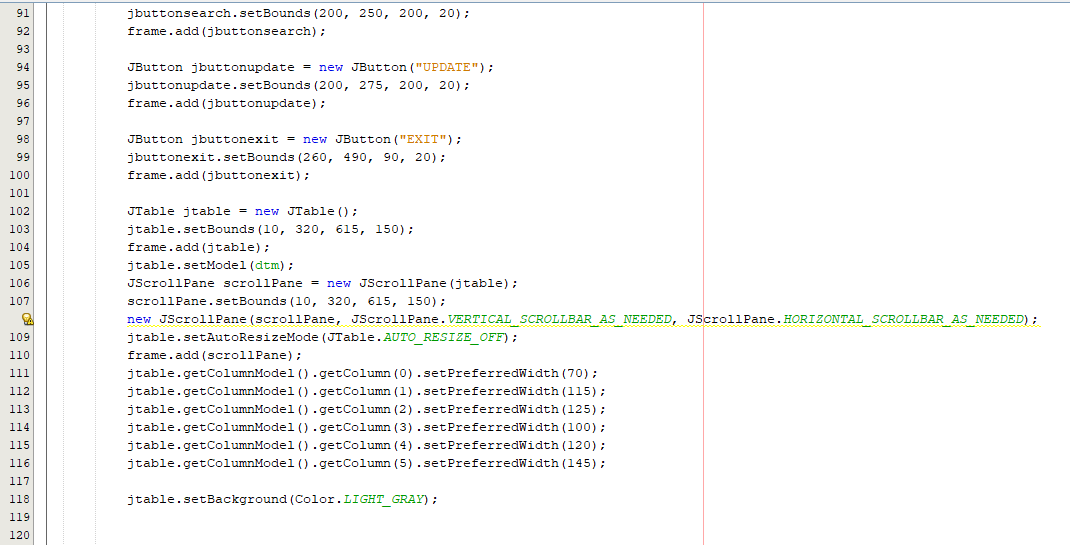
**Screenshot**

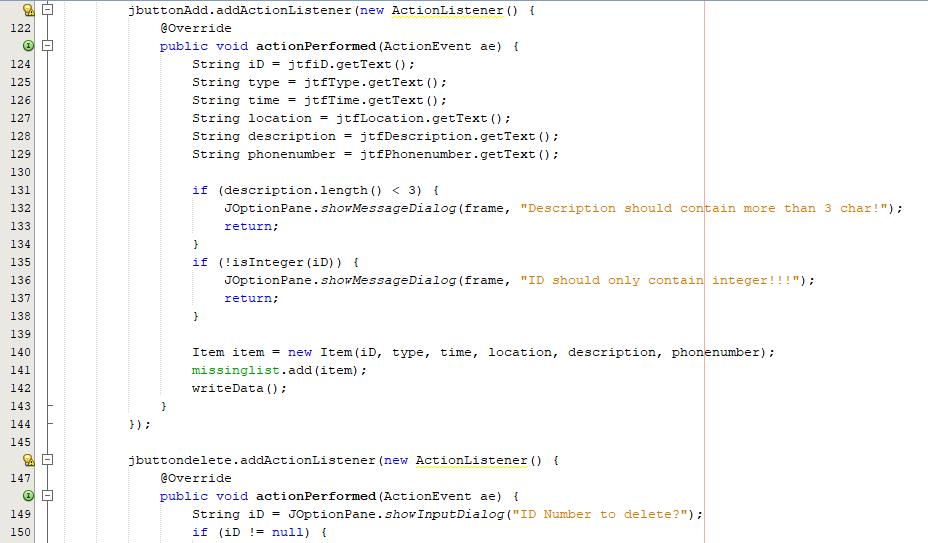
**Source code**

****

****

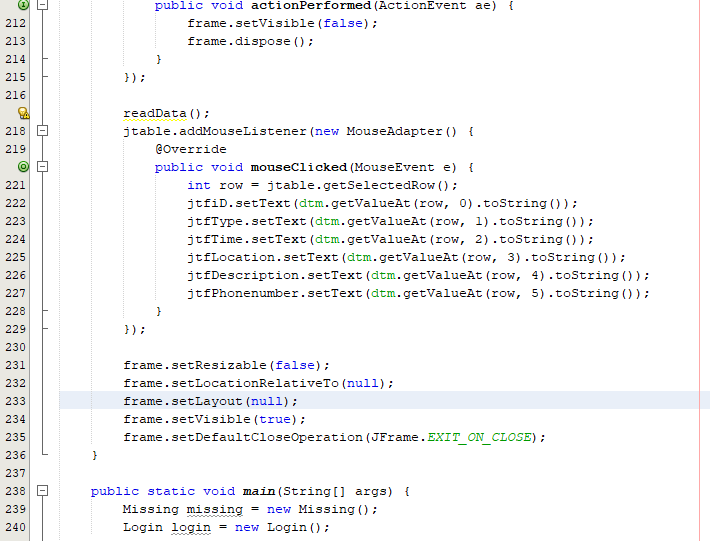
****

****

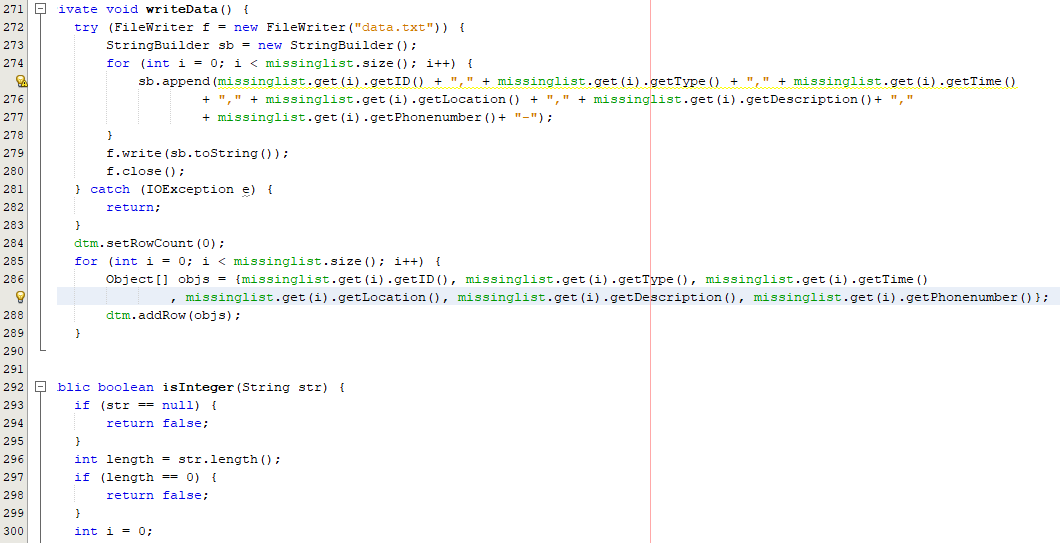
****

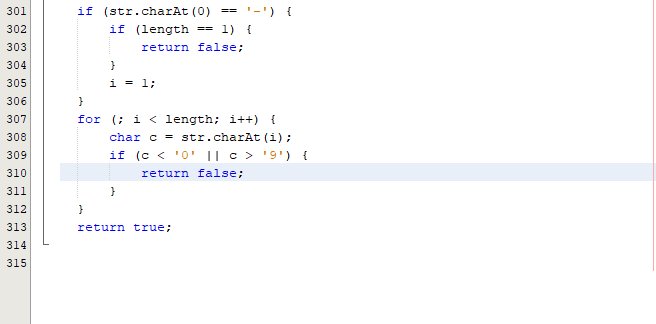
****

****

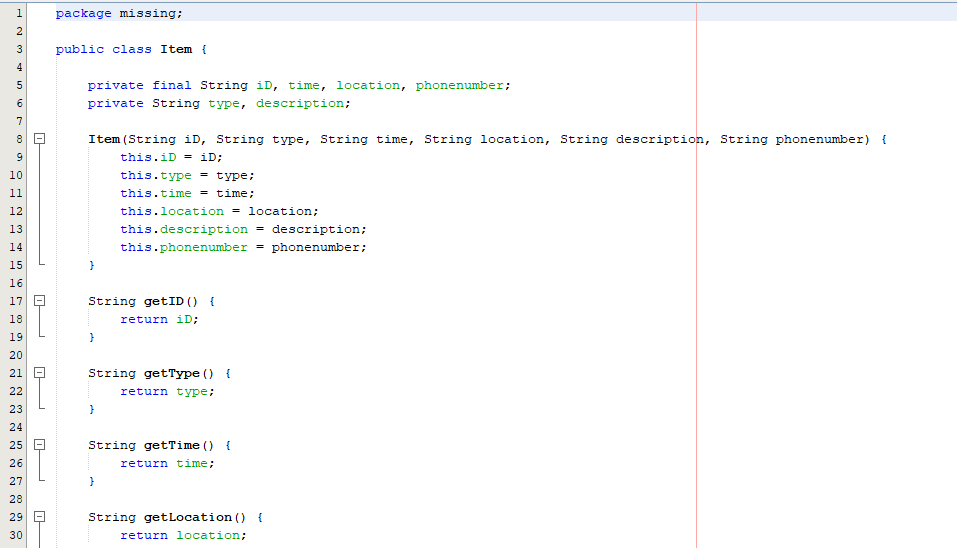
****

****

****

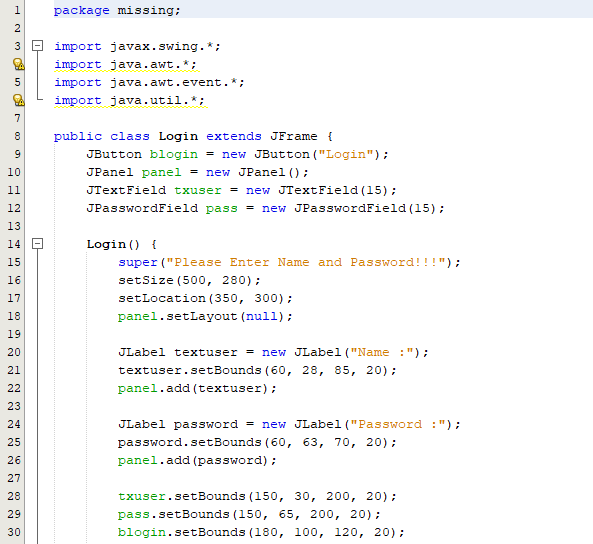
****

**Screenshot class Item**

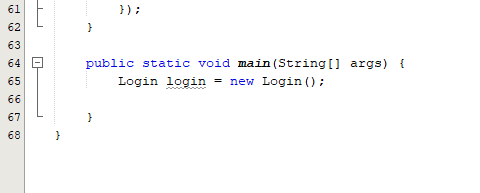
****

****

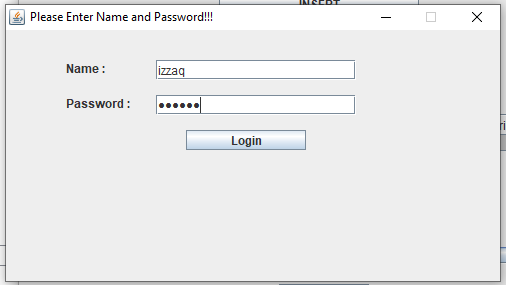
**Screenshot class Login**

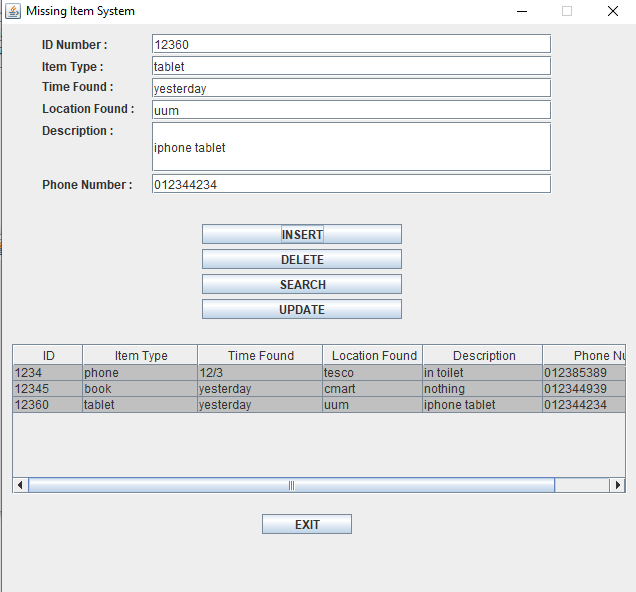
****

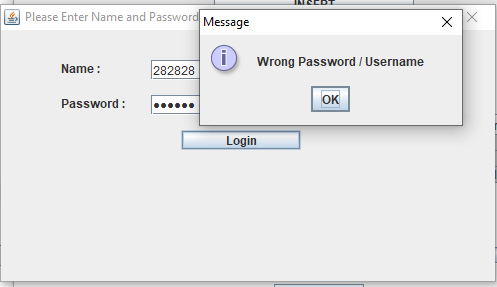
****

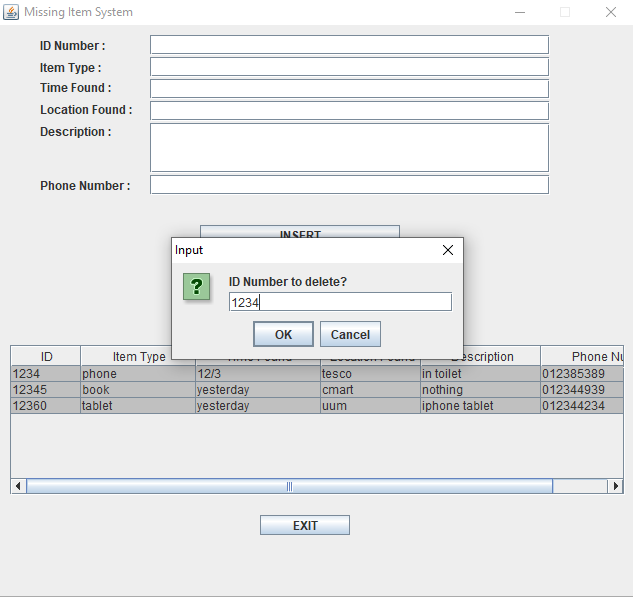
****

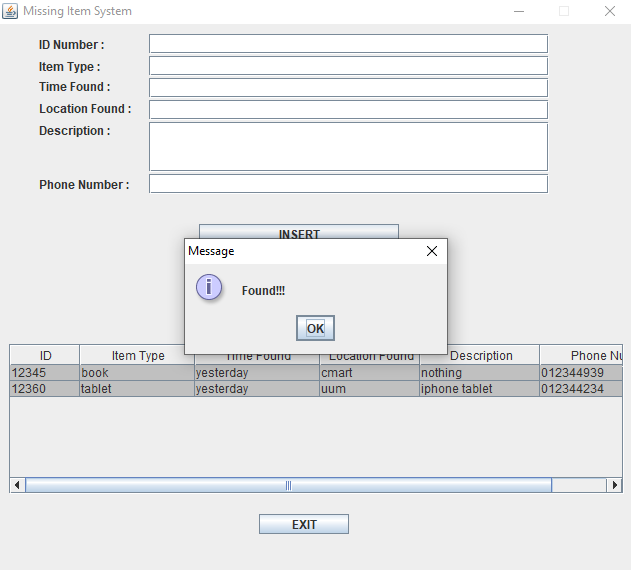
**Screenshot interface**

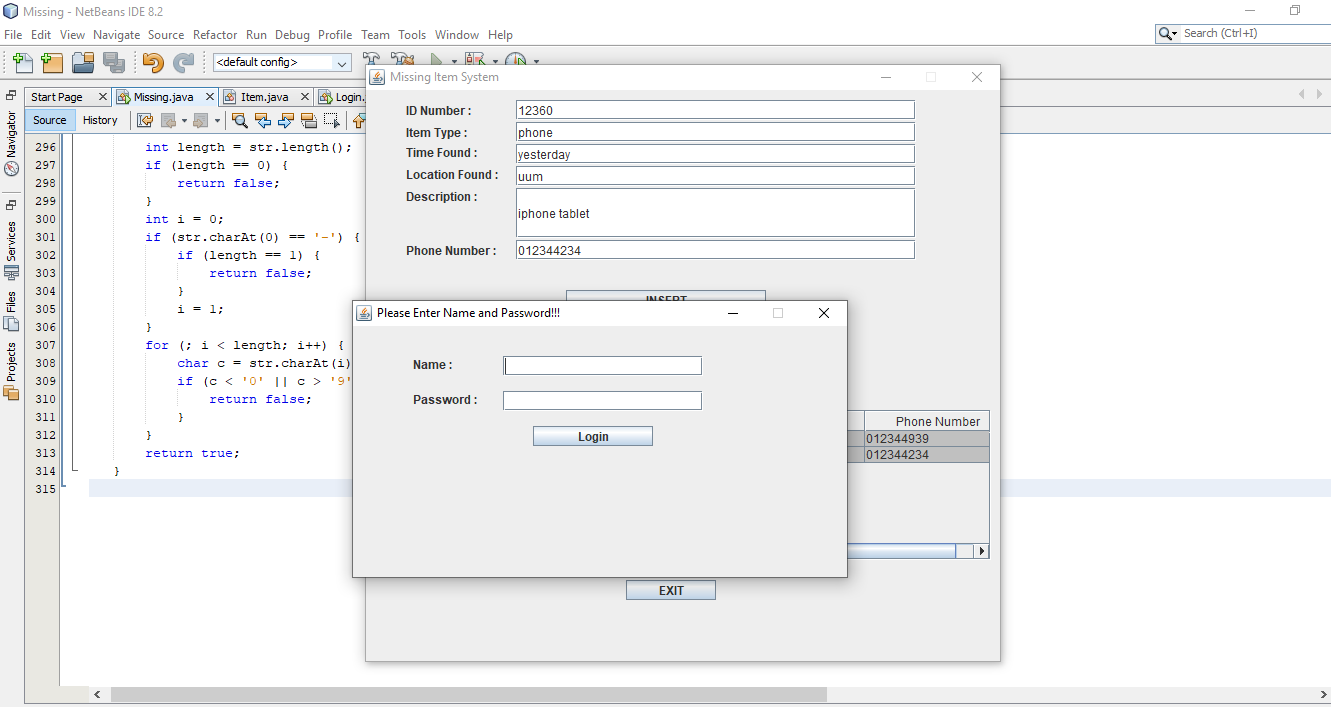
****

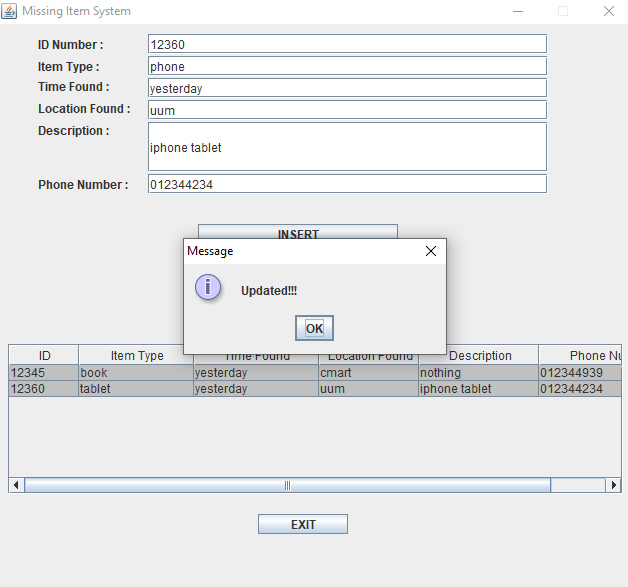
****

****

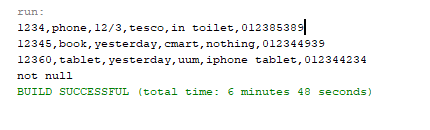
****

****

****

****

**Output**

****